Complex Event Processing und maschinelle Lernverfahren

Entwicklung eines Hybrid-Modells zur Erkennung von Identitätsdiebstahl beim Online-Banking

Inaugural-Dissertation zur Erlangung der Doktorwürde der Fakultät für Sprach-, Literatur- und Kulturwissenschaften der Universität Regensburg

vorgelegt von

Alexander Widder

aus

Teublitz

2010

Regensburg, 2011

Erstgutachter: Prof. Dr. Christian Wolff

Zweitgutachter: Prof. Dr. Rainer Hammwöhner

DANKSAGUNG

Für das Gelingen meiner Doktorarbeit schulde ich vielen Menschen meinen herzlichen Dank. Vor allem möchte ich mich bei meinem Doktorvater Herrn Prof. Dr. Christian Wolff bedanken, der mit sehr viel Geduld und vielen wertvollen Ratschlägen stark dazu beigetragen hat, dass diese Arbeit ihren Weg ins Ziel fand. Ein besonderes Wort des Dankes möchte ich in diesem Zusammenhang auch an meinen langjährigen Mentor Herrn Dr. Rainer von Ammon richten, der immer ein offenes Ohr für mich hatte und mir mit seinem Rat zur Seite stand. Ebenso danke ich meinen Interviewpartnern ohne deren Fach- und Insiderwissen im Bereich Bankbetrug das Ergebnis dieser Arbeit in dieser Form nicht zustande gekommen wäre. Mein Dank geht ebenfalls an Frau Sarah Schumann, die mich bei der Literaturrecherche unterstützt hat.

Abschließend möchte ich noch meinen Eltern Renate und Maximilian Widder sowie meiner Frau Gabriele Widder ein herzliches Dankeschön sagen, die mich stets bestärkt haben, wenn ich an mir gezweifelt habe. Meine Frau Gabriele zeigte immer großes Verständnis, wenn ich meine Zeit dieser Arbeit gewidmet habe und hat mir die ganze Zeit stets den Rücken frei gehalten. Daher widme ich ihr und unserem Sohn Tobias diese Arbeit.

ZUSAMMENFASSUNG

Betrugserkennung ist im Bankenbereich ein wichtiges Thema. Betrug tritt in diesem Umfeld in unterschiedlichen Formen auf, wie beispielsweise organisierte Geldwäsche, Geldautomatenmanipulation oder Identitätsdiebstahl beim Online-Banking. Die Analyse der Banktransaktionen kann aufgrund des hohen täglichen Transaktionsvolumens der Kreditinstitute nicht von Hand bewältigt werden. Das Ziel dieser Arbeit ist in diesem Zusammenhang die automatische Entdeckung von Betrugstransaktionen innerhalb der kompletten Transaktionsmenge bei Banken mit Hilfe des Einsatzes eines Hybrid-Modells aus maschinellen Lernverfahren und *Complex Event Processing*-Technologie in Echtzeit. Die Betrugstransaktionen sind die Folge von erfolgreich durchgeführten Identitätsdiebstählen beim Online-Banking durch Betrugsmethoden wie Phishing, Pharming oder Trojanische Pferde. Die Anzahl dieser Betrugsfälle in diesem Bereich nimmt auch in Deutschland von Jahr zu Jahr stetig zu.

Diese Transaktionen bzw. Transaktionsevents beinhalten bestimmte Attribute, deren Werte bei der Betrugsanalyse untersucht werden, wie z.B. Transaktionsbetragshöhe, Verhältnis von Transaktionsbetrag zum maximal verfügbaren Betrag oder das Vorliegen einer Auslandsüberweisung. Konkret soll erforscht werden, mit welcher Wahrscheinlichkeit eine aktuell untersuchte Online-Banking Transaktion eines Kunden im Vergleich mit historischen Transaktionen dieses Kunden, einen Betrugsfall darstellt. Bei dieser Arbeit wird die Kombination von drei maschinellen Lernverfahren (Entscheidungsbaum, Diskriminanzanalyse und neuronales Netzwerk) als Betrugserkennungskomponente verwendet. Diese Algorithmen werden im Zusammenspiel mit der relativ neuen Technologie *Complex Event Processing* in Form eines Hybrid-Modells eingesetzt um solche Betrugsszenarien in Echtzeit erkennen und präventiv handeln zu können. Die *Complex Event Processing*-Technologie dient dabei als Daten- bzw. Eventversorgungskomponente für die Betrugserkennungskomponente.

Im Rahmen dieser Arbeit wurde das gesamte Hybrid-Modell zur Betrugsanalyse als Prototyp mit Hilfe der Programmiersprache Java unter Verwendung der *Complex Event Processing*-Lösung des Herstellers StreamBase Inc. namens *StreamBase Studio* implementiert und getestet. Auf Basis simulierter Trainings- und Testevents erreichte das Modell im Optimalfall eine Erkennungsgenauigkeit von über 99% richtig klassifizierter Transaktionsevents.

ABSTRACT

In the context of online banking, identity fraud based on phishing, pharming, trojan horses etc. is a growing problem in Germany. The fraud analysis can not be performed by a human operator because of the high amount of transactions per day. The aim of this work is to identify fraud transactions out of the amount of all transactions in real time by using a hybrid model consisting of event processing technology and machine learning algorithms. A transaction includes attributes just as the transaction amount or destination account etc. that values have to be analysed for fraud detection. By using a combination of the machine learning algorithms decision tree, discriminant analysis and neural network the probability of a transaction of being a fraud attempt in comparison to historic transactions of a specific customer will be calculated. In this context, complex event processing technology is needed to deliver the transaction events for the analysis and to be able to react on identified fraud transactions in real time.

In order to obtain experimental results, a prototype of the hybrid model is generated by programming the machine learning algorithms with java and integrating them into the CEP engine of the vendor StreamBase Inc. called StreamBase Studio. The results show a recognition accuracy of more than 99% on the base of simulated fraud and non-fraud transactions.

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

API	Application Programming Interfaces
BAM	Business Activity Monitoring
BART	Bayesian Additive Regression Trees
BI	Business Intelligence
BIC	Bank Identifier Code
BPM	Business Process Management
CAAMAT	Comprehensive Account Activity Monitoring and Analysis Tool
CALDB	Central Activity Log Database
CART	Classification and Regression Trees
CBE	Common Base Event
CCL	Continuous Computation Language
CEP	Complex Event Processing
CITT	Centrum für Informations- und Technologie Transfer
CHAID	Chi Sqare Automatic Interaction Detection
CORBA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
CSV	Character Separated Values
DKG	Dynamic Key Generation
DNS	Domain Name Service
edBPM	event driven Business Process Management
EP	Event Processing
EPA	Event Processing Agent
EPL	Event Processing Language
EPM	Event Processing Model
EPN	Event Processing Network
EPTS	Event Processing Technical Society
ESP	Event Stream Processing
eTAN	elektronische Transaktionsnummer
FinTS	Financial Transaction Services
GK	Group Key
HBCI	Home Banking Computer Interface
HMM	Hidden Markov Model
IBAN	International Bank Account Number
IBPP	Internet Banking Payment Protocol
ID3	Iterative Dichotomiser 3
IDE	Integrated Development Environment
IM	Instant Messaging
IODEF	Incident Object Description Exchange Format
IP	Internet Protocol
iTAN	indizierte Transaktionsnummer
jar	Java Archive
JDL	Joint Directors of Laboratories
JMS	Java Message Service
LR	Logistic Regression
MQ	Message Queueing
mTAN	mobile Transaktionsnummer
NNet	Neural Networks
PIN	Persönliche Identifikationsnummer
RF	Random Forests
RFID	Radio Frequency Identification
SARESA	Sense and Response Service Architecture

SARI	Sense and Respond Infrastructure
SEPA	Single Euro Payments Area
SMS	Short Message Service
SNNS	Stuttgart Neural Network Simulator
SOA	Service Oriented Architecture
SQL	Structured Query Language
SSL	Secure Sockets Layer
SVM	Support Vector Machine
TAN	Transaktionsnummer
TCP	Transmission Control Protocol
TDIDT	Top Down Induction of Decision Trees
TLS	Transport Layer Security
UML	Unified Modeling Language
URL	Uniform Resource Locator
USB	Universal Serial Bus
VIRT	Valued Information at Right Time
WHO	World Health Organisation
XML	Extensible Markup Language
ZELESSA	Zero Latency Event Sensing and Responding Architecture

1 Einleitung

Betrüger haben das Ziel, andere Personen materiell zu schädigen um sich selbst dabei zu bereichern. Zur Erreichung dieses Ziels werden heute keine „langen Finger“ mehr benötigt. Das Internet bietet viele Möglichkeiten, virtuell auf Raubzug zu gehen. Phishing oder Pharming sind nur zwei von vielen Methoden einer immer stärker werdenden Internetmafia. Dem gegenüber stehen viele Ansätze um Online-Diebstahl zu erkennen und zu verhindern. Diese Ansätze können beispielsweise in regelbasierte und statistische Verfahren unterteilt werden. In diesem Zusammenhang ist vor allem die Zeit, die vom Start bis zur Identifikation einer Betrugstransaktion vergeht, ein entscheidendes Kriterium. *Ex post* (deut.: aus danach) Auswertungen z.B. im Rahmen des *Data Mining* helfen, bekannte Betrugsmuster zu erkennen oder neuartige zu identifizieren, wobei zwischen überwachten und unüberwachten Verfahren unterschieden wird, siehe [Jain00, S. 1; Bolt02, S. 2; Kou04, S. 2]. Betrugserkennung ist mittlerweile die etablierteste *Data Mining*-Anwendung im Regierungs- und Industrieumfeld [Phua05, S. 1]. Allerdings kann die Betrugsaktion selbst durch den nachträglichen Einsatz von *Data Mining*-Techniken nicht mehr verhindert werden. Zur sofortigen Unterbindung eines identifizierten Betrugsversuchs ist die Echtzeitfähigkeit eines Betrugserkennungssystems von entscheidender Bedeutung [Bose06, S. 1]. In dieser Thematik bildet *Complex Event Processing* (CEP) eine neuartige Technologie, welche die auftretenden *events* eines Systems auf Basis von *event patterns* miteinander korreliert und auf diese (neu) entstandenen *complex events* in Echtzeit bzw. *„on-the-fly“* reagieren kann. In dieser Arbeit wird auf Basis einer CEP *engine* ein Verfahren entwickelt, welches einen Entscheidungsbaum, die Diskriminanzanalyse sowie ein neuronales Netzwerk miteinander kombiniert um Betrugsversuche aus dem Bankbereich – im Speziellen Identitätsdiebstahl beim Online-Banking – zu entdecken und in Echtzeit zu verhindern. Dabei dienen Diskriminanzwerte als Inputparameter für das neuronale Netzwerk. Das Ergebnis ist ein fertig konzipiertes und implementiertes Hybrid-Modell, das Überweisungen im Rahmen des browserbasierten Online-Banking auf verdächtige Muster in den Transaktionen untersucht. Ein Hybrid-Modell stellt allgemein eine Kombination aus mehreren analytischen Verfahren zur Lösung eines Problems dar [Rich03, S. 67; Krah98, S. 74] und soll in Form seiner konkreten Implementierung im Rahmen dieser Arbeit verhindern, dass Identitätsbetrüger erfolgreich das Bankkonto ihrer Opfer plündern.

1.1 Zieldefinition dieser Arbeit

Bei der Betrugsbekämpfung wird zwischen Betrugsprävention und Betrugserkennung unterschieden. Betrugsprävention umfasst Verfahren, die verwendet werden um mögliche Betrugsfälle bereits im Ansatz zu verhindern. Dagegen wird Betrugserkennung zu einem Zeitpunkt durchgeführt, nachdem die Betrugspräventionsmaßnahmen nicht funktioniert haben und eine Betrugstransaktion erfolgreich gestartet wurde und diese identifiziert werden soll. [Bign06, S. 2]

Im Umfeld des Centrum für Informations- und Technologie Transfer (CITT) sowie der Universität Regensburg wurde von einer Arbeitsgruppe ein *event driven Business Process Management* (edBPM)-Referenzmodell zur Betrugserkennung mittels CEP-Technologie in der Bankenbranche erstellt. Der Begriff edBPM beschreibt das Szenario des Auslösens von Prozessen bzw. das Verändern des Status eines Prozesses auf Basis des Eintretens von *events* oder Kombinationen von *events*, für weiterführende Literatur zu edBPM siehe [Ammo09a; Ammo09b; Ammo08]. Das Referenzmodell ist in Abbildung 1 darstellt.

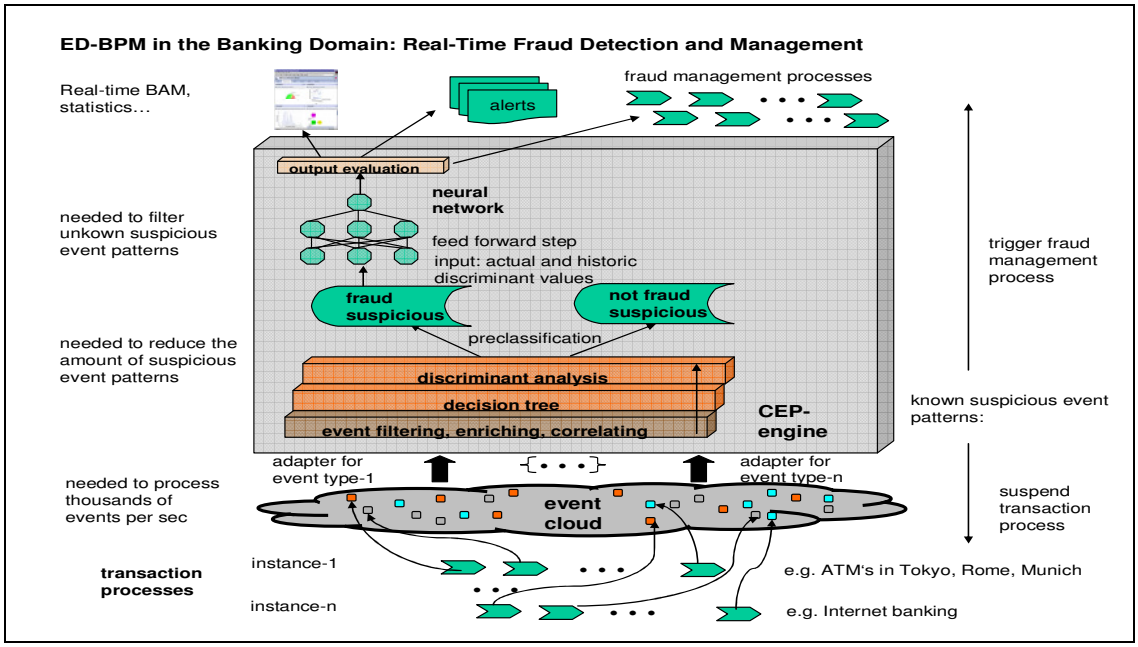

Abbildung 1: edBPM-Referenzmodell für Betrugserkennung im Bankenumfeld aus [Ammo09b, S. 6]

Ein Referenzmodell ist in [Beck03, S. 127] definiert als ein für eine Anwendungsdomäne allgemeingültiges Modell, das Strukturen, Eigenschaften, Verhalten und Beziehungen zwischen Objekten darstellt.

Das edBPM-Referenzmodell verfolgt das Ziel, auf Basis von CEP-Technologie die auftretenden *events* eines Kreditinstituts mittels Einsatz maschineller Lernverfahren in Echtzeit auf ihren Betrugsstatus (Betrugs- oder Nicht-Betrugsfall) zu analysieren. Es stellt dabei

die Verschmelzung von Ereignis- und Prozessverarbeitung dar, da auf Basis von *events* Prozesse und deren Ablauf beeinflusst werden können. Die *events* stammen aus der *Event Cloud* der betreffenden Organisationen und werden mittels A*dapter* auf das benötigte Format der CEP *engine* angepasst (eine Erklärung dieser Begriffe erfolgt in den Abschnitten 4.1, 4.2 und 4.3). Je nach Analyseergebnis werden neue Prozesse gestartet oder laufende Prozesse unterbrochen. Dabei gilt das Referenzmodell in Abbildung 1 allgemein für alle Arten von *events* (z.B. *events* aus Online-Transaktionen, Kontenbewegungen oder Geldautomaten usw.) der verschiedenen Arten von Betrugserkennungsanwendungen und ist durch *Complex Event Processing*-Technologie in der Lage, tausende von *events* pro Sekunde zu verarbeiten.

Im Rahmen dieser Arbeit wird der Mittelteil des Referenzmodells d.h. die Betrugserkennungskomponente in der CEP *engine* speziell für den Anwendungsfall Erkennung von Identitätsdiebstahl beim Online-Banking entwickelt sowie die Versorgung mit den notwendigen Daten bzw. *events* diskutiert.

Die Notwendigkeit für die Erstellung dieser Arbeit ergibt sich aufgrund einer immer stärker ansteigenden Zahl von Identitätsdiebstählen im Online-Banking, wie Statistiken im Abschnitt 2.3 belegen.

In diesem Zusammenhang soll die Frage beantwortet werden, wie effektiv der Einsatz von *Complex Event Processing*-Technologie in Kombination mit maschinellen Lernverfahren ist, um Betrugstransaktionen im Online-Banking aus der Gesamtmenge der Transaktionen zu identifizieren. Die korrekte Identifikation sowohl von Betrugs- als auch von Nicht-Betrugsfällen muss dabei so exakt und so schnell wie möglich erfolgen. Der Beitrag dieser Arbeit zur Beantwortung dieser Fragestellung ist, das zu Beginn dieses Kapitels erwähnte Hybrid-Modell aus einer Kombination aus Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk auf Basis von CEP-Technologie umzusetzen und die Resultate zu diskutieren. Zu diesem Ziel wird das verwendete Modell beschrieben, die Auswahl der eingesetzten Verfahren begründet und mittels Experimenten die Qualität des entstandenen Analysemodells für den konkreten Anwendungsfall der Identifikation von Identitätsdiebstahl beim Online-Banking auf Basis von *Complex Event Processing* aufgezeigt. Folglich ist diese Arbeit in die Kategorie der Betrugserkennung einzuordnen.

In diesem Zusammenhang soll im Rahmen dieser Arbeit primär erreicht werden, ein Problem der bankfachlichen Praxis zu lösen, Betrugsszenarien des immer stärker ansteigenden Identitätsdiebstahls im Online-Banking in Echtzeit zu erkennen und somit unmittelbar handeln zu können. Ein theoretischer Beitrag zum Forschungsgebiet der *Complex Event Processing*-Technologie oder der maschinellen Lernverfahren soll daher erst sekundär erstellt werden.

Darüber hinaus bezieht sich die hier entwickelte Lösung rein auf Identitätsdiebstahl beim Online-Banking mittels eines Internetbrowsers. Die Thematiken des Identitätsdiebstahls z.B. beim Durchführen von Zahlungen mittels Kreditkarten, bei Geldautomatenabhebungen oder bei der Teilnahme an Online-Auktionen werden im Rahmen dieser Arbeit nicht behandelt. Der Grund dafür ist, dass diese Betrugsarten eigenständige Themengebiete mit anderen Rahmenbedingungen (z.B. andere Betrugsmuster, andere Basisdaten oder andere Prozessabwicklung) darstellen und somit in gesonderten Arbeiten diskutiert und bearbeitet werden können bzw. müssen.
Der geografische Fokus dieser Arbeit in Bezug auf gesetzliche Vorschriften, Angaben zu Betrugsmustern, Statistiken usw. liegt auf Deutschland, da die an der Entwicklung des Hybrid-Modells beteiligten Personen und Institute in Deutschland angesiedelt und tätig sind.

1.2 Vorgehensweise bei der Erstellung dieser Arbeit

Diese Arbeit ist inhaltlich in zwei Hauptteile untergliedert. Die ersten fünf Kapitel gehen allgemein auf bestehende relevante Literatur ein und geben Erläuterungen im Umfeld dieser Arbeit, dagegen diskutieren die letzten fünf Kapitel den hier entstandenen neuen Ansatz zur Betrugserkennung im Bereich Online-Banking und dessen praktische Ergebnisse.
Zu Beginn wird zunächst auf die Formen bzw. Quellen des Betrugs im Bankenumfeld und die gesetzliche Lage eingegangen, wobei hier der Fokus auf Identitätsdiebstahl liegt. Statistisch nachgewiesene Entwicklungen in diesem Bereich und aktuelle Gegenmaßnahmen bei Kreditinstituten runden das Kapitel ab.
Da die Algorithmen im Mittelteil des edBPM-Referenzmodells Methoden der Wissensrepräsentation bzw. der maschinellen Lernverfahren darstellen, werden im Kapitel 3 die wichtigsten Verfahren in diesem Bereich genannt und in einer wertungsfreien Form beschrieben.
Kapitel 4 geht anschließend auf die in dieser Arbeit verwendete Basistechnologie *Complex Event Processing* ein und gibt einen Überblick über die Entwicklungen in dieser noch relativ jungen Technologie. Hierbei werden unter anderem die im Vorgängerabschnitt genannten Begriffe wie *event, Event Cloud* oder CEP *engine* ebenfalls wertungsfrei beschrieben.
Im nächsten Kapitel wird der *State-of-the-Art* diskutiert. Im Rahmen dieser Arbeit handelt es sich beim relevanten Stand der Forschung konkret um CEP-Implementierungen im Betrugserkennungsumfeld, die bereits heute aktiv verwendet werden sowie um Publikationen, die den Einsatz von maschinellen Lernverfahren im Bereich des Identitätsdiebstahls

beim Online-Banking betreffen. Mit diesem Kapitel ist der allgemeine Teil abgeschlossen. Die nachfolgenden Kapitel beschäftigen sich mit der Entwicklung des Ansatzes dieser Arbeit.

Im Kapitel 6 werden die Problemstellung und die Zielvorgaben dieser Arbeit genau beschrieben und die Auswahl der verwendeten Methoden des maschinellen Lernens begründet.

Das siebte Kapitel beschreibt das im Rahmen dieser Arbeit entstandene Hybrid-Modell als Kombination von Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk in Verbindung mit *Complex Event Processing*-Technologie als Ansatz zur Betrugserkennung von Identitätsdiebstahl beim Online-Banking.

Kapitel 8 erläutert die Umgebung zur Durchführung empirischer Experimente zur praktischen Validierung des entwickelten Hybrid-Modells. Hierbei wird die CEP *engine* des Herstellers `StreamBase Inc.` beschrieben, die für die Implementierung der Anwendung und für die Durchführung von Experimenten verwendet wird. In diesem Zusammenhang wird ebenfalls auf die Umsetzung der Betrugserkennungskomponente mit Hilfe der Programmiersprache Java eingegangen.

Das nachfolgende neunte Kapitel stellt die experimentellen Ergebnisse des entwickelten Modells auf Basis von simulierten Online-Überweisungen dar. Dabei werden simulierte Transaktionen bzw. *events* verwendet, da keine realen *events* zur Durchführung der erwähnten Experimente zur Verfügung stehen. Die Simulation wurde dabei auf Basis von Angaben aus durchgeführten Interviews mit Experten aus den Bereichen *Compliance* (deut.: Einhaltung von Vorschriften) und Betrugsmanagement mehrerer Kreditinstitute sowie Bankverbänden durchgeführt. Aus Gründen der Vertraulichkeit und des Datenschutzes dürfen hierbei die Namen der beteiligten Personen und Organisationen nicht genannt werden (die kompletten Interviews sind im Anhang 1 einzusehen). Zum Abschluss werden im letzten Kapitel Schlussfolgerungen aus dem Ansatz gezogen und ein Ausblick auf die Zukunft und Weiterentwicklungsmöglichkeiten dieser Arbeit gegeben.

Zu Beginn wird im nachfolgenden Kapitel erläutert, mit welchen Betrugsproblemen die Bankenbranche im Bereich Online-Banking konfrontiert wird und wie sie heute bereits damit umgeht.

2 Identitätsdiebstahl beim Online-Banking

In diesem Kapitel werden die bekannten Arten von Betrugstypen bzw. Identitätsdiebstahl bei Banken beschrieben, die mit dem in dieser Arbeit beschriebenen Ansatz aufgedeckt werden sollen. Des Weiteren werden Statistiken zu dem Problem vorgestellt, die typi-

schen Betrugsmuster mit Beispielfällen aufgezeigt sowie die gängigen Sicherheitsmechanismen der Kreditinstitute erläutert und die aktuelle Rechtslage diskutiert.
Identitätsdiebstahl ist allgemein definiert als unberechtigtes Annehmen einer fremden Identität zum Zweck des Durchführens betrügerischer Handlungen. Diese Handlungen beziehen sich größtenteils auf die Bereiche Kreditkartenzahlung, Mobiltelefonie, Online-Auktionen sowie Online-Banking. [Wagn07, S. 1; Bose06, S. 1; Lync05, S. 1]
Der Fokus dieser Arbeit liegt dabei – wie im Abschnitt 1.1 erwähnt – beim Online-Banking und somit im Bankenumfeld. Betrug tritt innerhalb von Kreditinstituten in verschiedenen Formen, aber oft in typischen Mustern auf, siehe dazu [Bign06, S. 4 - 6; Vikr04, S. 2 - 4]. Alarmierend sind hierbei die jüngsten Entwicklungen, die im letzten Abschnitt dieses Kapitels vorgestellt werden und die u.a. auch als Anlass genommen wurden, diese Arbeit anzufertigen und in diesem Rahmen eine Methodik zu entwickeln, Betrugstransaktionen beim Online-Banking zu identifizieren.

2.1 Formen des Identitätsdiebstahls beim Online-Banking

Im Bankenumfeld treten Betrugsfälle sowohl von innerhalb als auch von außerhalb eines Kreditinstituts auf. Die Quellen des Betrugs von innerhalb sind die eigenen Mitarbeiter, wie beispielsweise in einem Fall der Sparkasse Lüneburg aus dem Jahr 2006. Dort bereicherte sich ein Bankangestellter mittels fingierter Auszahlungen an den Konten der Sparkassenkunden um insgesamt 200.000 Euro, siehe dazu [Schu06]. Der Bereich interner Betrug durch Bankmitarbeiter und dessen typische Muster wie z.B. Schattenkonti ist in [Reol07, S. 7 - 13] diskutiert.
Dagegen basiert der Betrug von außerhalb auf vielfältigen Varianten. Eine externe Betrugsart ist beispielsweise die Geldwäsche. Bei der Geldwäsche handelt es sich um den Prozess, erworbene Gelder aus illegalen bzw. kriminellen Quellen z.B. aus dem Drogenhandel in „saubere“ Finanzsysteme zu transferieren, da in Zeiten des überwiegend bargeldlosen Zahlungsverkehrs große Bargeldmengen sehr stark auffallen. Dabei werden die zu „waschenden“ Geldbeträge sehr oft von Konto zu Konto transferiert. Das Ziel dabei ist, diese Gelder danach im legalen Finanzsystem wieder gewinnbringend investieren zu können bzw. weitere kriminelle Delikte zu finanzieren und in diesem Zusammenhang die wahre Herkunft der Finanzmittel zu verschleiern. [Schn06, S. 16 - 17; Assi02, S. 134]
Für weiterführende Literatur zu statistischen Methoden zur Bekämpfung der Geldwäsche siehe [Bolt02].
Der Fokus dieser Arbeit liegt – wie im Abschnitt 1.1 erwähnt – aufgrund aktueller Entwicklungen, die im Abschnitt 2.3 genauer erläutert werden, beim Identitätsdiebstahl im Rahmen von Online-Überweisungen. Im Rahmen dieser Arbeit werden alle Überweisungsar-

ten beim Online-Banking als Transaktion bezeichnet (für die einzelnen Überweisungsarten wie z.B. *Single Europe Payment Area*-Überweisung (SEPA), Inlands- oder Auslandsüberweisung siehe [Grun08, S. 85 - 141]). Beim Online-Banking kann sich der Bankkunde von zuhause aus unter Verwendung eines Internetbrowsers auf der Homepage seiner Bank mit seiner Kontonummer und seiner Persönlichen Identifikationsnummer (PIN) einloggen und von dort aus Überweisungen durchführen. Zu diesem Zweck werden auf der Internetseite der Bank die Überweisungsparameter wie Empfänger, Zielkontonummer, Zielbank, Überweisungsbetrag, Verwendungszweck usw. sowie eine Transaktionsnummer (TAN) eingetragen und anschließend die Überweisung gestartet. Die Kommunikation findet dabei über das Internet bzw. über das Netz des Internet-Providers statt. [Jano06, S. 162]

Zur Standardisierung des internationalen Zahlungsverkehrs wird für Auslandsüberweisungen die *International Bank Account Number* (IBAN) für die Zielkontonummer verwendet, die in Blöcken von vier Zeichen geschrieben wird. Der *Bank Identifier Code* (BIC) bildet das internationale Format für die Bankleitzahlen, der aus acht oder elf Zeichen besteht. [Maur08, S. 524; Grun08, S. 132]

Identitätsdiebstahl ist speziell im Bankenumfeld definiert als das Stehlen bzw. das unerlaubte Verwenden der persönlichen Daten und Passwörtern einer anderen Person um unter deren Namen für den Zweck der eigenen Bereicherung Waren einzukaufen, neue Konten zu eröffnen oder Überweisungen durchzuführen [Lobe04, S. 5 - 6; Vacc03, S. 4 - 7; May04, S. 2 - 3].

Zur Erlangung dieser Daten für das Online-Banking entwickelten die Identitätsbetrüger, die in vielen Fällen das Profil eines finanziell motivierten Computerspezialisten aufweisen [Wagn07, S. 7], verschiedene Möglichkeiten, die in den nachfolgenden Abschnitten erläutert werden.

2.1.1 Phishing

Phishing ist eine weitverbreitete Methode des Identitätsdiebstahls. Der Phisher sendet gefälschte Emails an willkürliche oder namentlich bekannte Empfänger, deren Identitäten die Betrüger aus dem Internet, Journalen oder Zeitungen beziehen [Ohay06, S. 1]. Diese Email täuscht beispielsweise eine Bank oder eine Kreditkartenfirma als Absender vor und enthält die Aufforderung, die persönlichen Daten auf einer entsprechend gefälschten Website zu aktualisieren [Kraf07, S. 309; Pate07, S. 1; Yu08, S. 1]. Abbildung 2 enthält eine solche Phishing-Email, die als Absender das Kreditinstitut `Wells Fargo` vorgibt.

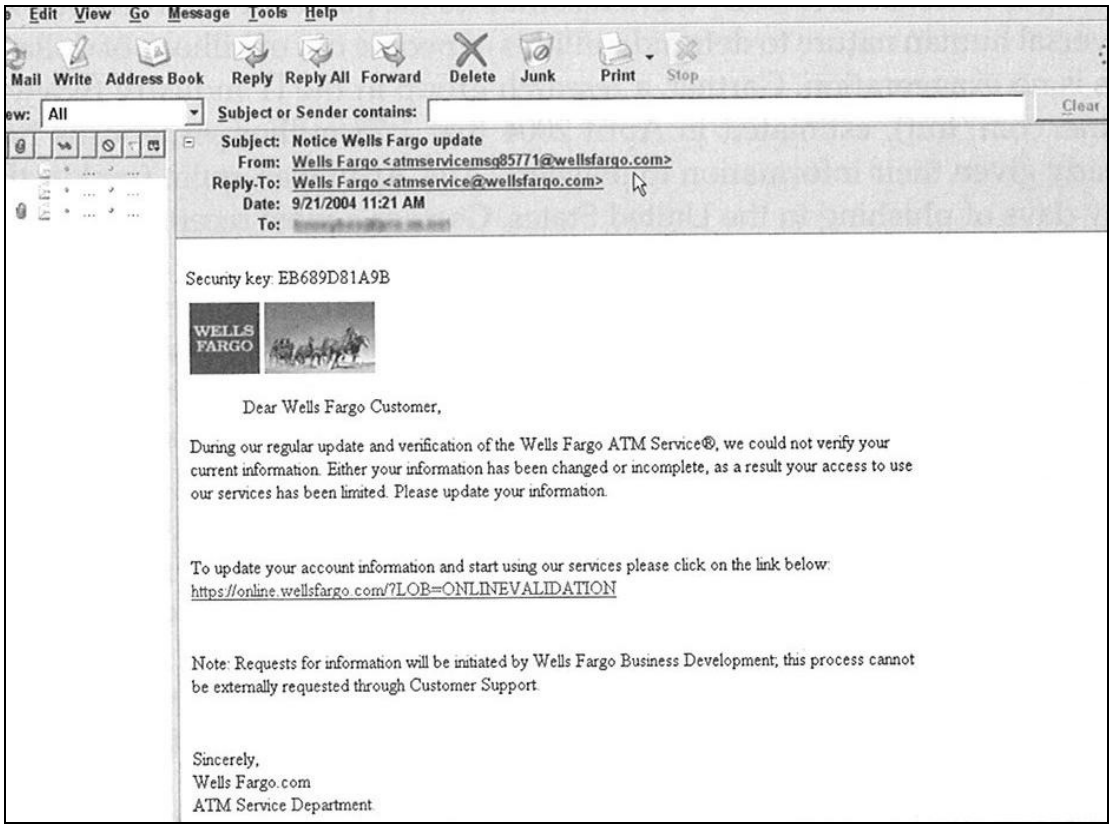

Abbildung 2: Beispiel für eine Phishing-Email aus [Lini05, S. 56]

Eine Phishing-Email enthält einen Link (in Abbildung 2 beispielsweise lautet dieser folgendermaßen: https://online.wellsfargo.com/?LOB=ONLINEVALIDATION) zu einer Phishingseite, die vom Layout her der Online-Banking-Seite eines Kreditinstituts sehr ähnlich sieht [Kraf07, S. 309; Pate07, S. 1; Yu08, S. 1].

Eine gut strukturierte Phishingseite kann lt. einer Studie von [Dham06, S. 1] bis zu 90% aller Seitenbesucher täuschen. In Abbildung 3 ist eine gefälschte Internetseite der `Bank of the West` dargestellt. Die Betrüger hosten in diesem Beispiel die nachgemachte Seite indem sie in der URL statt dem „w“ von bankofthe**w**est zwei „v“ verwenden. Dadurch werden die Opfer getäuscht, denn die Phishingseite entspricht ansonsten in allen anderen Details der Originalseite. [Dham06, S. 7]

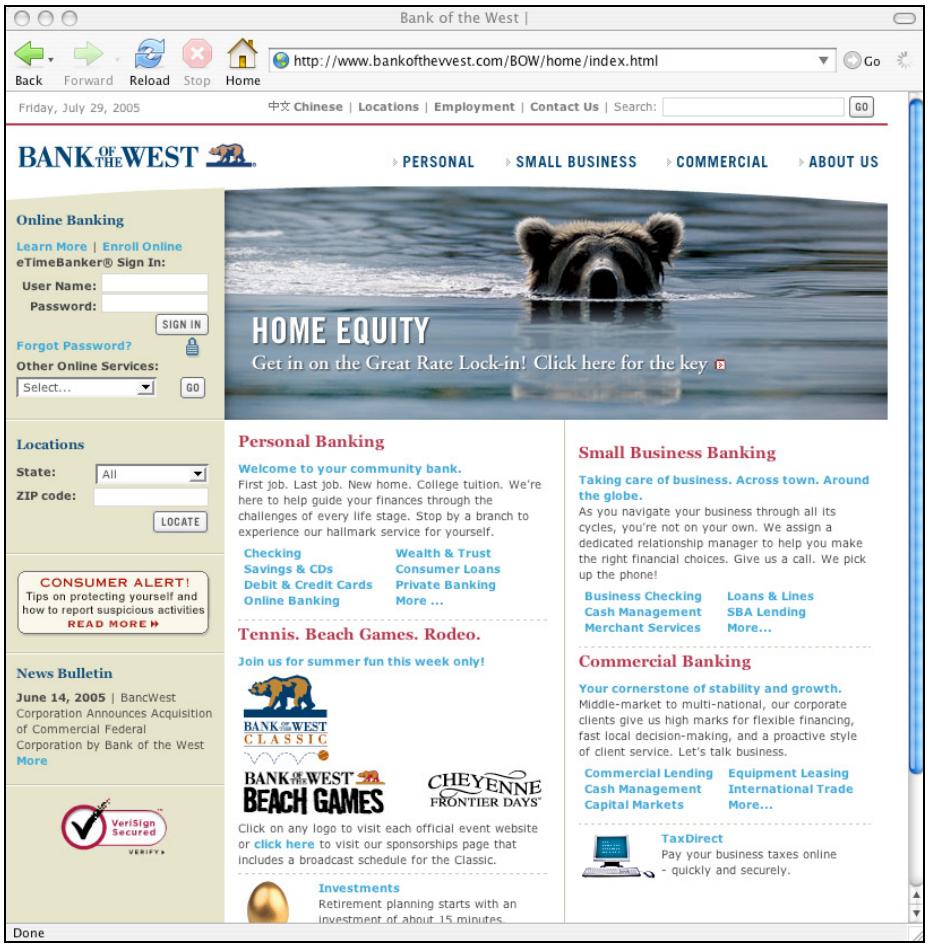

Abbildung 3: Beispiel einer original Phishingseite der Bank of the West aus [Dham06, S. 7]

Phishingseiten können von verschiedenen Systemen verwaltet werden, z.B. bei *Web Hosting*-Firmen oder auch auf privaten Webservern. Die meisten Phishingseiten existieren in den USA, China und Süd Korea [Lini05, S. 19], wobei Ende 2008 China die USA von der „Weltspitze“ der am meisten gehosteten Phishing-Internetseiten verdrängt hat [Müll09].

Der ahnungslose Kunde gibt – wie oben erwähnt – auf einer solchen gefälschten Website seine persönlichen Daten, wie z.B. Persönliche Identifikationsnummer (PIN), Transaktionsnummern (TAN) und Passwörter ein. Diese persönlichen Informationen benutzt der Betrüger im Anschluss um Überweisungen im Namen des Phishingopfers auf ein vom Betrüger kontrolliertes Konto bzw. auf das Konto eines Mittelsmanns durchzuführen [Lini05, S. 10; Kraf07, S. 309]. Ca. 90% aller Phishingattacken richten sich dabei gegen Kunden von Finanzinstituten [Müll09].

Das Ziel der Transaktionen bilden oftmals Mittelsmänner (in [Jano06, S. 250] auch als Finanzkuriere bezeichnet) im Inland oder Konten im Ausland. Erbeutete Identitätsdaten werden oftmals auch auf geheimen Online-Börsen gehandelt, die für die Kriminalbeamten schwer zu finden sind [Tecc06]. Die verschiedenen Phishingvarianten sowie die diesbe-

züglichen Schwächen der heutigen Internetbrowser sind in [Yu08], [Ohay06], [Herz08] und [Badr07] diskutiert. Die Methoden, mit denen die gefälschten Webseiten die Opfer täuschen, sind in [Dham06, S. 3 - 4] erläutert. Der Hauptgrund für den Erfolg von Phishing ist aber lt. [Down06, S. 1], dass die Opfer zu wenig über die Möglichkeiten aufgeklärt sind, diese gefälschten Emails zu identifizieren.
Was die Bankenbranche bereits heute gegen Phishing und andere Varianten des Identitätsbetrugs beim Online-Banking unternimmt um Ihre Kunden zu schützen, ist im Unterabschnitt 2.4.1 erläutert.

2.1.2 Trojanisches Pferd

Ein Trojanisches Pferd ist ein Programm, das sich z.B. beim Besuch einer Internetseite oder durch das Öffnen einer *Spam*-Email unbemerkt auf dem Rechner des Opfers installiert. Wenn sich der Bankkunde auf seiner Online-Banking-Seite einloggt, fängt der Trojaner die Eingaben ab und sendet diese an den Betrüger und täuscht oftmals das Opfer durch die Meldung (teilweise sogar in der original Online-Banking-Seite des Kreditinstituts), dass die Anmeldeinformationen falsch sind. Diese Form der Trojanischen Pferde registriert die Tastatureingaben und wird auch als *Keylogger* bezeichnet. *Keylogger* können sowohl eine Softwarekomponente darstellen, als auch in Form eines Chips in die Tastatur des Rechners des Opfers eingebaut sein [Kraf07, S. 307]. Bekannte *Keylogger* sind z.B. *ZeuS* oder *Nethell*, die von den Betrügern individuell konfiguriert werden können und auf dem Schwarzmarkt für 2.000 bis 3.000 Dollar erhältlich sind [Schü08]. In Abbildung 4 ist ein PS2 *Keylogger* am Tastaturanschluss eines Rechners dargestellt.

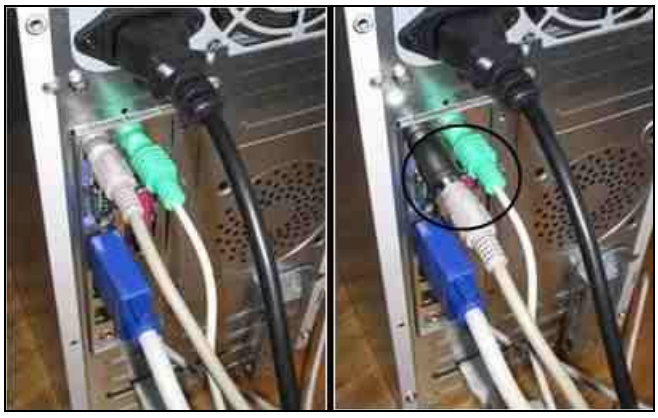

Abbildung 4: Keylogger an der Tastatur eines Rechners aus [Cybe08]

In der rechten Bildhälfte von Abbildung 4 befindet sich ein *Keylogger* innerhalb des schwarzen Kreises, die linke Seite enthält zum Vergleich das gleiche Bild ohne *Keylogger*.

Andere Formen der Trojanischen Pferde öffnen dem Betrüger direkt den Zugang zum Rechner seines Opfers, d.h. einen bestimmten Port, ähnlich wie das Trojanische Pferd bei der Eroberung von Troja durch die Griechen [Trae02, S. 404 - 405; Lini05, S. 11].
Ein Trick dabei ist, dass der Trojaner oftmals den Rechner des Opfers zum Absturz bringt. So hat der Betrüger Zeit, mit Hilfe der erbeuteten Daten eine Transaktion durchführen zu können oder der Trojaner selbst führt die Transaktion in Sekundenschnelle aus. [Jano06, S. 163]

2.1.3 Pharming

Pharming ist eine Weiterentwicklung des klassischen Phishings. Beim Pharming wird das Opfer ebenfalls auf eine gefälschte Internetseite gelockt. Allerdings nicht mittels einer Email sondern durch Manipulation der *Domain Name Service* (DNS) Information der Host-Datei auf dem Rechner des Opfers unter Zuhilfenahme eines Trojanischen Pferdes oder eines Computervirus (mit DNS ist der Dienst im Internet bezeichnet, der das Umwandeln (engl.: *Mapping*) einer Internet-Adresse (URL) von Klartext in die passende IP-Adresse des Zielhost übernimmt, mit der im Anschluss die Weiterleitung (engl.: *Routing*) zum gewünschten Ziel erfolgen kann, siehe dazu [Kuro02, S. 137 - 148]). Aufgrund dieser Manipulation erfolgt beim nächsten Aufruf der Internetseite eine automatische Weiterleitung zur gefälschten Seite, auf der sich der Bankkunde anschließend mit seiner PIN und TAN anmeldet. Dadurch gelangen die Betrüger an die gewünschten Informationen. Der Begriff Pharming rührt daher, dass die Betrüger oft umfangreiche Serverfarmen benötigen, um die vielen gefälschten Internetseiten verschiedener Banken bereitzustellen. [Tyna05, S. 51; Frau06]
Pharming und Phishing werden in der Literatur oftmals auch als *Man in the Middle*-Attacke bezeichnet, da sich bei diesen Verfahren der Betrüger in der Mitte des Kommunikationsweges zwischen dem Opfer und dem Kreditinstitut befindet und dort die relevanten Daten abfängt [Jano06, S. 252; Lini05, S. 148 - 149; Kraf07, S. 307].

2.1.4 Pretexting

Beim Pretexting verwenden die Betrüger das Telefon als Medium um an die Kontodaten ihrer Opfer zu gelangen. Dabei geben sich die Betrüger am Telefon gegenüber dem Bankkunden als Bankmitarbeiter aus und befragen ihn nach seinen Zugangsdaten. Gibt der Betroffene diese preis, wird im Anschluss daran der Kontenraub durchgeführt. [Sull04, S. 175 - 176; May04, S. 24]

Eine Variante des Pretexting ist das *Voice over IP Phishing* (auch Vishing genannt). Hierbei nutzen die Betrüger die günstigen Tarife sowie die räumlich uneingeschränkte Verfügbarkeit der Internettelefonie. Ein *Dialer* ruft dabei automatisch die Opfer an, wobei sich eine Stimme vom Band als ein Mitarbeiter bzw. eine Mitarbeiterin eines Kreditinstituts ausgibt und versucht, vertrauliche Daten wie PIN, TAN usw. vom möglichen späteren Opfer zu erlangen. [Müll08, S. 471; Grif08, S. 2]
Eine weitere Form des Identitätsdiebstahls im Bankenumfeld ist das Skimming. Betrüger installieren dabei heimliche Lesegeräte auf der Oberfläche von Geldautomaten in Einkaufszentren oder Bankfilialen. Diese Geräte lesen die Karteninformationen aus den Magnetstreifen der Girokarten der Automatenbediener aus. Anschließend werden von den Betrügern identische Karten angefertigt, mit denen die Konten der Opfer geplündert werden. [Alba07, S. 6; Bidg04, S. 332; May04, S. 27 - 28]
Allerdings erfolgt beim Skimming der Diebstahl direkt durch eine Bargeldabhebung am Geldautomaten und nicht durch eine Online-Überweisung und fällt somit nicht in den Fokus dieser Arbeit.

2.2 Typische Betrugsmuster von Identitätsdiebstahl im Bankenumfeld

Die im Abschnitt 2.1 genannten Formen des Identitätsbetrugs bei Überweisungen hinterlassen bei Banken Spuren in Form von wiederkehrenden bekannten Mustern. In diesem Fall liegt das Hauptaugenmerk auf Geschäftsbanken wie Volks- und Raiffeisenbanken oder Sparkassen, weil diese z.B. nach [Stat08] in der Regel die Anbieter für Girokonten darstellen und Online-Banking ermöglichen.
Die nachfolgenden Betrugsmuster wurden im Rahmen der zu Beginn genannten Interviews mit den Betrugsexperten aus der Bankenbranche genannt (siehe Anhang 1):

a) Verdächtig sind Transaktionsbeträge nahe an der maximalen Verfügbarkeitsgrenze des Girokontos (Kontostand + Dispolimit). Allerdings kann in seltenen Fällen auch die Situation eintreten, dass die Betrüger absichtlich keinen so hohen Betrag transferieren um nicht aufzufallen.

b) Verdächtig sind Transaktionsbeträge mit einem vielfachen Verhältnis zum durchschnittlichen Transaktionsbetrag des Kunden.

c) Verdächtig sind Überweisungen ins Ausland, wobei aber mittlerweile ca. 80% der Betrugstransaktionen zu Mittelsmännern im Inland gehen, welche die gestohlenen Beträge anschließend ins Ausland transferieren oder abheben und in Bar weiterleiten. Diese Mit-

telsmänner werden oft – besonders in wirtschaftlich schwierigen Zeiten – mit der Aussicht auf einen einfachen Hinzuverdienst angeworben, meistens ohne das Wissen dieser Personen nach den Quellen des transferierten Geldes. Die Zielländer der Betrugstransaktionen stellen oftmals osteuropäische, afrikanische, asiatische oder südamerikanische Staaten, aber auch England dar. Auslandsüberweisungen weisen im Privatkundenbereich allgemein einen sehr geringen Anteil auf, weshalb diese verdächtiger sind.

d) Verdächtig sind ausländische IP-Adressbereiche als Quelle der Online-Überweisung, wobei viele Kreditinstitute die IP-Adresse des Quellrechners nicht auswerten bzw. nicht wissen (die IP-Adresse identifiziert einen Rechner eindeutig im Internet, für weiterführende Literatur hierzu siehe [Kuro02, S. 301 - 319]). Der Grund dafür ist, dass die Betrüger oftmals im Ausland angesiedelt sind und mit den erbeuteten Daten von dort aus ihre Betrugstransaktion starten. Auch führen Inlandskunden in der Regel keine Überweisungen aus dem Ausland durch und benutzen somit immer den gleichen inländischen IP-Adressraum.

e) Verdächtig ist, wenn die Zeit, die für die Durchführung einer Online-Überweisung benötigt wird so kurz ist, dass diese von einem Menschen (im Normalfall) in dieser Geschwindigkeit nicht durchgeführt werden kann. Das deutet darauf hin, dass die Überweisung evtl. von einem Trojanischen Pferd durchgeführt wurde, wobei viele Kreditinstitute diese Zeitmessungen nicht durchführen.

f) Verdächtig sind Überweisungen zu einem unbekannten Empfänger d.h. Überweisungen auf ein Konto, auf welches vorher von diesem Kunden noch nie überwiesen wurde und es sich bei dem Empfänger um keine bekannte Organisation oder Behörde handelt. Allerdings kann dieser Fall häufiger, z.B. bei Neukunden, auftreten.

g) Gefährdet sind Kunden mit durchschnittlich wenigen Online-Transaktionen bzw. insgesamt sehr wenigen Aktivitäten im Bereich Online-Banking, da bei dieser Personengruppe mit einer höheren Wahrscheinlichkeit davon ausgegangen wird, dass sie weniger mit den Gefahren des Identitätsdiebstahls vertraut sind.

h) Verdächtig sind Transaktionsbeträge, die größer als 1.500 Euro sind. Konten, die einen geringeren maximal verfügbaren Transaktionsbetrag aufweisen, werden von den Betrügern in der Regel in Ruhe gelassen, weil es sich nicht lohnt, für kleinere Beträge einen Betrugsfall und folglich eine mögliche Aufmerksamkeit zu riskieren. Betrugstransaktionen mit Beträgen unter 1.500 Euro sind tendenziell selten.

i) Unverdächtig dagegen sind regelmäßige Transaktionen auf das gleiche Zielkonto bzw. bekannte, vertrauenswürdige Empfänger wie bekannte Organisationen oder Behörden.
Diese Muster gelten ausschließlich für Online-Transaktionen. Überweisungen, die persönlich am Schalter aufgegeben wurden oder Kredit- bzw. Debitkartenzahlungen werden hierbei nicht mit einbezogen. Für die Betrugsanalyse ist zu berücksichtigen, dass bei unterschiedlichen Kunden eine bestimmte Transaktion bei einem Kunden verdächtiger ist als bei einem anderen Kunden. Daher ist für die Durchführung der Transaktionsanalyse auch die Transaktionshistorie eines Kunden von Bedeutung, was im Abschnitt 6.1 diskutiert wird.
Die entwickelte Lösung dieser Arbeit setzt bei der Erkennung derartiger Transaktionsmuster an. Da die Banken oder Rechenzentrumsdienstleister aus Sicherheitsgründen keine Originaldaten von Betrugs- und Nicht-Betrugsüberweisungen preisgeben, wurden im Rahmen des experimentellen Teils dieser Arbeit die Transaktionen nach den oben angegebenen Mustern simuliert. Der folgende Beispieldatensatz bzw. Beispielevent aus der Simulation zeigt einen typischen Betrugsfall:

Attribut	Wert
KundenID	11258
TransaktionsID	12491
Transaktionsbetrag	-12.771,76 €
Transaktionsdatum	25.02.2009
Transaktionsuhrzeit	18:36:12
Quellkontonummer	1619134
Quellrechner IP-Adresse	87.115.18.115
Empfängerkontonummer	3001040
Empfängerbankleitzahl	165050 (= Bank of America, London)
Kontostand	+5.488,76 €
Dispolimit	7.300,00 €
Durchschnittlicher Transaktionsbetrag	-241,87 €
Durchschnittliche Transaktionszahl (Monat)	3,0
Auslandstransaktion	ja
Bekannter Empfänger	nein
Inländische IP-Adresse	nein
Zeit für die Online-Überweisung	56 sek.

Tabelle 1: Beispiel für einen verdächtigen Betrugsfall

Bei diesem Betrugsfall in Tabelle 1 sind von den oben genannten Charakteristika eines Betrugsfalls viele erfüllt, z.B. ein hohes Verhältnis von Transaktionsbetrag zum maximal verfügbaren Betrag (99,09% = 12.771,76 * 100 / (5.488,76 + 7.300,00)) sowie ein überdimensional hohes Verhältnis von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag (5.280,42% = 12.771,76 * 100 / 241,87). Darüber hinaus handelt es sich um eine Auslandstransaktion von einem Quellrechner mit ausländischer IP-Adresse zu einem zuvor nicht bekannten Empfänger. Dies zeigt einen offensichtlichen Betrugsfall in dem der Phisher über eine gefälschte Internetseite an die Kontodaten seines Opfers gelangt ist. Der Betrüger nutzt die temporäre Sicherheitslücke sofort um fast den maximalen Betrag zu stehlen. Im Rahmen der Simulation wurden auch Betrugsfälle generiert, bei denen die Werte nicht so offensichtlich auf einen Betrugsfall hindeuten. Beispiele dafür sind Betrugstransaktionen mit geringeren Verhältnissen von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag bzw. zum maximal verfügbaren Betrag oder auch Betrugstransaktionen, die ein Konto im Inland zum Ziel haben. Ein solcher Betrugsfall ist in Tabelle 2 dargestellt:

Attribut	Wert
KundenID	8916
TransaktionsID	11239
Transaktionsbetrag	-7.900,26 €
Transaktionsdatum	27.02.2009
Transaktionsuhrzeit	00:56:34
Quellkontonummer	1394561
Quellrechner IP-Adresse	88.101.58.15
Empfängerkontonummer	1587335
Empfängerbankleitzahl	13050000 (= Sparkasse Rostock)
Kontostand	+13.844,59 €
Dispolimit	9.000,00 €
Durchschnittlicher Transaktionsbetrag	-1.029,78 €
Durchschnittliche Transaktionszahl (Monat)	8,0
Auslandstransaktion	nein
Bekannter Empfänger	nein
Inländische IP-Adresse	ja
Zeit für die Online-Überweisung	8 sek.

Tabelle 2: Beispiel für einen unverdächtigen Betrugsfall

Bei diesem Betrugsfall ist das Verhältnis von Transaktionsbetrag zu maximal verfügbaren Betrag nicht so hoch (34,58% = 7.900,26 * 100 / (13.844,59 + 9.000,00)), gleiches gilt für das Verhältnis von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag (767,18% = 7.900,26 * 100 / 1.029,78). Des Weiteren handelt es sich hierbei nicht um eine Auslandsüberweisung und auch der Quellrechner befindet sich im Inland. Ebenso weist dieser Nutzer bei durchschnittlich acht Online-Transaktionen im Monat eine gewisse Erfahrung auf, dennoch liegt hier ein Betrugsfall vor. Anhand der Zeit für die Online-Überweisung kann davon ausgegangen werden, dass hier ein Trojanisches Pferd aktiv war. Der Virus ist in diesem Fall so programmiert, nicht die maximal mögliche Summe zu transferieren um (möglicherweise) länger unentdeckt zu bleiben.
Im Vergleich dazu sind in Tabelle 3 die jeweiligen Ausprägungen eines typischen Nicht-Betrugsfalles aufgelistet.

Attribut	Wert
KundenID	9031
TransaktionsID	14008
Transaktionsbetrag	-34,01 €
Transaktionsdatum	26.02.2009
Transaktionsuhrzeit	10:06:28
Quellkontonummer	1900912
Quellrechner IP-Adresse	77.25.68.25
Empfängerkontonummer	1201040
Empfängerbankleitzahl	70070010 (= Deutsche Bank, München)
Kontostand	+5.759,01 €
Dispolimit	4.300,00 €
Durchschnittlicher Transaktionsbetrag	-173,66 €
Durchschnittliche Transaktionszahl (Monat)	10,0
Auslandstransaktion	nein
Bekannter Empfänger	nein
Inländische IP-Adresse	ja
Zeit für die Online-Überweisung	117 sek.

Tabelle 3: Beispiel für einen unverdächtigen Nicht-Betrugsfall

Das Verhältnis von Transaktionsbetrag zum maximal verfügbaren Betrag ist bei diesem Nicht-Betrugsfall nicht sehr hoch (0,34% = 34,01 * 100 / (5.759,01 + 4.300,00)), ebenso das Verhältnis von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag

(19,58% = 34,01 * 100 / 173,66). Quelle und Ziel der Überweisung befinden sich im Inland. Da der Empfänger nicht bekannt ist, könnte es sich hierbei z.B. um die Bezahlung einer Buchsendung handeln, wobei der Kunde bei diesem Verkäufer zuvor noch nie etwas bestellt hat. Auch bei den Nicht-Betrugsfällen ist der Betrugsstatus nicht immer so offensichtlich. Ein verdächtigerer Nicht-Betrugsfall ist in Tabelle 4 eingetragen.

Attribut	Wert
KundenID	10959
TransaktionsID	13790
Transaktionsbetrag	-147,06 €
Transaktionsdatum	27.02.2009
Transaktionsuhrzeit	13:26:18
Quellkontonummer	1934569
Quellrechner IP-Adresse	101.134.129.19
Empfängerkontonummer	2387121
Empfängerbankleitzahl	0403001 (= Bank of America, Tokyo)
Kontostand	-3.233,05 €
Dispolimit	3.800,00 €
Durchschnittlicher Transaktionsbetrag	-99,26 €
Durchschnittliche Transaktionszahl (Monat)	9,0
Auslandstransaktion	ja
Bekannter Empfänger	nein
Inländische IP-Adresse	nein
Zeit für die Online-Überweisung	37 sek.

Tabelle 4: Beispiel für einen verdächtigen Nicht-Betrugsfall

Bei dem hier angeführten verdächtigen Nicht-Betrugsfall handelt es sich um eine Auslandsüberweisung mit einem für Nicht-Betrugsfälle relativ hohen Verhältnis von Transaktionsbetrag zum maximal verfügbaren Betrag (25,94% = 147,06 * 100 / (-3.233,05 + 3.800,00)). Das Verhältnis von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag ist bei diesem Nicht-Betrugsfall ebenfalls höher (148,16% = 147,06 * 100 / 99,26). Hierbei könnte es sich z.B. um die Online-Bezahlung einer Urlaubsrechnung bereits am Urlaubsort selbst handeln, da die IP-Adresse des Quellrechners aus dem Ausland stammt.

Die angeführten Fallbeispiele aus den Tabellen 1 bis 4 werden in dieser Form für die Betrugsanalyse verwendet, die in den Kapiteln 6, 7, 8 und 9 beschrieben wird.

Die reale Anzahl an simulierten Betrugs- und Nicht-Betrugsfällen und der genaue Wertebereich jedes Attributs mit den möglichen Ausprägungen wird in Verbindung mit der Erläuterung des Aufbaus der Simulation der Transaktionen im Unterabschnitt 9.1.2 diskutiert.

2.3 Rechtslage und Statistiken zu Identitätsdiebstahl

Betrug ist ein Straftatbestand nach § 263 StGB. Der § 263a StGB Computerbetrug, der nachfolgend zitiert ist, erweitert § 263 StGB um die Bereiche Internetkriminalität und Computerbetrug, siehe [Fisc09, § 263a]:

> *(1) Wer in der Absicht, sich oder einem Dritten einen rechtswidrigen Vermögensvorteil zu verschaffen, das Vermögen eines anderen dadurch beschädigt, dass er das Ergebnis eines Datenverarbeitungsvorgangs durch unrichtige Gestaltung des Programms, durch Verwendung unrichtiger oder unvollständiger Daten, durch unbefugte Verwendung von Daten oder sonst durch unbefugte Einwirkung auf den Ablauf beeinflusst, wird mit Freiheitsstrafe bis zu fünf Jahren oder mit Geldstrafe bestraft.*
> *(2) § 263 Abs. 2 bis 7 gilt entsprechend.*
> *(3) Wer eine Straftat nach Absatz 1 vorbereitet, indem er Computerprogramme, deren Zweck die Begehung einer solchen Tat ist, herstellt, sich oder einem anderen verschafft, feilhält, verwahrt oder einem anderen überlässt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.*
> *(4) In den Fällen des Absatzes 3 gilt § 149 Abs. 2 und 3 entsprechend.*

In der Randnummer 11a zum § 263a StGB ist die Anwendung von § 263a Absatz 3 für die in dieser Arbeit diskutierten Betrugsart der Verwendung von Zugangscodes für den Kontenzugriff beim Online-Banking gegen den (erkennbaren) Willen des Berechtigten ausgeführt, siehe [Fisc09, § 263a Rdnr. 11a]. Für das Erreichen des Ziels, das vorangehende Abfangen bzw. das Vorbereiten zum Ausspähen von fremden Daten ebenfalls unter Strafe stellen zu können, wurden im Jahr 2007 zusätzlich zum bereits bestehenden § 202a StGB die §§ 202b und 202c StGB neu in das Strafgesetzbuch eingefügt [Fisc08, Vorwort zur 55. Auflage]. Aus den §§ 202a, 202b und 202c StGB ist ersichtlich, dass allein das Ausspähen und Abfangen der geheimen Daten von Bankkunden bzw. dessen Vorbereitung mit Freiheitsstrafen von bis zu einem, zwei oder drei Jahren bestraft werden kann.
Auch die Mittelsmänner, die das erbeutete Geld z.B. ins Ausland weiterleiten, gehen nicht straffrei aus. Diese Personen gelten aufgrund mangels hinreichend konkretisiertem Vor-

satzes nicht als Gehilfen im Sinne des § 263a StGB [Fisc09, § 263a Rdnr. 25], aber wie ein Urteil des Amtsgerichts Neunkirchen vom 13.03.2007 zeigt, werden diese Mittelsmänner auch bestraft. In diesem konkreten Fall wurde ein Ehepaar wegen gemeinschaftlicher Geldwäsche in Tateinheit mit gemeinschaftlicher fahrlässiger Erbringung von Finanzdienstleistungen ohne Erlaubnis zu einer Geldstrafe verurteilt, siehe [Arbe07].
Die zivilrechtliche Haftung nach einer illegalen Kontoplünderung durch einen erfolgreichen Identitätsdiebstahl ist wie folgt geregelt: Bei jeder Online-Überweisung erwirbt die Bank einen vertraglichen Erstattungsanspruch gegenüber dem Kunden in Höhe des Transaktionsbetrags. Dieser Erstattungsanspruch entsteht nur bei einer rechtmäßigen Überweisung, die aber bei einer Transaktion durch einen unberechtigten Dritten nicht zustande kommt. Um diesen Erstattungsanspruch dennoch geltend machen zu können, muss die Bank nachweisen, dass der Kunde den Missbrauch schuldhaft zu vertreten hat, indem er seine Sorgfaltspflicht verletzt hat. Die Kreditinstitute haben in der Regel Schwierigkeiten, diesen Beweis gegenüber ihren betrogenen Kunden zu führen. Die Bank hat in einem solchen Fall lediglich einen Erstattungsanspruch gegen den Betrüger, falls dieser ausfindig gemacht werden kann. Ist dies nicht der Fall, trägt das betroffene Kreditinstitut den Schaden. Allerdings muss der Kunde vor der Rückzahlung durch das Kreditinstitut Anzeige bei der Polizei gegen den Betrüger bzw. Anzeige gegen Unbekannt erstatten. [Spei07, S. 304 - 307]
Hierzu existiert ein Urteil des Amtsgerichts Wiesloch vom 20.06.2008, wonach die Bank für den Schaden nach einer geglückten Phishingattacke voll haftet, siehe [Arbe08].
Darüber hinaus entstehen Banken Bearbeitungskosten aufgrund auftretender Betrugstransaktionen und nicht zuletzt schaden erfolgreiche Identitätsdiebstähle ebenfalls der Reputation der Kreditinstitute [Reol07, S. 5; Nie05, S. 1] sowie dem Vertrauen der Kunden in die Bankenbranche und Internetgeschäften [Bign06, S. 7; Pate07, S. 3].
Trotz der oben genannten Gesetze gegen Identitätsdiebstahl steigt die Zahl der in diesem Umfeld durchgeführten Betrugsdelikte. Statistiken des Bundesverbands Informationswirtschaft Telekommunikation und neue Medien zeigen eine bedrohliche Entwicklung in diesem Bereich. Die Zahlen stammen aus dem Jahr 2007 und spiegeln damit den Zeitraum des Beginns dieser Arbeit wieder, siehe [Bitk08]:

- Im Jahr 2007 gab es in Deutschland 4.100 erfolgreiche Phishingfälle mit einem Schaden von insgesamt 19 Mio. Euro.

- Der durchschnittliche Schaden bei einer illegalen Überweisung lag in Deutschland im Jahr 2007 bei 3.200 Euro.

- Die Gesamtschadenssumme in Deutschland ist im Jahr 2007 um 25% im Vergleich zum Jahr 2006 gestiegen.

- Im Jahr 2007 wurden in Deutschland 25.000 Phishingversuche registriert, wobei die Betrüger dabei eine ebenso hohe Zahl an gefälschten Websites unterhielten.

Die folgenden Zahlen aus der Polizeilichen Kriminalstatistik für 2007 untermauern diesen Trend, siehe [Bund08]:

- Kontoeröffnungs- und Überweisungsbetrug stieg in Deutschland im Vergleich der Jahre 2006 zu 2007 um 36,2% von 13.297 auf 18.116 Fälle. Die Aufklärungsquote verschlechterte sich von 74,8% auf 72,5%.

- Das Ausspähen von Daten nach den §§ 202a, 202b und 202c StGB stieg in Deutschland im Vergleich der Jahre 2006 zu 2007 um 61,5% von 2.990 auf 4.829 Fälle. Die Aufklärungsquote verschlechterte sich von 43,8% auf 32,8%.

- Computerbetrug nach § 263a StGB stieg in Deutschland im Vergleich der Jahre 2006 zu 2007 um 0,4% von 16.211 auf 16.274 Fälle. Die Aufklärungsquote verschlechterte sich von 48,9% auf 37,2%.

- Insgesamt stieg die Computerkriminalität in Deutschland im Vergleich der Jahre 2006 zu 2007 um 6,4% von 59.149 auf 62.944 Fälle. Die Aufklärungsquote verschlechterte sich von 47,1% auf 42,3%.

Im Jahr 2008 hat sich die Lage noch verschärft, wie die nachfolgenden Zahlen aus der Polizeilichen Kriminalstatistik 2008 belegen, siehe [Bund09]:

- Kontoeröffnungs- und Überweisungsbetrug sank in Deutschland im Vergleich der Jahre 2007 zu 2008 um 11,5% von 18.116 auf 16.039 Fälle. Die Aufklärungsquote verschlechterte sich von 72,5% auf 69,2%.

- Das Ausspähen von Daten nach den §§ 202a, 202b und 202c StGB stieg in Deutschland im Vergleich der Jahre 2007 zu 2008 um 60,0% von 4.829 auf 7.727 Fälle. Die Aufklärungsquote verschlechterte sich von 32,8% auf 29,0%.

- Computerbetrug nach § 263a StGB stieg in Deutschland im Vergleich der Jahre 2007 zu 2008 um 4,5% von 16.274 auf 17.006 Fälle. Die Aufklärungsquote verschlechterte sich von 37,2% auf 37,1%.

- Insgesamt stieg die Computerkriminalität in Deutschland im Vergleich der Jahre 2007 zu 2008 um 1,1% von 62.944 auf 63.642 Fälle. Die Aufklärungsquote verschlechterte sich von 42,3% auf 40,3%.

Aber nicht nur Deutschland, auch das Ausland hat Probleme mit Identitätsbetrug, wie die nachfolgenden Zahlen zeigen, siehe [Idth08]:

- Eine von zehn Personen in Großbritannien wurde bereits Opfer von Identitätsdiebstahl.

- Identitätsdiebstahl wuchs um 500% seit dem Jahr 2000 in Großbritannien.

- Identitätsdiebstahl kostet der Wirtschaft Großbritanniens jährlich 1,7 Mrd. Pfund.

- Identitätsdiebstahl kostet der Wirtschaft und den Banken der USA jährlich 56 Mrd. Dollar.

- In Australien wurden bereits 1,1 Mio. Menschen Opfer von Identitätsdiebstahl.

- In den USA werden jedes Jahr 10 Mio. Menschen Opfer von Identitätsdiebstahl.

In Deutschland wurde im Jahr 2006 bei 1.817,97 Mio. Online-Überweisungen insgesamt ein Betrag von 1.685,58 Mrd. Euro via Online-Banking durch Nichtbanken transferiert [Deut08a]. In diesem Zusammenhang nutzten im Jahr 2008 ca. 22 Mio. Bankkunden das Online-Banking Angebot bei 35 Mio. online geführten Konten, wobei 52,0% der Kunden die bestehenden Sicherheitsvorrichtungen nicht für ausreichend halten [Hamb08; Bank08]. Trotz dieses hohen Anteils an Personen mit Sicherheitsbedenken zeigt sich eine wachsende Beliebtheit von Online-Banking, da im Jahr 2007 insgesamt 2.476,3 Mio. Überweisungen in Deutschland via Online-Banking durch Nichtbanken getätigt wurden [Deut08b], dagegen waren es im Jahr 2002 lediglich 767,65 Mio. Transaktionen [Deut08a].

Nach einer Studie von TNS Infratest Shared Services und dem Anbieter für Bankrechenzentrumsdienstleistungen Fiducia IT AG ist von 1.000 befragten Nutzern von Online-Banking insgesamt einem Anteil von 96,7% der Frauen und 92,0% der Männer die Sicherheit der wichtigste Faktor beim Durchführen von Online-Überweisungen. Auf der anderen Seite nutzen von den Befragten lediglich 38,9% die als sicher geltenden iTAN- und mTAN-Verfahren und 5,6% das HBCI-Verfahren (siehe Unterabschnitt 2.4.1). [Fidu08]
Aufgrund dieser genannten Betrugszahlen ist die Notwendigkeit zur Durchführung dieser Arbeit gegeben um auch von Seiten der Wissenschaft einen Beitrag zu leisten, diese alarmierenden Betrugszahlen zu verringern und somit die Situation zu verbessern.

2.4 Maßnahmen gegen Identitätsdiebstahl beim Online-Banking

Aufgrund der im vorangegangen Abschnitt genannten Zahlen wurden im Bankenumfeld in den letzten Jahren verschiedene Maßnahmen durchgeführt um den Identitätsdiebstahl einzudämmen. Methoden wie Phishing lassen sich nach Meinung von [Lini05, S. 151] und [Frau06] nicht komplett stoppen und es gibt dagegen auch keinen 100%-Schutz. Allerdings können sowohl die betroffenen Organisationen bzw. Banken als auch der Anwender selbst etwas unternehmen um die Gefahr zu verringern.

2.4.1 Maßnahmen von Bankenseite gegen Identitätsdiebstahl

Für die Erreichung des Ziels, die Kommunikation zwischen Kunden und Bank so sicher wie möglich zu gestalten, setzt die Bankenbranche Verschlüsselungsmethoden zur Betrugsprävention ein, wofür es zwei grundlegende Ansätze gibt [Jano06, S. 162].
Das am weitesten verbreitete Verfahren ist in diesem Zusammenhang das PIN/TAN-Verfahren. Zum Durchführen von Online-Transaktionen muss sich der Bankkunde – wie im Abschnitt 2.1 erwähnt – mittels seiner Persönlicher Identifikationsnummer (PIN) sowie seiner Kontonummer auf der Online-Banking-Seite der Bank anmelden. Für jede Transaktion benötigt der Kunde zusätzlich eine Transaktionsnummer (TAN), die als Einmalpasswort pro Überweisung gilt. Die TAN's und eine initiale PIN, die nach dem ersten Einloggen geändert werden muss, werden dem Kunden per Post von seiner Bank zugesandt. Diese Nummern bieten nur Schutz, solange sie keinem Unbefugten bzw. Betrüger in die Hände fallen. Sie sind daher das Ziel von Phishingangriffen, bei denen der Kunde aufgefordert wird, eben diese Nummern in eine gefälschte Website einzugeben. Daher gilt das PIN/TAN-Verfahren als unsicher. [Kraf07, S. 308 - 309; Jano06, S. 162 - 163; Lini05, S. 167 - 168]

Zum Ausgleich der Schwächen des PIN/TAN-Verfahrens entwickelten die Kreditinstitute folgende vier Erweiterungen des Verfahrens, siehe [Müll08, S. 464 - 465]:

a) iTAN: Beim iTAN-Verfahren handelt es sich um die Verwendung einer indizierten TAN. Es kann dabei nicht jede beliebige TAN für eine Transaktion aus der TAN-Liste ausgewählt werden, sondern die Bank gibt für jede Überweisung einen bestimmten Index an. Der Kunde muss für diese bestimmte Transaktion exakt die TAN auswählen, die mit dem angegeben Index versehen ist. Dafür hat der Kunde maximal fünf Minuten Zeit. Alle anderen TAN's sind für diese Überweisung ungültig. Das ursprüngliche Verfahren wurde zum iTAN plus weiterentwickelt. Hierbei muss der Kunde die Transaktionsdaten zusammen mit einem Kontrollbild vor dem Ausführen der Überweisung nochmals bestätigen.

b) eTAN: Beim eTAN-Verfahren stellt das Kreditinstitut dem Kunden einen TAN-Generator in Form eines zusätzlichen Geräts zur Verfügung. Vor der Durchführung einer Überweisung erzeugt der Kunde mit diesem Gerät eine TAN, die nur eine bestimmte Zeit lang gültig ist. Dieses Gerät wird zur Erhöhung der Sicherheit regelmäßig ausgetauscht. Auch dieses Verfahren wurde vom ursprünglichen Modus zum eTAN plus weiterentwickelt. Hierbei müssen die Transaktionsdaten vom Kunden vor Ausführen der Überweisung nochmals bestätigt werden.

c) mTAN: Beim mTAN-Verfahren bekommt der Kunde per SMS seine TAN für eine Überweisung auf sein Mobiltelefon zugeschickt, die anschließend fünf Minuten lang gültig ist.

d) smartTAN: Beim smartTAN-Verfahren wird die TAN ebenfalls mit einem TAN-Generator erzeugt. Hierbei ist die generierte TAN mit dem jeweiligen Kunden logisch verknüpft, z.B. mit Informationen der Giro- oder Kreditkarte, welche die Bank dem Kunden zur Verfügung stellt.

Das andere, nicht so weit verbreitete Verschlüsselungsverfahren ist das *Home Banking Computer Interface* (HBCI). Beim HBCI-Verfahren wird eine Chipkarte mit einem Chipkartenlesegerät verwendet. Beim Online-Banking wird die Chipkarte von dem Lesegerät ausgelesen. Ohne diese Chipkarte, die den Benutzer eindeutig identifiziert, kann keine Online-Transaktion ausgeführt werden. Dabei existieren drei verschiedene Klassen, siehe [Kraf07, S. 312 - 313; Jano06, S. 163 - 165]:

Klasse 1: Ein Lesegerät kann mittels USB an den Rechner angeschlossen werden, wobei die PIN-Eingabe über die Tastatur des Rechners erfolgt.

Klasse 2: Die Daten werden direkt in den Chipkartenleser eingegeben.

Klasse 3: Der Chipkartenleser besitzt ein eigenes Display, worauf nochmals die Gültigkeit der eingegebenen Daten angezeigt wird. Der Chipkartenleser ist mit einem Signaturmodul ausgestattet, mit welchem dem Empfänger zusätzlich die Identität des Benutzers garantiert wird.

Nach dem Beenden der Online-Banking-Sitzung wird die Chipkarte wieder aus dem Kartenleser gezogen. Somit können nachträglich keine Angriffe auf die Chipkarte erfolgen. Die Chipkarte selbst ist wiederum durch eine PIN geschützt um eventuellen Missbrauch bei Verlust der Karte zu vermeiden. Das HBCI-Verfahren gilt im Vergleich zum PIN/TAN-Verfahren als sichere Methode, da bei Geräten ab Klasse 1 Betrüger keine Chance haben, über den PC an die vertraulichen Daten zu gelangen. Der Grund dafür ist, dass alle entscheidenden Sicherheitsprozeduren außerhalb des PC's ablaufen. Die sicheren HBCI-Geräte konnten sich aber aufgrund der Kosten (zwischen 45 und 400 Euro pro Gerät) bislang nicht gegen das PIN/TAN-Verfahren durchsetzen. [Kraf07, S. 312 - 313; Jano06, S. 163 - 165; Lini05, S. 168]
Die Möglichkeiten des HBCI-Verfahrens werden zusätzlich durch den Standard FinTS V4.0 erweitert, der die Übertragung der Daten im XML-Format ermöglicht (die Abkürzung XML steht für *Extensible Markup Language* und bezeichnet einen Standard zur Definition von Auszeichnungssprachen mit dem Ziel der Trennung von Daten und Repräsentation, für weiterführende Informationen zu XML siehe [Haro04]). Für weiterführende Literatur zu FinTS V4.0, siehe [Jano06, S. 163 – 165; Haub04].
Einen ähnlichen Ansatz zur Betrugsprävention mit HBCI verwendet [Puen05]. In diesem Artikel wird ein Gerät beschrieben, das über USB oder *Bluetooth* (als *Bluetooth* werden miteinander interagierende Funkkomponenten für die drahtlose Datenübertragung bezeichnet, siehe [Jano06, S. 45]) mit dem Rechner verbunden ist und die Signatur einer Transaktion empfängt, die anschließend verschlüsselt an den Rechner zurückliefert wird. Das Gerät kann nach seiner ersten Aktivierung nicht mehr umprogrammiert werden, siehe [Puen05, S. 2].
Dagegen setzt [Hu09] bei der Authentifizierung auf eine Kombination von *secret key*- und *smart cart*-Techniken. Das in dem Artikel diskutierte System generiert in diesem Zusammenhang digitale Signaturen und wird nach Aussage der Autoren eine wichtige Rolle für die Sicherheit beim Online-Banking spielen, siehe [Hu09, S. 3].
Ein weiteres Authentifizierungsverfahren, das auf *Dynamic Key Generation* (DKG) und *Group Key* (GK) basiert, ist in [Dand07a] diskutiert sowie dessen Weiterentwicklung mit der Integration des *Internet Banking Payment Protocol* (IBPP), das eine separate Authen-

tifizierung für jede einzelne Transaktion beinhaltet, siehe [Dand07b]. Eine zusätzliche Authentifizierung für jede einzelne Transaktion ist ebenfalls in [Alzo08] beschrieben. Ein ähnliches Protokoll zur Authentifizierung ist das *Transport Layer Security* (TLS) Protokoll, das z.B. in [Badr07] oder [Herz08] erläutert wird. In den Artikeln ist kein Hinweis enthalten, ob das Verfahren bereits in der Praxis eingesetzt wird.

Einen anderen Ansatz zur Authentifizierung beinhaltet die Arbeit von [Dham05]. Hierbei muss der Anwender beim Einloggen sowie vor dem Ausführen einer Transaktion bestimmte vertraute Bilder verifizieren, die ihm vom System mittels Zufallsgenerator vorgeschlagen werden, siehe [Dham05, S. 4]. Diese Methode wird beispielsweise bei der `Bank of America` eingesetzt [Jako08, S. 1].

In [Pate07, S. 3] dagegen wird Authentifizierung mittels Fingerabdruckerkennung vorgeschlagen.

Die Autoren von [Edge07] entwickelten ein Baumstrukturmodell, das zum Modellieren von Angriffsszenarien auf Kreditinstitute sowie zum Auffinden der effektivsten und kostengünstigsten Schutzmethodik dient. Die Bäume sind im Rahmen der Arbeit von [Edge07] so aufgebaut, dass im obersten Knoten das allgemeine Ziel definiert ist. In den darunterliegenden Schichten sind jeweils Unterziele zum Erreichen des übergeordneten Ziels aufgeführt. In den Endknoten sind die jeweils zur Zielerreichung notwendigen Aktionen genannt. Dabei werden in dem Artikel ein *Attack Tree*, der die Zielerreichung aus Sicht des Angreifers eines Online-Banking-Systems darstellt sowie ein *Protection Tree*, der die Ziele aus Sicht der Sicherheitsbeauftragten eines Online-Banking-Systems definiert, beschrieben. Die einzelnen Knoten der Bäume beinhalten drei Metriken. Diese Metriken sind die Wahrscheinlichkeit der jeweiligen Zielerreichung als Wert zwischen 0 und 1, die Kosten der Zielereichung in US Dollar sowie die Auswirkungen auf das Online-Banking-System auf einer Skala mit einer Größenordnung von 1 - 10. Die Werte dieser Metriken werden von entsprechenden Experten für jeden Knoten manuell festgelegt. Anhand dieser Strukturen können z.B. die Kosten für eine Schutzwahrscheinlichkeit von 50% des kompletten Systems ermittelt werden oder die Schutzwahrscheinlichkeit bei einem Sicherheitsbudget von 900.000 Dollar. Eine Schwäche dieses Modells ist lt. den Autoren, dass hierbei nur bekannte Angriffs- und Schutzszenarien modelliert werden können, siehe [Edge07, S. 2].

Für das Ziel, dem Problem des Identitätsdiebstahls beim Online-Banking und bei Kreditkartenzahlungen besser begegnen zu können, tauschen Online-Banken aus mittlerweile 130 Ländern im dafür gegründeten *RSA eFraudNetwork* Informationen zu identifizierten Betrügern wie z.B. verdächtige IP-Adressen aus, siehe [Zhan09, S. 2]. Zu diesem Zweck wurde z.B. das *Incident Object Description Exchange Format* (IODEF) erweitert (das IODEF-Format ist ein einheitliches Format für *Computer Security*-Beauftragte zum Zweck des Austauschs von Informationen über Sicherheitslücken, deren Beteiligte und Gegen-

maßnahmen, siehe [Cove08]). Das Ziel ist, den Informationsaustausch bezüglich Phishingfällen und anderen Arten von Internetbetrug auf Basis einer XML-Struktur zu unterstützen [Cain09].

Laut den Aussagen aus den Experteninterviews (siehe Anhang 1) setzen Kreditinstitute in Deutschland bezüglich ihrer Betrugsbekämpfungsstrategie größtenteils auf Betrugsprävention mittels der genannten Verschlüsselungs- bzw. Authentifizierungsverfahren anstatt auf aktive Betrugserkennung mittels Analyse der durchgeführten Finanztransaktionen.

Dennoch existieren kommerzielle Systeme zur Transaktionsüberwachung auf dem Markt. Diese Systeme können in regelbasierte und statistische Verfahren, wie z.B. auf Basis von neuronalen Netzwerken, unterteilt werden [Slew04, S. 4].

Das amerikanische Unternehmen `Fair Isaac Corp.` benutzt z.B. mit seiner Lösung *Falcon Fraud Manager* ein neuronales Netzwerk zur Überprüfung von Kreditkartenzahlungen [Fico09a]. Eine Erweiterung der Lösung mit der Bezeichnung *Falcon ID* besteht aus einer Kombination von benutzerbasierten Fragen zur Authentifizierung (Betrugsprävention) sowie Transaktionsanalyse mittels einem neuronalen Netzwerk (Betrugserkennung) zur Bekämpfung von Identitätsdiebstahl beim Online-Banking [Fico09b]. Ein anderes Unternehmen aus den Vereinigen Staaten namens `ID Analytics Inc.` verwendet dagegen feste Muster, die das normale Transaktionsverhalten beim Online-Banking repräsentieren. Passt eine Transaktion nicht in das Muster, wird sie als Anomalie und somit als Betrug eingestuft. Beide Systeme haben das Problem, dass oftmals „gutartige" Transaktionen fälschlicherweise als Betrug eingestuft werden, was als *false positive* (d.h. fälschlicherweise positiv klassifiziert. Umgekehrt wird bei *false-negative*-Bewertung ein Objekt negativ klassifiziert, das aber positiv klassifiziert werden müsste, zur genauen Begriffserklärung siehe [Mcdo05, S. 24; Reol07, S. 69; Bose06, S. 3]) oder VIRT-Problem bezeichnet wird, das im Abschnitt 6.1 genauer erläutert wird. [Lini05, S. 183 - 184]

Das irische Unternehmen `Norkom Technologies` bietet eine Lösung zur Betrugsbekämpfung im Online-Banking, die auf Basis von bekannten Informationen das Transaktions- sowie das Webverhalten des jeweiligen Kunden in Echtzeit abprüft. Mit diesem Ansatz wird ermittelt, ob das gegenwärtige Verhalten mit der normalen Vorgehensweise des Benutzers übereinstimmt oder nicht. [Nork09]

In Deutschland hat sich die `Inform GmbH` mit ihrer Lösung *Risk Shield* auf Transaktionsüberwachung bei kartenbasierter Zahlung spezialisiert. Dieses System basiert ebenfalls auf Vergleichsverfahren mit bekannten Betrugsmustern. [Info08]

Falls bei der Transaktionsüberwachung eine Überweisung als betrugsverdächtig eingestuft wird, wird diese gestoppt und manuell überprüft. Besteht in diesem Zusammenhang z.B. der Verdacht der Geldwäsche, werden die gesamten Gelder des Kunden eingefroren und der Fall der Staatsanwaltschaft übergeben. Falls von dieser Instanz kein kriminelles

Verhalten nachgewiesen werden kann, werden die Gelder des Kunden im Anschluss wieder freigegeben. Andernfalls wird ein Strafverfahren eingeleitet. [Fina08]
Nach den Aussagen aus den Interviews würde bei einem Betrugsverdacht speziell beim Online-Banking die Transaktion gestoppt und der Kunde anschließend kontaktiert. Es ist dabei aufgrund des (möglichen) Reputationsverlusts problematisch, den Kunden zu fragen ob die Transaktion in dieser Form gewollt ist. Falls es sich um einen Betrugsfall handelt, wird die Transaktion – sofern noch möglich – rückabgewickelt und der Fall an die Staatsanwaltschaft übergeben.
Die Lösung dieser Arbeit untersucht bzw. überwacht die Transaktionen im Zahlungsverkehr, wobei hier – wie im Abschnitt 1.1 erwähnt – die Analyse zur Betrugsidentifikation auf Basis von *Complex Event Processing*, Entscheidungsbäumen, Diskriminanzanalyse und neuronalen Netzwerken in einem Hybrid-Modell zur Betrugserkennung erfolgt. Dieses Modell wird mit seinen Bestandteilen in den Kapiteln 6, 7, 8 und 9 erläutert.

2.4.2 Maßnahmen von Kundenseite gegen Identitätsdiebstahl

Zur Verringerung der Gefahr des Diebstahls der eigenen Identität durch Betrüger sollten von Seiten der Anwender die folgenden Hinweise beachtet werden, siehe [Kraf07, S. 308; Frau06; Lini05, S. 151 - 163]:

a) Emails, die angeblich von Banken stammen und eine Überprüfung ihrer persönlichen Daten beinhalten, sollten sofort gelöscht werden. Solche wichtigen Nachrichten würden Banken nur über den klassischen Postweg versenden.

b) Die Homepage der Bank zum Online-Banking sollte nicht über einen Link geöffnet werden. Stattdessen sollte die Internetadresse manuell in den Browser eingegeben werden.

c) Die Browser- und Betriebssystemupdates sollten regelmäßig durchgeführt werden.

d) Für die lokale Firewall und den Antivirenscanner sollten regelmäßig Updates durchgeführt werden. Wenn diese Sicherheitsprogramme nicht installiert sind, sollte dies dringend nachgeholt werden.

e) Emails von unbekannten Absendern sollten nicht geöffnet werden. Auch sollten Dateien mit den Endungen .exe und .bat aus Email-Anhängen nicht ausgeführt werden.

f) Nach Möglichkeit sollte ein vertrauenswürdiger und anerkannter Internetprovider gewählt werden.

g) Die Browseroption „Ausführen von Java Script“ sollte deaktiviert sein.

h) Die Übertragung von Daten sollte mit SSL-Verschlüsselung erfolgen (SSL steht für *Secure Sockets Layer* und dient der Erstellung von sicheren Kommunikationskanälen innerhalb von Netzwerken, siehe [Jano06, S. 313]).

i) Browser ab der Version *Internet Explorer 7* oder *Mozilla Firefox* sollten nach Möglichkeit verwendet und der darin integrierte Phishing-Blocker aktiviert werden. Diese Funktion soll nach Angaben der Hersteller Phishingseiten erkennen und blocken.

j) Manche Email-Programme, wie z.B. *Thunderbird*, sind nach Angabe des Herstellers in der Lage, gefährliche Phishing-Emails entsprechend zu markieren und sollten daher bevorzugt verwendet werden.

k) Für Online-Überweisungen sollte nur der eigene Heimcomputer, keine fremden PC’s verwendet werden.

l) Für die Höhe von Online-Transaktionsbeträgen sollte ein Tages- oder Transferlimit mit der Hausbank vereinbart werden.

m) Verdächtige Emails, Webseiten oder Kontenbewegungen sollten sofort der Hausbank gemeldet werden.

Nach Aussage von [Frau06] ist eine gewisse Skepsis und Wachsamkeit bereits ein erster bedeutsamer Schritt, um den Missbrauch der eigenen Identität für Betrugszwecke zu vermeiden.
Die in diesem Unterabschnitt genannten Maßnahmen dienen zur Betrugsprävention. Diese Arbeit dagegen setzt mit ihrem Ansatz der Transaktionsüberwachung zur Betrugserkennung zu einem späteren Zeitpunkt an. Dieser tritt ein, nachdem ein Betrüger erfolgreich die relevanten Zugangsdaten erlangt und die Betrugstransaktion gestartet hat.
Trotz aller gegebenen Sicherheitsmaßnahmen, sowohl von Seiten der Bankenbranche als auch der Anwender, besteht immer noch die Gefahr, Opfer von Identitätsdiebstahl zu werden, wie die aufgeführten Statistiken im Abschnitt 2.3 zeigen. Die Gründe dafür sind, dass die Betrugsmethoden zum einen von Seiten der Betrüger permanent weiterentwi-

ckelt werden und zum anderen, dass viele Kunden beim Online-Banking die gegebenen bzw. angebotenen Sicherheitsvorkehrungen nicht oder nicht vollständig einsetzen. Dieses Kapitel zeigt, dass trotz vorhandener Sicherheitsvorkehrungen noch immer viele Betrugsversuche gelingen. Daraus entsteht die Notwendigkeit, diesem Problem mit weiteren Lösungsmethoden zu begegnen, z.B. mit einzelnen oder mehreren kombinierten Verfahren der Wissensrepräsentation, die im nächsten Kapitel erläutert werden.

3 Methoden und Algorithmen der Wissensrepräsentation

In diesem Kapitel werden die gängigen Methoden der Wissensrepräsentation und Datenanalyse vorgestellt. Die Begründungen für die Verwendung bzw. Nicht-Verwendung der, in diesem Kapitel beschriebenen Methoden, werden für die Anforderungen dieser Arbeit im Abschnitt 6.2 gegeben.

Zu den Verfahren der Wissensrepräsentation zählen u.a. regelbasierte Ansätze bzw. deduktive maschinelle Lernverfahren und induktive maschinelle Lernverfahren [Beie06, S. 3; Krah98, S. 59 - 60]. Regelbasierte Verfahren gehören zu den ältesten Formen der Wissensrepräsentation und basieren auf erprobten fixen (deterministischen) Regeln. [Beie06, S. 3]

Sie sind durch Bedingungen, Regeln und Schlussfolgerungen definiert, z.B. „wenn ‚a' und ‚b' auftreten folgt daraus ‚c'". In diesem Fall ist ‚a' eine Bedingung, ‚b' ist eine Bedingung, ‚a' und ‚b' ist die Regel und ‚c' ist eine Schlussfolgerung daraus. [Lust02, S. 91; Krah98, S. 72]

Eine Form der regelbasierten Systeme sind die Expertensysteme, die mit Hilfe von Expertenwissen und Schlussfolgerungstechniken versuchen, Antworten auf Benutzeranfragen zu bestimmten Problemen zu finden. Diese Schlussfolgerungen werden deterministisch oder stochastisch ermittelt. Bei deterministischen Schlussfolgerungstechniken wird versucht, ein Problem mit Hilfe von logischem Schließen aus definierten Regeln zu beheben. Bei stochastischen Schlussfolgerungstechniken hingegen werden ebenfalls deterministische Regeln verwendet, wobei hier zusätzlich ein Maß für die Ungewissheit mit berücksichtigt wird. Die daraus entstehenden Schlussfolgerungen werden anschließend mit bestimmten Werten, welche die Wahrscheinlichkeit des Zutreffens einer Schlussfolgerung beurteilen, versehen. [Bode03, S. 133]

Expertensysteme können zusätzlich mit Fuzzy-Logik erweitert werden [Beie06, S. 426]. Fuzzy-Logik dient zur Modellierung unscharfer Zuordnungen zu Gruppen. Ein Beispiel bildet die Körpergröße von Menschen. Bei „scharfer" Klassifizierung in große und kleine Menschen kann das Körpermaß von 1,85m als hartes Grenzmaß verwendet werden. Mit

Hilfe von Fuzzy-Systemen lässt sich ein „weicher“ Übergang erstellen, z.B. ein Mensch mit einer Größe von 1,83m ist „ziemlich groß“. Somit kann der Fuzzy-Ansatz auch Ungenauigkeiten z.B. bei Messergebnissen berücksichtigen. Für weiterführende Literatur zu Fuzzy-Systemen sei auf [Krus93] oder [Muna08, S. 121 - 161] verwiesen. Im Rahmen von Expertensystemen können unscharfe Fuzzy-Regeln, wie z.B. „große Menschen sind meistens schwer“, verwendet werden um innerhalb des Expertensystems vages Wissen auszudrücken und Schlussfolgerungen aufgrund dieses unscharfen Wissens zu ziehen. Die exakte Vorgehensweise ist in [Beie06, S. 426 - 429] beschrieben.
Der in dieser Arbeit vorgestellte Ansatz zur Betrugserkennung basiert auf der Verarbeitung von Massendaten. Hierfür sind maschinelle Lernverfahren gut geeignet, wie z.B. die Arbeit von [Nime07] zeigt. Maschinelles Lernen lässt sich – wie oben erwähnt – in zwei Gruppen von Lernmethoden einteilen, siehe [Krah98, S. 59 - 60]:

a) Deduktives Lernen: Beim deduktiven Lernen wird das in den Wissensbasen vorhandene Wissen analysiert um es zukünftig effizienter und automatisch einsetzen zu können, wie z.B. Expertensysteme.

b) Induktives Lernen: Beim induktiven Lernen werden aus dem in den Wissensbasen bestehenden Wissen neue Zusammenhänge erkannt. Induktive Lernverfahren lassen sich in überwachte und unüberwachte Lernverfahren einteilen. Bei überwachten Lernverfahren ist das gewünschte Lernergebnis bereits im Vorfeld festgelegt, wodurch der Lernerfolg gut überprüft werden kann. Bei unüberwachten Lernverfahren steht das Lernziel zu Beginn der Analyse nicht fest. Bei diesen Algorithmen wird innerhalb einer bestehenden Datenmenge nach Strukturen und Zusammenhängen gesucht. Hierbei gibt es keine Garantie, dass gefundene Strukturen auch hilfreich zur Lösung eines Problems sind und müssen somit von einem Menschen fachlich geprüft werden [Reol07, S. 2].

Die einzelnen Funktionen sowie der Aufbau des maschinellen Lernens sind in Abbildung 5 graphisch dargestellt.

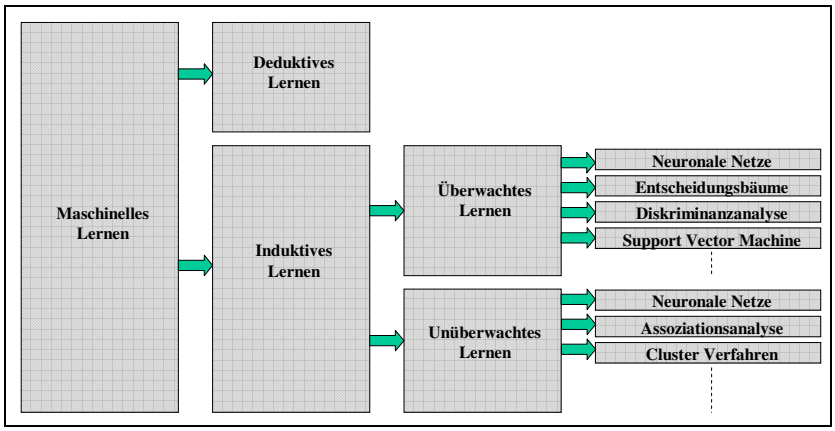

Abbildung 5: Überblick über die Methoden des maschinellen Lernens in Anlehnung an [Krah98, S. 40]

Die Verfahren des maschinellen Lernens können lt. [Beie06, S. 98 - 99] nach den drei nachfolgend genannten Dimensionen klassifiziert werden.

a) Klassifikation auf Basis der zugrunde liegenden Lernstrategie, wobei unterschieden wird, in welchem Maße Informationen bereits vorliegen und das System eigene Schlussfolgerungen zieht.

b) Klassifikation gemäß der benutzten Repräsentation des Wissens.

c) Klassifikation gemäß dem Anwendungsbereich des lernenden Systems.

Im Rahmen dieser Arbeit wird – wie im Abschnitt 1.1 erwähnt – der Fokus auf die zugrunde liegende Analysemethodik zur Betrugserkennung gelegt. Die Erkennungsleistung von maschinellen Lernverfahren wird nach Erkennungsgenauigkeit sowie nach den Faktoren *false positive* und *false negative* – wie im Unterabschnitt 2.4.1 definiert – ermittelt [Abu07, S. 2], wobei dies im Abschnitt 6.1 genauer erläutert wird. Die Versorgung mit den auszuwertenden Daten bzw. *events* erfolgt mittels *Complex Event Processing*-Technologie, die im nächsten Kapitel beschrieben wird.
In den folgenden Abschnitten werden die wichtigsten Verfahren des maschinellen Lernens vorgestellt. Da in dieser Arbeit aufgrund der vorhandenen zu analysierenden Datenstrukturen (siehe Beispiele für Betrugs- und Nicht-Betrugsfälle im Abschnitt 2.2) Entscheidungsbäume, Diskriminanzanalyse und neuronale Netzwerke zur Betrugserkennung eingesetzt werden, sind diese Verfahren umfangreicher beschrieben. Diese Methodiken sind, wie in Abbildung 5 zu erkennen ist, den überwachten Lernverfahren zugeteilt. Überwachte Lernverfahren werden benötigt, da die Struktur der Trainingsdaten auch ein boolesches Datenfeld beinhaltet, das den Betrugsstatus (Betrugs- oder Nicht-Betrugsfall) des Daten-

satzes bzw. der Transaktion enthält. Die Eignung überwachter Verfahren für Datenstrukturen mit bekanntem Lernstatus wird von [Jain00, S. 30], [Bolt02, S. 9] und [Kou04, S. 2] bestätigt. Des Weiteren sind signifikante Betrugsmuster im Bereich Betrugstransaktionen durch Identitätsdiebstahl, die im Abschnitt 2.1 erläutert wurden, bereits bekannt. Aus diesen Gründen werden nachfolgend verstärkt überwachte Lernverfahren, wie z.B. Entscheidungsbäume beschrieben. Die Begründung für die Auswahl einer Kombination dieser drei genannten Verfahren im Rahmen dieser Arbeit wird im Abschnitt 6.2 gegeben.

3.1 Entscheidungsbaumverfahren

Entscheidungsbäume gehören zu den überwachten Lernverfahren und basieren auf Wenn-Dann-Regeln [Krah98, S. 69 - 72; Beie06, S. 107]. Eine Wenn-Dann-Regel ist z.B. „WENN der Hahn kräht, DANN ändert sich das Wetter und WENN der Hahn schweigt, DANN bleibt das Wetter“ [Lust02, S. 96]. In Abbildung 6 ist diese Regel graphisch dargestellt.

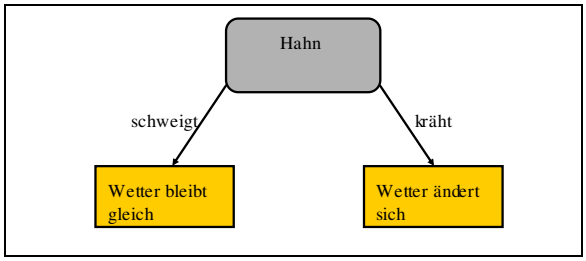

Abbildung 6: Einfacher Entscheidungsbaum in Anlehnung an [Lust02, S. 96]

Ein Entscheidungsbaum kann – wie in Abbildung 6 dargestellt – einstufig sein. Falls es bei einem oder mehreren Ästen weitere Unterteilungsmöglichkeiten gibt, handelt es sich um einen mehrstufigen oder zusammengesetzten Entscheidungsbaum. Abbildung 7 bildet einen mehrstufigen Entscheidungsbaum in einer allgemeinen Form ab.

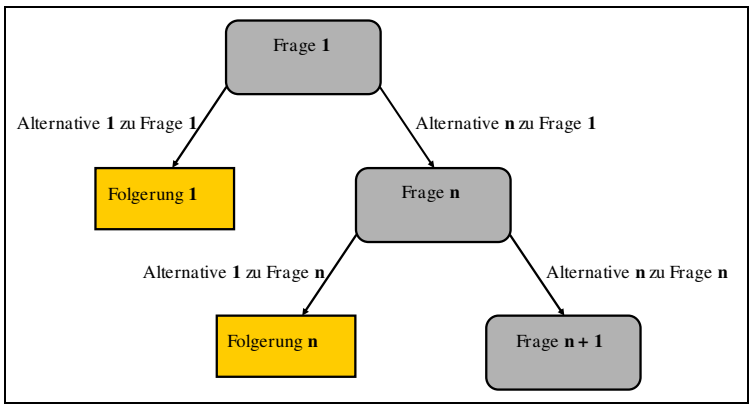

Abbildung 7: Mehrstufiger Entscheidungsbaum in Anlehnung an [Lust02, S. 97]

Als Attribute an den Knoten bzw. Trennstellen eines Entscheidungsbaums können sowohl metrische Werte (siehe Abbildung in [Lust02, S. 86]) als auch nichtmetrische Werte (siehe Abbildung in [Krah98, S. 70]) verwendet werden.
Als nichtmetrische Attribute werden alle Attribute der Nominalskala (ohne Bildung einer qualitativen Reihenfolge) und Ordinalskala (mit Bildung einer qualitativen Reihenfolge) bezeichnet, die zur Klassifizierung qualitativer Eigenschaftsausprägungen dienen. Unter diese Typen fallen beispielsweise Attribute wie Geschlecht (männlich, weiblich), Farbe (rot, gelb, grün usw.), Religion (römisch-katholisch, evangelisch-lutherisch usw.) oder Postleitzahlen. Im Gegensatz dazu werden alle Attribute der Intervallskala (ohne natürlichem Nullpunkt) und der Ratioskala (mit natürlichem Nullpunkt) als metrische Attribute bezeichnet. Sie zeichnen sich durch die Eigenschaft aus, dass mit ihnen die vier Grundrechenarten und Mittelwertbildung durchgeführt werden können. Beispiele für diese Attributtypen sind Einkommen (Monats-/ Jahreseinkommen in Euro), Körpergröße (in cm) oder Kosten (Betrag in Euro). [Back06, S. 4 - 6]
Die Positionierung einer Wenn-Dann-Regel bzw. eines Attributs innerhalb des Entscheidungsbaums vom Informationsgehalt des Attributs in Bezug auf die Zielgröße abhängig ist. Je größer der Informationsgehalt, desto weiter oben im Baum wird das Attribut platziert. [Krah98, S. 71; Beie06, S. 109]

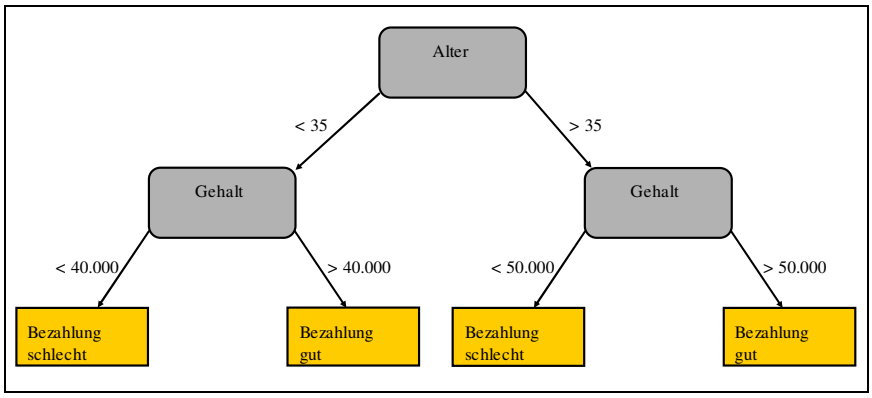

Abbildung 8: Mehrstufiger Entscheidungsbaum zur Gehaltsklassifikation in Anlehnung an [Krah98, S. 70]

Das Beispiel in Abbildung 8 zeigt, dass in dem Fall der Einordnung der Bezahlung in „gute Bezahlung“ bzw. „schlechte Bezahlung“ in erster Linie das Alter einen höheren Informationsgehalt besitzt als die Höhe des Gehaltes an sich, da aufgrund der Gehaltshöhe allein noch keine eindeutige Aussage getroffen werden kann, ob die Bezahlung gut oder schlecht ist. Dieses Beispiel aus Abbildung 8 ist sehr einfach gewählt, da neben dem Alter auch noch Attribute wie „Bildungsabschluss“ oder „Führungsverantwortung“ usw. eine wichtige Rolle spielen und somit bei der Frage nach der Güte der Bezahlung ebenfalls Berücksichtigung finden müssten.

Zur Ableitung exakter Regeln aus einer Trainingsmenge mit verschiedenen Attributen müssen von der Wurzel bis zum untersten Knoten die Wenn-Dann-Regeln so konstruiert werden, dass alle Elemente der Trainingsmenge beim Testen in die entsprechende Gruppen eingeteilt werden, deren Zugehörigkeit bei den Trainingsdaten vorher bekannt ist (überwachter Ansatz) [Beie06, S. 108].
Diese Regeln, welche die Struktur eines Entscheidungsbaums bilden, können entweder manuell auf Basis von Erfahrungswerten aufgestellt oder automatisch aus einer Trainingsmenge generiert bzw. gelernt werden. Ein vereinfachtes heuristisches Verfahren zum automatischen Erzeugen eines Entscheidungsbaums aus [Lust02, S. 303 - 304] lautet:

Schritt 1: Ermittlung, wie gut jedes Attribut allein die Elemente der Trainingsmenge klassifiziert.

Schritt 2: Positionierung des Attributs auf der aktuellen Baumebene, welches die Trainingsmenge am besten klassifiziert.

Schritt 3: Wiederholung der ersten beiden Schritte für die unteren Baumebenen, bis ein zuvor definiertes Abbruchkriterium erreicht oder die komplette Trainingsmenge korrekt klassifiziert ist.

Für diese Aufgabe existieren in der Praxis eine Reihe von induktiven Lernalgorithmen, die *Top Down Induction of Decision Trees* (TDIDT) Verfahren. Beispiele dafür sind *Classification And Regression Trees* (CART) für die Verarbeitung stetiger unabhängiger Attribute [Lust02, S. 302; Pete05, S. 182 - 183] oder *Chi Sqare Automatic Interaction Detection* (CHAID) für nicht-binäre Entscheidungsbäume [Lust02, S. 302; Pete05, S. 160 - 162]. Die etabliertesten TDIDT-Algorithmen bzw. Entscheidungsbaumlernsysteme sind *Iterative Dichotomiser 3* (ID3) und dessen Weiterentwicklungen C4.5 bzw. C5.0 [Beie06, S. 115; Pete05, S. 143]. Diese Verfahren berechnen für die Erstellung des optimalen Entscheidungsbaums anhand der vorhandenen Attribute und der bekannten Gruppenklassifikationen der Trainingsdaten den mittleren Informationsgehalt eines Attributs, der in der Literatur auch als Entropie bezeichnet wird [Beie06, S. 115; Lust02, S. 305].
Die Entropie Ent stellt eine Verbindung zwischen Wahrscheinlichkeit und Information her. Sie misst dabei die Unsicherheit der Information eines zu erwartenden Elementarereignisses w in Kenntnis der Wahrscheinlichkeitsverteilung P [Beie06, S. 447 - 448]. Der Begriff Entropie wurde in der Literatur zum ersten Mal von C. Shannon und W. Weaver erwähnt, siehe dazu [Shan63].

Der ID3-Algorithmus hat folgenden Ablauf, siehe [Lust02, S. 304 - 309; Pete05, S. 143 - 148]:

Schritt 1: Für jedes mögliche Attribut der Unterknoten wird die Entropie Ent_{sub} auf Basis der vorhandenen Attributausprägungen und der Klassifikationsverteilung der Trainingsmenge in die verschiedenen Gruppen in Abhängigkeit der Häufigkeit der Ausprägungen mit folgender Formel berechnet:

$$Ent_{sub} = -\sum_{w} P(w) * \log_2 P(w)$$ [Beie06, S. 448; Lust02, S. 305]

Schritt 2: Auf Basis der ermittelten Entropien wird der Erwartungswert bzw. die gewichteten Entropien Ent_{gew} des Attributs berechnet. Der Erwartungswert ergibt sich aus der Summe der mit der Häufigkeit der Ausprägungen im Verhältnis zur Trainingsmenge gewichteten Entropien der einzelnen Ausprägungen des Attributs. Dies wird durch folgende Formel realisiert:

$$Ent_{gew} = \sum_{w} P(w) * Ent_{sub}(w)$$ [Lust02, S. 307]

Schritt 3: Auf Basis der Verteilung der Attributausprägungen w wird die Entropie Ent_{root} für den Wurzelknoten bzw. das aktuelle Attribut, welches den Wurzelknoten der aktuellen Ebene bilden könnte, berechnet. Hierbei wird im Vergleich zum Schritt 1 die Häufigkeit des Auftretens der Gruppenklassifizierungen der Trainingsmenge im Verhältnis zur Attributausprägung nicht berücksichtigt. Für die Berechnung der Entropie des Wurzelknotens liegt folgende Formel zugrunde:

$$Ent_{root} = -\sum_{w} P(w) * \log_2 P(w)$$ [Lust02, S. 307]

Schritt 4: Der absolute Klassifikationsgewinn bzw. Informationsgewinn G_A eines Attributs A wird mit folgender Formel berechnet:

$$G_A = Ent_{root} - Ent_{gew}$$ [Lust02, S. 307]

Diese Schritte werden iterativ für jedes Attribut wiederholt, bis das Attribut mit dem höchsten Klassifikationsgewinn gefunden wurde. Dieses wird danach auf der aktuellen Ebene positioniert. Anschließend beginnt der Algorithmus erneut mit der nächst tieferen Baumebene. Das Verfahren ist beendet, wenn alle Elemente bzw. ein vorher festgelegter Anteil

der Trainingsmenge korrekt klassifiziert ist. Die Güte der Klassifizierung des Entscheidungsbaums kann anschließend mittels einer Testmenge, die nicht Teil der Trainingsmenge ist, überprüft werden. [Lust02, S. 308 - 309]
In der Weiterentwicklung C4.5 des ID3-Algorithmus wird statt dem absoluten Informationsgewinn ein um die Entropie des aktuellen Attributs normierter Informationsgewinn verwendet [Beie06, S. 117; Pete05, S. 156]. Für weiterführende Literatur zu den von J. R. Quinlan entwickelten ID3- und C4.5-Algorithmen siehe [Quin83] und [Quin93].
Die Fehlerrate von Entscheidungsbäumen nimmt bei steigender Knotenzahl monoton ab. Allerdings besteht bei steigender Knotenzahl die Gefahr einer zu starken Spezialisierung des Baums und damit einer Abnahme der Generalisierungsfähigkeit, was auch als *Overfitting* bezeichnet wird. [Pete05, S. 149]
Aus diesem Grund werden beim *Pruning*-Prozess Knoten und Blätter des Entscheidungsbaums abgeschnitten, wobei je nach Algorithmus das *Pruning* während oder nach der Baumgenerierung erfolgen kann. Für die Beschreibung der einzelnen *Pruning*-Methoden wie beispielsweise *Cost-Complexity Pruning*, *Reduced Error Pruning* oder *Error-Complexity Pruning* siehe [Pete05, S. 148 - 152].
Entscheidungsbäume besitzen den Vorteil, dass sie sehr einfach und verständlich präsentiert werden können. Allerdings kann durch zufällige Elemente das Herauskristallisieren von exakten Regeln erschwert werden, was den Entscheidungsbaum in einem solchen Fall schnell unübersichtlich, komplex und übermodelliert werden lässt. Um dies zu verhindern sollte der Baum auf eine bestimmte Tiefe und eine maximale Anzahl von Verzweigungen an den Knoten begrenzt werden. [Krah98, S. 74]

3.2 Diskriminanzanalyse

Die Diskriminanzanalyse wurde das erste Mal von R. A. Fisher im Jahr 1936 vorgestellt bzw. verwendet [Bahr03, S. 318]. Sie gehört zur Familie der Strukturen-prüfenden multivariaten Analysemethoden. Diese Methoden werden im Allgemeinen zur Durchführung von Kausalitätsanalysen verwendet um in Erfahrung zu bringen, ob und wie stark Variablen voneinander abhängen. [Back06, S. 8 - 10]
Die Diskriminanzanalyse im Besonderen ist ein Verfahren, das zwei generelle Aufgaben erfüllt. Zum einen werden anhand metrischer Attribute zwei oder mehrere Gruppen gleichartiger Objekte voneinander getrennt bzw. eine vorhandene Trennung überprüft (Diskrimination). Zum anderen werden neue Objekte in eine der bereits vorhandenen Gruppen eingeteilt (Klassifikation). [Back06, S. 159; Bahr03, S. 316; Ecke02, S. 289].
Im Gegensatz zur Clusteranalyse, bei der die Gruppen durch Ausführung des Verfahrens erst gebildet werden, wird bei der Diskriminanzanalyse von bereits bestehenden Gruppen

ausgegangen [Back06, S. 157; Bahr03, S. 318] und zählt somit ebenfalls zu den überwachten Lernverfahren [Bolt02, S. 3; Jain00, S. 1; Krah98, S. 75 - 78].
Bei der Diskriminanzanalyse wird dabei in folgenden Schritten vorgegangen, siehe [Bahr03, S. 318 - 324], ausgehend davon, dass zwei voneinander zu trennende, gleich große Gruppen vorliegen:

Schritt 1: Festlegen des Grundproblems d.h. welche Art von Gruppen sollen voneinander getrennt werden und welche Attribute stehen zur Verfügung. Ein klassisches Beispiel für den Einsatz von Diskriminanzanalyse ist die Prüfung der Kreditwürdigkeit in die Risikoklassen „hoch“ und „niedrig“ [Back06, S. 158].

Schritt 2: Festlegen der metrischen Objektvariablen bzw. -attribute, die Einfluss auf die Trennungsgüte der Gruppen vorweisen. Dies ist von Anwendungsfall zu Anwendungsfall verschieden, z.B. Einkommens- und Spareinlagenhöhe bei der Prüfung der Kreditwürdigkeit.

Schritt 3: Berechnen der Diskriminanzfunktion, welche die Gruppen optimal voneinander trennt. Diese Funktion wird beim zwei Gruppen - zwei Variablen-Fall auch als Diskriminanzachse oder Trennachse zwischen den Gruppen bezeichnet. Im mehr Gruppen-Fall existieren mehrere Trennachsen. In Abbildung 9 ist diese Trennachse graphisch dargestellt.

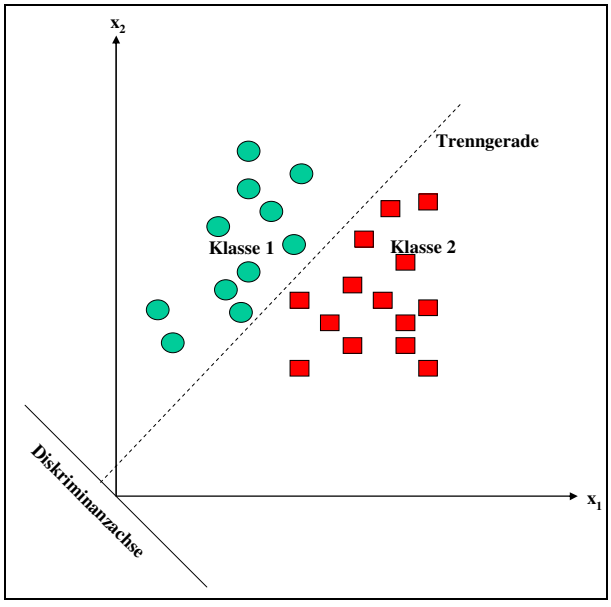

Abbildung 9: Diskriminanzachse und Trenngerade in Anlehnung an [Bahr03, S. 320]

Die Diskriminanzfunktion bildet eine Linearkombination der relevanten Attribute und weist für den zwei Gruppen - zwei Variablen-Fall folgende Form auf:

$y = v_1 * x_1 + v_2 * x_2$ [Bahr03, S. 319; Back06, S. 161]

Hierbei ergeben y den Diskriminanzwert und x_1 bzw. x_2 die einzusetzenden relevanten Attributwerte eines zu klassifizierenden Objekts. Die optimalen Koeffizienten v_1 und v_2 der Diskriminanzfunktion werden durch die Maximierung des Diskriminanzkriteriums nach folgender Formel berechnet:

$\Gamma = d^2/s^2$ [Bahr03, S. 323]

Hierbei stellen d^2 den quadrierten Abstand der Gruppenmittelwerte zueinander (= Streuung zwischen den Gruppen) und s^2 die quadrierten Abstände der Gruppenelemente zu ihrem jeweiligen Gruppenmittelpunkt (= Streuung innerhalb der Gruppen) dar. In diesem Zusammenhang sollten d^2 möglichst groß und s^2 möglichst klein sein um eine bessere Trennqualität zu erreichen. [Bahr03, S. 323; Back06, S. 165]
Zur Ermittlung der Diskriminanzkoeffizienten muss das Diskriminanzkriterium Γ nach v_1 und v_2 partiell abgeleitet werden. Dabei wird in folgenden Schritten vorgegangen, siehe [Bahr03, S. 318 - 324]:

Schritt 1: Errechnen der Mittelwerte $\tilde{Y}$ pro Variable y für jede Gruppe d.

Schritt 2: Berechnen der Abweichungen der Mittelwerte der Gruppen d für jede Variable y. Im zwei Gruppen - zwei Variablen-Fall werden hierfür folgende Formeln angewandt:

$d_1 = \tilde{Y}_{1A} - \tilde{Y}_{1B}$
$d_2 = \tilde{Y}_{2A} - \tilde{Y}_{2B}$ [Bahr03, S. 323]

Schritt 3: Berechnen der Summe der quadratischen Abweichungen eines jeden einzelnen Attributwerts zum Mittelwert der Gruppe, der das Objekt zu Beginn zugeordnet ist. Anschließend erfolgt die Bildung der Summe s über die Gruppen für jede einzelne Variable.

Im zwei Gruppen - zwei Variablen-Fall werden zu diesem Zweck folgende Formeln angewandt:

$s_1 = s_{11A} + s_{11B}$
$s_2 = s_{22A} + s_{22B}$ [Bahr03, S. 323]

Schritt 4: Berechnen der Kreuzproduktsumme s_{12} über alle Attribute eines Objekts mit den Abweichungen eines jeden einzelnen Attributwerts vom Mittelwert der Gruppe, der das Objekt zu Beginn zugeordnet ist. Anschließend erfolgt das Bilden der Summe über die Gruppen für jede einzelne Variable. Im zwei Gruppen – zwei Variablen-Fall werden die Kreuzprodukte über die zwei Attributwerte eines Objekts berechnet. Für das Ermitteln der Kreuzproduktsumme s_{12} wird folgende Formel angewandt:

$s_{12} = s_{12A} + s_{12B}$ [Bahr03, S. 323]

Schritt 5: Berechnen der Koeffizienten v der Diskriminanzfunktion durch Bilden und Errechnen eines linearen Gleichungssystems der folgenden Form:

$d_1 = v_1 * s_{11} + v_2 * s_{12}$
$d_2 = v_1 * s_{12} + v_2 * s_{22}$ [Bahr03, S. 323]

Ein bekanntes Verfahren zum Berechnen linearer Gleichungssysteme nach zwei oder mehreren Unbekannten (hier v_1 und v_2) ist das Gaußsche Eliminationsverfahren, siehe dazu [Köni99, S. 377 - 378].

Schritt 4: Mit Hilfe der ermittelten Diskriminanzfunktion erfolgt das Berechnen des kritischen Diskriminanzwerts y_T durch Einsetzen der Attributwerte der vorhandenen Objekte in die Diskriminanzfunktion. Der kritische Diskriminanzwert y_T bildet das arithmetische Mittel der durchschnittlichen Diskriminanzwerte $\tilde{Y}$ der Trainingsdaten der beiden Gruppen und berechnet sich mit folgender Formel:

$y_T = (\tilde{Y}_A + \tilde{Y}_B) / 2$ [Bahr03, S. 327]

Bei dieser Berechnung wird – wie oben erwähnt – davon ausgegangen, dass die beiden Gruppen die gleiche Anzahl an Elementen besitzen. Ist dies nicht der Fall, müssen die Mittelwerte durchschnittlichen Diskriminanzwerte $\tilde{Y}$ mit der Anzahl der Elemente n der Gruppen gewichtet werden, was folgende Formel ergibt:

$$y_T = (n_A * \tilde{Y}_A + n_B * \tilde{Y}_B) / (n_A + n_B)$$ [Bahr03, S. 327]

Auf Basis des kritischen Diskriminanzwerts erfolgt die Klassifikation von Objekten in eine bestimmte Gruppe. Im zwei Gruppen-Fall wird das Objekt der Gruppe mit den kleineren Diskriminanzwerten zugeordnet, falls der Diskriminanzwert des Objekts kleiner als der kritische Diskriminanzwert ist. Analog wird mit den Objekten verfahren, deren Diskriminanzwert größer als der kritische Diskriminanzwert ist. Diese werden der Gruppe mit den größeren Diskriminanzwerten zugeteilt. Auf diese Weise werden bei Banken beispielsweise Kreditantragssteller in die beiden Gruppen Kreditwürdig und Nicht-Kreditwürdig eingeteilt. [Bahr03, S. 328]
In [Erb90, S. 19 - 26] ist ein Beispiel aus der Geographie angeführt. Es beschreibt die Klassifizierung von Gemeinden in urbanen Raum und suburbanen Raum mit Hilfe der Attribute „Veränderung der Bodenpreise“ und „Durchschnittliche Gehaltssumme“. Dieses Beispiel repräsentiert den zwei Gruppen – zwei Variablen-Fall. Für den zwei Gruppen – n Variablen-Fall (n>2) werden die Kreuzprodukte und das Gleichungssystem aus Schritt 5 linear für die weiteren Variablen erweitert, siehe dazu [Ecke02, S. 292 - 307]. Da im Rahmen dieser Arbeit die Diskriminanzanalyse nur mit zwei Gruppen durchgeführt wird (Betrugsverdächtig und Nicht-Betrugsverdächtig) sei für die Beschreibung des rechenintensiveren mehr Gruppen-Falls auf [Back06, S. 177 – 188] und [Bahr03, S. 329 - 340] verwiesen.

3.3 Neuronale Netzwerke

Neuronale Netzwerke (auch künstliche neuronale Netzwerke genannt) bilden in ihren verschiedensten Ausprägungen Algorithmen mit dem allgemeinen Ziel, die Informationsverarbeitung der neuronalen Struktur des menschlichen Gehirns nachzubilden. Den ersten Artikel in dieser Richtung veröffentlichten W. McCulloch und W. Pitts im Jahr 1943 mit ihrer Arbeit über neurologische Netzwerke, die auf dem McCulloch-Pitts Neuron basierten, siehe [Mccu43]. Biologische neuronale Netze speichern Informationen an den Übergängen von einem Neuron zu einem anderen. Abbildung 10 bildet ein Neuron ab.

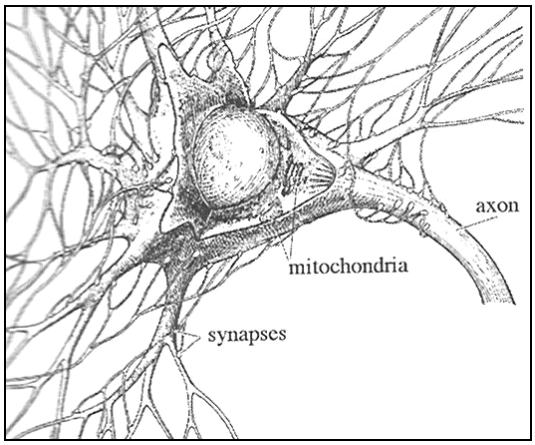

Abbildung 10: Neuron des menschlichen Gehirns in [Stev88, S. 5]

Ein Neuron empfängt und produziert Signale. Es besteht aus einem Zellkörper, der mittels der Mitochondrien mit Energie versorgt wird. Die Übermittlung der ausgehenden Signale erfolgt durch das Axon, eingehende Signale werden durch die Dentriten empfangen. An den Übergängen zwischen Axon und Dentriten befinden sich die Synapsen, die bei entsprechendem Aktionspotenzial Signale an die Nachbarzellen weiterleiten [Lämm08, S. 196 - 197; Zöll07, S. 102; Haun98, S. 4 - 7]. Für weiterführende Literatur zu diesem Thema siehe [Thom01, S. 29 - 94].
Wie eingangs zu diesem Abschnitt erwähnt, handelt es sich bei künstlichen neuronalen Netzen um ein Verfahren mit dem Ziel, die natürlichen neuronalen Netze des menschlichen Gehirns technisch nachzubilden um damit auftretende Muster erkennen zu können. Dabei haben sich in der Forschung verschiedene Ausprägungen von neuronalen Netzen entwickelt, die nachfolgend beschrieben werden.

3.3.1 Arten von neuronalen Netzwerken

In diesem Abschnitt werden die wichtigsten Varianten der neuronalen Netzwerke vorgestellt. Welche Art von neuronalem Netzwerk und welcher Lernalgorithmus für die Problemstellung in dieser Arbeit verwendet wird und warum, wird im Abschnitt 6.2 diskutiert.
Die am häufigsten angewandte Form der neuronalen Netzwerke sind die Assoziationsnetzwerke (auch *Feedforward*-Netzwerke genannt) [Dorf91, S. 38]. Assoziationsnetzwerke bestehen aus Knoten, die direkt miteinander verbunden sind. Ein Knoten wird in einem neuronalen Netzwerk durch die Ebene bzw. Schicht, in der er sich befindet, definiert. Bei den Knotenarten wird zwischen Eingabeknoten (Inputschicht), versteckten Knoten (Hiddenschicht) und Ausgabeknoten (Outputschicht) unterschieden. Assoziationsnetzwerke bestehen mindestens aus Input- und Outputschicht. [Dorf91, S. 37]

Eingabeknoten beziehen ihre Daten von der Außenwelt, wohingegen Ausgabeknoten die Ergebnisse des neuronalen Netzwerks an die Außenwelt weiter- bzw. zurückgeben. Zwischen Eingabe- und Ausgabeknotenabschnitt können Schichten mit versteckten Knoten integriert werden, die keine Verbindung zur Außenwelt besitzen, weswegen die neuronalen Netzwerke oft als *„Black Box“* angesehen werden. [Roja96, S. 29; Lust02, S. 325 - 326; Rey08, S. 17 - 18]

Neuronale Netzwerke mit einer oder mehreren Hiddenschichten werden auch als *Multi Layer Perceptron* bezeichnet. Die Knoten der verschiedenen Schichten sind in den *Multi Layer Perceptrons* durch Kanten verbunden, wobei jede Kante einen Gewichtswert repräsentiert. Die Gewichtswerte stellen in ihrer Gesamtheit das Wissen eines neuronalen Netzwerks dar. [Dorf91, S. 39]

Abbildung 11 zeigt die grundsätzliche Topologie eines *Multi Layer Perceptrons*.

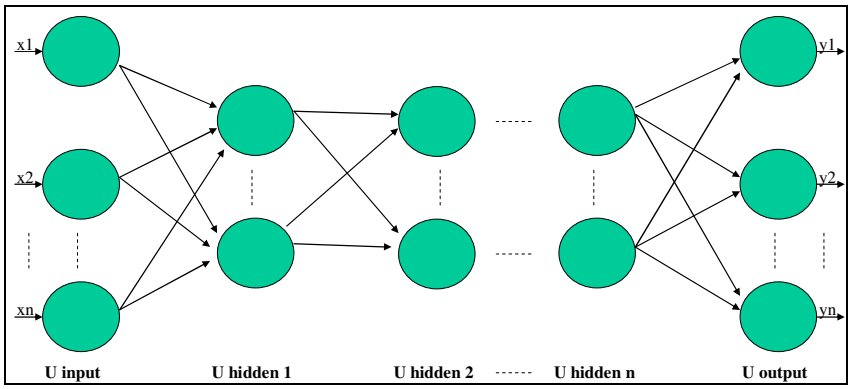

Abbildung 11: Allgemeine Topologie eines Multi Layer Perceptrons in Anlehnung an [Borg03, S. 40]

Um aus einer Kombination von Inputdaten bzw. Inputwerten an den Eingabeknoten zu einem Ergebnis an den Ausgabeknoten zu gelangen, müssen die einzelnen Knoten „aktiviert“ werden. Dies geschieht mit Hilfe von Outputwerten der Vorgängerknoten und den Gewichten an den Kanten der Knotenübergänge von den Vorgängerknoten. Der Outputwert eines Knotens wird folgendermaßen berechnet, siehe [Dorf91, S. 15 - 17; Lust02, S. 336 - 339; Rey08, S. 20 - 21]:

Schritt 1: Multiplizieren der Outputwerte der Vorgängerknoten mit den speziellen Gewichten zwischen Zielknoten und dessen Vorgängerknoten ($o_i * w_i$).

Schritt 2: Bilden der Summe aus den in Schritt 1 ermittelten Produkten. Diese Summe wird als Nettoinput eines Knotens bezeichnet ($net_n = \sum o_i * w_i$).

Schritt 3: Berechnen des Aktivierungswerts des Knotens durch Einsetzen des berechneten Nettoinputs in die Aktivierungsfunktion ($f(net_n)$).

Schritt 4: Berechnen des Outputwerts des Knotens durch Einsetzen des Aktivierungswerts eines Knotens in die Outputfunktion des Knotens ($g(f)$). Oftmals werden Output- und Aktivierungsfunktion zu einer Funktion zusammengefasst, z.B. wenn nur der Outputwert in Bezug auf die Problemstellung als interessant angesehen wird [Dorf91, S. 97].

In Abbildung 12 sind die Schritte zur Ermittlung des Outputwerts eines Knotens graphisch zusammengefasst.

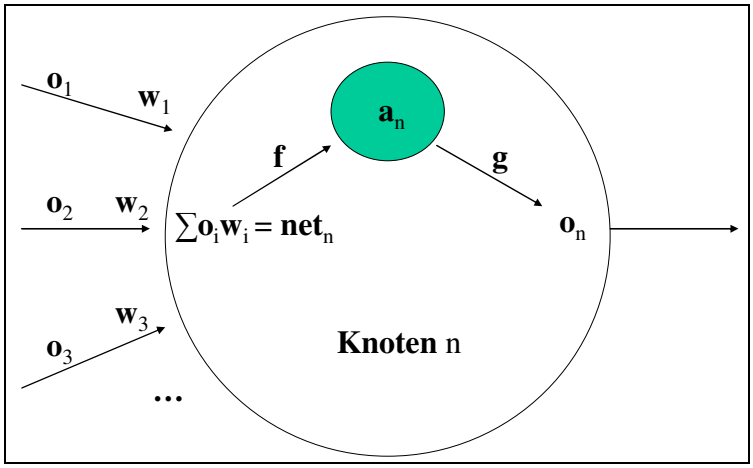

Abbildung 12: Ablauf einer Knotenaktivierung in Anlehnung an [Dorf91, S. 17]

Der ermittelte Outputwert eines Knotens wird wiederum für die Berechnung der Aktivierungswerte der Nachfolgerknoten verwendet. Wenn der Outputwert ungleich 0 ist, wird dies auch als „feuern“ bezeichnet. [Dorf91, S. 17]
Es existieren verschiedene Output- und Aktivierungsfunktionen für unterschiedliche Zwecke. Welche Funktion für die Problemstellung in dieser Arbeit am besten geeignet ist, wird im Abschnitt 6.2 diskutiert. Generell werden Output- und Aktivierungsfunktionen in die folgenden vier Funktionstypen unterteilt, siehe [Lämm08, S. 198 - 199; Roja96, S. 150]:

a) Schwellwertfunktion: Die Schwellwertfunktion gibt in Abhängigkeit vom Nettoinput entweder 0 oder 1 als Outputwert weiter, aber keinen Wert, der innerhalb dieses Intervalls liegt.

b) Identitätsfunktion: Die Identitätsfunktion gibt den Inputwert direkt und unverändert als Outputwert weiter.

c) Lineare Schwellenfunktion: Die lineare Schwellenfunktion erzeugt einen Outputwert in linearer Abhängigkeit vom Eingabewert.

d) Sigmoidfunktion: Die Sigmoidfunktion ist nichtlinear und durch folgende Formel gekennzeichnet:

$s_c(x) = 1 / 1 + e^{-cx}$ [Roja96, S. 150; Lämm08, S. 199]

In Abbildung 13 sind der Verlauf der Schwellwertfunktion (Links), der linearen Schwellenfunktion (Mitte) und der Sigmoidfunktion für den Wert $c = 1$ (Rechts) abgebildet.

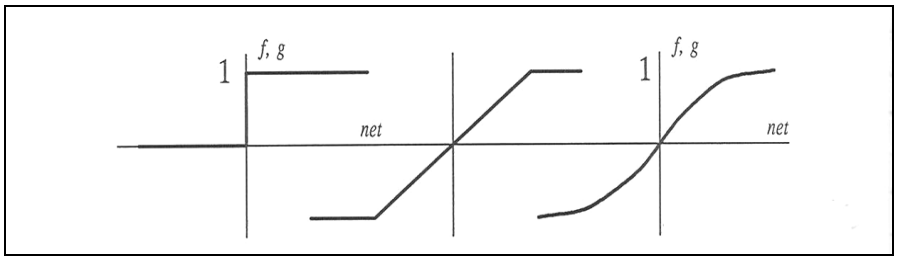

Abbildung 13: Verlauf der Outputfunktionen eines Knotens in [Dorf91, S. 18]

Die oben aufgezählten Funktionen bilden die klassischen Outputfunktionen in Bezug auf neuronale Netzwerke. Für weitere Funktionstypen siehe [Borg03, S. 41; Rey08, S. 22 - 24].

Eine besondere Form der Knoten sind die Bias-Knoten. Sie zeichnen sich durch die Eigenschaft aus, dass sie keine Vorgängerknoten besitzen und immer mit dem Wert 1 aktiviert werden. Das bedeutet, dass das Gewicht zwischen einem Bias-Knoten und seinem Nachfolgerknoten immer direkt als Summand in den Nettoinput dieses Nachfolgerknotens eingeht. Dies ist beispielsweise notwendig, wenn sichergestellt werden soll, dass ein bestimmter Knoten immer feuert, also ein bestimmter Schwellwert einer Schwellwertfunktion immer überschritten sein soll. [Roja96, S. 63; Rey08, S. 29]

Werden die vier oben beschriebenen Schritte für alle Knoten eines *Multi Layer Perceptrons* ausgeführt, entstehen aus den Eingabewerten der Inputschicht bestimmte Ausgabewerte der Outputschicht, anhand derer die Klassifizierung der Eingabewerte vorgenommen wird. In [Dorf91, S. 21 - 23; Lämm08, S. 222 - 224] ist die Vorwärtsberechnung zur Lösung des XOR-Problems auf Basis einer Schwellwertfunktion als Beispiel durchgerechnet. Die Ausbreitungsrichtung verläuft dabei von der Inputschicht über die Hiddenschicht(en) zur Outputschicht. Diese Vorwärtsaktivierungen, bei der jeder Knoten einer Schicht eine Verbindung mit jedem Knoten seiner Nachfolgeschicht aufweist, wird als *Feedforward*-Vollverbindung [Dorf91, S. 36] bezeichnet. *Feedforward*-Vollverbindungen

werden für die Klassifikation von numerischen Mustern (z.B. XOR-Problem) eingesetzt. [Dorf91, S. 36 - 42]
Feedforward-Netzwerke werden für eine breite Palette von Anwendungen eingesetzt wie z.B. Klassifizierungen, Mustererkennungen, Prognosen und Maschinensteuerungen [Lämm08, S. 227 - 228].
Bei *Recurrent*-Netzwerken dagegen sind - anders als bei der strengen Vorwärtsaktivierung - Rückkopplungen zwischen den Knoten einer Schicht möglich [Roja96, S. 42 - 47; Rey08, S. 67]. Damit können zeitlich codierte Informationen (zeitabhängige Muster) in den Inputdaten entdeckt werden bzw. Muster, die eine zeitliche Verzögerung aufweisen können [Köni99, S. 698 - 699; Rey08, S. 67]. Dabei existieren drei Arten von Rückkopplungen, die nachfolgend aufgelistet sind, siehe [Rey08, S. 68]:

a) Direkte Rückkopplungen (engl.: *Direct Feedback)*: Bei direkten Rückkopplungen dient der Outputwert eines Knotens wieder als Inputwert für sich selbst.

b) Indirekte Rückkopplungen (engl.: *Indirect Feedback)*: Bei indirekten Rückkopplungen dient der Outputwert eines Knotens als Inputwert eines Knotens der Vorgängerschicht.

c) Seitliche Rückkopplungen (engl.: *Lateral Feedback)*: Bei seitlichen Rückkopplungen dient der Outputwert eines Knotens als Inputwert eines Knotens der gleichen Schicht.

Recurrent-Netzwerke werden häufig zum Treffen von Vorhersagen und der Simulation von menschlichen Verhaltensweisen eingesetzt [Rey08, S. 67].
Ein weiterer Typ der neuronalen Netzwerke sind die Kohonen-Netzwerke. Sie besitzen keine Hiddenschicht und bestehen aus einer Inputschicht und einer zweidimensionalen Outputschicht. Ihr Lernalgorithmus (*Kohonen Learning*) gehört zur Familie des unüberwachten Lernens, d.h. der Outputwert der Trainingsmenge steht nicht fest, was im folgenden Unterabschnitt diskutiert wird. Diese Art der neuronalen Netzwerke wurde erstmals von T. Kohonen erwähnt und basiert auf einem selbstorganisierenden Prozess, siehe dazu [Koho82]. Das Modell ist der menschlichen Fähigkeit, Muster bzw. Strukturen von selbst (d.h. ohne externen Lehrer) zu erlernen, sehr ähnlich [Roja96, S. 390 - 392]. Jedes Neuron der Inputschicht ist mit jedem Neuron der Outputschicht verbunden. Dabei werden nur die Neuronen der Inputschicht aktiviert, welche die größten Ähnlichkeiten zum Inputvektor bzw. Inputmuster aufweisen (Wettbewerb zwischen den Neuronen). Während des Trainingsprozesses wird die topographische Abbildung des Netzes verändert, wodurch sich sog. Karten ergeben. [Lämm08, S. 260 - 264; Rey08, S. 78 - 79; Pete05, S. 73 - 76]

Kohonen-Netzwerke werden häufig für Spracherkennung oder Anwendungen der Kinematik (z.B. mechanische Roboterarme) eingesetzt [Rey08, S. 89 - 90]. Im folgenden Unterabschnitt wird u.a. auf das Trainieren von Kohonen-Netzwerken eingegangen.

3.3.2 Arten von Lernalgorithmen für neuronale Netzwerke

Lernende Systeme oder Programme sind definiert als Systeme, die in der Lage sind, neues Wissen aufzunehmen, es sich zu merken und daraus Verhalten zu adaptieren und Feedback zu geben [Dorf91, S. 248; Rey08, S. 100 - 101].
Die wichtigste Eigenschaft der neuronalen Netzwerke ist in diesem Zusammenhang ihre Generalisierungsfähigkeit, d.h. aus bekannten Mustern neue Strukturen zu erkennen und zu lernen und somit auch nichtlineare Zusammenhänge zu identifizieren [Dorf91, S. 37 - 38; Rey08, S. 101].
Zu diesem Zweck muss das Netzwerk trainiert werden, wofür es – wie zu Beginn von Kapitel 3 erwähnt – zwei verschiedene Typen von Lernmethoden gibt. Das sind zum einen die überwachten Lernverfahren (engl.: *supervised*) und zum anderen die unüberwachten Lernalgorithmen (engl.: *unsupervised*). Der Unterschied zwischen den beiden Varianten ist, dass bei überwachten Lernverfahren der Outputwert der Trainingsmenge bzw. des jeweiligen Trainingsmusters (explizites Feedback) bereits im Vorfeld feststeht [Reol07, S. 2; Jain00, S. 1; Phua05, S. 5]. Bei unüberwachtem Lernen herrschen dagegen nur Restriktionen vor, die durch die Architektur vorgegeben sind [Dorf91, S. 30; Phua05, S. 8]. Im Forschungsgebiet der neuronalen Netzwerke werden beide Lernverfahren mit spezifischen Lernalgorithmen verwendet und in diesem Abschnitt beschrieben, siehe dazu [Krah89, S. 61 - 84].
Eine einfache überwachte Lernregel stellt die Delta-Regel dar. Diese Lernregel ist in ihrer ursprünglichen Form nicht für Hiddenschichten anwendbar, sondern lediglich für neuronale Netzwerke, die aus einer Inputschicht und einer Outputschicht bestehen. Bei dieser Lernregel wird der berechnete Outputwert eines Knotens mit dem gewünschten Outputwert verglichen. Auf Basis dessen wird mit folgender Formel die Anpassung der Gewichte zwischen den Knoten ermittelt, siehe [Dorf91, S. 33 - 34; Rey08, S. 40 - 42]:

$$\Delta w(jk) = \gamma * x(j) * (y'(k) - y(k))$$

$\Delta w(jk)$ = Anpassungswert der Gewichte zwischen dem Vorgängerknoten j und dem Nachfolgerknoten k
$y'(k)$ = Gewünschter Outputwert des Nachfolgerknotens k
$y(k)$ = Tatsächlicher Outputwert des Nachfolgerknotens k
$x(j)$ = Outputwert des Vorgängerknotens j
γ = Vordefinierte Lernkonstante für das gesamte neuronale Netzwerk ($0<\gamma<1$)

Eine erweiterte Form der Delta-Regel nach [Rume86, S. 5 - 11] ergibt das häufig eingesetzte Backpropagationsverfahren, mit dem zusätzlich die Gewichte der Hiddenschicht(en) verändert werden können [Dorf91, S. 34 - 35].
Der Name *Backpropagation* besagt, dass entgegengesetzt zur Vorwärtsaktivierung der Knoten die Gewichte in Rückwärtsrichtung während eines Lernschritts angepasst werden. Dieser Algorithmus basiert auf dem Gradientenabstiegsverfahren (engl.: *Gradient Descent*) [Roja96, S. 149; Lämm08, S. 218], das bei Minimierungsproblemen eingesetzt wird um lokale und globale Minima von Funktionen zu finden. Für weiterführende Literatur zum Gradientenabstiegverfahren siehe [Borg03, S. 55 - 71].
In Anwendung auf neuronale Netzwerke soll bei diesem Verfahren die Fehlerfunktion der Outputwerte minimiert werden. Diese Funktion errechnet den Fehler der Outputwerte für eine bestimmte übergebene Kombination von Inputwerten. Der Gradient gibt die Richtung des stärksten Abstiegs der Fehlerfunktion an. Die Höhe der Gewichtsanpassung wird anschließend durch die Lernkonstante (oder auch Lernfaktor genannt) beeinflusst. [Köni99, S. 700 - 702; Lämm08, S. 219]
Der Algorithmus startet mit einer zufällig ausgewählten Gewichtskombination. Für jeden Trainingssatz an Inputwerten ergibt sich für jede Gewichtskombination ein bestimmter Outputwert des neuronalen Netzwerks. Da bei überwachten Lernverfahren der Sollwert des Netzwerkoutputs für einen bestimmten Trainingssatz bereits feststeht, ist das Ziel des Verfahrens, die Abweichung zwischen Ist- und Solloutputwert zu minimieren. Beim Durchlauf der Rückwärtspropagation werden die Gewichte zwischen den Knoten in Abhängigkeit von der Größe der Lernkonstante γ (= $0<\gamma<1$) verändert bzw. angepasst [Roja96, S. 86; Lämm08, S. 219]. Dieser Vorgang wird bei jedem Lernzyklus durchgeführt bis eine vorgegebene Menge der Lernzyklen abgelaufen oder sich der Wert der Fehlerfunktion nicht mehr verringert und somit ein Minimum gefunden ist. Ein Lernzyklus des Backpropagationsverfahrens läuft in folgenden Schritten ab, siehe [Roja96, S. 153 - 167; Lämm08, S. 218 - 222]:

Schritt 1 – Durchführen des *Feedforward*-Prozesses:
Ausführen des Vorwärtsprozesses durch die Verwendung einer Kombination von Trainingsdaten als Inputwerte um die Outputwerte der Outputschichtknoten des neuronalen Netzwerks zu berechnen (Reizung des Netzwerks).

Schritt 2 – Rückwärtspropagierung für die Outputschichtknoten:
Vergleich der Outputwerte der Outputschichtknoten (Istwerte) mit den vordefinierten Outputwerten (Sollwerte) für die übergebenen Trainingsinputwerte und Berechnen der Fehlerwerte der Outputschichtknoten durch Verwendung der folgenden Formel:

$$\delta(k) = o(k) * (1 - o(k)) * (o(k) - t(k))$$

$\delta(k)$ = Fehlerwert des Outputschichtknotens k
$o(k)$ = Outputwert (Istwert) des Outputschichtknotens k
$t(k)$ = Vordefinierter Outputwert (Sollwert) des Outputschichtknotens k

Schritt 3 – Rückwärtspropagierung der Hiddenschichtknoten:
Berechnen des Fehlerwerts für die Hiddenschichtknoten auf Grundlage der bestehenden Gewichte zur Nachfolgerschicht mithilfe der folgenden Formel:

$$\delta(j) = o(j) * (1 - o(j)) * \sum(w(jk) * \delta(k))$$

$\delta(j)$ = Fehler des Hiddenschichtknotens j
$o(j)$ = Outputwert (Istwert) des Hiddenschichtknotens j
$w(jk)$ = Gewicht zwischen dem Hiddenschichtknoten j zum Nachfolgerknoten k
$\delta(k)$ = Fehler des Nachfolgerknotens k

Schritt 4 – Anpassung der Gewichtswerte zwischen dem aktuellen Knoten und den Knoten der Nachfolgerschicht:
Berechnen des Anpassungswerts (Delta-Wert) der Gewichte des Netzwerks auf Grundlage des aktuellen Netzwerkfehlers und der Lernkonstante mithilfe der folgenden Formel:

$$\Delta w(jk) = -\gamma * o(j) * \delta(k)$$

$\Delta w(jk)$ = Anpassungswert der Gewichte zwischen dem Hiddenschichtknoten j und dem Nachfolgerknoten k
$o(j)$ = Outputwert (Istwert) des Hiddenschichtknotens j
$\delta(k)$ = Fehlerwert des Nachfolgerknotens k
γ = Vordefinierte Lernkonstante für das gesamte neuronale Netzwerk ($0<\gamma<1$)

Das berechnete $\Delta w(jk)$ wird danach zu dem ursprünglichen Gewicht $w(jk)$ addiert. Anschließend werden die Schritte 3 und 4 für alle Knoten und den Gewichtsübergängen wiederholt. Diese Wiederholung erfolgt solange, bis diese Berechungen für die Gewichtsübergänge zwischen der Inputschicht und der ersten Hiddenschicht abgeschlossen und somit die Ausgangspunkte des neuronalen Netzwerks erreicht sind.

Anschließend wird der Algorithmus mit dem nächsten Trainingssatz erneut bei Schritt 1 gestartet. [Roja96, S. 153 - 167; Muna08, S. 11 - 17; Rutk08, S. 210 - 218]
Abbildung 14 fasst das Backpropagationsverfahren graphisch in den drei wesentlichen Schritten Vorwärtsaktivierung, Fehlerberechnung und Gewichtsanpassung zusammen.

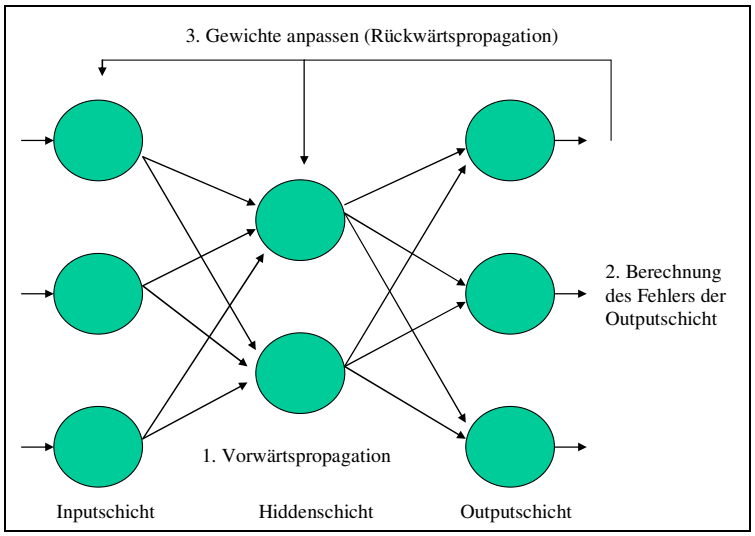

Abbildung 14: Ablauf des Backpropagationsverfahrens in Anlehnung an [Lust02, S. 369]

Falls die Anpassung der Gewichte sofort nach dem Durchlauf eines Trainingsmusters vorgenommen wird, handelt es sich um Online-Training. Im Gegensatz dazu existiert die Variante des Offline-Trainings, bei der die Anpassung der Gewichte im Schritt 4 erst nach dem abgeschlossenen Durchlauf aller Trainingsmuster erfolgt. Dafür müssen die jeweiligen Gewichtsänderungen für die Trainingssätze aufsummiert werden bis nach dem letzten Trainingssatz die Gewichtsänderung erfolgt. [Roja96, S. 167; Lämm08, S. 219]
Online-Training ist nach [Schi93, S. 15 - 24] die effektivere Variante, da sowohl die Wahrscheinlichkeit in ein lokales Minimum der Fehlerfunktion „zu fallen“ geringer ist als auch bei größer werdender Anzahl von Trainingsmustern die Performance des Lernprozesses besser ist als beim Offline-Training. Bei einer geringen Zahl von Trainingssätzen nimmt die Fehlerfunktion beim Online- und beim Offline-Training den gleichen Verlauf an [Borg03, S. 62 - 63].
Wenn alle Trainingssätze einmal durchgelaufen sind, wird ein neuer Durchlauf der Trainingsmenge begonnen. Das Ziel ist, die Mustererkennungsfähigkeit des Netzes zu verbessern und somit den Gesamtfehler des neuronalen Netzwerks zu verringern. [Roja96, S. 149].
Der Backpropagationsalgorithmus ist aufgrund der Tatsache, dass die Inputmuster und dessen Outputwerte bekannt sind, sehr gut für die Anwendung auf Assoziationsnetzwerke bzw. *Multi Layer Perceptrons* geeignet [Dorf91, S. 39].

Für zeitabhängige *Recurrent*-Netzwerke wird der Backpropagationsalgorithmus für die Gewichte der Rückkopplungen erweitert, siehe dazu [Roja96, S. 172 - 175].
Allgemein wird die Erkennungsgenauigkeit eines *Multi Layer Perceptrons* höher, je größer die Trainingsmenge ist [Roja96, S. 143]. Wenn die Anzahl der Lerndurchläufe zu hoch ist, kann der Fall eintreten, dass das Netzwerk übertrainiert (engl.: *overfitted*) wird. Das bedeutet, das Netzwerk lernt die Trainingsmenge perfekt, ist aber nicht (mehr) in der Lage, Inputmuster korrekt zu klassifizieren, die zuvor nicht trainiert wurden. In diesem Fall oszilliert das Verfahren um das Minimum der Fehlerfunktion. Abbildung 15 verdeutlicht dies.

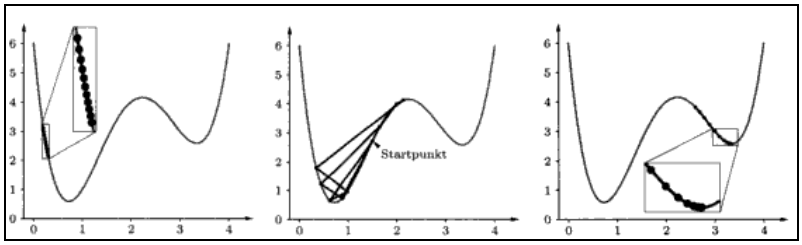

Abbildung 15: Verlauf einer Fehlerfunktion eines neuronalen Netzwerks in Anlehnung an [Borg03, S. 65 - 73]

Die drei Kurven in Abbildung 15 zeigen eine zweidimensionale Fehlerfunktion eines neuronalen Netzwerks, die sich aus dem Backpropagationsverfahren ergibt. Die X-Achse gibt hierbei das Gewicht an, die Y-Achse als Funktionswert den Fehler eines Knotens des Netzwerks bei diesem Gewicht. Anhand der Lernkonstante wird das Gewicht verändert, wobei der Gradient die Richtung des Abstiegs angibt. Im linken Bild ist ein Startwert für das Gewicht von 0,2 und einer Lernkonstante von 0,001, im mittleren Bild ist der Startwert mit 1,5 und einer Lernkonstante von 0,25 und im rechten Bild ist der Startwert mit 2,6 und einer Lernkonstante mit 0,05 festgelegt. Es ist zu erkennen, dass sich im linken Bild das Gewicht langsam dem globalen Minimum der Fehlerfunktion nähert. Im mittleren Bild oszilliert der Gradient um das globale Minimum, es ändert sich lediglich das Vorzeichen der Gewichtsanpassung. Im rechten Bild nähert sich der Gradient langsam einem lokalen Minimum, das globale Minimum wird aber nie gefunden. [Borg03, S. 63 - 65]
Zur Vermeidung des Oszillierens des Gradienten sollte die Lernkonstante möglichst klein gewählt werden. Dies führt aber zum dem Nachteil, dass mehr Lernschritte bzw. Backpropagationsdurchläufe für das Training notwendig sind, wodurch sich wiederum die Performance des Verfahrens verringert. [Rey08, S. 49 - 50]
Hierbei kann neben der kleinen Lernkonstante ebenfalls der Momentumterm verwendet werden. Der Momentumterm α kann als zusätzlicher Faktor den Anpassungswert der Gewichte $\Delta w(jk)$ verkleinern, so dass das ursprüngliche Gewicht nicht mit 100% des Anpassungswerts verändert wird. [Rey08, S. 50 - 51; Rutk08, S. 218; Lämm08, S. 225]

Um das Verfehlen des globalen Minimums aufgrund eines lokalen Minimums der Fehlerfunktion zu vermeiden und die Güte der Netzwerktopologie besser bewerten zu können, sollte das Verfahren mit mehreren Startgewichten initialisiert werden [Rey08, S. 49; Pete05, S. 231].
In [Lang03, S. 48] ist exemplarisch ein Verlauf einer Fehlerentwicklung eines neuronalen Netzwerks in Abhängigkeit von der Anzahl der Backpropagationsdurchläufe beschrieben. Abbildung 16 zeigt diesen Verlauf.

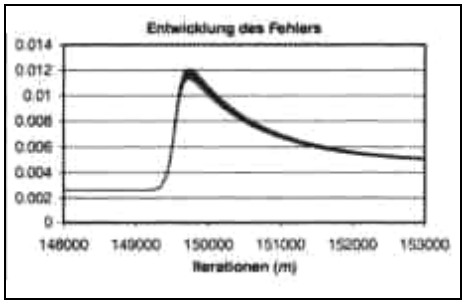

Abbildung 16: Entwicklung des Fehlers eines neuronalen Netzwerks in [Lang03, S. 48]

Es ist zu erkennen, dass diese Funktion bei ca. 150.000 Durchläufen zu oszillieren beginnt, sich aber der Fehler im weiteren Verlauf monoton fallend dem globalen Minimum nähert [Lang03, S. 48].
Im Gegensatz zum Backpropagationsalgorithmus und zur Delta-Regel gehört das *Kohonen Learning* zur Familie der unüberwachten Lernverfahren, da der korrekte Outputwert im Vorfeld nicht *a priori* (deut.: vom Früheren her) feststeht [Roja96, S. 389]. Diese Methode dient – wie im Unterabschnitt 3.3.1 erwähnt – als Lernalgorithmus für die gleichnamigen Kohonen-Netzwerke. Der Algorithmus ist durch folgenden Ablauf gekennzeichnet, siehe [Roja96, S. 393 - 394; Rey08, S. 80; Lämm08, S. 264 - 268]:

Schritt 1: Festlegung der Werte der n-dimensionalen Gewichtsvektoren zwischen der Inputschicht und den n-Outputschichten nach dem Zufallsprinzip. Zusätzlich erfolgt die Festlegung der beiden vordefinierten Komponenten Lernkonstante η und *Neighborhood Function* $ǿ(i,k)$.

Schritt 2: Auswahl eines geeigneten Inputmusters als Trainingswerte.

Schritt 3: Ermittlung des Knotens, der die größte Ähnlichkeit mit dem Inputmuster aufweist. Das ist in diesem Fall der Knoten, der den geringsten Abstand vom Gewichtsvektor zum Inputmuster besitzt.

Schritt 4: Anpassung des Gewichts nach folgender Formel:

$$\Delta w_i = \eta * ó(i,k) * (\zeta - w_i) \text{ für } i = 1,\ldots,m$$

Δw_i = Anpassungswert der Gewichte zwischen dem Inputschichtknoten k und dem Outputschichtknoten i
w_i = Alter Gewichtswert zwischen dem Inputschichtknoten k und dem Outputschichtknoten i
$ó(i,k)$ = Funktionswert der *Neighborhood Function* für den Inputschichtknoten k und dem Gewichtsübergang zum Outputschichtknoten i
ζ = Wert des entsprechenden Inputmustervektors
m = Anzahl der Outputschichtknoten, die mit dem Inputschichtknoten k verbunden sind
η = Vordefinierte Lernkonstante für das gesamte neuronale Netzwerk

Schritt 5: Abbruch, wenn eine bestimmte Anzahl an Durchläufen erreicht ist, andernfalls Modifikation von η und $ó$ und Fortsetzung mit Schritt 1, falls der gewünschte Trainingszustand noch nicht erreicht ist.

Eine weitere Lernmethode namens *Hebbian Learning* begründet sich aus der Biologie nach der Methode des Psychologen D. O. Hebb aus dem Jahr 1949. Die Idee dieses Verfahrens ist, dass zwei Neuronen, die gleichzeitig aktiv sind, einen höheren Grad an Interaktion besitzen als zwei unkorrelierte Neuronen, für weiterführende Literatur siehe [Miln93]. Diese Lernmethode ist nicht für Hiddenschichten ausgelegt, sondern nur für Netzwerke mit einer Input- und einer Outputschicht. Die Gewichte werden nur verändert, wenn beide Knoten gleichzeitig feuern [Hebb49, S. 62]. Die Anpassung der Gewichte erfolgt in einem einzigen Schritt mit folgender Formel, siehe [Roja96, S. 314 - 315; Rey08, S. 38 - 39]:

$$\Delta w(jk) = \gamma * o(j) * o(k)$$

$\Delta w(jk)$ = Anpassungswert der Gewichte zwischen dem Hiddenschichtknoten j und dem Nachfolgerknoten k
$o(j)$ = Outputwert des Vorgängerknotens j
$o(k)$ = Outputwert des Nachfolgerknotens k
γ = Vordefinierte Lernkonstante für das gesamte neuronale Netzwerk ($0<\gamma<1$)

Die in diesem und im vorhergehenden Unterabschnitt beschriebenen Netzwerktypen und Lernverfahren sind die in der Praxis am häufigsten eingesetzten Varianten. Für weitere Typen von Netzwerken und Lernalgorithmen siehe [Roja96], [Rey08] oder [Dorf91].

3.4 Support Vector Machine

Das Verfahren *Support Vector Machine* (SVM) ist der Diskriminanzanalyse ähnlich. Diese Methode separiert ebenfalls Gruppen von Objekten, die in Form von Vektoren repräsentiert werden und kann auf Basis dessen neue Elemente klassifizieren. Dafür ermittelt der Algorithmus eine Hyperebene (*Margin*), welche die Objekte der Gruppen optimal trennt. Diese Trennung ist in Abbildung 17 für zwei Dimensionen dargestellt. Dabei wird diejenige Trenngerade als optimal bezeichnet, bei welcher der Abstand zwischen den beiden am nächsten zueinander liegenden Punkten der beiden Trainingsgruppen maximal ist und somit den geringsten Generalisierungsfehler aufweist. Diese beiden Punkte werden auch als *Support Vectors* bezeichnet. [Pal04, S. 83 - 88]

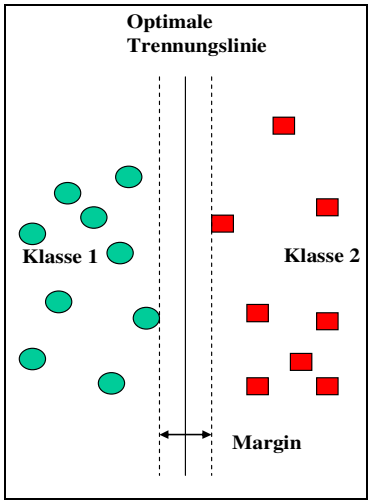

Abbildung 17: Optimale zweidimensionale Trennlinie und Margin in Anlehnung an [Wang05, S. 18]

Diese Hyperebene wird durch die jeweils vorhandene Menge von Trainingsbeispielen bestimmt:

$(x_1,y_1), (x_2,y_2), \ldots, (x_i,y_i)\ (x_n,y_n),\ x_i\ \varepsilon\ R^{P,}\ y_i\ \varepsilon\ \{+1, -1\}$ [Pal04, S. 86]

Da der Wert y die bereits definierte Gruppenklassifikation der Trainingselemente angibt, handelt es sich bei *Support Vector Machine* ebenfalls um ein überwachtes Lernverfahren [Pal04, S. 83].

Während des Trainings wird mit Hilfe der Trainingsobjekte eine Funktion entwickelt, welche die Trainingsgruppen voneinander trennt. Dabei werden die Parameter $w[w_1, w_2, \ldots, w_n]^T$ und b aus der Trainingsmenge mittels folgender Funktion verwendet:

$$d(x,w,b) = w^T * x + b = \sum_{i=1}^{n} w_i * x_i + b$$ [Wang05, S. 12; Cris00, S. 9]

Der Parameter w stellt dabei den Normalenvektor dar, der die Richtung senkrecht zur Trennlinie angibt (ein Normalenvektor ist ein Vektor, der senkrecht zu einer Geraden oder Fläche steht, siehe dazu [Henz04, S. 33 - 34]). Der Parameter b bildet als der Bias-Wert eine Konstante, welche den Abstand vom Ursprungspunkt senkrecht zur Trennlinie angibt. [Cris00, S. 10]
Ein neues Objekt x wird durch Einsetzen dieser Parameter in die oben genannte Entscheidungsfunktion $d(x,w,b)$ klassifiziert. Anschließend wird die folgende Funktion ausgeführt:

$i_F = sign(d(x,w,b)$ [Wang05, S. 12]

Die Funktion $sign()$ gibt das Vorzeichen des Werts d zusammen mit dem Wert 1 zurück, somit -1, +1 oder 0. Die Gruppenzuteilung eines neuen Objekts wird durch folgende Regel durchgeführt. Ist der Wert i_F

a) größer als 0 (d.h. +1), dann wird er der Klasse 1 zugeteilt.
b) kleiner als 0 (d.h. -1), dann wird er der Klasse 2 zugeteilt.
c) gleich 0, dann liegt er genau auf der Trennlinie und kann keiner Klasse zugeteilt werden.

[Wang05, S. 12]

Die Voraussetzung für diese Art der Klassifikation ist eine lineare Trennbarkeit der Daten. Oftmals sind in der Praxis eine solche zweidimensionale, linear trennbare Trainingsmenge und somit auch eine lineare Trennungslinie nicht gegeben. Dann können neue Objekte unter Umständen falsch klassifiziert werden. [Pal04, S. 87]
In Abbildung 18 ist diese Situation graphisch dargestellt.

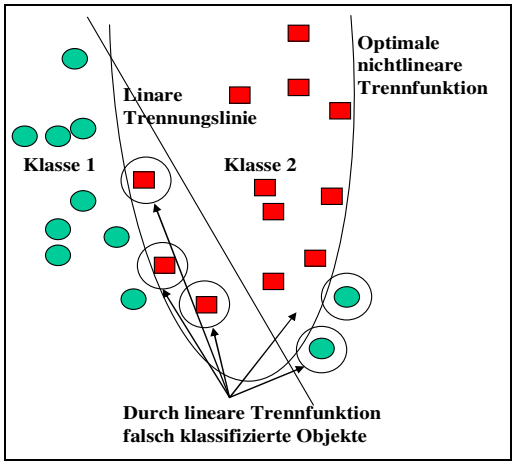

Abbildung 18: Nichtlineare Trennfunktion im Vergleich zur linearen Trennfunktion in Anlehnung an [Wang05, S. 24]

Zur Lösung dieses Problems wird eine höherdimensionale nichtlineare Trennfunktion bestimmt (im Beispiel in Abbildung 18 ein Polynom zweiten Grades). Innerhalb dieses höherdimensionalen Raums F sind die einzelnen, regional begrenzten, Trennfunktionen wieder linear. Dabei werden die zweidimensionalen Vektoren w zu höherdimensionalen Vektoren $\Phi(x)$. Zur Ermittlung der Entscheidungsfunktion wird das Skalarprodukt der Vektoren gebildet. [Wang05, S. 24 - 25]
Dieses Skalarprodukt ist in der folgenden Kernel-Funktion $k(x_i,x_j)$ dargestellt:

$$k(x_i,x_j) = \Phi^T(x_i)\Phi(x_j)$$ [Wang05, S. 26]

Die Entscheidungsfunktion $i_F(x)$ für die Klassifizierung eines neuen Objekts x nimmt die folgende Form an:

$$i_F(x) = \text{sign}(\sum_{i=1}^{n} v_i * k(x_i,x) + b)$$ [Wang05, S. 25]

Bei der Anwendung dieser Formel werden alle Elemente der Trainingsmenge n mit ihren Skalarprodukten mit dem neu zu klassifizierenden Objekt aufsummiert. Der Parameter v_i repräsentiert in diesem Zusammenhang einen Wert für die Gewichtung dieses speziellen Elements innerhalb der Trainingsmenge. Der Wert b entspricht auch in diesem Fall dem Bias-Wert. Analog zum linearen Fall erfolgt die Eingruppierung eines Objekts durch den Rückgabewert der $\text{sign}()$-Funktion. [Wang05, S. 25 - 26]
Das Bilden der Trennfunktionen, das Berechnen der Hyperebene sowie das Klassifizieren neuer Objekte werden immer komplexer und rechenintensiver je höher der Raum bzw. die Trennfunktion dimensioniert ist [Wang05, S. 26].
Für weiterführende Literatur zu *Support Vector Machine* siehe [Wang05], [Pal04] und [Cris00].

3.5 Probabilistische Netzwerke

Probabilistische Netzwerke haben die Aufgabe der Repräsentation von unsicherem Wissen in Form einer Graphenstruktur. Die graphische Modellierung ermöglicht das Berechnen von Wahrscheinlichkeiten für das Eintreten von vorher definierten Zuständen. [Beie06, S. 359 - 360]
Hierbei wird zwischen gerichteten und ungerichteten Graphen unterschieden. Gerichtete Graphen dienen zur Abbildung kausaler Beziehungen, während ungerichtete Graphen zur Repräsentation von allgemeinen Beziehungen verwendet werden. Die Bayes-Netzwerke

bilden gerichtete Graphen und werden aus diesem Grund oftmals als kausale Netzwerke bezeichnet. [Beie06, S. 376]

Ein Beispiel für ein einfaches Bayes-Netzwerk ist in Abbildung 19 wiedergegeben. Das Ereignis R steht für „es hat geregnet“, das Ereignis S bedeutet „der Rasensprenger von Herrn B ist eingeschaltet“. Das Ereignis A_r bedeutet „der Rasen von Herrn A ist nass“, dagegen heißt das Ereignis B_r „der Rasen von Herrn B ist nass“. Wenn bekannt ist, dass es geregnet hat, erhöht sich die Wahrscheinlichkeit für die Ereignisse „nasser Rasen“. Sowohl der Regen als auch der Rasensprenger beeinflussen, ob der Rasen von Herrn B nass ist. Falls der Rasen von Herrn A nicht nass ist, wohl aber der von Herrn B, erhöht sich die Wahrscheinlichkeit, dass der Rasensprenger von Herrn B eingeschaltet ist. Sind beide Rasen nass, erhöht diese Information wiederum die Wahrscheinlichkeit für Regen (Ereignis R) und stellt zusätzlich die Wahrscheinlichkeit für das Laufen des Rasensprengers von Herrn B (Ereignis S) auf ein normales Maß ein. Die Pfeile in dem Netzwerk aus Abbildung 19 geben die Richtung der Abhängigkeiten wieder. [Beie06, S. 375]

Verbundene Knoten sind direkt abhängig, während zwei Knoten, die nur über einen Zwischenknoten miteinander verbunden sind als indirekt abhängig bezeichnet werden [Beie06, S. 361].

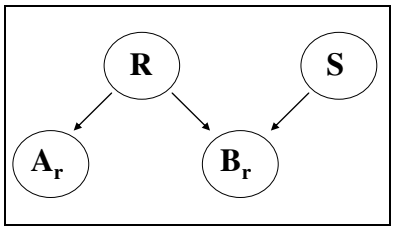

Abbildung 19: Einfaches Bayes-Netzwerk in Anlehnung an [Beie06, S. 375]

Die kausalen Vorgängerknoten werden als Elternknoten bezeichnet. Ein Bayes-Netzwerk liegt nach der Definition vor, wenn alle Variablen oder Ereignisse nur direkt von ihren Elternknoten beeinflusst werden können und somit von allen anderen Knoten bedingt unabhängig sind. [Beie06, S. 375 - 376]

Eine Menge von Variablen A, B und C sind bedingt unabhängig wenn gilt: $P(A,B|C) = P(A|C) * P(B|C)$, siehe dazu [Beie06, S. 442 - 446].

Zur Berechnung der exakten Wahrscheinlichkeiten innerhalb eines Bayes-Netzwerks werden für jeden Knoten die Wahrscheinlichkeiten der jeweiligen Elternknoten unter Unabhängigkeitsannahme multipliziert.

Dafür wird für alle Knoten die Wahrscheinlichkeit $P(V)$ unter Verwendung der Wahrscheinlichkeiten der Elternknoten $pa(V)$ mittels folgender Formel berechnet:

$P(V) = \prod P(V \mid pa(V))$ [Beie06, S. 377]

Beim Trainieren eines Bayes-Netzwerks bzw. eines probabilistischen Netzwerks werden die Wahrscheinlichkeiten der Zustände der Knoten verändert. Die Wahrscheinlichkeiten der Knoten werden auf Basis der λ-Nachrichten (Nachricht vom Kindknoten zu den Elternknoten) und der Π-Nachrichten (Nachrichten vom Elternknoten zu den Kindknoten) angepasst. Das geschieht, wenn sich der Zustand eines Knotens auf Basis eines neuen (externen) Hinweises verändert. Der genaue Ablauf des *Message Passing* bzw. des Algorithmus von Lauritzen und Spiegelhalter ist beschrieben in [Tzaf97, S. 423 - 425].
Bayes-Netzwerke werden beispielsweise in den Versionen des Betriebssystems Windows für die Hilfefunktionen z.B. bei der Behebung von Druckerproblemen eingesetzt, siehe [Heck95, S. 1 - 7; Köni99, S. 120]. Für weiterführende Literatur zu gerichteten Graphen bei Bayes-Netzwerken sei auf [Jens96] verwiesen.
Genau wie die Bayes-Netzwerke zählen die Markov-Netzwerke zur Familie der probabilistischen Netzwerke. Sie sind von der Struktur her den Bayes-Netzwerken ähnlich, wobei hier im Gegensatz zu den Bayes-Netzwerken ungerichtete Graphen repräsentiert werden. Das bedeutet, es liegen keine kausalen Beziehungen zwischen den Knoten vor, sondern allgemeine Beziehungen [Görz03, S. 329; Beie06, S. 376]. Lt. [Beie06, S. 365, S. 376] sind gerichtete Netzwerke im Allgemeinen für zahlreiche Anwendungen besser geeignet, jedoch nicht unbedingt ausdrucksstärker.
Nach der Definition ist ein Markov-Netzwerk ein minimaler Unabhängigkeitsgraph G einer Wahrscheinlichkeitsverteilung P, wenn gilt, dass G keine überflüssigen Kanten enthält und somit alle Verbindungen bedingt unabhängig sind, d.h. im Umkehrschluss ist G nach dem Entfernen einer beliebigen Kante kein Unabhängigkeitsgraph mehr. Zum Aufbau eines Markov-Netzwerks müssen somit von einem bestehenden Graphen V alle Kanten $\{A,B\}$ entfernt werden, die nicht bedingt unabhängig sind. Eine andere Möglichkeit ist, mit einem leeren Graphen zu starten und nur diejenigen Knoten zu verbinden, für welche die bedingte Unabhängigkeit gilt. [Baie06, S. 365 - 366]
Ein Markov-Netzwerk wird in der Literatur auch als *Hidden Markov Model* (HMM) bezeichnet, wobei für Markov-Netzwerke ebenfalls der Algorithmus von Lauritzen und Spiegelhalter zum Berechnen der Wahrscheinlichkeiten verwendet werden kann, siehe dazu [Baie06, S. 400 - 402].

Ein Problem der probabilistischen Netzwerke liegt lt. [Baie06, S. 360] darin, dass die Wahrscheinlichkeitsverteilungen exponentiell mit der Anzahl der Knoten bzw. Variablen wachsen und somit schnell unübersichtlich werden.

3.6 Logistische Regression

Die logistische Regression zählt genau wie die Diskriminanzanalyse zur Familie der Strukturen-prüfenden Verfahren [Back06, S. 8 - 12]. Sie dient – im Gegensatz zur Diskriminanzanalyse – zur Untersuchung der Fragestellung, mit welcher Wahrscheinlichkeit ein Ereignis eintritt und welche Attribute bzw. Variablen diese Wahrscheinlichkeit wie stark beeinflussen. Die logistische Regression besitzt gegenüber der Diskriminanzanalyse den Vorteil, dass sowohl metrische als auch nichtmetrische Attribute (siehe Abschnitt 3.1) zu Analysezwecken verwendet werden können [Back06, S. 8 - 12; Sart06, S. 205]. Der logistische Regressionsansatz berechnet die Wahrscheinlichkeit des Eintretens eines Ereignisses mit Hilfe der nichtlinearen logistischen Funktion:

$p = 1 / 1 + e^{-z}$ [Back06, S. 431; Sart06, S. 206; Bahr03, S. 136]

Die nichtlineare logistische Funktion, welche auch als Sigmoidfunktion bezeichnet wird, kann so oder in ähnlicher Form ebenfalls für die Aktivierung von Knoten eines neuronalen Netzwerks verwendet werden (siehe Unterabschnitt 3.3.1).
Bei der Verwendung dieses Verfahrens wird in folgenden Schritten vorgegangen, siehe [Back06, S. 433; Chri97, S. 170 - 177]:

Schritt 1 – Modellformulierung:
Im ersten Schritt werden die abhängigen Variablen sowie deren Einflussgrößen (unabhängigen Variablen) bestimmt, die für die jeweilige Anwendung relevant sind.

Schritt 2 – Schätzung der logistischen Regressionsfunktion:
In diesem Schritt wird – genau wie bei der Diskriminanzanalyse – eine Funktion ermittelt. Hierbei wird allerdings auf Basis bekannter empirischer Beobachtungen die Ausprägung und Skalierung der logistischen Funktion geschätzt.

Schritt 3 – Interpretation der Regressionskoeffizienten:
Dieser Schritt ist notwendig, aufgrund oftmals vorliegender, erschwerter inhaltlicher Interpretation der Ergebnisse.

Schritt 4 – Prüfung des Gesamtmodells:
In diesem Schritt erfolgt eine nochmalige Überprüfung der Qualität des Gesamtmodells.

Schritt 5 – Prüfung der Merkmalsvariablen:
In diesem letzten Schritt erfolgt eine Prüfung des Einflusses der unabhängigen Variablen auf die abhängigen Variablen des Modells.

Bei der Schätzung der logistischen Funktion werden die Parameter mit Hilfe der *Maximum Likelihood*-Methode auf Basis der bekannten empirischen Beobachtungswerte maximiert. Die *Maximum Likelihood*-Methode ist ein Verfahren aus der Statistik, mit dem auf Basis von Stichprobenergebnissen mit mehreren Variablen Rückschlüsse auf die Wahrscheinlichkeitsverteilung der Gesamtpopulation gezogen werden können. Für weiterführende Literatur zur *Maximum Likelihood*-Methode siehe [Blob98, S. 183 - 188]. Das Ziel dieses Schätzverfahrens ist, die Parameter der logistischen Funktion so zu ermitteln, dass die Wahrscheinlichkeit, die beobachteten Erhebungsdaten exakt zu erhalten, maximal wird. [Back06, S. 436]
Die Regressionsverfahren zählen demnach zu den überwachten Lernverfahren [Wied03, S. 284]. In Abbildung 20 ist ein Beispiel einer logistischen Regressionsfunktion abgebildet, die Aussagen über die Kaufwahrscheinlichkeit von Margarine auf Basis der Beobachtungswerte für die Parameter Streichfestigkeit und Haltbarkeit trifft.

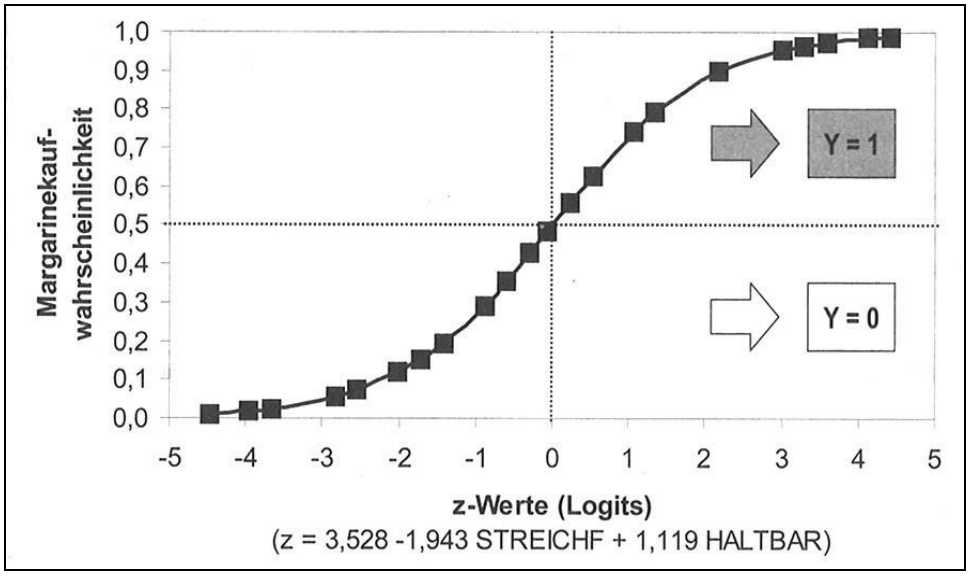

Abbildung 20: Beispielhafter Verlauf einer logistischen Regressionsfunktion aus [Back06, S. 439]

Wie in Abbildung 20 zu erkennen ist, gibt die Y-Achse die Höhe der Wahrscheinlichkeit für das Ereignis „Margarinekauf“ wieder, wobei die Grenze zwischen Kauf und Nicht-Kauf bei einer Wahrscheinlichkeit von 0,5 gezogen ist. Die X-Achse ist nach den z-Werten – auch

Logits genannt – skaliert. In diesem Beispiel beträgt die Regressionsgleichung z_k zur Bestimmung des Logitwerts für die Beobachtung k:

$$z_k = 3{,}528 - 1{,}943 * \text{Streichfestigkeit}_k + 1{,}119 * \text{Haltbarkeit}_k$$ [Back06, S. 438]

Diese Gleichung wird in einem iterativen Prozess ermittelt. Bei diesem Prozess werden durch Einsatz des Newton-Raphson Algorithmus die Parameterschätzungen so lange verändert, bis die Übereinstimmung der Ergebnisse mit den Beobachtungswerten maximal ist. [Back06, S. 438]
Bei diesem Algorithmus werden zunächst die Startwerte der Logitkoeffizienten angenommen. Anschließend wird ein beliebiger Beobachtungswert k als Wahrscheinlichkeitswert der Regressionsfunktion verwendet und der *LogLikelihood*-Wert LL mit Hilfe der folgenden Formel berechnet:

$$LL = \sum_{k=1}^{K} [y_k * \ln(1 / 1 + e^{-zk})] [(1-y_k) * \ln(1 / 1 + e^{-zk})]$$ [Back06, S. 437]

Diese Berechnung wird anschließend mit allen Beobachtungswerten durchgeführt um daraus die Summe, d.h. die Gesamtlogit-*Likelihood*-Funktion zu bestimmen. Als nächster Schritt erfolgt eine Wiederholung mit anderen Startwerten. Dieser Vorgang wird beendet, wenn keine deutliche Steigerung der Gesamtlogit-*Likelihood*-Funktion mehr erkennbar ist. [Back06, S. 437; Chri97, S. 346 - 347]
Bei alternativen Parameterschätzungen bzw. Koeffizientenschätzungen kann die logistische Funktion aus dem obigen Beispiel folgende Formen annehmen, wie in Abbildung 21 dargestellt:

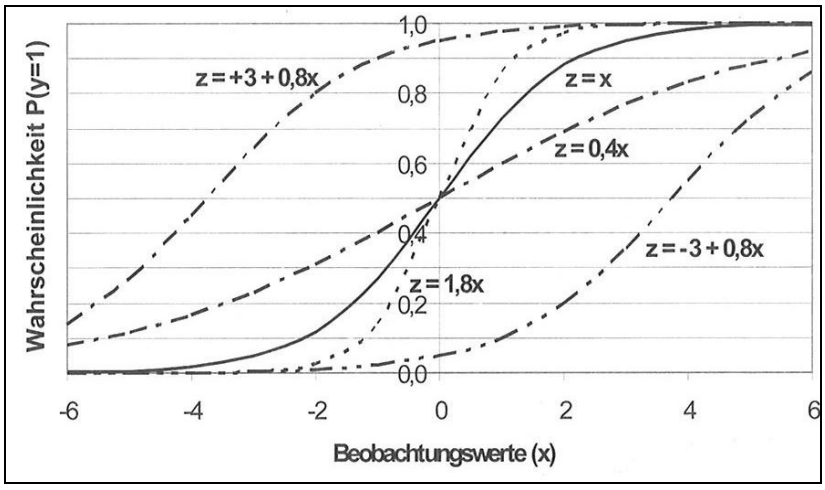

Abbildung 21: Logistische Funktionsverläufe bei verschiedenen Parametereinstellungen aus [Back06, S. 440]

Zur Erreichung des Ziels der Erstellung eines aussagekräftigen Modells sollte die Anzahl der Beobachtungen nach Aussage von [Sart06, S. 206] größer als 100 sein, einschließlich einer minimalen Anzahl an Beobachtungen von 25 pro Klassifikationsgruppe.
Bei der Prüfung des Gesamtmodells ist zusätzlich zu begutachten, ob die Parameterschätzungen das Modell in ihrer Gesamtheit bzw. die Beobachtungen gut widerspiegeln. Darüber hinaus ist zu bewerten, ob und in welchem Umfang extreme Ausreißer in den Beobachtungswerten vorliegen. Bei häufigem Auftreten von Ausreißern ist eine Anpassung des Modells erforderlich bzw. bei einem geringen Anteil im Verhältnis zur Gesamtbeobachtungsmenge sollten die Ausreißer für die Ermittlung der Regressionsgleichung nicht berücksichtigt werden. [Back06, S. 445]
Typische Anwendungsfälle der logistischen Regression sind beispielsweise die Ermittlung der Einflussfaktoren für das Wahlverhalten von Bürgern, für die Sterbewahrscheinlichkeit in der Intensivstation oder für das Geburtsgewicht von Babys. [Back06, S. 427]
Bei der klassischen linearen Regressionsanalyse ist die grundsätzliche Vorgehensweise identisch zur logistischen Regression. In diesem Fall wird eine lineare Funktion ermittelt. Bei der linearen Regressionsanalyse sind lediglich eine abhängige und eine unabhängige Variable möglich, da die Regressionsgleichung in Form einer Geraden dargestellt ist. Somit wird auf der Y-Achse keine Eintrittswahrscheinlichkeit für ein Ereignis skaliert. [Back06, S. 58 - 59; Sart06, S. 205]
In diesem Kapitel wurden die wichtigsten Verfahren des maschinellen Lernens beschrieben. Alle diese genannten Methoden benötigen zur Lösung eines Problems eine für die Analyse geeignete, problemrelevante und verfügbare Datenbasis. Daher wird im nächsten Kapitel mit *Complex Event Processing* die im Rahmen dieser Arbeit verwendete Technik zur Daten- bzw. Eventbereitstellung beschrieben.

4 Complex Event Processing

Dieses Kapitel beschreibt mit *Complex Event Processing* (CEP) die Basiskomponente des in dieser Arbeit beschriebenen Ansatzes zur Erkennung von Identitätsdiebstahl im Bereich Online-Banking. Ebenso wie die maschinellen Lernverfahren in Kapitel 3 wird auch in diesem Kapitel das Thema *Complex Event Processing* ohne Wertung beschrieben. Eine Begründung für die Verwendung von CEP wird ebenfalls im Abschnitt 6.2 gegeben. Da *Complex Event Processing* eine noch junge Disziplin ist, wird sie im Rahmen dieser Arbeit etwas ausführlicher beschrieben als die im Kapitel 3 diskutierten Verfahren. Die offizielle Gründung von CEP als Disziplin sowie die Bildung einer *Community* fand im März 2006 während des ersten *Event Processing Symposiums* in Yorktown Heights, NY, USA statt,

siehe dazu [Comp08]. Die ersten Veröffentlichungen zu diesem Thema wurden bereits zu Beginn dieses Jahrhunderts publiziert, wie beispielsweise [Luck02]. Mittlerweile bildete sich aus der anfangs lose zusammengeschlossenen *Community* die *Event Processing Technical Society* (EPTS) (siehe [Even09]), welche diese Technologie permanent weiterentwickelt und das *Event Processing Glossary* definiert (siehe [Luck08] in der aktuellen Version 1.1). Die Grundlage dieser Technologie bilden *events*, die nachfolgend definiert werden.

4.1 Events in Bezug auf CEP

Ein *event* im Sinne von CEP ist abstrakt definiert als eine Beobachtung von Interesse innerhalb eines Computersystems [Mühl06, S. 2]. Die Definition des Begriffs *event* ist in [Luck02, S. 88] angeführt als:

> *An event is an object that is a record of an activity in a system. The event signifies the activity. An event may be related to other events.*

Bei [Luck02, S. 88] entspricht ein *event* daher einer Aufzeichnung einer Aktivität eines Systems, das in diesem Zusammenhang folgende drei Aspekte aufweist, siehe [Luck02, S. 88]:

a) *Form*: *Form* ist die Struktur eines *events*. Ein *event* ist ein Objekt. Als solches weist es Datenkomponenten bzw. Attribute auf. Diese können einfache String- oder Integer-Parameter sein oder ein Zeitstempel, der aussagt, wann das *event* entstand usw. Der folgende Java-Code aus [Luck02, S. 89] zeigt beispielhaft die *Form* eines *events* als Java-Klasse:

```
Class InputEvent { Name  NewOrder;
                   Event_Id  E_Id;
                   Customer  Id;
                   OrderNo  O_Id;
                   Order  (CD X, Book …);
                   Time T;
                   Causality (Id1, Id2, …);
                  }
```

b) *Significance*: *Significance* ist die Aktion, welche das *event* kennzeichnet bzw. die das *event* ausgelöst hat. Ein *event* beinhaltet oft ein Attribut, das diese Aktion benennt.

c) *Relativity*: *Relativity* bezeichnet die Beziehungen eines *events* zu anderen *events.* Diese Beziehungen basieren entweder auf dem Prinzip von Ursache und Wirkung oder zeitlichen Abfolgen von *events*. Die Beziehungen von *events* zueinander sind in Form von bekannten *event patterns* beschreiben (siehe Abschnitt 4.2).

Mittlerweile haben sich bei der Struktur der *events* verschiedene Formate bzw. Formatvorschläge gebildet, wie z.B. das von D. Ogle und A. Kreger entwickelte *Common Base Event*-Format (CBE) von `IBM Corp`. [Ibm04]. Das *Event Processing Glossary* definiert ein *event* allgemeiner als [Luck02, S. 88] in folgender Form, siehe [Luck08; Chan09, S. 111]:

> *Anything that happens, or is contemplated as happening.*

Dahingehend erfolgt eine weitere Definition als *event object* für die computergestützte Verarbeitung, siehe [Luck08; Chan09, S. 111]:

> *An object that represents, encodes, or records an event, generally for the purpose of computer processing.*

Events stammen innerhalb einer Organisation aus den verschiedensten Quellen. In [Luck02, S. 91] sind die drei grundsätzlichen Quellen genannt:

a) *IT Layer: IT Layer* bezeichnet *events* für die Kommunikation zwischen den IT-Schichten eines Unternehmens. Diese Nachrichten werden von der CEP *engine* beobachtet und mittels *Adapter* in CEP-fähige *events* transformiert.

b) *Instrumentation: Instrumentation* bezeichnet Komponenten eines Systems, die darauf konfiguriert sind, *events* als Folge von Aktionen bzw. Situationen zu erzeugen. Beispiele hierfür sind Statusprüfungen in Betriebssystemen, Transaktionsevents oder *alerts*, die von einem Netzwerkmanagementsystem generiert wurden.

c) *CEP:* CEP bezeichnet *events*, die von CEP *engines* generiert werden als Folge der Verarbeitung beobachteter Systemevents, d.h. diese *events* werden von der CEP *engine* erzeugt.

In einem Blog-Eintrag hat T. Bass die Eventquellen – wie nachfolgend aufgelistet – konkretisiert, siehe [Thec07b]:

- Geräte als Eventquellen:
  - Router, Hubs und andere Telekommunikations- und Netzwerkgeräte
  - Netzwerkgeräte für die Überwachung, wie z.B. Firewall, Sniffer usw.
  - Sensoren und Sensornetzwerke mit RFID-Lesegeräten

- Log-Dateien als Eventquellen:
  - Log-Dateien von Anwendungen (einschließlich syslog)
  - Andere *flat file event*-Logs

- Datenbanken als Eventquellen:
  - Datensatzmanipulation (*insert, update, delete*)
  - Transaktionsevents bzw. Datenbankinhalte

- *Data Feeds* als Eventquellen:
  - Marktdaten *feeds*
  - Nachrichten *feeds*
  - Andere *feeds* von entsprechenden Anbietern

- *Message-oriented middleware* als Eventquellen:
  - *Java Message Service* (JMS)
  - *Tibco Rendezvous*
  - *IBM Websphere Message Queueing* (MQ)

- Analysemethoden bzw. deren ausführende Anwendungen als Eventquellen:
  - *Rule engines*
  - Statistische Analysewerkzeuge (z.B. *Bayesian Inference engines*)
  - Künstliche neuronale Netzwerke
  - Filter und Signal Prozessoren (z.B. *Kalman filters*)
  - CEP und *Event Stream Processing* (ESP) *engines*, die eine oder mehrere der genannten Methoden enthalten können

- Prozess- oder *Ressource Management*-Systeme als Eventquellen:
  - *Business Process Management* (BPM) Systeme
  - *Customer Relationship Management* (CRM) Systeme

- Anwender als Eventquellen:
  - Email
  - *Instant Messaging* (IM)

Für den in dieser Arbeit entwickelten Lösungsansatz sind vor allem Datenbanken der Transaktionssysteme von Kreditinstituten bzw. Bankrechenzentren relevante Eventquellen, was im Abschnitt 6.1 erläutert wird.
Zusammenfassen lassen sich die verschiedenen Eventquellen bzw. Eventtypen und deren Auftreten mit dem von D. Luckham geprägten Begriff *Event Cloud*. Die Definition einer *Event Cloud* im *Event Processing Glossary* lautet, siehe [Luck08]:

> *A partially ordered set of events (poset), either bounded or unbounded, where the partial orderings are imposed by the causal, timing and other relationships between the events.*

Mit dieser Definition wird ausgedrückt, dass Organisationen heute komplexe und heterogene IT-Landschaften mit vielen internen und externen Schnittstellen besitzen [Luck02, S. 28 - 31]. Dadurch treten die *events* oftmals zeitlich ungeordnet in Form von *posets* (*partially ordered sets*) in Erscheinung. Den Gegensatz dazu bildet *Event Stream Processing* (ESP), wobei die *events* in zeitlich geordneten Strömen auftreten, die *tosets* (*totally or linearly ordered sets*) genannt werden. [Thec07a; Luck08]
Des Weiteren werden *events* nicht nur nach ihrer Quelle sondern auch nach ihrer Eventhierarchie unterschieden. Bezüglich dieser Hierarchien existieren zum einen die *low level events* wie z.B. *JMS Messages*, siehe [Haas02], Netzwerknachrichten wie *TCP Acknowledges*, siehe [Kuro02, S. 220 - 222] oder *SNMP Traps*, siehe [Kuro02, S. 618 - 620]. *Low level events,* die in [Chan09, S. 119] auch als *simple base events* bezeichnet werden, besitzen für sich alleine keine semantische Signifikanz und sind unveränderlich [Howa06]. Den Gegensatz dazu bilden die hierarchisch höher gestellten *complex events,* die sich aus einer definierten Kombination von *low level events* oder wiederum *complex events* zusammensetzen oder durch das Auftreten dieser Kombinationen erzeugt bzw. korreliert werden [Chan09, S. 118 - 120].

Im *Event Processing Glossary* ist diese Form von *events* folgendermaßen definiert, siehe [Luck08]:

> *Complex event: an event that is an abstraction or aggregation of other events called its members.*

Innerhalb der CEP *Community* herrscht (noch) Unentschlossenheit zum Begriff der *complex events*. In [Mühl06, S. 235] sind sie als *composite events* bezeichnet, mit folgender Definition:

> *Every composite event "c" has a composite event type "Tc" and belongs to the composite event space "C". A composite event type "Tc" corresponds to a valid expression "C" in a composite event language. A composite event "c" consists of an interval timestamp "tc" and a set of composite subevents {c1, c2,..., ck}.*

In dieser Arbeit wird nachfolgend die Definition von *complex events* verwendet, wie in [Luck08] beschrieben.
Das Korrelieren von *low level event*s zu *complex events* über die verschiedenen IT-Schichten hinweg ist in Abbildung 22 verdeutlicht.

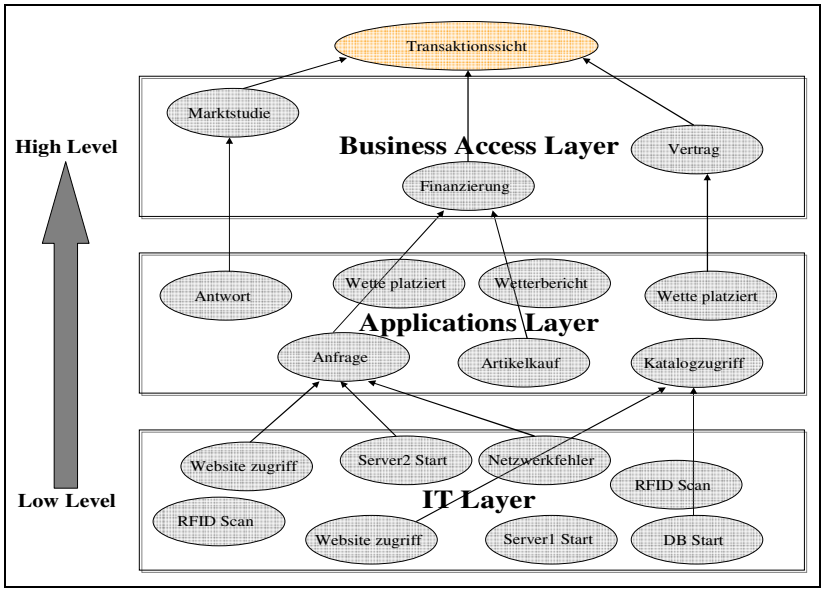

Abbildung 22: Exemplarisches Korrelieren von low level events zu complex events durch die verschiedenen Schichten in Anlehnung an [Luck05; Chan09, S. 119]

Ein *complex event* wird von der CEP *engine* erzeugt, wenn mit Hilfe von *event patterns* eine bestimmte Eventkombination erkannt wird.

4.2 Event Patterns

Event patterns definieren aus Kombinationen von *low level events* sog. *complex events*. In [Luck02, S. 114] ist ein *event pattern* folgendermaßen definiert:

> *An event pattern is a template that matches certain sets of events – the set you want to find. It describes precisely not only the events but also their causal dependencies, timing, data parameters, and context. So an event pattern is a template for posets.*

Abstrakte Beispiele für *event patterns* aus [Luck02, S. 16] sind:

- Eine Menge von Netzwerkevents, die aus dem gleichen Quellgerät stammen und das Eröffnen eines Zugangs zu verschiedenen Zielgeräten erreichen wollen, löst ein *alert event* aus. Dies deutet auf einen Einbruchsversuch in das Netzwerk hin.

- Eine Menge von *stock trading events,* die vom gleichen Account stammen und in einem bestimmten Zeitintervall auftreten, lösen ein *alert event* aus. Dies deutet auf die Verletzung einer Richtlinie hin.

Im *Event Processing Glossary* ist ein *event pattern* ähnlich definiert, siehe [Luck08]:

> *A template containing event templates, relational operators and variables. An event pattern can match sets of related events by replacing variables with values.*

Um von einer abstrakten Ebene aus ein *event pattern* mit den Operatoren und Variablen konkret beschreiben bzw. definieren zu können, wird eine Entwicklungssprache oder *Event Processing Language* (EPL) benötigt [Chan09, S. 121]. Die offizielle Definition einer EPL aus dem *Event Processing Glossary* lautet, siehe [Luck08]:

> *A high level computer language for defining the behavior of event processing agents.*

Eine EPL muss nach [Luck02, S. 146] die vier folgenden Eigenschaften aufweisen:

a) *Power of Expression: Power of Expression* definiert die Notwendigkeit nach komplexen Vergleichsoperatoren wie z.B. *and, or, causes* oder *is independent of*.

b) *Notational Simplicity: Notational Simplicity* definiert die Notwendigkeit einer EPL nach einer einfachen Notation, welche die *patterns* schnell programmieren lässt.

c) *Precise Semantics: Precise Semantics* definiert die Notwendigkeit einer EPL nach einem mathematisch eindeutigen Prüfkonzept.

d) *Scalable Pattern Matching: Scalable Pattern Matching* definiert die Notwendigkeit einer EPL nach einem Sprachdesign, wodurch die CEP *engine* in der Lage ist, große Mengen an *events* in Echtzeit verarbeiten zu können.

Eine in der Praxis verwendete EPL ist STRAW EPL, beschrieben in [Luck02, S. 116 - 126]. Sie ist eine sehr einfache EPL, deren *patterns* die folgenden Sprachelemente aufweisen:

- Eine Liste der Variablen des *patterns (variables).*
- Eine Liste der Eventtypen (*event types*).
- Eine Definition der Beziehung der *events* zueinander (*relational operators*).
- Eine Definition des *event patterns (pattern)*.
- Eine Bedingung, die erfüllt sein muss, damit das *pattern* zutrifft (*context test).*
- Eine Aktion, die bei *„pattern match“* ausgelöst wird (*action*).

Folgendes *event pattern* ist in STRAW EPL geschrieben, siehe [Luck02, S. 122]:

Element	***Declarations***
Variables	*node N1, node N2, data D, bit B, time T1, time T2, time T3, time T4;*
Event Types	*send (node N1, node N2, data D, bit B, time T1),* *receive (node N1, node N2, data D, bit B, time T1),* *ack (node N1, node N2, bit B, time T1),* *recack (node N1, node N2, bit B, time T2),* *warning (node N1, node N2, time T1, time T2);*
Rational operators	*-> (causes);*
Pattern	*send (N1, N2, D, B, T1) -> receive (N2, N1, D, B, T2) ->* *ack (N2, N1, B, T3) -> recack (N1, N2, B, T4);*
Context test	*T4 – T1 >= 1 hour;*
Action	*create warning (N1, N2, D, T1, T4);*

Dieses *event pattern* beschreibt eine Datenübertragung auf der *Transmission Control Protocol* (TCP) Ebene. Für weiterführende Informationen über das Netzwerkprotokoll TCP siehe [Kuro02, 212 - 234]. Als Variablentypen werden *node*, *data*, *bit* und *time* verwendet, als Eventtypen *send event*, *receive event*, *ack event* und *recack event*. Das *event pattern* besagt, dass bei Überschreitung einer definierten Zeitspanne (1 Stunde) zwischen dem *send event* und dem *reckack event* ein *complex event* namens *warning* erzeugt wird. [Luck02, S. 121 - 122].
STRAW EPL oder RAPIDE EPL sind Standardsprachen für *event patterns*. Komponenten bzw. *Application Programming Interfaces* (API) zum Programmieren von und mit *events* sind mittlerweile ebenso in klassischen objektorientierten Programmiersprachen wie Java, C#, Visual Basic oder .NET vorhanden, für weiterführende Literatur siehe [Fais06]. Zusätzlich entwickelten Hersteller von CEP *engines*, wie beispielsweise `StreamBase Inc.` oder `TIBCO Software Inc.` oftmals eigene Sprachen oder graphische Codegeneratoren, siehe hierzu Tabelle 5 und [Howa06].

Event Processing Languages werden ebenfalls für die Programmierung von *Event Processing Agents* (EPA) verwendet. In [Luck02, S. 176] ist ein EPA nachfolgend definiert:

> *An event processing agent is an object that monitors an event execution to detect certain patterns of events. When those patterns occur in its input, it reacts by executing actions. An EPA can monitor an event execution online, in real time as the events are being created, or offline, postmortem, using a log of the execution.*

Im *Event Processing Glossary* ist ein EPA beschrieben als ein Softwaremodul zur Verarbeitung von *events*, siehe [Luck08]. Ein EPA besitzt als Eingangsschnittstelle die Definition eines *event patterns*, das den Agenten aktiviert und als Ausgangsschnittstelle die Definition von Eventtypen bzw. *complex events*, die von dem Agenten erzeugt werden. Ein EPA ist so konzipiert, dass er Funktionen ausführen kann, z.B. Attribute der Eingangsevents abfragen, die wiederum die Ausgabeevents beeinflussen [Chan09, S. 97 - 98]. Abbildung 23 zeigt die Struktur eines EPA.

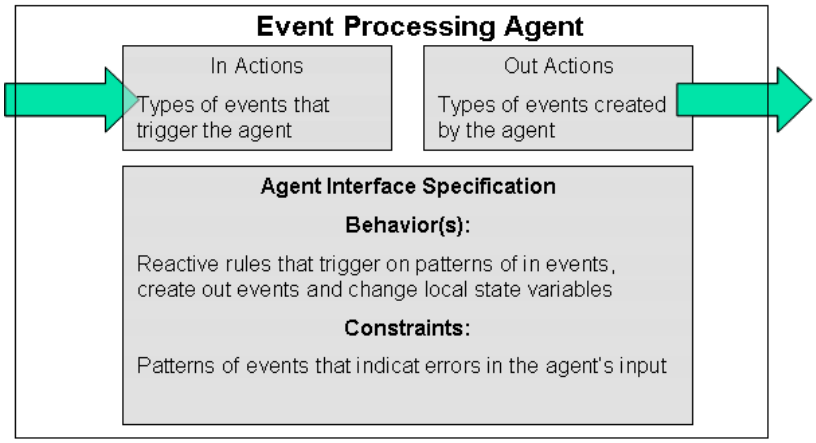

Abbildung 23: Struktur eines EPA in Anlehnung an [Luck02, S. 176]

Nachfolgend sind ein *pattern* bzw. die Funktionen eines EPA mit RAPIDE EPL beschrieben:

```
class Sensor
in action activate(), deactivate();
out action light (enum {ON,Off} X;);
{activate    => generate light(On);
deactivate => generate light(Off);
};
```

Dieser EPA stellt ein *event pattern* eines Sensors dar, das die *events* an der Input-Schnittstelle auf ein *activate event* und *deactivate event* abprüft und mittels einem *generate light event* reagiert, in Abhängigkeit vom Eingangsevent. Bei dem EPA in dem oben genannten Beispiel handelt es sich um eine *Event Pattern Map*. Bei EPA's lassen sich generell die folgenden drei Typen unterscheiden:

a) *Event Pattern Filter*: Ein *Event Pattern Filter* ist ein Agent, der aus Eventmengen nach bestimmten Regeln Untermengen herausfiltert [Luck02, S. 187 - 192].

b) *Event Pattern Map:* Eine *Event Pattern Map* implementiert den klassischen CEP-Ansatz des Korrelierens bzw. Aggregierens von *low level events* und des Generierens von *complex events* [Luck02, S. 193 - 195].

c) *Event Pattern Constraints:* Ein *Event Pattern Constraints* ist eine spezielle Form der *Event Pattern Map. Constraints* überwachen die Einhaltung von definierten Bedingungen oder *Security Policies*. Beispiele hierfür sind die Überwachung des Eintreffens oder Ausbleibens bestimmter Eventtypen. [Luck02, S. 195 - 204]

EPA's stehen innerhalb einer CEP *engine* nicht für sich allein, sondern bilden Netzwerke, sog. *Event Processing Networks* (EPN). In [Luck02, S. 207] ist ein EPN nachfolgend definiert:

> *Event Processing Networks can be represented graphically as networks consisting of EPA's at the nodes and communication between the EPA's by means of events flowing on the network paths between the nodes.*

Auch [Chan09, S. 95] sowie das *Event Processing Glossary* [Luck08] bezeichnen den Zusammenschluss von EPA's zu einem Netzwerk als *Event Processing Network*.
Die allgemeine Struktur der EPN's sieht eine Kommunikation von *Adapter* zu *Event Pattern Filter* und von dort weiter zu *Event Pattern Maps* als EPA-Typen vor. Diese Struktur ist in Abbildung 24 ersichtlich.

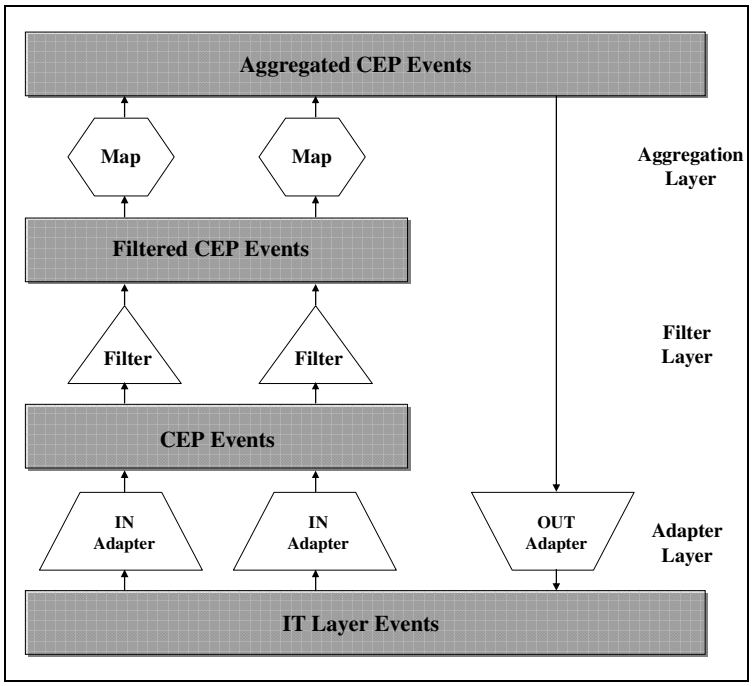

Abbildung 24: Klassische Struktur eines EPN in Anlehnung an [Luck02, S. 208]

Darüber hinaus existieren die drei folgenden Arten der Verbindungen von EPA's innerhalb eines EPN, siehe [Luck02, S. 212 - 217]:

a) *Basic Connection*: Bei einer *Basic Connection* entsprechen die *output actions* bzw. *output events* eines EPA's den *input actions* bzw. *input events* eines anderen EPA's.

Syntax in RAPIDE EPL:

connect
(variable_declarations)
EPA1.out_Action(parameters) => EPA2.in_Action(parameters);

b) *Guarded Connection*: Eine *Guarded Connection* ist ein spezielle Form der *Basic Connection*, die nur ausgeführt wird, wenn vorher eine Überprüfung (= *guard*) stattgefunden hat und diese bestanden wurde.

Syntax in RAPIDE EPL:

connect
(variable_declarations)
EPA1.out_Action(parameters) where C(parameters) =>
EPA2.in_Action(parameters);

c) *Multiple Basic Connections*: Eine *Multiple Basic Connection* ist eine Verbindung von einem Quell-EPA zu mehreren Ziel-EPA's. Hierbei wird zwischen sequenzieller *Connection* (= Verbindungsaufnahme zu den Ziel-EPA's in einer festgelegten Reihenfolge) und

paralleler *Connection* (= gleichzeitige Verbindungsaufnahme zu den Ziel-EPA's) unterschieden.

Syntax in RAPIDE EPL:

```
connect   // sequenzielle Connection
(variable_declarations)
EPA1.out_Action(parameters) => EPA2.in_Action(parameters);
                            => EPA3.in_Action(parameters);

connect   // parallel Connection
(variable_declarations)
EPA1.out_Action(parameters) ||> EPA2.in_Action(parameters);
                            ||> EPA3.in_Action(parameters);
```

Neben den genannten Verbindungen können EPN's wiederum in Architekturen miteinander verbunden werden. Für weiterführende Informationen hierzu siehe [Luck02, S. 221 - 230].

Die Vorteile bei der Verwendung von EPN's liegen in der Verteilung der Komplexität auf viele Agenten. Dies führt zu einer sehr flexiblen Struktur, bei der ein Austausch von EPA's bei Bedarf auch zur Laufzeit ohne großen Aufwand durchführbar ist. [Luck02, S. 211 - 212]

4.3 Eventbasierte Systeme

Ein eventbasiertes System verwendet *events* um z.B. das Prinzip von *Publish and Subscribe* zu implementieren. Dieses Prinzip besagt, dass ein Sender (engl.: *Producer*) und ein Empfänger (engl.: *Consumer*) Nachrichten mittels *events* übermitteln. Sender und Empfänger sind dabei voneinander entkoppelt. Das bedeutet, der Empfänger registriert (*engl.: subscribe*) sich für den Empfang von bestimmten gewünschten Nachrichten bei einem *Event Notification Service* (auch *Publish/Subscribe Middleware* oder kurz *Pub/Sub* genannt). Dieser *Event Notification Service* dient als Vermittler zwischen Sender und Empfänger und verteilt (*engl.: publish*) eine gesendete Nachricht an die Empfänger, die sich für diesen bestimmten Nachrichtentyp registriert haben. [Mühl06, S. 11 - 14; Chan09, S. 108 - 109]

Eine Möglichkeit der Implementierung des *Publish and Subscribe*-Prinzips bietet der *Java Message Service* (JMS), siehe [Haas02] oder der *CORBA Notification Service*, siehe [Tark05]. In Abbildung 25 ist die Architektur eines *Publish and Subscribe*-Systems graphisch dargestellt.

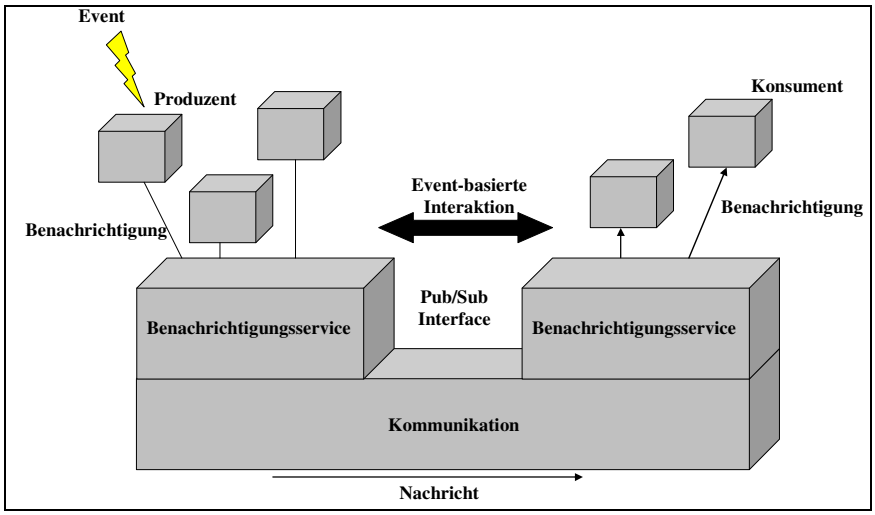

Abbildung 25: Architektur eines Publish and Subscribe-Systems in Anlehnung an [Mühl06, S. 12]

In Abbildung 25 ist zu erkennen, dass sich sowohl verschiedene *Producer* als auch verschiedene *Consumer* bei einem *Notification Service* registrieren, die wiederum über *messages* bzw. *events* miteinander kommunizieren. Für die Registrierung stellen die *Notification Services* spezielle *Pub/Sub Interfaces* zur Verfügung.
Eine andere Form der eventbasierten Systeme bildet *Complex Event Processing* in Form von CEP *engines*. CEP *engine* ist ein Begriff, der durch Hersteller aus dem CEP-Umfeld geprägt wurde. In einem *White Paper* der `IBM Corp.` ist der Begriff CEP *engine* beispielsweise folgendermaßen beschrieben, siehe [Nech06]:

> *During runtime, events of different format and from multiple sources are fed in. The CEP engine analyzes them based on current rules and conditions. Upon detection of the predefined patterns, alerts are sent out as messages and actions are triggered. These messages are also fed back into the engine and treated as incoming events, enabling nested patterns.*

Bei `Coral8 Inc.`, einem amerikanischem CEP-Hersteller lautet die Definition von CEP *engine* folgendermaßen, siehe [Tsim06]:

> *A CEP engine is a platform for building and running applications, used to process and analyze large numbers of real-time events.*

Diese Definition wurde von `Coral8 Inc.` später konkretisiert, siehe [Cora07]:

> *A software product used to create and deploy applications that process large volumes of incoming messages or events, analyze those messages or events in various ways, and respond to conditions of interest in real-time.*

Der Unterschied von *Complex Event Processing*-Technologie zu *Publish and Subscribe*-Systemen ist, dass CEP in der Lage ist, unterschiedliche Eventtypen gleichzeitig zu verarbeiten. Bei *Publish and Subscribe* dagegen werden die *events* unabhängig von anderen *events* verarbeitet. [Chan07]
In [Luck02, S. 329 - 351] wird eine CEP *engine* auch als CEP *infrastructure* bezeichnet. Sie ist die technische Umgebung, in welcher der Mustererkennungsprozess ausgeführt wird. Die CEP *engine* besitzt Schnittstellen zu den Quellsystemen einer Organisation mittels *Adapter*. In Abbildung 26 sind die Komponenten eines *Adapter* abgebildet.

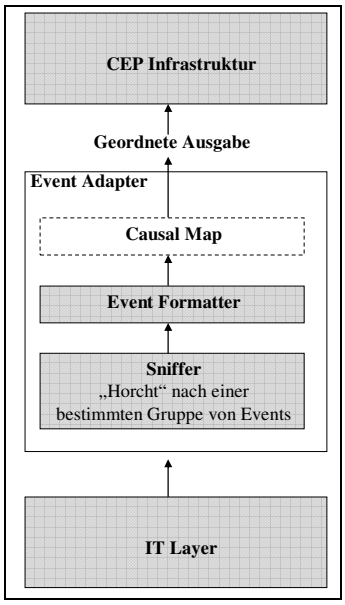

Abbildung 26: Aufbau eines Adapter in Anlehnung an [Luck02, S. 337]

Innerhalb des *Adapter* werden durch den Sniffer diejenigen *events*, die für die Mustererkennung nicht relevant sind, aus dem Eventstrom herausgefiltert. Das Filtern von *events* basiert auf den Eventtypen und deren Attributen [Tsim06]. Der *Event Formatter* ist für das Umwandeln (engl.: *Mapping*) der *events* aus dem Format des Quellsystems in ein Format, das die CEP *engine* interpretieren bzw. analysieren kann, zuständig. Mittels einer *Causal Map* können die *events* zeitlich (nach ihrem *Timestamp*-Attribut) geordnet werden. Die

Funktion der *Causal Map* ist optional, d.h. nicht jeder *Adapter* beinhaltet diese Funktionalität. [Luck02, S. 336 - 338]

Die CEP *infrastructure* oder CEP *engine* besitzt eine Vielzahl von *Adapter* zu den jeweiligen Quellsystemen der *events* z.B. zu einer *Publish/Subscribe Middleware* [Luck02, S. 338]. Abbildung 27 zeigt eine vereinfachte Architektur einer CEP *infrastructure*.

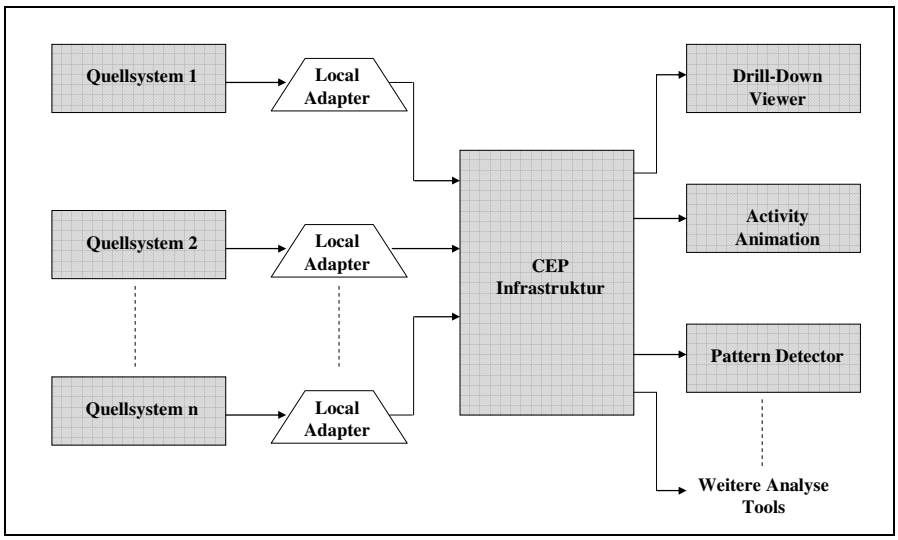

Abbildung 27: Exemplarischer Aufbau einer CEP-Umgebung in Anlehnung an [Luck02, S. 336]

In dieser Architektur werden die einzelnen *events* aus den verschiedenen Quellsystemen über die *Adapter* in der CEP *engine* gesammelt und dort von den einzelnen Analysewerkzeugen wie *Pattern Detector* oder *Activity Animation* verarbeitet. Die CEP *infrastructure* bildet in diesem Zusammenhang die Basis, von der aus *Business Activity Monitoring* (BAM) betrieben werden kann, siehe dazu [Luck04a].

Eine CEP *infrastructure*, die auf EPN's gestützt ist, ist in Abbildung 28 abgebildet.

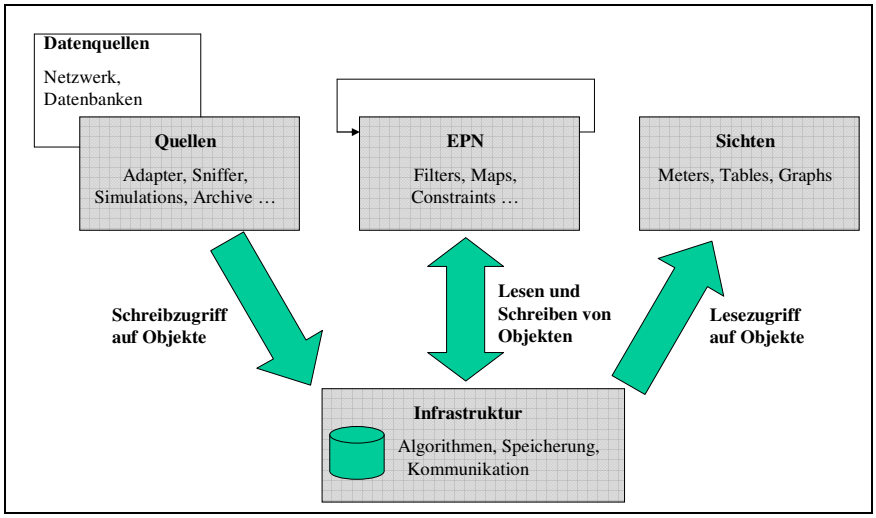

Abbildung 28: CEP infrastructure mit EPN in Anlehnung an [Luck02, S. 339]

Die Quellsysteme sind in dieser CEP *infrastructure* ebenfalls mittels *Adapter* angebunden. Die ankommenden *events* werden in Echtzeit von den EPN's verarbeitet, die wiederum einen Bestandteil der CEP *engine* bilden. Die CEP *engine* besitzt zusätzlich Schnittstellen für graphische Auswertungswerkzeuge. Eine weitere wichtige Eigenschaft von CEP *engines* ist die Echtzeitfähigkeit. Das bedeutet, dass die *events* zum Zeitpunkt ihres Auftretens analysiert werden, gleichzeitig der Mustererkennungsprozess ausgeführt wird und anschließend bei Bedarf mit entsprechenden, vordefinierten Aktionen reagiert werden kann. Die genauen Zahlen an *events*, die von den Lösungen der einzelnen Hersteller pro Sekunde verarbeitet werden können, sind im Abschnitt 4.4 genannt.
In einem Blog-Eintrag hat T. Puzak die Eigenschaften beschrieben, die eine CEP *engine* erfüllen muss, um optimal in einem Unternehmen eingesetzt werden zu können, siehe [Thec08]. In diesem Zusammenhang wird davon ausgegangen, dass die Grundfunktionalitäten wie Echtzeitfähigkeit und *event aggregation* zu *complex events* erfüllt werden und sind daher nicht aufgeführt:

- Der Erkennungsmechanismus muss in der Lage sein, *event patterns* zu identifizieren, bei denen die einzelnen *events* innerhalb einer bestimmten definierten Zeitperiode (auch längere Zeitperioden) auftreten oder nicht auftreten.

- Der Erkennungsmechanismus muss in der Lage sein, *events* nach der Reihenfolge ihres Auftretens unterscheiden zu können um Mehrfacherkennungen zu vermeiden.

- Die CEP *engine* muss in der Lage sein, die *complex events* bzw. die auszulösenden Aktionen in der bestimmten definierten Reihenfolge auszuführen.

- Die CEP *engine* muss in der Lage sein, die *complex events* bzw. die auszulösenden Aktionen zu einem bestimmten definierten Zeitpunkt ausführen zu können.

- Die CEP *engine* muss in der Lage sein, bestimmte *events* beim Start oder Stopp der Anwendung auszulösen.

- Die CEP *engine* muss in der Lage sein, auftretende *events* nicht nur mit gleichen *events* zu *complex events* zu korrelieren, sondern auch mit anderen *events*, die beispielsweise durch SQL-Abfragen aus Datenbanken gewonnen werden.

- Die CEP *engine* muss in der Lage sein, auftretende *events* bei Bedarf abzuspeichern und somit für spätere Auswertungen verfügbar zu machen.

- Die CEP *engine* muss in der Lage sein, mit externen Datenbanken interagieren zu können und somit *Adapter* für die gängigen Lösungen der bekannten Datenbankhersteller (z.B. `Oracle Corp.`) zur Verfügung zu stellen.

- Die CEP *engine* muss Entwicklungsmöglichkeiten bzw. eine Entwicklungsumgebung bereitstellen.

Für diese Arbeit ist vor allem die Fähigkeit einer CEP *engine*, auftretende *events* mit *events* aus SQL-Abfragen korrelieren zu können, sehr wichtig. Die Begründung hierfür ist, dass die Transaktionsevents mit bestimmten Daten aus Bestandssystemen erweitert werden müssen um die Durchführung der weiteren Untersuchungen mittels des in dieser Arbeit beschriebenen Ansatzes gewährleisten zu können (siehe Abschnitt 6.1).

4.4 Praktische Einsatzgebiete von CEP

Im Zuge der Verbreitung von CEP haben sich viele praktische Anwendungsgebiete herauskristallisiert. Nachfolgend sind die wichtigsten Bereiche aufgelistet:

a) *IT-Blindness*: *IT-Blindness* ist – genau wie *Event Cloud* – ein von D. Luckham geprägter Begriff. Er beschreibt die Tatsache, dass tausende von *events* Minute für Minute in den IT-Schichten der Unternehmen entstehen, ohne dass die Organisationen wissen, welche Muster sich hinter den *events* verbergen und welche Auswirkungen diese auf die Geschäftsprozesse der Unternehmen haben, siehe [Luck04b]. Durch den Einsatz von CEP bzw. *event patterns* können diese *low level events* über die IT-Schichten hinweg zu *complex events* korreliert und diese wiederum z.B. über BAM-Module überwacht werden. Diesen Prozess bezeichnet D. Luckham als *IT-Insight,* siehe [Luck04c].

b) *Algorithmic Trading*: Beim *Algorithmic Trading* werden auf Basis bekannter Muster automatisch Kauf- oder Verkaufstransaktionen an Finanzmärkten getätigt. Hierbei ist die Echtzeitfähigkeit von CEP *engines* besonders wichtig um auf die *events* im Börsen- und Marktumfeld sofort reagieren zu können. Der Algorithmus legt dabei den Zeitpunkt der Transaktionen anhand vordefinierter Parameter fest, wobei sowohl historische als auch aktuelle Marktdaten mit einbezogen werden. [Bate07]

c) *Predictive Business*: Bei *Predictive Business* handelt es nicht nur um die Reaktion auf bestimmte auftretende *event patterns*. Das Ziel ist dieser Anwendung ist, die *event patterns* bzw. das Auftreten der definierten Eventkombinationen bereits zu einem früheren Zeitpunkt vorauszusagen. Dies funktioniert durch das Korrelieren von historischen *events*, die aufgrund von Beobachtungen in deterministischen Ketten zusammengesetzt sind, mit aktuell auftretenden *events*. Somit können mit Hilfe bekannter deterministischer Ablaufketten Schlussfolgerungen in Echtzeit gezogen werden. Diese Ketten können in der Realität durch unvorhergesehene Ereignisse unterbrochen werden, wodurch die zukünftigen Ereignisse nur mit einer bestimmten Wahrscheinlichkeit vorausgesagt werden. [Rana06, S. 39 - 59]

d) *Global Epidemic Warning*: Bei *Global Epidemic Warning*-Systemen handelt es sich um Anwendungen, die verschiedene Quellen aus dem Internet (z.B. *News Feeds*, Zeitungsberichte oder lokale medizinische Quellen in mehreren Sprachen) benutzen um mittels CEP-Technologie Muster in Echtzeit abzuleiten, die auf Epidemien schließen lassen. Das bekannteste System in diesem Umfeld ist das kanadische *Global Public Health Intelligence Network*, das von der *World Health Organisation* (WHO) eingesetzt wird. Wenn das System ein Seuchenmuster identifiziert (unabhängig davon ob natürliche Epidemie oder Bio-Terrorismus), informiert das System sofort lokale Krankenhäuser, Gesundheitsämter, Behörden und Fluggesellschaften. [Luck07]

e) *Intrusion Detection*: Im Bereich der Thematik *Intrusion Detection* existieren viele klassische Lösungsansätze, die z.B. in Richtung *Data Mining* gehen, siehe hierzu [Brug04]. Im CEP-Umfeld werden mittels bekannter *event patterns* auftretende *events* in einem Netzwerk nach Einbruchsversuchen in Echtzeit untersucht und bei Bedarf sofort reagiert. In [Luck02, S. 333 - 335] ist beispielsweise ein *event pattern* in RAPIDE EPL beschieben, das einen Einbruchsversuch identifiziert, bei dem Dateien mit Passwörtern von verschiedenen Rechnern aus an eine unbekannte Email-Adresse gesendet werden.

f) *Fraud Detection*: *Fraud Detection* ist ein weiteres Feld für die Anwendung von CEP. In diesem Umfeld werden *event patterns* gesucht, die auf Betrugsversuche hindeuten, z.B. in der Bankenbranche. Der konkrete Anwendungsfall dieser Arbeit betrifft *Fraud Detection*, daher wird die relevante Literatur dieses Anwendungsfalls im Abschnitt 5.1 diskutiert.

Für weitere Anwendungsbereiche wie z.B. Risiko- und *Compliance*-Anwendungen, *Healthcare*, *Customer Relationship Management* (CRM) usw., siehe [Chan09, S. 78 - 89] sowie die Beschreibungen zum ersten und zweiten *Event Processing Symposium* bei

[Comp08]. Diese oben genannten Anwendungsbereiche werden von den verschiedenen CEP-Herstellern in ihren entsprechenden Lösungen abgedeckt. In Tabelle 5 sind die bekanntesten Vertreter der Branche mit ihren CEP-Produkten aufgelistet und näher beschrieben. Die Kriterien, nach denen sich die Hersteller unterscheiden, sind die verfügbare Entwicklungsumgebung, die Betriebssysteme, auf denen die angebotenen Lösungen installiert werden können, der Durchsatz von *events* in einer Sekunde sowie die Anwendungsgebiete, in denen die jeweilige CEP-Lösung zum Einsatz kommt.

Hersteller	Produkt	Entwicklungsumgebung	Unterstützte Betriebssysteme	Event Durchsatz	Anwendungsgebiete	Bemerkung	Quelle
Progress Software Corp.	Progress Apama	Apama Studio (Eclipse basiert)	Linux, Windows, Solaris	mehrere 10.000 events/sek.	Algorithmic Trading, Risikomanagement, RFID	Beinhaltet eine eigene Datenbank (Eventstore)	[Prog09]
TIBCO Software Inc.	Tibco Business Events™	UML basierte Modellierung	Windows, Linux, HP-UX, Solaris	*Keine Angabe*	Supply Chain Management, Logistik, Sicherheit	Tendiert in Richtung Predictive Business	[Tibc09]
Stream Base Inc.	Stream Base Event Processing Platform	StreamBase Studio, Stream SQL, Stream Base Server	Linux, Windows, Solaris	500.000 events/sek.	Algorithmic Trading, Finanzanwendungen, Telekommunikation, Online Gaming	Sowohl graphische als auch programmiertechnische Entwicklung möglich, Java Schnittstelle integriert	[Stre09]
Coral8 Inc.	Continuous Intelligence™	CCL (Continuous-Computation Language), basiert auf SQL	Linux, Windows	400.000 events/sek.	Kundenmanagement, Telekommunikation, RFID	CEP Startup Company, Gründung im Jahr 2003, Kooperation mit Aleri Inc.	[Cora09]
Aleri Inc.	Aleri Streaming Platform	Aleri Studio (Eclipse basiert) mit Aleri Splash als EPL	Linux, Windows, Solaris	mehrere 100.000 events/sek.	Algorithmic Trading, Risikomanagement	Excelschnittstelle für Auswertungen konfigurierbar	[Aler09]
Agent Logic Inc.	RulePoint ®	Webbasiertes Entwicklungstool	Linux, Windows, Solaris	*Keine Angabe*	Gesamte operative Unternehmenssteuerung	Keine Clientinstallation notwendig	[Agen09]

Tabelle 5: Lösungen der bekanntesten CEP-Hersteller

Weitere Lösungen werden z.B. von den großen Softwarefirmen wie `Oracle Corp.` oder `IBM Corp.` angeboten. Für eine konkrete Auflistung der aktuellen CEP-Hersteller, wie zum Beispiel das deutsche Unternehmen RTM (Realtime Monitoring), siehe [Gild09]. Für weitere Informationen zu den Herstellern und deren Lösungen siehe [Gual09]. Insgesamt ist lt. [Gual09] zu erkennen, dass sich der Markt für CEP-Technologie von einem Nischen- zu einem *Mainstream*-Markt wandelt.
Aufgrund dieser Zahlen an verarbeiteten *events* pro Sekunde aus Tabelle 5 bezeichnen die Hersteller ihre CEP *engines* als echtzeitfähig. Für die Umsetzung des Ansatzes dieser Arbeit wird die Lösung *StreamBase Studio* des Herstellers `StreamBase Inc.` als CEP *engine* verwendet, da hierfür sowohl eine kostenlose Lizenz zur Verfügung steht als auch nach Aussage von [Nguy07, S. 10] mit dieser Lösung eine leistungsstarke Eventverarbeitung möglich ist.
Zusammenfassend ist zu erwähnen, dass CEP eine relativ junge Technologie ist, die durch die genannten Anwendungsgebiete großes Entwicklungspotenzial in sich birgt. Nachfolgend wird im letzten allgemeinen Kapitel dieser Arbeit der Stand der Forschung im Rahmen von Betrugserkennung im Bereich von CEP und maschinellen Lernverfahren im Bankenumfeld diskutiert.

5 State of the art im Bereich Betrugserkennung mit CEP und im Bankenumfeld

Die ausgewertete Literatur mit ähnlichen Entwicklungen im Vergleich zu dem Ansatz dieser Arbeit betrifft zum einen den Bereich Betrugserkennung im Umfeld von CEP. Da CEP eine neue Technologie ist, werden in diesem Kapitel alle Artikel, die allgemein über das Thema Betrugserkennung (unabhängig von der Betrugsart) in Zusammenhang mit dieser Technik veröffentlicht wurden, diskutiert.
Eine weitere Art von relevanten Publikationen bezieht sich auf den Bereich Erkennung und Prävention von Identitätsdiebstahl beim Online-Banking mit Hilfe maschineller Lernverfahren. Um den Fokus gezielt auf diese Fälle zu richten, werden hierbei keine Publikationen mit den Themen Analyse von Kreditkartentransaktionen oder Benutzeridentifikation beim Online-Banking mittels bestimmter Protokolle bzw. Authentifizierungsmethoden diskutiert.
Zusätzlich werden in diesem Kapitel Artikel über den kombinierten Einsatz von Diskriminanzanalyse und neuronalen Netzwerken als Hybrid-Modell diskutiert um herauszufinden, ob das Verfahren der Übergabe von Diskriminanzwerten an ein neuronales Netzwerk bereits in der Literatur verwendet wird.

5.1 Betrugserkennung im Umfeld von CEP

Ein Beispiel eines regelbasierten Systems für Betrugserkennung liefert die Arbeit von [Rozs07]. Die Autoren entwickelten ein eventbasiertes Betrugserkennungssystem namens SARI für den Bereich Online-Wetten. Die Abkürzung steht für *Sense and Respond Infrastructure*. Die Architektur von SARI besteht aus *Adapter*, welche die *events* wie z.B. *betplaced* oder *sporteventstarted* aus den *Operational Systems* wie *Betting Engine* oder *Financial System* einlesen. Die Analyse der *events* und deren Zusammenhänge erfolgt mit Hilfe von *Event Processing Maps*. Die *Event Processing Maps* bilden ein Regelwerk zur Klassifizierung von Betrugsmustern wie z.B. Platzierung von Wetten mit ungewöhnlich hohen Einsätzen auf einen sportlichen Außenseiter oder Überweisungen aus verdächtigen Quellen. Die *Event Processing Maps* können vom Anwender durch ein *User Interface* graphisch modelliert werden. Die Speicherung der Regeln, *events* und deren Strukturen ist die Aufgabe der *Event-Base*. Für nachhaltige Auswertungen mittels *Data Mining*-Technologie dient das *Business Intelligence* (BI) Werkzeug *Event Tunnel*. Dort werden kausale Zusammenhänge und Beziehungsmuster zwischen den *events* identifiziert. Die gewonnenen Erkenntnisse können anschließend vom Benutzer zu neuen *Event Processing Maps* umgesetzt werden. Für weiterführende Informationen über das BI-Werkzeug *Event Tunnel*, siehe [Sunt08]. In [Sunt08] wird ebenfalls der in [Rosz07] beschriebene Anwendungsfall der Betrugserkennung im Bereich Online-Wetten und der Zusammenhang bzw. die Visualisierung der Betrugsevents mit *Event Tunnel* diskutiert. Darüber hinaus besteht SARI aus einer Komponente namens *Real-Time Management Cockpit*, die beispielsweise dem Benutzer ein Sportereignis anzeigt, bei dem betrugsverdächtige Wetten entdeckt wurden. Die Architektur von SARI ist in Abbildung 29 angezeigt.

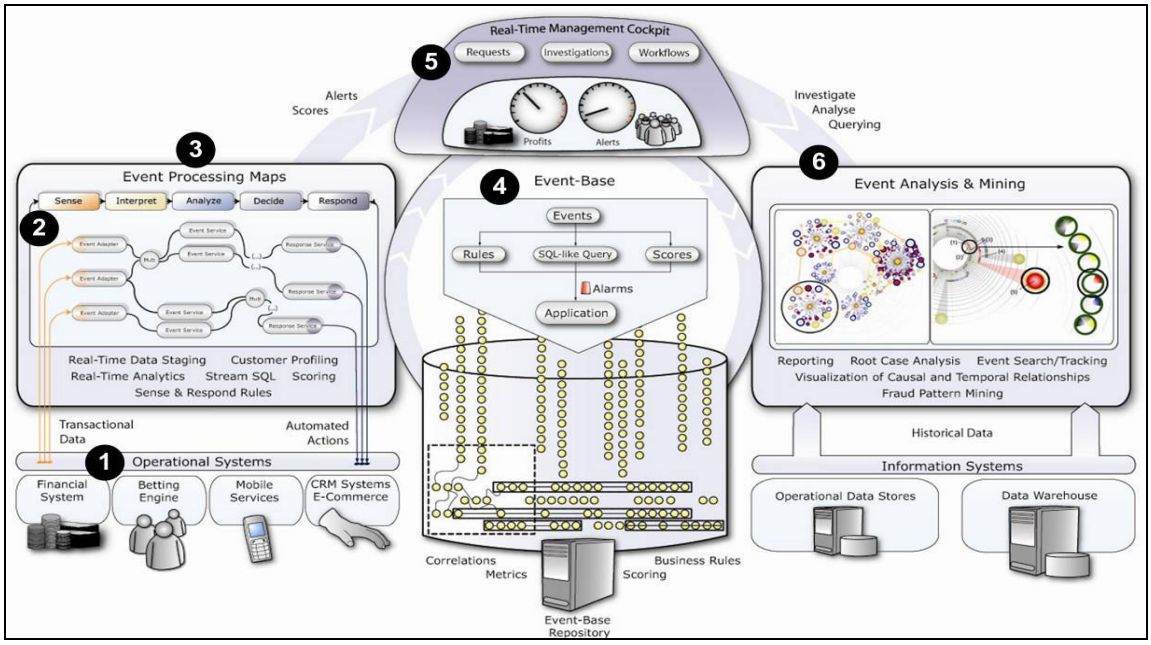

Abbildung 29: Betrugserkennungsarchitektur von SARI in [Rozs07, S. 3]

Im Gegensatz zu der hier vorliegenden Arbeit wird bei [Rozs07] ein Ansatz bestehend aus fixen Regeln verwendet, der bekannte Betrugsmuster im Bereich Online-Wetten abprüft. Das bedeutet, es wird keine Echtzeitüberprüfung der *events* mittels maschineller Lernverfahren durchgeführt, sondern es erfolgt in [Rozs07] ein Vergleich mit bekannten Mustern, die in den *Event Processing Maps* hinterlegt sind. Neue Betrugsmuster werden dabei *ex post* mittels *Data Mining*-Techniken in dem Werkzeug *Event Tunnel* identifiziert.

Eine weitere Betrugserkennungsarchitektur im *Event Processing*-Umfeld beinhaltet die Arbeit von [Nguy07]. Dieser Artikel beschreibt einen Forschungsprototyp namens ZELESSA. Die Abkürzung steht für *Zero Latency Event Sensing and Responding Architecture.* Diese Architektur dient als Infrastruktur für die Generierung von echtzeitfähigen BI-Anwendungen auf Basis eines *Data Warehouse* und der CEP *engine* des Herstellers `StreamBase Inc.`, die nach Aussage der Autoren eine sehr leistungsfähige Eventverarbeitung garantiert [Nguy07, S. 10]. Ein bereits implementierter Anwendungsfall von ZELESSA ist die Identifikation von Betrugsfällen im Umfeld Mobiltelefonie auf Basis von *Event Streams*. In dieser Anwendung werden *phonecall events* auf Basis vordefinierter Regeln analysiert, die in XML implementiert sind. Ein typisches *event pattern* in diesem Bereich lautet: Wenn ein Anruf mit einem Mobiltelefon von Österreich nach Asien acht Minuten dauert, wird dieser Anruf nicht als Betrug klassifiziert, wenn die durchschnittliche Anrufdauer der letzten drei Monate nach Asien mit diesem Mobiltelefon 15 Minuten beträgt. Überschreitet ein solcher Anruf aber einen bestimmten definierten Grenzwert (z.B. 60 Minuten), wird er als Betrugsversuch gekennzeichnet und abgebrochen. Das System

überwacht zu diesem Zweck die *events phonecallstarted* und *phonecallhangup* bzw. die Zeitspanne nach dem Eintreten des *phonecallstarted events*. Im Falle des Abbruchs des Anrufs durch das System tritt das *phonecallhangup event* nicht ein. Der ZELESSA Prototyp ist mit der Entwicklungsumgebung Visual Studio .NET 2005 programmiert und basiert auf einer MS SQL Server 2005-Datenbank. Ein zentraler Server (hier *Dispatcher Host*) empfängt die *events* aus unterschiedlichen heterogenen Quellen und übergibt sie dem *Event Processing Model* (EPM), das in Abbildung 30 dargestellt ist.

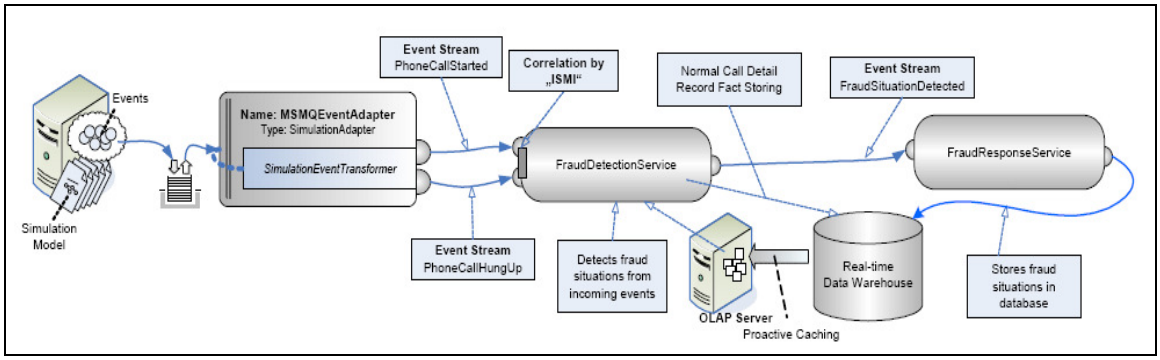

Abbildung 30: Event Processing Model von ZELESSA in [Nguy07, S. 7]

Über den *MSMQEventAdapter* gelangen die Ströme der *phonecallstarted* und *phonecallhangup events* in den zentralen Kern der Architektur, dem *FraudDetectionService*. Hier wird auf Basis der historischen Anrufdaten des dazugehörigen Kunden für jedes *phonecallstarted event* ein Grenzwert ermittelt, nach dem das Gespräch als Betrugsversuch gewertet wird. Falls innerhalb dieses ermittelten kundenspezifischen Anrufzeitraums kein *phonecallhangup event* eintritt, erzeugt der *FraudDetectionService* ein *fraudsituationdetected event* für den *FraudResponseService*, der das Gespräch automatisch abbricht. Die Betrugssituation wird innerhalb des *Real-time Data Warehouse* gespeichert. Aus diesem *Data Warehouse* bezieht wiederum der *FraudDetectionService* die Datengrundlage für die Ermittlung des Grenzwerts für die Anrufdauer eines Kunden. Für diesen Forschungsprototyp werden die *events* von einem Simulator erzeugt.

Im Vergleich zu der beschriebenen Lösung dieser Arbeit wird bei [Nguy07] ebenfalls auf Basis historischer *events* gearbeitet und die Lösung des Herstellers `StreamBase Inc.` als CEP *engine* verwendet. Allerdings ist hierbei ein regelbasierter Ansatz für die Betrugserkennung im Bereich Mobiltelefonie gewählt.

In einem früheren Artikel hat die gleiche Autorengruppe eine Vorgängerversion von ZELESSA mit der Bezeichnung SARESA vorgestellt, siehe [Nguy05]. SARESA steht für *Sense and Response Service Architecture* und beinhaltet mit Betrugserkennung im Umfeld Mobiltelefonie den gleichen implementierten Anwendungsfall mit einem exakt über-

einstimmenden *Event Processing Model* wie in Abbildung 30 abgebildet ist. Allerdings sind in dem Vorgängermodell die *event patterns* noch nicht so konkret definiert.

Eine weitere CEP-Architektur für Betrugserkennung ist in [Bass06] beschrieben. Das entwickelte *Joint Directors of Laboratories* (JDL) Modell ist in Abbildung 31 dargestellt.

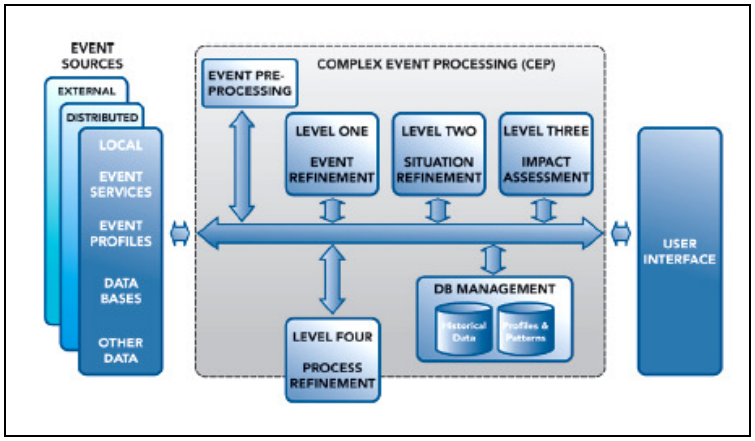

Abbildung 31: JDL Complex Event Processing Modell in [Bass06]

Das JDL-Modell beschreibt den Prozess des *Event Processing* (EP) in einem festgelegten Ablauf mit folgenden Komponenten:

Schritt 1 – *Event Sources*:
Die *events* stammen aus unterschiedlichen externen und internen Quellen, welche die globale *Event Cloud* einer Organisation bilden.

Schritt 2 – *Event Preprocessing*:
Die *events* aus heterogenen Quellen werden mittels *Adapter* normalisiert, so dass im Anschluss alle *events* eine einheitliche Struktur für die weitere Verarbeitung aufweisen.

Schritt 3 – *Level One Event Refinement*:
Die *events* werden in Abhängigkeit vom jeweiligen Anwendungsfall (hier Betrugserkennung) gefiltert. Die *events*, die in diesem Zusammenhang nach der Definition nicht bzw. nur mit einer geringen Wahrscheinlichkeit relevant für die Betrugserkennung sind, werden aus dem *Event Stream* entfernt.

Schritt 4 – *Level Two Situation Refinement*:
In diesem Schritt wird die Betrugsrelevanz der *events* analysiert. Auf Basis historischer und aktueller *events* aus dem *Event Stream* werden Betrugssituationen identifiziert bzw.

in Echtzeit korreliert. Die Identifizierung erfolgt mit Hilfe eines Bayes-Netzwerks, indem das gesammelte Wissen über Ursache- und Wirkungsbeziehungen von Phishingevents repräsentiert ist. Abbildung 32 zeigt das verwendete Bayes-Netzwerk.

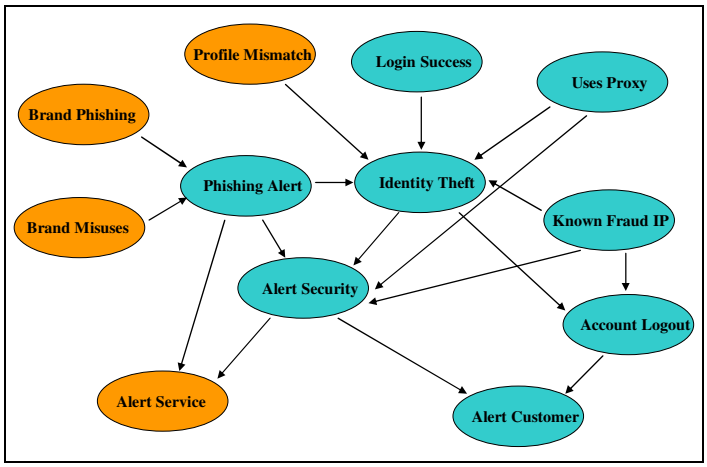

Abbildung 32: Verwendetes Bayes-Netzwerk zur Betrugserkennung in [Bass06]

Die hellbraun gezeichneten Knoten in Abbildung 32 zeigen *events* aus externen Eventquellen, die blauen dagegen deuten auf interne Quellen hin. Die Pfeile repräsentieren die kausalen Verbindungen zwischen den Knoten. Die genauen Wahrscheinlichkeiten der einzelnen Knotenverbindungen sind in dem Artikel nicht genannt.

Schritt 5 – *Level Three Impact Assessment*:
Die Auswirkungen der identifizierten Situationen auf die Organisation in Form von Schäden, Gewinnen oder Verlusten wird geschätzt.

Schritt 6 – *Level Four Process Refinement*:
Die gewonnenen Informationen aus *Level Three* werden verwendet um die Prozesse und die bekannten Muster zu verfeinern.

Schritt 7 – *Database Management*:
Die historischen *events* für *Level Two* sowie die bekannten Muster werden in der Datenbank gespeichert.

Schritt 8 – *User Interface*:
Das *User Interface* wird zur Visualisierung des *Event Processing*-Prozesses und der Betrugserkennungsergebnisse verwendet.

Das JDL-Modell ist nach Aussage von [Bass06] Teil der CEP-Lösung des Herstellers TIBCO Software Inc. namens *BusinessEvents*. Konkrete Instanzen der beschriebenen Komponenten sind als sog. *ScoreCards* implementiert, die als Container für lokale Parametereinstellungen fungieren.
Dieser Artikel beschreibt eine prozessorientierte CEP-Architektur, die ebenfalls auf historische *events* zurückgreift. Im Gegensatz zum Ansatz dieser Arbeit basiert die Betrugserkennung in [Bass06] auf einem Bayes-Netzwerk. Darüber hinaus werden in [Bass06] keine Transaktionsevents untersucht, sondern *login events* oder *logout events*, die z.B. auf einen Einbruchsversuch hindeuten.

Eine weitere Publikation im CEP-Umfeld mit dem Anwendungsfall Identitätsdiebstahl stammt von D. Luckham, dem Begründer der CEP-Technologie. In [Luck04d] beschreibt der Autor keine konkrete CEP-Lösung zur Betrugserkennung, macht aber auf die Problematik des wachsenden Identitätsdiebstahls aufmerksam und zeigt auf, dass dieses Thema ein wichtiger Anwendungsfall für *Complex Event Processing* ist. In diesem Artikel beschreibt D. Luckham ein typisches *event pattern* aus einem Phishingfall auf der Internetseite einer Bank mit der Sequenz *account login*, gefolgt von *password change*, und anschließendem *new auto pay*. Als Lösung schlägt [Luck04d] vor, dieses Muster mittels *Business Activity Monitoring*-Technologie (BAM) zu überwachen (die Kombination von BAM- und CEP-Technologie wird ebenfalls in [Gual09] und [Chan09, S. 172] als Einsatzszenario im CEP-Umfeld erwähnt). Der Autor bemerkt in seinem Artikel außerdem, dass ca. 1% aller Bankkunden alleine durch grob fahrlässiges und leichtsinniges Handeln stark gefährdet sind, Opfer von Identitätsbetrügern zu werden.
In [Luck04e] benutzt D. Luckham Identitätsdiebstahl beim Online-Banking als Beispiel zur Beschreibung der z.B. in Kreditinstituten vorherrschenden *IT-Blindness* (siehe Abschnitt 4.4) indem er anführt, dass eine Bank keine Kenntnis hat, ob ein Kunde Opfer von Identitätsbetrug wurde bis es der Kunde selbst der Bank meldet.

Eine ähnliche Argumentation wie [Luck04d] verwendet A. Lundberg in [Lund06]. Auch dieser Artikel stellt keine konkrete Betrugserkennungslösung im *Event Processing*-Umfeld vor, erwähnt aber, dass durch den Einsatz von CEP im Bereich Online-Banking betrügerische Transaktionen bzw. Anomalien in Echtzeit identifiziert und sofort über weitere *events* bzw. ausgelöste Aktionen entsprechend reagiert werden kann.

5.2 Maschinelle Lernverfahren zur Erkennung und Prävention von Identitätsdiebstahl im Online-Banking

Dieser Abschnitt unterscheidet die diskutierten Ansätze zur Bekämpfung von Identitätsdiebstahl beim Online-Banking mit Hilfe von maschinellen Lernverfahren in Methoden zur Betrugsprävention und Methoden zur Betrugserkennung.

5.2.1 Maschinelle Lernverfahren zur Prävention von Identitätsdiebstahl

In einem Artikel von [Abu07] zur Prävention von Identitätsdiebstahl beim Online-Banking werden Emails analysiert um herauszufinden, ob es sich um Phishing-Emails handelt oder nicht. Zu diesem Zweck vergleichen die Autoren die Erkennungsleistung von maschinellen Lernverfahren nach den Faktoren *false positive* und *false negative*. Der Begriff *false negative* bedeutet im Rahmen der Arbeit von [Abu07], dass eine Phishing-Email fälschlicherweise als Nicht-Phishingfall klassifiziert wurde. Bei *false positive*-Bewertung ist die Situation umgekehrt. Bei den verwendeten maschinellen Lernverfahren in [Abu07] handelt es sich um:

- *Logistic Regression* (LR) (siehe Abschnitt 3.6)
- *Classification and Regression Trees* (CART) (siehe Abschnitt 3.1)
- *Bayesian Additive Regression Trees* (BART) (BART ist ein Bayesscher Regressionsansatz, der auf dynamisch anpassbaren Elementen basiert. Für weiterführende Literatur zu *Bayesian Additive Regression Trees* siehe [Chip06])
- *Support Vector Machine* (SVM) (siehe Abschnitt 3.4)
- *Random Forests* (RF) (RF ist ein Entscheidungsbaumalgorithmus, der auf vielen Entscheidungsbäumen basiert und als Ergebnis den Wert ausgibt, der am häufigsten eine Trainingsmenge über individuelle Bäume hinweg richtig klassifiziert. Für weiterführende Literatur zu *Random Forests* siehe [Pavl00])
- *Neural Networks* (NNet) (siehe Abschnitt 3.3)

Diese genannten Verfahren gehören zur Familie der überwachten Lernverfahren und wurden auf eine Datenbasis von insgesamt 2.889 Emails angewandt. Diese Menge ist in 1.171 Phishing-Emails und 1.718 Nicht-Phishing (= *legitimate*) Emails aufgeteilt. Die Autoren von [Abu07] identifizierten in den Emails insgesamt 43 Variablen, die zur Identifikation von Phishing-Emails verwendet wurden. Diese Variablen repräsentieren die Häufigkeiten des Auftretens bestimmter englischer Begriffe im Emailtext sowie im Betreff. Die Variablen

bilden zusammen einen Vektor, der jeweils eine Email darstellt. Zusätzlich existiert zu jeder Email eine Zielvariable, welche die boolesche Information enthält, ob diese Email eine Phishing-Email ist oder nicht.
Das neuronale Netzwerk in [Abu07] besteht aus insgesamt drei Schichten, wobei die Hiddenschicht insgesamt zehn Knoten aufweist, bei einem Output- und 43 Inputknoten (d.h. ein Knoten pro Variable). Trainiert wurde das Netzwerk mit einem Lernfaktor von 0,1, wobei die Autoren keine Aussagen über die Aufteilung der Trainings- und Testmenge und der Anzahl der Lerndurchläufe getroffen haben.
Tabelle 6 enthält nach Abschluss der Analysen mit den erwähnten Verfahren die folgenden Ergebnisse.

	false positive	false negative
LR	4,89%	17,04%
CART	7,68%	12,93%
SVM	7,92%	17,26%
NNet	5,85%	21,72%
BART	5,82%	18,92%
RF	8,29%	11,12%

Tabelle 6: Identifikationsergebnisse verschiedener maschineller Lernverfahren aus [Abu07, S. 7]

Tabelle 6 zeigt, dass auf Basis der analysierten Emails bezüglich der *false positive*-Rate *Logistic Regression* (LR) die optimale Methode ist und im Gegensatz dazu bei der *false negative*-Quote *Random Forests* (RF) das beste Resultat erzielt. Das neuronale Netzwerk schneidet bei [Abu07] im Vergleich zu den anderen Verfahren bei der *false negative*-Quote mit einem Wert von 21,72% am schlechtesten ab, erzielt aber bei der *false positive*-Rate mit einem Anteil von 5,85% ein vergleichsweise gutes Ergebnis. Allerdings weist bei [Abu07] das neuronale Netzwerk mit 43 Inputparametern eine größere Zahl auf als im Rahmen dieser Arbeit, wobei in dem diskutierten Artikel das neuronale Netzwerk nur mit einer Hiddenschicht getestet wurde.

Eine ähnliche Untersuchung wie [Abu07] führten die Autoren von [Chan06] durch. Sie analysierten in ihrer Arbeit 200 Emails, die aus 100 Phishing- und 100 Nicht-Phishingfällen zusammengesetzt waren. Als Ergebnis identifizierten sie 25 relevante Attribute, von denen ein Teil – genau wie bei [Abu07] – aus den Häufigkeiten bestimmter Wörter im Text und Betreff bestehen. Zusätzlich verwendeten die Autoren Variablen, die Informationen über die Struktur des Betreffs und des Emailtextes beinhalten. Als maschinel-

les Lernverfahren setzten die Autoren von [Chan06] *Support Vector Machine* (siehe Abschnitt 3.4) ein. Sie erreichten dabei ein Resultat von 95,0% richtig klassifizierter Emails.

Eine weitere Arbeit in dieser Richtung wurde von [Fett07] durchgeführt. Diese Autoren verwendeten zehn relevante Attribute, die ebenfalls auf den Häufigkeiten des Vorhandenseins bestimmter englischer Wörter im Emailtext basieren. In [Fett07] wurden 860 Phishing-Emails und 6.950 Nicht-Phishingfälle analyiert. Als maschinelles Lernverfahren implementierten die Autoren einen selbstentwickelten Algorithmus namens PILFER, der auf *Random Forests* (siehe oben) basiert. Als Ergebnis wurde eine Klassifikationsgenauigkeit von 96,0% richtig klassifizierter Emails erreicht, bei einer *false positive*-Quote von 0,1%. Nach Aussage der Autoren kann dieser Algorithmus auch zum Identifizieren von Phishingseiten verwendet werden.
Im Vergleich zum Ansatz dieser Arbeit verwenden die Autoren von [Abu07], [Chan06] und [Fett07] keine Hybrid-Architektur, sondern vergleichen die Ergebnisse verschiedener überwachter maschineller Lernverfahren miteinander bzw. benutzen alleingestellte Analysemethoden. In diesem Zusammenhang setzen die drei Arbeiten für die Betrugsbekämpfung nicht bei den Betrugstransaktionen (d.h. Betrugserkennung) sondern einen Schritt vorher bei den Phishing-Emails (d.h. Betrugsprävention) an. Auch wird in keinem der Artikel CEP-Technologie eingesetzt oder erwähnt.

Mit den Emails sollen die Phishingopfer zu den Betrugsseiten gelockt werden. Zur Erreichung des Ziels, diese Betrugsseiten von legitimen Internetseiten einer Bank unterscheiden zu können, entwickelten die Autoren von [Medv08] eine Methode zur Ähnlichkeitsanalyse von falschen und echten Webseiten. Folgende Grundkomponenten von Internetseiten werden bei dem Verfahren verglichen:

- Textfragmente
- Eingebettete Bilder
- Komplettes visuelles Erscheinungsbild einer Internetseite

Bei der Ähnlichkeitsanalyse wird in folgenden Schritten vorgegangen:

Schritt 1 – Auffinden einer verdächtigen Internetseite w:
Hierbei wird der Analyseprozess gestartet, sobald eine verdächtige Internetseite registriert wird.

Schritt 2 – Berechnen der Signaturwerte $S(w)$ der verdächtigen Internetseite:
Eine Signatur ist im Rahmen der Arbeit von [Medv08] ein quantifizierter Wert aus Text- und Bildinformationen einer Internetseite. Der Signaturwert für Textstellen setzt sich aus den Einzelkomponenten Schriftfarbe, Hintergrundfarbe, Textinhalt, Schriftgröße, Schriftname und der Position innerhalb der Seite (ausgehend vom linken oberen Pixel des Textes) zusammen. Bei Bildern dagegen definiert sich der Signaturwert aus der Quellenadresse des Bildes, der Höhe, der Breite, der Farbe sowie der *2D Haar wavelet transformation* (*2D Haar wavelet transformation* ist eine effiziente, gering-auflösende Bildanalysetechnik, für weiterführende Informationen siehe [Stan03]). Der Signaturwert des kompletten visuellen Erscheinungsbildes besteht aus den Einzelkomponenten Farbe und *2D Haar wavelet transformation*.

Schritt 3 – Vergleich der Signaturwerte $S(w)$ der verdächtigen Internetseite mit den Signaturwerten $S(w')$ der legitimen Internetseite:
Bei diesem Schritt werden gleiche Einzelkomponenten von zwei Internetseiten miteinander verglichen. Bei Textelementen wird zu diesem Zweck die Levenshtein-Distanz berechnet (die Levenshtein-Distanz ist ein Maß, das die minimal notwendige Anzahl der Operationen Einfügen, Löschen und Ersetzen ermittelt um eine Zeichenkette in die andere zu überführen, für weiterführende Literatur siehe [Leve66]). Farbunterschiede werden mittels der 1-Norm-Distanz ermittelt, bei Positionsunterschieden dagegen wird die Euklidische-Distanz verwendet (die 1-Norm-Distanz bezeichnet die tatsächliche Entfernung, die innerhalb einer quadratischen Blockstruktur zurückgelegt werden muss um von einem Ausgangsort zu einem Zielort zu gelangen, z.B. die Distanz, die ein Taxi zurücklegt um in Manhattan von A nach B zu kommen, daher auch der Name Manhattan-Distanz. Die Euklidsche-Distanz dagegen, oder auch 2-Norm-Distanz genannt, ist der Abstand zweier Punkte innerhalb eines Raumes. Für weiterführende Literatur zu den genannten Distanzen siehe [Deza09]). Für die drei Grundvergleichskomponenten Text, Bilder und visuelles Erscheinungsbild wird der Signaturwert jeweils durch Addition der Distanzen der Einzelkomponenten berechnet.

Schritt 4 – Ausgabe einer Alarmmeldung falls die Signaturwerte zu ähnlich sind:
Die Signaturwerte sind zu ähnlich, wenn der Scorewert s, der aus einer Linearkombination der Signaturwerte der drei Grundvergleichskomponenten ermittelt wird, einen bestimmten Grenzwert t überschreitet ($s>t$). Der Scorewert s wird mit folgender Formel berechnet:

$$s = a^t * s^t + a^i * s^i + a^o * s^o$$

a^t = Koeffizient für Textfragmente
s^t = Signaturwert für Textfragmente
a^i = Koeffizient für eingebettete Bilder
s^i = Signaturwert für eingebettete Bilder
a^o = Koeffizient für das visuelle Erscheinungsbild
s^o = Signaturwert für das visuelle Erscheinungsbild

Falls diese Situation $s>t$ eintritt, wird die verdächtige Seite registriert und eine definierte Alarmmeldung gesendet.
Die Koeffizienten der Funktion werden mit Hilfe einer Trainingsmenge von 35 Internetseitenpaaren, wobei bei 14 Paaren eine Phishingseite enthalten ist, ermittelt. Die Berechnung der Koeffizienten erfolgt unter Verwendung des Simplex-Verfahrens (Simplex ist eine Methode der Numerik zur Lösung linearer Optimierungsprobleme, für weiterführende Literatur siehe [Klee72]).
In dem Artikel sind sowohl die exakten Werte der Koeffizienten als auch der Grenzwert nicht angegeben. Die Autoren von [Medv08] führten zur Validierung ihres Ansatzes Experimente mit einer Testmenge von 41 Paaren aus realen Phishing- und legitimen Internetseiten durch. Dabei kamen sie zu dem Ergebnis, dass zwei Phishingseiten nicht als solche erkannt wurden und keine legitime Seite als *false positive* klassifiziert wurde.
Im Vergleich zum Ansatz dieser Arbeit setzen die Autoren von [Medv08] nicht auf Betrugserkennung sondern auf Betrugsprävention. Dabei wird in [Medv08] ein selbstentwickeltes Verfahren verwendet, das auf dem Simplex-Algorithmus zur Generierung der Klassifikationsfunktion mit Hilfe einer Trainingsmenge basiert. Die Klassifikation erfolgt anschließend – ähnlich wie bei der Diskriminanzanalyse – auf Basis eines Vergleichs des ermittelten Funktionswerts mit einem zuvor berechneten Grenzwert. Allerdings wird bei [Medv08] weder ein neuronales Netzwerk noch ein Entscheidungsbaum zur Analyse benötigt, da die Basisattribute mit einer Kombination aus Distanzwerten anders strukturiert sind als im Rahmen dieser Arbeit. Des Weiteren wird bei [Medv08] die Verwendung von CEP-Technologie nicht genannt, wobei in diesem Artikel ein Anspruch auf Echtzeitfähigkeit des Verfahrens ebenfalls nicht erwähnt wird. Für weitere Vorgängerarbeiten zum Thema Ähnlichkeitsanalyse von Phishing- und legitimen Internetseiten mittels entsprechender Distanzberechnungen, siehe [Rosi07], [Fu06], [Weny06] und [Weny05].

5.2.2 Maschinelle Lernverfahren zur Erkennung von Identitätsdiebstahl

Einen Ansatz zur Betrugserkennung mittels maschineller Lernverfahren verwenden die Autoren von [Vikr04]. In dieser Arbeit werden neuronale Netzwerke benutzt, die Informationen aus einer Vielzahl von Datenquellen untersuchen um verdächtige Kontoaktivitäten zu identifizieren. In diesem Zusammenhang erwähnen die Autoren, dass Banken oftmals nur Betrugsuntersuchungen durchführen, wenn eine verdächtige Aktivität gemeldet wird. Es erfolgt in vielen Kreditinstituten keine Echtzeitanalyse der Transaktionen, was auch mit den Aussagen aus den geführten Interviews (siehe Anhang 1) übereinstimmt. Die Lösung in [Vikr04] basiert auf der Annahme, dass eine Verbindung zwischen den *Predictor Variables* (deut.: Wirkungsvariablen, wie z.B. Transaktions- und Zugriffsaktivität auf das Konto) und den *Predicted Variables* (deut.: erwartete Variablen, wie z.B. Betrugsrisiko und Anteil ungewöhnlicher Aktivitäten des Kontos) besteht. Neuronale Netzwerke werden bei [Vikr04] eingesetzt, um auf Basis von trainierten Mustern den Risikograd von Finanztransaktionen zu bewerten. Für diesen Zweck identifizierten die Autoren folgende Attribute als relevant:

- Kontoinformation (Kontoinhaber)
- Quelle bei Kontoänderungen (z.B. Kunde, Mitarbeiter, Administrator)
- Netzwerkinformationen über die Quelle der Transaktion (IP-Adresse, Routing Informationen)
- Empfängerinformationen (Zielkonto, Vertraulichkeit des Empfängers, geographische Angaben zur Zielbank)

Aufgrund der Tatsache, dass diese Informationen auf verschiedene Quellen innerhalb einer Finanzorganisation verteilt sind, stellte sich für die Autoren die Frage nach der Vereinigung der verteilten Informationen für die Analyse. Die Antwort darauf bildeten intelligente Agenten für jede Architekturplattform, die alle mit der gleichen Datenbank namens *Central Activity Log Database* (CALDB) kommunizieren (ein Agent ist in diesem Zusammenhang ist ein Prozess, der lokal Aktionen durchführt und mit der verwaltenden Einheit kommuniziert, siehe dazu [Kuro02, S. 615 - 618]).
Für die Analyse der gesammelten Daten entwickelten die Autoren eine Lösungsarchitektur namens *Comprehensive Account Activity Monitoring and Analysis Tool* (CAAMAT). Diese Architektur ist in Abbildung 33 dargestellt.

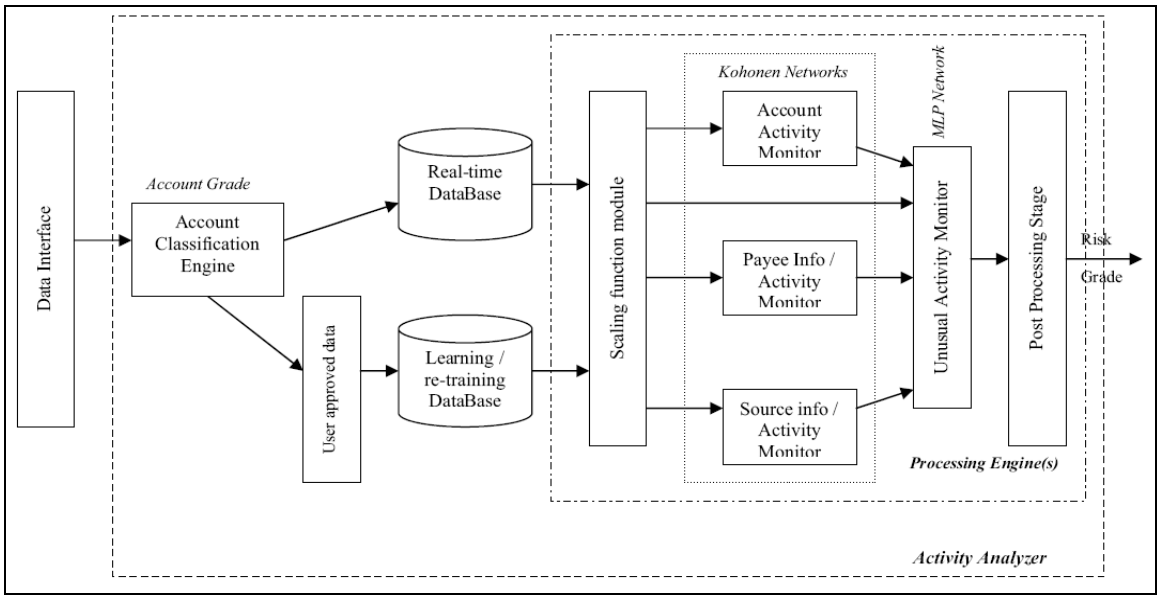

Abbildung 33: Lösungsarchitektur CAAMAT zur automatisierten Betrugserkennung aus [Vikr04, S. 10]

Das *Data Interface* bildet die Schnittstelle zur zentralen Datenbank CALDB, welche die zu analysierenden Daten aus den verschiedensten Quellen enthält. Die Hauptkomponente der Lösungsarchitektur bildet der *Activity Analyzer*. Die erste Komponente ist die *Account Classification Engine*, welche die bestehenden Konten dynamisch in verschiedene Kategorien (*Account Grade*) klassifiziert. Die Ermittlung des *Account Grade* erfolgt mittels folgender Formel:

Account Grade = Account Rating * Activity Level

Der Faktor *Account Rating* setzt sich zusammen aus einer Kombination von Attributen wie Initialrisiko, Einkommenslevel des Kontoinhabers usw., dagegen besteht der Faktor *Activity Level* aus historischen Daten des Kunden für eine bestimmten Zeitspanne, wie z.B. die Häufigkeit der Kontenzugriffe. Nach der *Account Classification Engine* folgen innerhalb der Architektur zwei Datenbanken. Die *Real-time Database* beinhaltet gegenwärtige Log-Informationen, die *Learning/Re-training Database* dagegen enthält historische Log-Informationen des Kunden, die auch zum Training der neuronalen Netzwerke verwendet werden. Diese Informationen fließen in die *Processing Engine*, welche die Anomalieanalyse der Log-Daten durchführt. Für jede ermittelte Ausprägung des *Account Grade* existiert eine eigene *Processing Engine*, die wiederum in Unterkomponenten aufgeteilt ist. Das *Scaling Function Module* wandelt die Log-Informationen in numerische Werte um, die anschließend in dieser Form von einem neuronalen Netzwerk analysiert werden können. Die folgenden Komponenten *Account Activity Monitor*, *Payee Info/Activity Monitor* und *Source Info/Activity Monitor* beinhalten Kohonen-Netzwerke (siehe Unterabschnitt 3.3.1).

Diese unüberwachten neuronalen Netzwerke besitzen je nach Komponente sowohl für das Training als auch für die Analyse zur Laufzeit folgende verschiedene Inputdaten:

a) *Account Activity Monitor*: Der *Account Activity Monitor* bekommt Daten über die abgelaufene Kontoaktivität übergeben.

b) *Payee Activity Monitor*: Der *Payee Activity Monitor* analysiert Daten über das Ziel bzw. die Empfänger der Transaktion.

c) *Source Activity Monitor*: Der *Source Activity Monitor* untersucht Daten über die Quelle der Transaktion wie z.B. die IP-Adresse.

Die Kohonen-Netzwerke bilden die Kernkomponente der *Processing Engine* und dienen dem Zweck, die Verbindung zwischen *predictor variables* und *predicted variables* herzustellen. Sie empfangen ihre Trainingsdaten aus der *Learning/Re-training Database* und haben die Aufgabe, die n-dimensionalen Inputdaten auf ein zweidimensionales Ausgabemuster zu reduzieren. Diese zweidimensionalen Muster bilden wiederum die Inputdaten für den *Unusual Activity Monitor*. Diese Komponente besteht aus einem *Multi Layer Perceptron* (siehe Unterabschnitt 3.3.1), d.h. einem überwachten neuronalen Netzwerk. Dieses *Multi Layer Perceptron* wird mittels des Backpropagationsverfahrens trainiert, wobei die Autoren keine konkrete Netzwerktopologie nennen. Nach Aussage der Autoren ist die Netzwerktopologie abhängig von der Kundenstruktur des Kreditinstituts, das die Lösung implementiert. In der *Preprocessing Stage* wird abschließend ein Scorewert auf Basis des Outputwerts des *Multi Layer Perceptrons* berechnet, der den Risikograd der analysierten Transaktion wiedergibt.
Nach Angaben der Autoren liegt die Stärke ihrer Lösungsarchitektur CAAMAT darin, verschiedene Arten von Informationen über die Transaktionen im Umfeld eines Finanzinstituts zentral auf alle Arten ungewöhnlicher, betrugsverdächtiger Aktivitäten zu untersuchen. Die Kohonen-Netzwerke wurden in ersten Tests mit 8.400 Transaktionen trainiert und aus den Ergebnissen die Muster für das *Multi Layer Perceptron* abgeleitet. Das Training benötigte eine Durchlaufzeit von 17 Sekunden.
Im Rahmen des Artikels von [Vikr04] wurde die komplette CAAMAT-Architektur noch nicht auf Performance und Zuverlässigkeit getestet. Nach Angaben der Autoren ist dies eine Aufgabe für die zukünftige Forschungsarbeit, wobei im Zuge der Recherchen im Rahmen dieser Arbeit bislang keine weiteren Publikationen dieser Autorengruppe gefunden wurden. Es findet sich in dem Artikel ebenfalls kein Hinweis, dass diese Architektur bereits im Bankenumfeld eingesetzt wird.

Genau wie im Rahmen dieser Arbeit wird bei [Vikr04] eine Hybrid-Architektur zur Identifikation von Betrugstransaktionen verwendet, allerdings basiert diese Architektur nicht auf einer Kombination von Diskriminanzanalyse, Entscheidungsbaum und neuronalem Netzwerk, sondern besteht aus einer Kombination von unüberwachten Kohonen-Netzwerken und einem überwachten *Multi Layer Perceptron*. Auch in diesem Artikel bildet der Output eines Vorgängerverfahrens den Input eines Nachfolgerverfahrens, aber nicht in Form einer Übergabe von Diskriminanzwerten. Darüber hinaus werden in der Arbeit von [Vikr04] keine *Event Processing*-Technologien, sondern Agentensysteme zur Datenversorgung verwendet.

In einem weiteren Artikel [Bign06] wird ein Framework zur Entwicklung einer Sicherheitsstrategie beim Online-Banking diskutiert. Der Autor definiert seine Arbeit in diesem Zusammenhang als Betrugserkennungsmethode und grenzt sie von bestehenden Betrugspräventionsmaßnahmen ab. Die vorgeschlagene Strategie basiert auf zwei Säulen in Form von Einbruchserkennung und Transaktionsüberwachung. Bei der Einbruchserkennung wird die Überprüfung der IP-Adresse und ein Abgleich mit der normal verwendeten IP-Adresse des Quellrechners des Kunden vorgeschlagen. Für diesen Vorgang existieren bekannte Muster wie z.B. wenn für einen Prozess im Rahmen des Online-Bankings eines Kunden eine IP-Adresse aus Australien verwendet wird, kann nicht fünf Stunden später für den gleichen Kunden eine IP-Adresse aus Brasilien als Quelle ermittelt werden. Wenn doch, spricht dies für einen Betrugsfall. Zusätzlich kann die normale Zeitdauer des Online-Bankings eines Kunden analysiert und mit der aktuell benötigten Zeit verglichen werden, wobei hier lt. [Bign06, S. 7] die Gefahr einer *false positive*-Klassifizierung des Zugriffs sehr groß ist. Bei der Transaktionsüberwachung werden in [Bign06, S. 2 - 3] überwachte Lernmethoden allgemein als die günstigsten Verfahren zur Betrugsanalyse im Online-Banking genannt, aufgrund ihrer Fähigkeit, das Transaktionsverhalten der Kontoinhaber zu lernen bzw. permanent mitzulernen. In diesem Kontext nennt der Autor speziell mehrschichtige neuronale Netzwerke mit Backpropagation als Lernverfahren, weil diese Methode in der Lage ist, große Datenmengen mit vielen Inputparametern – wie im Anwendungsfall der Transaktionsanalyse – zu lernen und zu verarbeiten. Als besonders relevante Attribute nennt [Bign06, S. 7] das Verhältnis vom Transaktionsbetrag zum maximal verfügbaren Betrag sowie den Empfänger der Transaktion. In diesem Zusammenhang wird für jeden Kunden ein Profil der vertrauensvollen Empfänger hinterlegt, zu denen auch bekannte Organisationen wie Telekommunikations- oder Energieversorgungsunternehmen gezählt werden. Für das Training werden bei [Bign06] sowohl Betrugstransaktionen als auch Nicht-Betrugstransaktionen verwendet.

Als Erweiterung der Sicherheitsstrategie macht der Autor u. a. folgende Vorschläge:

a) Einführung eines fixen Transferlimits für jede Online-Überweisung, z.B. 500 Dollar.

b) Keine Echtzeitbuchung der Transferbeträge durchführen, sondern eine Verzögerung von 12 bis 24 Stunden einbauen. Somit kann bei einem Betrugsverdacht einer Transaktion bei dem Kunden explizit nachgefragt werden, ob er diese Transaktion tätigen will.

c) Wenn ein Betrugsverdacht bei einer Transaktion besteht sollte das Konto für alle weiteren Transaktionen gesperrt werden, bis der Betrugsverdacht entkräftet werden kann.

In dem Artikel wird ebenfalls erwähnt, dass eine Diskrepanz zwischen dem Komfort des Kunden und den Sicherheitsaspekten besteht, wobei der Autor glaubt, dass diese Einschränkungen für die Mehrheit der Kunden akzeptabel sind, wenn dafür die Sicherheit beim Online-Banking erhöht wird.
In der Arbeit von [Bign06] werden – genau wie bei dem Ansatz dieser Arbeit – die Transaktionen beim Online-Banking analysiert. Es werden diesbezüglich auch bei [Bign06, S. 7] die Attribute vertrauensvoller Empfänger und das Verhältnis vom Transaktionsbetrag zum maximal verfügbaren Betrag als betrugsrelevant deklariert. Allerdings werden in diesem Artikel keine konkrete Implementierung und keine exakten Parameter (wie z.B. Netzwerktopologie des vorgeschlagen neuronalen Netzwerks) sowie keine experimentellen Ergebnisse beschrieben. Im Rahmen der Diskussion einer geplanten Implementierung des vorgeschlagenen neuronalen Netzwerks schreibt der Autor, dass er die Erfahrung gemacht hat, dass Banken keine Testdaten für akademische Zwecke zur Verfügung stellen [Bign06, S. 2]. Dieser Umstand führte ebenfalls im Rahmen dieser Arbeit dazu, dass Transaktionen simuliert wurden, anstatt Echtdaten zu verwenden. Im Gegensatz zu dem Hybrid-Modell dieser Arbeit fungiert in [Bign06] das neuronale Netzwerk als alleingestellte Betrugserkennungskomponente. Des Weiteren wird in diesem Artikel keine *Event Processing*-Technologie erwähnt bzw. im Rahmen der Sicherheitsstrategie diskutiert.

5.3 Kombination von Diskriminanzanalyse und neuronalen Netzwerken

Ein kombinierter Einsatz von Diskriminanzanalyse und neuronalen Netzwerken ist bei [Chen97] beschrieben. Der Artikel diskutiert Sprechererkennung sowie unabhängige Vokal-Identifikation. Für diese Anwendungsfälle entwickelten die Autoren eine baumstrukturartige Hybrid-Architektur, basierend auf dem Prinzip von *Divide and Conquer* (deut.: Teile und Herrsche). Dieses Prinzip besagt, dass „größere“ Probleme in „kleinere“ Probleme

zerlegt werden, die anschließend nacheinander oder parallel zueinander gelöst werden. Für weiterführende Informationen zu *Divide and Conquer* bzw. *Top Down*-Vorgehensweise bei der Problemlösung, siehe [Logo08, S. 145 - 167; Böhm02, S. 33 - 39].
In [Chen97] wird die Diskriminanzanalyse an den Knoten der Baumstruktur verwendet um „größere“ Probleme in „kleinere“ Probleme zu zerlegen. Hierbei wird an jedem Knoten eine *Splitting Rule* angewandt um eine Menge X in zwei Untermengen X_1 und X_2 aufzuteilen. In diesem Zusammenhang werden aus der Gruppe X die zwei Gruppen gebildet, deren Mittelwerte den maximalen Abstand zueinander aufweisen. Die einzelnen Elemente werden anschließend jeweils der Gruppe zugeteilt, zu deren Mittelwert der geringere Abstand (quadrierte Euklidische-Distanz) besteht. Zu diesem Vorgang siehe auch [Back06, S. 164 - 192].
An den Blättern bzw. Endknoten werden neuronale Netzwerke (*Feedforward*-Netzwerke mit drei Schichten) eingesetzt um die „kleineren“ Probleme zu lösen bzw. die an den Knoten erzeugten Untermengen X_1, X_2,…,X_n zu analysieren. Eine beispielhafte Darstellung der Architektur ist in Abbildung 34 dargestellt.

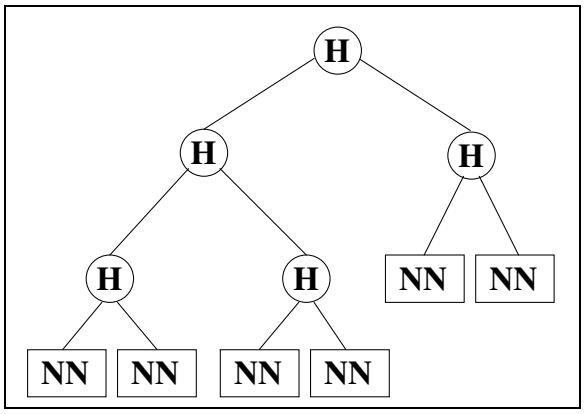

Abbildung 34: Baumstruktur mit Diskriminanzanalyse und neuronalen Netzwerken in der Arbeit von [Chen97, S. 24]

Das Trainieren der Baumstruktur erfolgt durch das Ausführen von *Constructive Learning Algorithm* (siehe [Pare00]) wie *Growing Algorithm* und *Credit Assignment Algorithm*. Der *Growing Algorithm* ist für die automatische Entwicklung der Baumstruktur verantwortlich. In diesem Zusammenhang werden sowohl die *Splitting Rule* als auch eine *Stopping Rule* rekursiv ausgeführt bis die vorher definierten *success*- oder *failure*-Bedingungen erfüllt sind. Das Training der neuronalen Netzwerke an den „Blättern“ erfolgt durch die Levenberg Marquat-Methode. Dieses Lernverfahren ist robust und konvergiert mit einer hohen Wahrscheinlichkeit auch bei schlechten Startbedingungen, für weiterführende Literatur siehe [Rutk08, S. 221 - 222; Mish05]. Das Ausgabeergebnis der Architektur (Identifizierung des Sprechers) wird durch Kombinieren der Ausgaben aller Netzwerkausgabeknoten

erzeugt, was durch den *Credit Assignment Algorithm* realisiert wird. Lt. Aussage der Autoren liefert die Aufteilung in kleinere Netzwerke exaktere Ergebnisse und geringere Trainingszeiten als der Einsatz eines *Multi Layer Perceptrons* zur Lösung des Gesamtproblems.
Im Vergleich zu dem Ansatz dieser Arbeit werden bei [Chen97] keine Diskriminanzwerte als Inputwerte der neuronalen Netzwerke verwendet, sondern die ursprünglichen Eingabewerte der Gesamtarchitektur. Die Diskriminanzanalyse wird an den Knoten der Baumstruktur zur Aufteilung der Gruppen in Untergruppen eingesetzt. Zusätzlich basiert die Arbeit von [Chen97] nicht auf *Event Processing*-Technologien. Es wird bei [Chen97] nicht mit *events* aus der IT-Landschaft einer Organisation gearbeitet, sondern mit den aufgezeichneten Klängen von Vokalen von 15 englisch sprechenden Testpersonen experimentiert.

Ein weiterer kombinierter Einsatz von Diskriminanzanalyse und neuronalen Netzwerken findet sich in der Arbeit von [Murc04]. Der Artikel beschreibt eine Methode zur Unterscheidung von aktiven und inaktiven antibakteriellen Molekülverbindungen auf Basis von topologischen Deskriptoren (topologische Deskriptoren sind einfache Integerwerte, die sich aus der Struktur von Molekülen ergeben. Sie werden benutzt um Molekülstruktur und biologische Aktivitäten zu korrelieren, für weiterführende Informationen siehe [Hu03]). Die Ziele der Forschungsarbeit von [Murc04] sind, neue antibakterielle Substanzen aus den Strukturen zu gewinnen und zwischen antibakteriellen und nicht-antibakteriellen Medikamenten unterscheiden zu können. Die eingesetzten Verfahren für diese Aufgabe sind Diskriminanzanalyse und neuronale Netzwerke. In ihrer Studie verwenden die Autoren 217 Moleküle mit bekannter antibakterieller Aktivität und 216 Verbindungen, bei denen keine antibakterielle Aktivität vorliegt. Die in [Murc04] entwickelte Methode für die Selektion neuer antibakterieller Verbindungen besteht aus dem folgenden mehrstufigen Prozess:

Schritt 1: Berechnung der topologischen Deskriptoren für jedes einzelne Molekül der Studie.

Schritt 2: Training eines neuronalen Netzwerks mit bekannten topologischen Deskriptoren mit gleichzeitiger Berechnung einer Diskriminanzfunktion für die Klassifikation von aktiven und inaktiven Verbindungen.

Schritt 3: Durchführung einer Datenbanksuche nach Strukturen mit möglicher antibakterieller Aktivität mit Hilfe der ermittelten Diskriminanzfunktion.

Schritt 4: Klassifizierung der Strukturen in aktiv und inaktiv durch Verwendung der Diskriminanzfunktion.

Schritt 5: Klassifizierung der Strukturen mit dem neuronalen Netzwerk zur Überprüfung der Klassifizierungsergebnisse der Diskriminanzfunktion.

Schritt 6: Durchführung von pharmalogischen und toxikologischen Tests zur Bestätigung der antibakteriellen Aktivität in den ausgewählten Verbindungen.

Das neuronale Netzwerk bildet in [Murc04] ein *Multi Layer Perceptron* mit insgesamt drei Schichten, das die Integerwerte der topologischen Deskriptoren als Inputwerte übergeben bekommt. Die Hiddenschicht beinhaltet zwei Knoten, die Outputschicht einen Knoten. Abbildung 35 zeigt das neuronale Netzwerk für 62 topologische Deskriptoren als Eingabeparameter. Als Trainingskomponente wird der *Stuttgart Neural Network Simulator* (SNNS) verwendet. Hierbei handelt es sich um eine Softwarekomponente für die Entwicklung von neuronalen Netzwerken, die in einem Gemeinschaftsprojekt der Universitäten Stuttgart und Tübingen entwickelt wurde. Für weiterführende Informationen zu SNNS, siehe [Univ08].

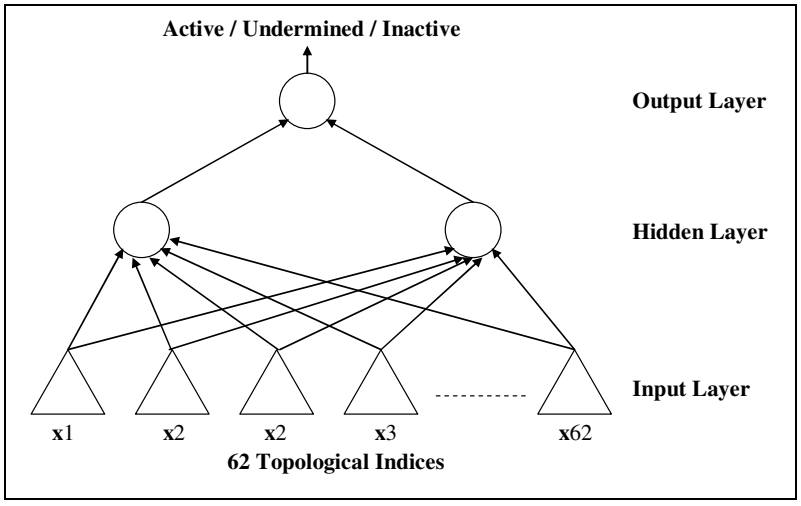

Abbildung 35: Verwendete neuronale Netzstruktur in der Arbeit von [Murc04, S. 3]

Das Ergebnis der Studie von [Murc04] mit 433 Molekülen ist, dass bei einem Diskriminanzwert größer als 0,5 die molekulare Verbindung von der Diskriminanzfunktion als aktiv klassifiziert wird. Dagegen ist bei einem Wert kleiner als 0,5 von einer inaktiven Verbindung auszugehen. Die Überprüfung dieser Klassifizierung der Diskriminanzanalyse mit dem beschriebenen neuronalen Netzwerk ergibt die nachfolgenden Ergebnisse.

Wenn durch die Diskriminanzanalyse als inaktiv klassifiziert, bedeuten die Outputwerte des neuronalen Netzwerks:

- Outputwert im Intervall zwischen -1,0 und -0,5 => Klassifikation korrekt.
- Outputwert im Intervall zwischen -0,5 und 0,0 => Klassifikation unbestimmt.
- Outputwert im Intervall zwischen 0,0 und 1,0 => Klassifikation falsch.

Wenn durch die Diskriminanzanalyse als aktiv klassifiziert, bedeuten die Outputwerte des neuronalen Netzwerks:

- Outputwert im Intervall zwischen 0,5 und 1,0 => Klassifikation korrekt.
- Outputwert im Intervall zwischen 0,0 und 0,5 => Klassifikation unbestimmt.
- Outputwert im Intervall zwischen -1,0 und 0,0 => Klassifikation falsch.

Für die Gruppe der aktiven Verbindungen weist die Diskriminanzanalyse eine Klassifikationsgenauigkeit von 75,00% auf, beim neuronalen Netzwerk liegt dieser Wert bei 95,31%. Bei den inaktiven Verbindungen liefert die Diskriminanzanalyse einen Ergebniswert von 90,63% Klassifikationsgenauigkeit, während das neuronale Netzwerk bei dieser Gruppe 87,50% der Verbindungen korrekt identifiziert. Bei allen genannten Prozentwerten sind die unbestimmten Gruppeneinteilungen mit einbezogen, werden aber nicht zu den exakt klassifizierten Verbindungen gezählt.
In [Murc04] werden Diskriminanzanalyse und neuronale Netzwerke unabhängig voneinander zur Klassifikation von molekularen Verbindungen eingesetzt. Im Gegensatz zu der hier vorliegenden Arbeit dienen die Integerwerte der topologischen Deskriptoren als Inputparameter für das neuronale Netzwerk. Somit werden keine Diskriminanzwerte an das neuronale Netz übergeben. Bei [Murc04] wird im Bereich der Pharmazie geforscht, wobei keine *Event Processing*-Technologie verwendet wird.
Durch diesen Abschnitt wurde gezeigt, dass die Übergabe von berechneten Diskriminanzwerten an ein neuronales Netzwerk in der Literatur bislang noch nicht erwähnt wurde, da die hier diskutierten Artikel der Architektur dieser Arbeit am Ähnlichsten sind.
Ein weiteres Ergebnis ist, dass die Forschung im Bereich Betrugserkennung mit *Complex Event Processing* noch in einem Anfangsstadium steckt. Dieser Umstand ist mit dem bisher relativ kurzen Bestehen von CEP-Technologie begründet. Aus diesen Gründen existiert ein großes Potenzial, in diesem Gebiet hilfreiche Forschungsarbeit leisten zu können. Im nächsten Kapitel wird daher die Auswahl der Verfahren zur Betrugserkennung auf Ba-

sis von *Complex Event Processing* begründet. An dieser Stelle ist der allgemeine Teil abgeschlossen und die Beschreibung der konkreten Entwicklung dieser Arbeit beginnt.

6 Diskussion des Ansatzes dieser Arbeit

In diesem Kapitel werden die in den Kapiteln 3 und 4 beschriebenen Algorithmen und Technologien in Bezug auf die Anwendung zur Lösung des im Rahmen dieser Arbeit behandelten Problems bewertet. In diesem Zusammenhang wird sowohl auf die Problemstellung eingegangen als auch das Gesamtmodell, die einzelnen Lösungskomponenten sowie alternative Verfahren diskutiert.

6.1 Spezifikation der Problemstellung dieser Arbeit

Nach [Böhm02, S. 32] ist ein Problem definiert als eine Diskrepanz zwischen der vorhandenen, feststellbaren Ist-Situation und der Soll-Vorstellung.
Diese Arbeit behandelt dabei das praktische Problem der Ist-Situation der auftretenden Betrugstransaktionen im Online-Banking im Anschluss an erfolgreiche Identitätsdiebstähle. Die Soll-Vorstellung ist, dass unter den auftretenden *events*, die durch die Transaktionen entstehen bzw. die Transaktionen repräsentieren, diejenigen in Echtzeit (oder nahe Echtzeit) herausgefiltert werden, die auf einen Betrugsversuch hindeuten. Diese *events* bzw. Transaktionen besitzen bestimmte Attribute, deren Werte zu diesem Zweck analysiert werden können, z.B. Transaktionsbetrag, Kontonummer des Begünstigten, Währungskennzeichen usw. [Zent07]. Konkret soll erforscht werden, wie Kombinationen (*patterns*) aus aktuellen und historischen *events* automatisiert (maschinell) erkannt werden, die zusammen aus der Transaktionshistorie heraus ein bestimmtes Betrugsmuster ergeben (wie z.B. Kontoplünderung durch Phishing, Pharming usw.) bzw. mit einer gewissen Wahrscheinlichkeit einen Betrugsfall darstellen.
Die Eventquellen der Transaktionen bilden – lt. den Aussagen aus Interviews mit Betrugsexperten – operative Datenbanken, in denen diese nach dem Anstoßen der Online-Überweisung durch den Kunden zur Weiterverarbeitung (z.B. durch das Rechenzentrum oder einem Rechenzentrumsdienstleister) zwischengespeichert werden.

Eine Transaktion selbst besteht u.a. aus den folgenden Parametern bzw. Attributen, die der Kunde beim Durchführen einer Online-Überweisung explizit angeben muss, siehe [Zent07]:

- Name des Begünstigten
- Kontonummer des Begünstigten
- Bankleitzahl des Begünstigten
- Überweisungsart
- Transaktionsbetrag
- Verwendungszweck
- Name des Kontoinhabers
- Kontonummer des Kontoinhabers
- Datum der Transaktion (optional)

Vervollständigt wird dieser Datensatz durch weitere Attribute, wie z.B. Währungskennzeichen oder Kreditinstitut des Begünstigten, die der Transaktion noch hinzugefügt werden, siehe dazu [Zent07]. Aus den Interviews mit den Betrugsexperten ergab sich, dass diese oben aufgezählten Attribute alleine nicht ausreichend sind um ein Urteil abgeben zu können, ob es sich bei einer bestimmten Transaktion um einen Betrugsfall handelt oder nicht. Der Grund dafür ist, dass sich nach einem erfolgreichen Einloggen ein Betrüger im Web oftmals genau so bewegt wie der eigentliche Kunde. Aus den Interviews ergab sich, dass die nachfolgend aufgezählte Attributteilmenge aus der Attributgesamtmenge einer Überweisung aussagekräftig ist, um einen Betrugsfall zu identifizieren, wobei zu diesem Zweck nur Online-Überweisungen zu untersuchen sind (auf Basis dieser Attribute ist auch die Simulation der Überweisungen aufgebaut, die im Unterabschnitt 9.1.2 beschrieben wird):

- KundenID
- TransaktionsID
- Transaktionsbetrag
- IP-Adresse des Quellrechners bzw. Inland IP-Adresse
- Zeit für die Online-Überweisung
- Kontostand
- Maximal verfügbarer bzw. übertragbarer Transaktionsbetrag
- Durchschnittliche Anzahl von Transaktionen (pro Monat) zum Zeitpunkt der Transaktion einschließlich der Zugriffe auf die Online-Banking Seite z.B. zur Kontostandsabfrage ohne Online-Überweisung

- Durchschnittlicher Transaktionsbetrag der Online-Überweisungen zum Zeitpunkt der Transaktion
- Kontonummer (Empfänger)
- Bankleitzahl (Empfängerbank)
- Vertrauenswürdiger bzw. bekannter Empfänger
- Bekannte, nicht vertrauenswürdige Zielbanken bzw. Auslandsüberweisung

Zur Erzeugung eines *complex events* mit diesen genannten Attributen muss von der CEP *engine* das ursprüngliche *event* mit einem *event*, das die zusätzlichen, abgefragten Daten zu einem Kunden enthält, korreliert werden.
Das Einbinden der Transaktionshistorie und die damit verbundene Bildung eines Transaktionsprofils eines Kunden ist lt. den Aussagen aus den Interviews von Bedeutung, weil der Fall auftreten kann, dass bei zwei Transaktionen mit identischen Attributausprägungen von zwei verschiedenen Kunden eine der beiden Transaktionen eine höhere Betrugswahrscheinlichkeit aufweist als die andere, ansonsten identische, Transaktion. Der Grund dafür liegt in der unterschiedlichen Verhaltens- bzw. Transaktionshistorie der beiden Kunden. Dazu folgendes Fallbeispiel: Ein Kunde überweist regelmäßig einen Betrag, der nahe an der maximalen Kapitalverfügbarkeitsgrenze des online geführten Kontos liegt, ein zweiter Kunde transferiert immer nur kleine Beträge im Verhältnis zu seinem verfügbaren Maximalbetrag. Bei dem zweiten Kunden ist die Wahrscheinlichkeit eines Betrugsfalls bei einer Überweisung in der Höhe seines Kontostands höher als bei dem erstgenannten Kunden. Das bedeutet, eine einzelne Transaktion für sich allein dargestellt reicht nicht aus, um einen Betrugsfall eindeutig zu identifizieren, was auch von [Reol07, S. 3] bestätigt wird. In [Reol07, S. 20] wird das Bilden von solchen Verhaltensmustern als „Relationalisierung der Transaktionen“ bezeichnet. Allerdings können diese Verhaltensmuster nicht für Neukunden erstellt werden, die noch keine historischen Transaktionen durchgeführt haben. Kundenspezifische Transaktionsprofile werden beispielsweise auch zur Identifikation von Kreditkartenbetrug auf Basis von Transaktionsmustern verwendet, siehe dazu [Xu06, S. 1], [Xu05, S. 1], [Kou04, S. 3], [Bolt02, S. 18] oder bei der Betrugserkennung im Wertpapierhandel, siehe [Conz07].
Für die Optimierung der Betrugsanalyse beim Online-Banking ist es wichtig, dass einerseits ein hoher Anteil der Betrugstransaktionen identifiziert wird und andererseits so wenig wie möglich Nicht-Betrugstransaktionen fälschlicherweise als Betrugstransaktionen klassifiziert werden (*false positive*).
Eine falsch klassifizierte Nicht-Betrugstransaktion ist ebenfalls kritisch für ein Kreditinstitut, da hierbei die Gefahr besteht, die Kundenzufriedenheit zu verringern oder den Kunden an

einen Mitbewerber zu verlieren, falls eine legale Transaktion aufgrund eines Betrugsverdachts gestoppt wird [Agge06, S. 2].
Die Identifikation der Betrugstransaktionen muss so schnell wie möglich, d.h. in Echtzeit oder nahe an Echtzeit erfolgen um sie vor dem Abschluss verhindern zu können, worauf z.B. auch [Conz07], [Bose06, S. 1] und [Xu06, S. 1] in ihren Arbeiten hinweisen. Dies ist ebenfalls notwendig, da die Opfer selbst einen Betrugsfall in der Regel viel zu spät (z.B. oftmals erst nach einer Woche) bemerken [Agge06, S. 1]. Diese Problemstellung, die richtige Information (in diesem Fall über einen Betrugsvorgang) komprimiert zur exakten Zeit an den benötigten Ort zu transferieren wird in den Artikeln von F. Hayes-Roth als *Valued Information at Right Time*-Problem (VIRT) bezeichnet, siehe [Haye06] und [Haye07], wobei in diesen Publikationen der militärische Sektor das untersuchte Anwendungsgebiet darstellt. Von den interviewten Betrugsexperten wird dieses Problem bei einer Erkennungsgenauigkeit von 99,0% und zeitnaher Verteilung der Information an die entsprechenden Stellen durch ein automatisiertes Verfahren als gelöst angesehen. Der Rest, d.h. die nicht exakt zuordenbaren Transaktionen würden lt. den befragten Betrugsexperten von den Mitarbeitern der Revisionsabteilungen der Kreditinstitute manuell geprüft werden. Somit soll im Rahmen dieser Arbeit ein inverses Problem nach [Bung08, S. 16] gelöst werden, da die genannten Anforderungen im Vorfeld feststehen und nachfolgend nach einem geeigneten Modell geforscht wird, welches diese Anforderungen erfüllt.

6.2 Lösungskomponenten und deren Verwendungsbegründung

Zur Erfüllung der Anforderungen, die sich aufgrund der im Vorgängerabschnitt sowie im Abschnitt 2.2 beschriebenen, zu analysierenden metrischen und nichtmetrischen Datenstrukturen sowie dem Anspruch der Echtzeitfähigkeit als auch einer hohen Erkennungsgenauigkeit ergeben, wird im Rahmen dieser Arbeit zur Lösung des Problems der Identifikation von Betrugstransaktionen beim Online-Banking ein Hybrid-Modell vorgeschlagen. Das Modell beinhaltet eine CEP *engine* um die Daten- bzw. Eventversorgung zu implementieren sowie die Echtzeitfähigkeit des Modells zu gewährleisten. Innerhalb der CEP *engine* werden die *events* zur Laufzeit um die oben genannten zusätzlichen Attribute erweitert. Das bedeutet, sie werden mit weiteren *events* korreliert, welche die benötigten Zusatzinformationen zum Transaktionsverhalten der Kunden (wie z.B. durchschnittlicher Transaktionsbetrag) als Abfrageergebnisse aus externen Wissensquellen z.B. Bestandsdatenbanken enthalten. Extern bedeutet in diesem Fall von außerhalb der CEP *engine*.
Darüber hinaus besteht die Hybrid-Architektur aufgrund der gegebenen metrischen und nichtmetrischen Attributstruktur der Transferevents – wie im Abschnitt 1.1 erwähnt – aus

einer Kombination von Diskriminanzanalyse, einem Entscheidungsbaum sowie einem neuronalen Netzwerk als überwachte maschinelle Lernverfahren.
Die Diskriminanzanalyse eignet sich für die Analyse von Attributen der Ratioskala (Attribute mit denen die vier Grundrechenarten und Mittelwertbildung logisch durchgeführt werden können, siehe Abschnitt 3.1), wie z.B. Transaktionsbetrag, maximal verfügbarer Transaktionsbetrag oder durchschnittliche Anzahl von Transaktionen (siehe Abschnitt 3.2). Dagegen werden Entscheidungsbäume für vorhandene Attribute der Ordinalskala (Attribute, die zur Klassifizierung qualitativer Eigenschaftsausprägungen verwendet werden können, siehe Abschnitt 3.1) wie z.B. IP-Adresse des Quellrechners, Empfängerkontonummer oder Empfängerbankleitzahl benötigt (siehe Abschnitt 3.1). Eine Begründung für die Verwendung dieser genannten Verfahren folgt weiter unten in diesem Abschnitt.
Nach Aussage von [Bign06, S. 2 - 3] sind überwachte Verfahren, aufgrund des durch die Bekanntheit des Betrugsstatus abprüfbaren Lernerfolgs, generell besser für die Transaktionsüberwachung beim Online-Banking geeignet als unüberwachte Methoden. Lt. [Muna08, S. 35] werden in der Forschung neuronale Netzwerke häufig mit anderen Analyseverfahren in Form eines Hybrid-Modells kombiniert, da die Vereinigung mehrerer Verfahren oftmals bessere Ergebnisse liefert als alleinstehende Analysemethoden. Dies zeigt sich z.B. in einer Studie von [Phua05] zum Thema Betrugserkennung auf Basis von *Data Mining*. In dieser Publikation werden unter anderem Hybrid-Modelle diskutiert, die zur Verbesserung von Erkennungsergebnissen geführt haben. Ein Beispiel dafür bildet das Modell eines Entscheidungsbaums, der zur Aufteilung der Daten für ein neuronales Netzwerk mit Backpropagation verwendet wird. Die Funktion $\tanh$ (*Tangens Hyperbolicus* oder auch Hyperbelfunktion genannt, siehe [Köni99, S. 419 - 422]) dient in diesem Beispiel als Gewichtsfunktion zur Generierung der Betrugsdichte der Testfälle. Das Ziel des Modells ist die Identifikation von Betrugsversuchen bei Kreditkartentransaktionen, wobei hier die Attributstruktur anders zusammensetzt ist und keine CEP-Technologie verwendet wird. Nach Aussage von [Chan09, S. 27 - 28] werden in der Zukunft Hybrid-Systeme – auch im *Event Processing*-Umfeld – eine immer weitere Verbreitung finden.
Ein weiteres Beispiel für die Verwendung eines Hybrid-Modells zeigt sich im Anwendungsgebiet der automatisierten Betrugserkennung im Bankenumfeld bei [Reol07, S. 71 - 72]. In dieser Publikation wird ein solches Modell bestehend aus einer Kombination des relationalen Graphenreduktionsalgorithmus SUBDUE (siehe [Reol07, S. 35 - 38]) und dem propositionalen Clusteralgorithmus YALE (siehe [Reol07, S. 52 - 54]) als unüberwachtes Verfahren zum Erkennen von internem Bankbetrug evaluiert. Allerdings ist der Autor mit der Auswahl der einzelnen Algorithmen seines Hybrid-Modells unzufrieden, weil sich im Rahmen der Auswertungen gezeigt hat, dass vor allem SUBDUE bereits bei geringen Datenmengen eine zu lange Laufzeit benötigt [Reol07, S. 92]. Aus diesem Grund

kam dieses Modell als Option im Rahmen dieser Arbeit nicht in Betracht. Aufgrund der Tatsache, dass der Betrugsstatus der Trainingsdaten (Betrugsfall oder Nicht-Betrugsfall) bereits bekannt ist, bietet sich – wie oben erwähnt – die Verwendung überwachter Verfahren für diese Arbeit an. Des Weiteren sind nach [Reol07, S. 98] überwachte Verfahren aus Sicht der Wirtschaftlichkeit besser zur Betrugserkennung geeignet als unüberwachte Verfahren, da die unüberwachten Methoden aufgrund der unbekannten Qualität der Ergebnisse ein größeres Fehlerrisiko bergen. Der Grund für das höhere Risiko ist, dass nicht gewährleistet ist, ob etwaige gefundene Muster zur Lösung des jeweiligen Problems geeignet sind.

Das Hybrid-Modell dieser Arbeit besteht aus zwei Grundkomponenten, der Datenversorgungs- und der Datenanalysekomponente bzw. Betrugserkennungskomponente. Für die Datenversorgung bzw. Eventversorgung wurde *Complex Event Processing*-Technologie verwendet. Eine CEP *engine* wird in diesem Zusammenhang benötigt um die, im Abschnitt 4.3 erwähnte, Echtzeitfähigkeitseigenschaft für den Betrugserkennungsprozess bereitzustellen. Auswertungen werden somit nicht *ex post* auf einem bestehenden *Data Warehouse* wie z.B. beim *Data Mining* vorgenommen [Krah98, S. 52], sondern in Echtzeit, d.h. beim Auftreten bzw. Auslösen der Transaktionsevents. Nach Aussage der interviewten Experten und [Reol07, S. 11] erfolgt eine Betrugsuntersuchung in Kreditinstituten größtenteils erst zu einem Zeitpunkt, nachdem ein Betrugsfall vom Kunden gemeldet wurde, nicht aber in Echtzeit. CEP wird ebenfalls dazu benötigt um das ursprüngliche Transaktionsevent mit einem abgefragten *event* aus einer Datenbank zu korrelieren, das weitere betrugsrelevante Attribute enthält (siehe Abschnitt 6.1). In einem Blog-Eintrag beschreibt G. Nelson, dass aufgrund der Schnelligkeit, bei der die Transaktionen in den heutigen Finanzsystemen verarbeitet werden, CEP eine sinnvolle Technik ist um die durchgeführten Überweisungen in Echtzeit nach ihrem Betrugsstatus zu analysieren, siehe [Even08]. Dies wird auch durch die Aussagen aus den Interviews bestätigt, dass Überweisungen zum Zielkonto beim Online-Banking „*on the fly*“ durchgeführt werden. Die gleichen Argumente verwenden [Luck04d] und [Lund06, S. 8] in ihren Artikeln, wobei diese Autoren (noch) keine konkrete Betrugserkennungslösung entwickelt haben (siehe Abschnitt 5.1). Auch erwähnt [Conz07], dass im Bankenumfeld Betrugsversuche so schnell wie möglich identifiziert werden sollten, z.B. unter Einsatz von Mustererkennung mittels CEP, wobei dieser Artikel ebenfalls keine konkreten Muster oder Lösungsarchitekturen diskutiert.

Die Betrugserkennungskomponente besteht – wie u.a. im Abschnitt 1.1 erwähnt – aus einer Kombination von Diskriminanzanalyse, Entscheidungsbaum und neuronalem Netzwerk. Der genaue Aufbau der Betrugserkennungskomponente wird in den Abschnitten 7.1 und 7.2 beschrieben. Nachfolgend wird eine Begründung der Auswahl der eingesetzten

Verfahren sowie für die Nichtberücksichtigung von alternativen maschinellen Lernverfahren gegeben:

Ein verwendetes Analyseverfahren bildet die Diskriminanzanalyse. Diese Methode eignet sich für die Verwendung im Rahmen dieser Arbeit aufgrund ihrer Fähigkeit, Elemente auf Basis ihrer Merkmalsausprägungen bzw. Attributwerte in Gruppen zu klassifizieren [Back06, S. 156]. Da die Struktur der Trainingsevents so angelegt ist, dass nur zwei voneinander zu trennende Gruppen (Betrugsverdächtig und Nicht-Betrugsverdächtig) existieren, ist für die Auswertungen dieser Arbeit der zwei Gruppen-Fall für die Diskriminanzanalyse relevant. Im zwei Gruppen-Fall kann die Diskriminanzfunktion schneller ermittelt werden als im mehr Gruppen-Fall, weil weniger Berechnungen nötig sind (siehe Abschnitt 3.2). Diese Tatsache kommt wiederum der Anforderung der Echtzeitfähigkeit entgegen. Ein weiteres Argument für die Diskriminanzanalyse ist, dass eine Transaktion bzw. ein korreliertes *event* mit allen Attributwerten durch einen einzigen Wert dargestellt wird und sich daher als komprimierter Eingabewert für ein neuronales Netzwerk eignet. Somit benötigt ein neuronales Netzwerk weniger Eingabeknoten als bei einer Übergabe aller Einzelattribute mit ihren Werten. Dadurch verringert sich die Laufzeit bei der Berechnung des Ausgabewerts eines neuronalen Netzwerks. Eine Schwäche der Diskriminanzanalyse ist, dass sie – wie im Abschnitt 3.2 erwähnt – lediglich metrisch skalierte Werte analysieren kann. Für das Ziel, auch nichtmetrische Attribute, wie z.B. Empfängerbankleitzahl in die Betrugsanalyse mit einbeziehen zu können, wurde für die Analysen dieser Arbeit zusätzlich ein Entscheidungsbaum eingesetzt. Der Grund für diese Auswahl ist, dass sich dieses Verfahren aufgrund seiner schnellen und einfachen Implementierbarkeit in der Praxis bewährt hat [Krah98, S. 73]. Darüber hinaus können im Entscheidungsbaum KO-Kriterien hinterlegt werden, nach denen der Betrugserkennungsprozess auf jeden Fall ausgeführt oder nicht ausgeführt werden soll. Diskriminanzanalyse und Entscheidungsbaum können flexibel für die möglichen Ausprägungen der Attribute eingesetzt werden. Die im Rahmen dieser Arbeit verwendeten Attribute und deren Ausprägungen werden in den Unterabschnitten 9.2.1 und 9.2.2 diskutiert.

Als finale Auswertungsmethode wird ein neuronales Netzwerk verwendet um den aktuellen Diskriminanzwert gleichzeitig in Verbindung mit historischen Diskriminanzwerten eines Kunden analysieren zu können. Der Ansatz zur Betrugserkennung mittels Diskriminanzanalyse ohne neuronalem Netzwerk ist in [Widd07] beschrieben. Allerdings ist es mit dieser Entwicklung nicht möglich, historische *events* in die Analyse zur Laufzeit mit einzubinden. Dies führt zu einer ungenauen Betrugsaussage, da kein Verhaltensmuster auf Basis historischer Werte berücksichtigt wird (siehe Abschnitt 6.1). Neuronale Netze werden in der Literatur häufig als erfolgreiches und stabiles Mittel zur Betrugserkennung genannt, z.B. bei [Phua05, S. 5], [Bolt02, S. 3], [Bose06, S. 2 - 5] oder [Kou04, S. 2 - 3] in ihren Studien

zur Erkennung von Kreditkartenbetrug oder bei [Vikr04, S. 2] zur Identifizierung illegaler Kontenaktivitäten im Bankenumfeld. In diesem Zusammenhang ist die Generalisierungsfähigkeit der neuronalen Netzwerke von entscheidender Bedeutung, d.h. die Fähigkeit auf Basis von gelernten Mustern unbekannte Muster auch bei nichtlinearen Zusammenhängen klassifizieren zu können (siehe Abschnitt 3.3). Als unbekanntes Muster (engl.: *unknown event pattern*) wird im Rahmen dieser Arbeit ein Muster definiert, das sich nicht unter den Trainingsmustern befindet.

Da die Outputwerte der Trainingsmuster bereits im Vorfeld feststehen, bietet sich für das neuronale Netzwerk die Backpropagationsmethode als überwachtes Lernverfahren an (siehe Unterabschnitt 3.3.2). In [Bign06, S. 3] werden überwachte neuronale Netzwerke mit Backpropagation als Trainingsalgorithmus aufgrund ihrer exakten Lernfähigkeit als die beste Methode zur Identifikation von Betrugstransaktionen beim Online-Banking bezeichnet.

Eine Kombination von Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk bietet den Vorteil einer Vorklassifizierung durch Diskriminanzanalyse und Entscheidungsbaum, d.h. es werden bereits im Vorfeld als harmlos identifizierte *events* bzw. Transaktionen aus dem Eventstrom in der CEP *engine* herausgefiltert (die genauen Filterkriterien werden in den Unterabschnitten 9.2.1 und 9.2.2 diskutiert). Dadurch sind insgesamt weniger Muster durch das neuronale Netzwerk zu untersuchen, was wiederum Vorteile bezüglich der Performance mit sich bringt. Die Übergabe von Diskriminanzwerten an ein neuronales Netzwerk hat – wie oben erwähnt – zur Folge, dass ein *event* bzw. eine Transaktion durch einen einzigen Wert (den Diskriminanzwert) dargestellt wird. Somit genügt die Bereitstellung eines Inputknotens pro *event*. Wäre dies nicht der Fall müsste bei einem *Multi Layer Perceptron* für sich alleinstehend – wie oben erwähnt – ein Inputknoten für jedes betrugsrelevante Attribut eines *events* vorhanden sein, wodurch sich die Netzstruktur automatisch erweitert. Dadurch würde das Trainieren des Netzwerks und die Vorwärtsaktivierung mehr Zeit benötigen, was in [Pete05, S. 230] diskutiert ist. Lt. Aussage der Autoren von [Chen97, S. 1] führt die Aufteilung in kleinere neuronale Netzwerke auf Basis einer Vorgruppierung durch die Diskriminanzanalyse zu exakteren Ergebnissen und kürzeren Trainingszeiten als der Einsatz eines alleinstehenden *Multi Layer Perceptrons* zur Lösung des Gesamtproblems. Neuronale Netzwerke sind darüber hinaus in der Lage, alle Arten von Zahlenmustern zu lernen, wodurch sie neu trainiert werden können, sobald sich die Muster der Diskriminanzwerte verändern.

Diese beschriebene Betrugserkennungskomponente des Hybrid-Modells besitzt in ihrer Zusammensetzung den Vorteil, dass – aufgrund der Vorselektion von metrischen Attributen (mittels Diskriminanzanalyse) sowie nichtmetrischen Attributen (mittels Entscheidungsbaum) – von einem neuronalen Netzwerk nur Diskriminanzwerte analysiert werden.

Diese Diskriminanzwerte können auch für andere Branchen und Anwendungsfälle gebildet und von einem neuronalen Netzwerk untersucht werden, z.B. für die Betrugserkennung in der Schadensfallabwicklung bei Versicherungen. Dadurch gestaltet sich das Modell als sehr flexibel und individuell anpassbar.

Neben den oben diskutieren Verfahren existieren weitere maschinelle Lernverfahren, die im Rahmen dieser Arbeit aus verschiedenen Gründen nicht verwendet werden. Als Alternative zur Vorselektion mittels Diskriminanzanalyse und Entscheidungsbaum wäre die logistische Regression ein mögliches Verfahren. Sie dient aber – wie im Abschnitt 3.6 erwähnt – größtenteils zur Untersuchung der Fragestellung, mit welcher Wahrscheinlichkeit ein Ereignis eintritt und welche Attribute diese Wahrscheinlichkeit wie stark beeinflussen. Die logistische Regression besitzt gegenüber der Diskriminanzanalyse den Vorteil, dass sowohl metrische als auch nichtmetrische Attribute zu Analysezwecken verwendet werden können [Back06, S. 10 - 11]. Allerdings gibt sie die Zugehörigkeit zu einer Gruppe als Wahrscheinlichkeitswert (entspricht dem Funktionswert der logistischen Funktion) [Back06, S. 439] an, was zur Folge hätte, dass ein neuronales Netzwerk mit diesen Wahrscheinlichkeitswerten trainiert werden müsste. Dies würde aber keine Verbesserung der Erkennungsgenauigkeit ergeben, da bei den Trainingssätzen bereits bekannt ist, ob es sich um einen Betrugsfall handelt oder nicht. Es bestünde somit ein linearer Zusammenhang zwischen dem Wahrscheinlichkeitswert und dem bekanntem Outputwert eines Trainingsfalls. Die logistische Regression ist gut als alleinstehendes Verfahren zur Betrugserkennung geeignet, was bei [Inte07] beschrieben ist. Allerdings wäre es bei Verwendung der logistischen Regression nicht möglich – genau wie bei der Diskriminanzanalyse als alleinige Analysemethode – die historischen Transaktionen des Kunden bewerten zu können.

Als weitere Alternative zu dem beschriebenen Hybrid-Modell sind probabilistische Netzwerke z.B. in Form von Bayes-Netzwerken oder Markov-Netzwerken zu nennen. In der Literatur sind diese Methoden häufig als Verfahren zur Betrugserkennung genannt, z.B. bei [Mukh08, S. 1], [Phua04, S. 2] oder [Sriv08, S. 1]. Bayes-Netzwerke kombinieren aktuelle Daten mit vorhandenem Expertenwissen zur Darstellung und Analyse der Beziehungen von Ursache und Wirkung innerhalb einer Graphenstruktur. Die Knoten des Graphen geben bestimmte Zustände wieder. An den Kanten zwischen den Knoten sind bedingte Wahrscheinlichkeiten angegeben. Wie im Abschnitt 3.5 beschrieben ist, treffen diese bestimmten Werte eine Aussage darüber, wie wahrscheinlich es ist, dass der Zielknoten einen bestimmten Zustand annimmt, ausgehend vom Zustand des Quellknotens. Sie dienen daher zum Treffen von semantischen Schlussfolgerungen auf Basis bekannter Wahrscheinlichkeiten und verfolgen somit einen anderen Ansatz als diese Arbeit. Dies ist darin begründet, dass in den Strukturen der Quellinformationen bzw. *events* des in dieser

Arbeit zu lösenden Problems keine solchen beschriebenen Übergangswahrscheinlichkeiten enthalten sind und auch nicht exakt angegeben werden können.
Ein weiterer Algorithmus, der für die Implementierung der Betrugserkennung im Rahmen dieser Arbeit diskutiert wurde, ist das *Support Vector Machine*-Verfahren. Bei dieser Methode tritt das gleiche Problem wie bei der Diskriminanzanalyse als alleinstehendes Verfahren ein. *Support Vector Machine* dient zur Klassifikation eines unabhängigen Falls, somit kann die notwendige Transaktionshistorie nicht berücksichtigt werden. SVM ist ein sehr rechenintensiver Algorithmus, vor allem bei größeren Datenmengen [Wang05, S. 26], wodurch sich Nachteile bezüglich der Performance des Betrugserkennungsmodells bei einem möglichen realen Einsatz ergeben könnten.
Eine andere Alternativmethode zu dem Hybrid-Modell dieser Arbeit besteht in der Formulierung von Wenn-Dann-Regeln als Implementierung eines regelbasierten Ansatzes. Die Regeln müssen exakt definiert sein. Sind sie das nicht, können entweder zu viele Betrugsversuche unerkannt bleiben (wenn Regel zu spezifisch) oder zu viele Nicht-Betrugsversuche als *false positive* klassifiziert werden (wenn Regel zu ungenau), siehe dazu [Widd07, S. 2]. Zusätzlich basieren die Regeln auf statischem Wissen, auf welches sich die Betrüger mit der Zeit einstellen könnten. Des Weiteren sind solche Systeme durch ihre starren Regeln nicht in der Lage, unbekannte Muster (d.h. Muster, die zuvor nicht durch entsprechende Regeln definiert wurden) zu entdecken. Im Ansatz dieser Arbeit wird dagegen für jede Transaktion eine Betrugswahrscheinlichkeit ausgegeben, daher können auch unbekannte Betrugsmuster eine höhere Betrugswahrscheinlichkeit aufweisen.

6.3 Neuartigkeit des Ansatzes dieser Arbeit

Die Neuartigkeit des Ansatzes dieser Arbeit für die Beantwortung der wissenschaftlichen Fragestellung aus dem Einleitungskapitel ist durch folgende fünf Punkte gekennzeichnet:

a) Die Kombination von Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk zur Erkennung von Betrugsfällen. Diese Kombination wurde in dieser Form zuvor nicht in der Literatur diskutiert.

b) Die Einbindung von *Complex Event Processing* und *events* als Informationslieferanten einer, aus maschinellen Lernverfahren kombinierten, automatischen Betrugserkennungskomponente beim Online-Banking.

c) Die Nutzung von *Complex Event Processing*-Technologie zur Transaktionsanalyse beim Online-Banking. Diese Zusammensetzung wurde in der Literatur als mögliche Lösung zur Betrugserkennung erwähnt (siehe [Even08], [Conz07] oder [Lund06, S. 8]), aber bis zu diesem Zeitpunkt noch kein konkretes Modell diskutiert oder praktisch umgesetzt. Der Grund dafür ist, dass die Betrugsbekämpfungsstrategie der Banken – nach den Aussagen aus den Interviews (siehe Anhang 1) – stärker auf Betrugsprävention z.B. mittels frühzeitiger Erkennung von Phishingseiten oder Weiterentwicklung der bestehenden Authentifizierungsverfahren als auf aktive Betrugserkennung mittels Transaktionsanalyse ausgelegt ist. Wobei nach einer Studie von [Moor07] die durchschnittliche Lebensdauer einer Phishingseite von 61,69 Stunden von der Aktivierung bis zur Deaktivierung durch den Internetprovider noch zu lang ist um das Problem vollständig zu lösen. Diese Lebensdauer ist von der Anzahl abhängig, wie oft die Internetseite eines Kreditinstituts imitiert wird. Beispielsweise hat eine gefälschte Internetseite der `Citybank` eine durchschnittliche Lebensdauer von ca. 100 Stunden, weil sie nicht so häufig nachgeahmt wird wie z.B. die Internetseite von `Wachovia`, siehe [Moor07, S. 11].

d) Eine Transaktion bzw. *event* wird mit seinen relevanten Attributen als ein einziger Wert (= Diskriminanzwert) dargestellt und direkt an das neuronale Netzwerk übergeben. Somit lernt das Netzwerk nicht die Originalwerte der Attribute, sondern die Ergebnisse eines Vorgängerverfahrens, d.h. Betrugs- und Nicht-Betrugsmuster bestehend aus Diskriminanzwerten.

e) Die Forschung in der Richtung Kombination von CEP-Technologie und maschinellen Lernverfahren bzw. statischer Klassifikation von *events* steht erst am Anfang. Die erste Konferenz, die exakt diese Thematik unter dem Begriff *Intelligent Event Processing* adressierte, war das AAAI Spring Symposium im März 2009, siehe [Aaai09].

In diesem Kapitel wurde die Auswahl der verwendeten Verfahren Entscheidungsbaum, Diskriminanzanalyse und neuronales Netzwerk auf Basis von CEP-Technologie begründet. Dabei wurde deren Eignung für den Anwendungsfall dieser Arbeit diskutiert und festgestellt sowie Argumente für die Neuartigkeit des Ansatzes zur Erkennung von Identitätsdiebstahl im Online-Banking genannt. Die eingesetzten Verfahren werden – wie u.a. im Abschnitt 6.2 erwähnt – in dieser Lösung nicht als alleinstehende Analysemethoden verwendet, sondern als Kombination in Form eines Hybrid-Modells. Die Trainings- und die Ablaufphase dieses Hybrid-Modells werden im nächsten Kapitel erläutert.

7 Erkennung von Identitätsdiebstahl beim Online-Banking

In diesem Kapitel wird der Ansatz dieser Arbeit, das Hybrid-Modell zur Betrugserkennung, das sowohl eine Datenversorgungs- als auch eine Betrugserkennungskomponente enthält, allgemein beschrieben. Auf die konkreten Ausprägungen des Modells und die spezifischen Parameter für das Online-Banking wird im Abschnitt 9.2 eingegangen. Das Modell kombiniert – wie u.a. im Abschnitt 6.2 erwähnt – einen Entscheidungsbaum mit Diskriminanzanalyse und einem neuronalen Netzwerk. Diese Kombination der Algorithmen wird nachfolgend als ein Teil des Hybrid-Modells namens Betrugserkennungskomponente bezeichnet. Erweitert werden diese Analysealgorithmen durch eine CEP *engine*, welche die *events* für die Analyse liefert und zuvor korreliert. Die zugrundeliegende *Event Cloud* beinhaltet die zu analysierenden Eingabeevents als Online-Überweisungen bzw. Transaktionen in Datenbanken als Eventquellen in der Bankenbranche. Den *event type* für die Analyse aus dem edBPM-Referenzmodell in Abbildung 1 stellen somit Transaktionsevents dar. Eine Transaktion kann nach der Definition des Begriffs *event* von D. Luckham (siehe Abschnitt 4.1) als *event* bezeichnet werden, da es eine Struktur (*Form*) in Form von Attributen wie z.B. Transaktionsbetrag besitzt. Die Aktion (*Significance*), die das *event* auslöst, ist durch das Anstoßen einer Online-Überweisung durch den Kunden gegeben genau wie Beziehungen (*Relativity*) zu anderen *events*, wie z.B. zu den historischen Transaktionsevents des Kunden. In [Chan09, S. 130] wird eine solche Transaktion als *withdrawal event* bezeichnet. Auch M. Chandi, S. Ramo und R. Schulte nennen in [Chan07] Online-Überweisungen als Beispiel für ein *event*. Die Transaktionen sind als *events* auf der Geschäftsprozessebene angesiedelt (engl.: *business level events*), im Gegensatz zu den *low level events* auf der Netzwerkebene wie z.B. *ICMP Traps*.

Die CEP *engine* erweitert die ursprüngliche Struktur der *events* – wie im Abschnitt 6.1 erwähnt – mit zusätzlichen Attributen um die Analyse mit allen benötigten Attributen durchführen zu können. Dazu werden mit den entsprechenden Befehlen der CEP *engine* die benötigten Daten aus einer externen Datenbank mit Kundenbestandsdaten geladen. Die *events*, die aus der Bestandsdatenbank zurückgesandt werden, werden mit den ursprünglichen Transaktionsevents korreliert. Dieser Vorgang wird im Abschnitt 8.1 sowie im Anhang 2 näher erläutert. Das gesamte entworfene Hybrid-Modell ist in Abbildung 36 abgebildet.

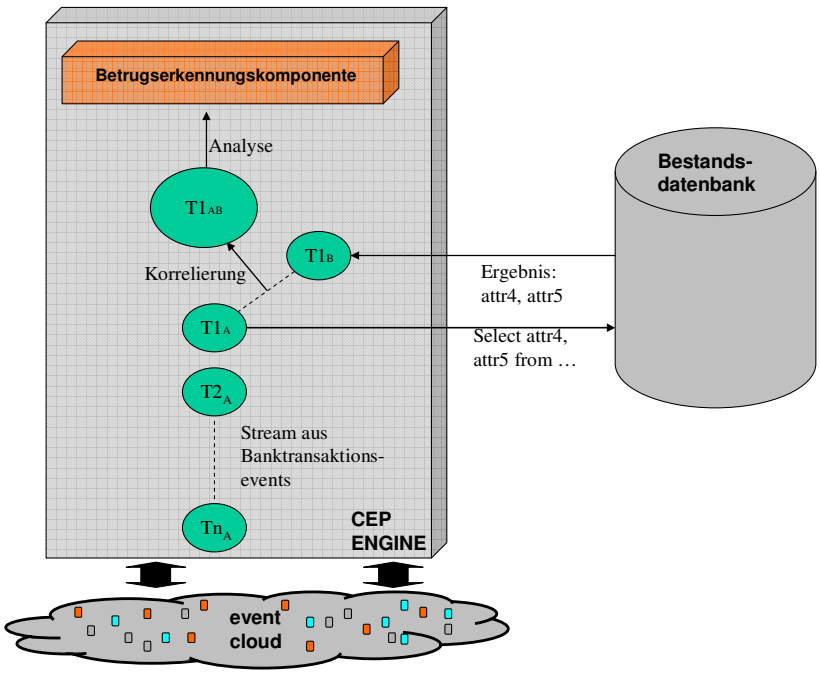

Abbildung 36: Architektur des Hybrid-Modells zur Betrugserkennung

Die CEP *engine* dient – wie im Abschnitt 6.2 erwähnt – als Datenversorgungskomponente des Hybrid-Modells, das als weiteren Bestandteil eine Betrugserkennungskomponente beinhaltet, wie in Abbildung 36 zu erkennen ist.

7.1 Trainingsprozess in der Betrugserkennungskomponente

Nach der Korrelation der *events* kann die Betrugsanalyse gestartet werden. Zur optimalen Ausführung der Analyse sind bestimmte Vorarbeiten bzw. Trainingsprozesse in der Betrugserkennungskomponente durchzuführen, die in Abbildung 37 als Modell dargestellt sind.

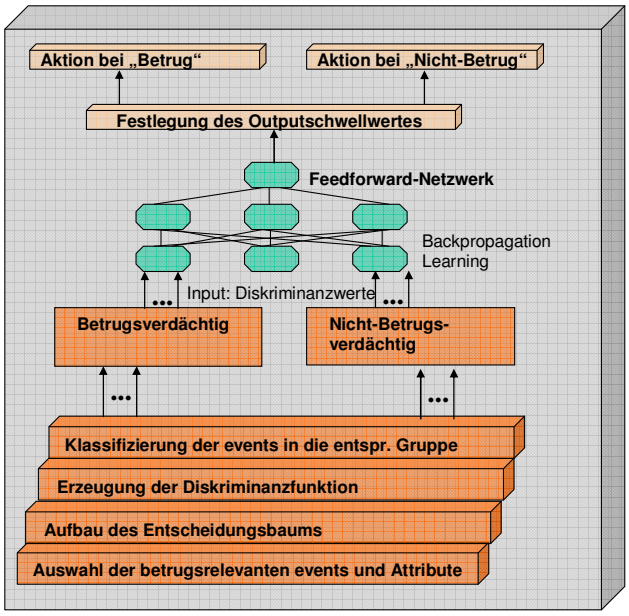

Abbildung 37: Ablaufmodell der Trainingsprozesse in der Betrugserkennungskomponente

Bei den Trainingsprozessen wird in den nachfolgenden Schritten vorgegangen:

Schritt 1 – Auswahl der betrugsrelevanten *events* und Attribute:
Bevor ein Problem gelöst werden kann, muss es identifiziert werden. Zur Erreichung dieses Ziels muss zunächst festgestellt werden, welche *events* Aussagen über (mögliche) Betrugsfälle zulassen. Sind die *events* selektiert ist der nächste Schritt, die betrugsrelevanten Attribute dieser *events* für die Analyse durch das Modell festzulegen. Dieser *Feature Selection*-Prozess kann sowohl auf Basis der fachlichen Erfahrung einer Person oder eines Teams als auch maschinell mittels Algorithmen ausgeführt werden. Für die automatische *Feature Selection* wird einerseits ein Verfahren benötigt um aus einer Gesamtmenge n eine Untermenge von r passenden *Features* zu selektieren und andererseits ein Kriterium um die Qualität der selektierten *Features* in Hinblick auf das zu lösende Problem zu beurteilen [Raud01, S. 224]. Ein dafür geeigneter Algorithmus ist die logistische Regression, die auf Basis von Eintrittswahrscheinlichkeiten für Ereignisse die Ausprägung der Attribute untersucht und somit den Einfluss der Attribute bestimmt [Back06, S. 426 - 432].
Ein Beispiel: Zwölf Personen werden zum Butterkauf befragt, sieben davon kauften Butter und deren Monatseinkommen liegt über 2.000 Euro, wobei bei den Nichtbutterkäufern das Einkommen unter 2.000 Euro liegt. In diesem einfachen Beispiel konnte (für diese spezielle Kundengruppe) ein Rückschluss getroffen werden, dass das Einkommen einen Einfluss auf die Kaufentscheidung für oder gegen Butter hat [Back06, S. 426 - 432].
Weitere Algorithmen für *Feature Selection* sind neben der logistischen Regression beispielsweise *Naive Bayes*, neuronale Netzwerke, Cluster-Verfahren oder Entscheidungs-

bäume [Sqls08]. Für weiterführende Literatur zu *Feature Selection* sei auf [Liu98] und [Jain00] verwiesen.
Im Rahmen dieser Arbeit wurde für die *Feature Selection* kein Algorithmus ausgeführt, sondern die Festlegung relevanter Attribute erfolgte auf Basis der Interviews mit den Betrugsexperten, da diese zum einen jahrelange Erfahrung in Betrugserkennung aufweisen und zum anderen keine echten Basisdaten für die *Feature Selection* zur Verfügung standen. Die selektierten Attribute sind in den Abschnitten 2.2 und 6.1 genannt.
Auf Basis der Erkenntnisse dieses Schritts ist aus dem relevanten Daten- bzw. Eventbestand der Organisation eine Trainingsmenge herzustellen, mit deren Hilfe die Komponenten der nachfolgenden Schritte entsprechend konzipiert werden können. Die Trainingsmenge wird – wie im Abschnitt 6.2 erwähnt – in zwei Gruppen (bekannte Betrugsfälle und bekannte Nicht-Betrugsfälle) aufgeteilt. Zur Vermeidung der Notwendigkeit zur Gewichtung der arithmetischen Mittel mit den jeweiligen Gruppengrößen bei der Diskriminanzanalyse (siehe Abschnitt 3.2), weisen die beiden Gruppen die gleiche Anzahl an Gruppenelementen auf. Der Umfang dieser Trainingsmenge wird im Abschnitt 9.2 erläutert.

Schritt 2 – Aufbau des Entscheidungsbaums:
Falls im Schritt 1 nichtmetrische Attribute als betrugsrelevant deklariert wurden, was bei der Festlegung der relevanten Attribute dieser Arbeit der Fall ist (siehe Abschnitt 6.1), wird für diese Attribute ein Entscheidungsbaum (siehe Abschnitt 3.1) angelegt. Der Grund dafür ist, dass die Diskriminanzanalyse als multivariates Verfahren zur Analyse von Gruppenunterschieden nur metrisch skalierte Attribute analysieren kann, siehe [Back06, S. 156]). Der Entscheidungsbaum wird so konzipiert, dass an den Ausgängen eine entsprechende Vorklassifizierung der untersuchten Transaktionen in Betrugsverdächtig und Nicht-Betrugsverdächtig erfolgt, d.h. die *events* werden im Vorfeld ausgesiebt oder durch das neuronale Netzwerk weiter analysiert. Die Tiefe und der Umfang des Entscheidungsbaums sind von der Anzahl der nichtmetrischen Attribute und der KO-Kriterien abhängig. Diese Anzahl sowie die festgelegten Attributausprägungen an den Ästen des Entscheidungsbaums werden wiederum von den fachlichen Rahmenbedingungen und den Datenbeständen der Organisation beeinflusst und im Abschnitt 9.2 für diesen Anwendungsfall diskutiert.

Schritt 3 – Berechnen der Diskriminanzfunktion und des kritischen Diskriminanzwerts:
Bei der Berechnung dieser beiden Komponenten werden auf Basis der betrugsrelevanten metrischen Attribute der Trainingsmenge die Koeffizienten der Diskriminanzfunktion berechnet. Im Rahmen dieser Arbeit wird zu diesem Zweck das Verfahren zur Diskriminan-

zanalyse aus [Bahr03, S. 316 - 329] und [Ecke02, S. 292 - 307] für den zwei Gruppen- (Betrugsverdächtig oder Nicht-Betrugsverdächtig) und n Variablen-Fall (n>2) verwendet (siehe Abschnitt 3.2). Durch Einsetzen der betrugsrelevanten Attribute in die ermittelte Diskriminanzfunktion werden im Anschluss die Diskriminanzwerte (= Funktionswert der Diskriminanzfunktion) für alle Datensätze bzw. *events* der Trainingsmenge berechnet. Danach wird auf Basis der berechneten Diskriminanzwerte der kritische Diskriminanzwert ermittelt. Wie im Abschnitt 3.2 erwähnt, bildet der kritische Diskriminanzwert das arithmetische Mittel der beiden Gruppenmittelwerte. Durch einen Vergleich der berechneten Diskriminanzwerte der Trainingsevents mit dem kritischen Diskriminanzwert werden die *events* in die Gruppen Betrugsverdächtig und Nicht-Betrugsverdächtig eingeteilt. Ist bei diesem Vergleich der ermittelte Diskriminanzwert des Elements kleiner als der kritische Diskriminanzwert, erfolgt eine Zuteilung in die Gruppe der betrugsverdächtigen *events*. Im gegenteiligen Fall wird das *event* der Gruppe der nicht-betrugsverdächtigen *events* zugeordnet.

Schritt 4 – Trainieren des neuronalen Netzwerks:
Die mittels der Diskriminanzfunktion berechneten Diskriminanzwerte werden in diesem Schritt einem neuronalen Netzwerk als Eingabewerte für die Inputknoten übergeben. Als Typ des neuronalen Netzwerks wird ein *Feedforward*-Netzwerk bzw. ein *Multi Layer Perceptron* ohne Rückkopplungen verwendet (siehe Unterabschnitt 3.3.1), da die übergebenen Muster zur Laufzeit keine Zeitabhängigkeit beinhalten. Die zu analysierenden Trainingsmuster an Diskriminanzwerten sind so konzipiert, dass einem bestimmten Inputknoten der Diskriminanzwert des aktuell untersuchten Transaktionsevents übergeben wird und den anderen Knoten die historische Diskriminanzwerte dieses Kunden. Das bedeutet, es herrscht keine Zeitabhängigkeit, da für jeden Inputknoten des neuronalen Netzwerks fest definiert ist, welcher Diskriminanzwert innerhalb der zeitlichen Reihenfolge der Transaktionen als Inputwert entgegengenommen wird. Als Beispiel: Knoten Nr. 1 bekommt den Diskriminanzwert des aktuellen *events*, Inputknoten Nr. 2 den Diskriminanzwert der jüngsten historischen Transaktion bzw. des jüngsten *events* des Kunden vor dem aktuellen *event*. Inputknoten Nr. 3 übernimmt dann wiederum den Diskriminanzwert des *events*, das zeitlich vor dem *event* des Inputknotens Nr. 2 auftrat usw. Somit wird auf Basis von aktuellen und historischen Diskriminanzwerten ein komplettes zeitliches Verhaltensmuster des Kunden dem neuronalen Netzwerk zum Training übergeben. Auf der Grundlage dieser Inputwerte wird durch den Outputwert des Ausgabeknotens des neuronalen Netzwerks bestimmt, ob es sich um einen Betrugsfall handelt oder nicht. Da es sich bei dieser Aufgabenstellung um eine boolesche Entscheidung (d.h. nur zwei Ausprägungen in Form von „ja“ oder „nein“ sind möglich) handelt, wird hierfür nur ein einziger Ausgabeknoten benö-

tigt [Krah98, S. 68]. Die optimale Anzahl an Inputknoten, sowie der Hiddenschichten und wiederum deren Knotenzahlen werden durch Experimente, die im Abschnitt 9.3 beschrieben sind, ermittelt. Die Trainingsmuster sind so konzipiert, dass sie bekannte Verhaltensmuster von Kunden widerspiegeln, d.h. bekannte Muster aus Diskriminanzwerten, die aus dem zeitlichen Ablauf heraus im Nachhinein als Betrugsfälle bzw. als Nicht-Betrugsfälle identifiziert wurden. Als unbekanntes Muster wird im Rahmen dieser Arbeit ein Muster definiert, das sich nicht unter den Trainingsmustern befindet, wie im Abschnitt 6.2 erwähnt.

Da für diese Arbeit die Bekanntheit, ob eine Transaktion ein Betrugsfall oder ein Nicht-Betrugsfall ist, für die Trainingsmuster vorausgesetzt wird, wird für das Training des neuronalen Netzwerks mit dem Backpropagationsverfahren (siehe Unterabschnitt 3.3.2) eine Methode des überwachten Lernens gewählt. Die Backpropagationsmethode ist lt. [Dorf91, S. 39] prädestiniert für die Anwendung auf Assoziationsnetzwerke zu denen auch ein *Multi Layer Perceptron* gehört, da die bekannten Outputwerte als *Teaching Input* des Backpropagationsverfahrens verwendet werden. Diese bekannten Outputwerte werden für das Training in den Ausprägungen 0,0 (Nicht-Betrugsfall) und 1,0 (Betrugsfall) angegeben. Als Aktivierungsfunktion in den Knoten wird die nichtlineare, durchgängig differenzierbare Sigmoidfunktion (siehe Unterabschnitt 3.3.1) verwendet, da Backpropagation bzw. das Gradientenabstiegsverfahren eine durchgängig differenzierbare Funktion voraussetzt [Rey08, S. 26]. Ein weiterer Grund für die Entscheidung für ein *Multi Layer Perceptron* mit mindestens einer Hiddenschicht und nichtlinearer Output- bzw. Aktivierungsfunktion ist, dass lt. [Rume86, S. 1 - 2] mit dieser Kombination beliebige Musterformen abgebildet werden können. Da der Ausgabewert der Sigmoidfunktion immer in dem offenen Intervall zwischen 0,0 und 1,0 liegt (siehe Abbildung 13 im Unterabschnitt 3.3.1) und die übergebenen Inputmuster mit den Zielwerten 0,0 und 1,0 trainiert werden, gibt der Wert des Ausgabeknotens die Betrugswahrscheinlichkeit der aktuellen Transaktion wieder. Die detaillierte Beschreibung der untersuchten Topologien des verwendeten neuronalen Netzwerks als Teil der Betrugserkennungskomponente erfolgt im Abschnitt 9.3 im Rahmen der Vorstellung der experimentellen Ergebnisse.

Schritt 5 – Festlegen des Schwellwerts für den Ausgabewert des Outputknotens und den daraus folgenden Aktionen:

Dieser Schwellwert bestimmt, wann eine bestimmte Aktion ausgelöst wird. In diesem Schritt wird festgelegt, wo dieser Schwellwert im offenen Intervall zwischen 0,0 und 1,0 liegt. Falls die Ausprägung der Trainingsmuster gleich verteilt ist, bietet sich ein Wert von 0,5 an. Wenn das Ziel der Anwendung die Vermeidung einer zu hohen Rate an fälschlicherweise als Betrugsfall klassifizierten Transaktionen ist, sollte der Schwellwert höher

angesetzt werden, z.B. bei 0,9. Darüber hinaus ist im Rahmen dieses Schritts zu definieren, mit welchen Aktionen bei Eintreten eines Betrugsfalls bzw. eines Nicht-Betrugsfalls reagiert werden soll. Bei einem Nicht-Betrugsfall wird die Transaktionsverarbeitung weiter fortgesetzt. Dagegen sind lt. [Bose06, S. 4] sowie nach der Definition des edBPM-Referenzmodells aus Abbildung 1 die möglichen Reaktionen im Fall einer als Betrug identifizierten Transaktion das Auslösen eines Alarms bei einem zuständigen Operator und/oder der Abbruch der Transaktion. Die Schwellwerte, die im Rahmen dieser Arbeit einen Betrugs- oder einen Nicht-Betrugsfall exakt identifizieren, werden in den Abschnitten 9.2 und 9.3 diskutiert. Nach dem Abschluss dieser beschriebenen Vorarbeiten kann das Modell für die Analyse unbekannter Transaktionsevents eingesetzt werden.

7.2 Prozessablauf in der Betrugserkennungskomponente

Der Ablauf beim Analysieren eines neuen bzw. unbekannten *events* in der Betrugserkennungskomponente des Hybrid-Modells ist in Abbildung 38 als Modell dargestellt.

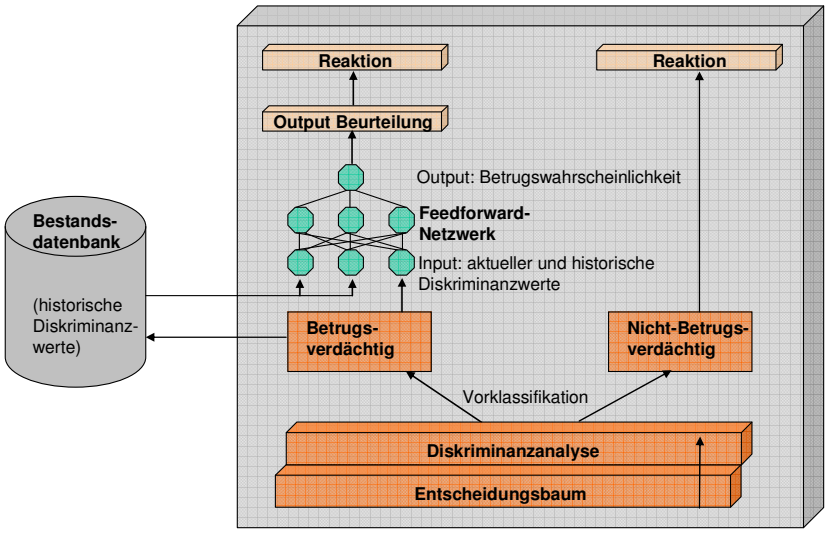

Abbildung 38: Ablaufmodell der Analyseprozesse in der Betrugserkennungskomponente

Im ersten Schritt werden die nichtmetrischen Attribute des korrelierten Transaktionsevents mit Hilfe des Entscheidungsbaums vorklassifiziert. Anschließend werden die metrischen Attribute mittels Diskriminanzanalyse ausgewertet und ebenfalls vorklassifiziert. Zu diesem Zweck werden die relevanten metrischen Attribute in die ermittelte Diskriminanzfunktion eingesetzt und der berechnete Diskriminanzwert – genau wie in der Trainingsphase – mit dem kritischen Diskriminanzwert verglichen.
Im Rahmen dieser Arbeit liegt der Fall vor, dass ein korreliertes Transaktionsevent sowohl metrische als auch nichtmetrische Attribute enthält. Wenn dies nicht der Fall wäre, könnte

z.B. bei Nichtvorhandensein von metrischen Attributen auf die Diskriminanzanalyse und das neuronale Netzwerk verzichtet werden. Umgekehrt könnte beim Fehlen von nichtmetrischen Attributen der Entscheidungsbaum ausgeblendet werden.
Die relevanten Attribute für die Diskriminanzfunktion und der Aufbau des Entscheidungsbaums sollten angepasst werden, wenn sich in der Praxis die Fehlklassifizierungen dieser beiden Methoden häufen.
Wenn mindestens eine der beiden Methoden (Entscheidungsbaum oder Diskriminanzanalyse) das *event* als betrugsverdächtig vorklassifiziert hat, wird im Anschluss sofort der Diskriminanzwert des aktuellen korrelierten *events* zusammen mit den historischen Diskriminanzwerten des betreffenden Kunden an die Inputknoten des neuronalen Netzwerks übergeben. Die historischen Diskriminanzwerte werden zu diesem Zweck aus einer Bestandsdatenbank ausgelesen. Jeder Knoten bekommt als Input einen Diskriminanzwert aus einer bestimmten zeitlichen Reihenfolge übergeben. Der aktuelle und die historischen Diskriminanzwerte werden den Intputknoten des neuronalen Netzwerks in der gleichen zeitlichen Reihenfolge für die Analyse übergeben wie zuvor für den Trainingsprozess. Aus diesem Grund bildet sich die Anzahl der benötigten historischen Diskriminanzwerte aus der Formel:

$$\text{Anzahl historische Transaktionen} = \text{Anzahl Inputknoten} - 1$$

Hat ein Kunde weniger historische Transaktionen aufzuweisen, z.B. ein Neukunde, muss zur Vermeidung von Fehlklassifikationen für diese Kundengruppen jeweils ein eigenes neuronales Netzwerk mit weniger Inputknoten trainiert werden.
Dieses komplette Muster eines Kunden wird im nächsten Schritt durch Vorwärtsaktivierung der Knoten vom neuronalen Netzwerk verarbeitet. Durch die Generalisierungsfähigkeit des neuronalen Netzwerks (siehe Unterabschnitt 3.3.1) ergibt sich für das analysierte unbekannte Muster ein Ausgabewert, der sich im offenen Intervall zwischen 0,0 und 1,0 befindet.
Anschließend wird der Outputwert des neuronalen Netzwerks beurteilt indem dieser Wert mit dem vordefinierten Schwellwert für den Outputwert verglichen wird. Falls der Outputwert größer ist als der definierte Schwellwert wird eine vorher festgelegte Aktion oder ein Prozess gestartet, da das *event* bzw. die Transaktion vom neuronalen Netzwerk als Betrugsfall identifiziert wurde. Lt. dem edBPM-Referenzmodell aus Abbildung 1 können sowohl *Alerts* versendet als auch Prozesse gestartet oder gestoppt werden, siehe [Ammo09b, S. 6 - 7]. In [Bign06, S. 3] wird vorgeschlagen, bei Betrugsverdacht nur zu reagieren, wenn der Transaktionsbetrag eine signifikante Höhe aufweist. Im Rahmen dieser Arbeit werden die Analyseergebnisse in eine Textdatei geschrieben, da es sich hierbei um

eine experimentelle Umgebung handelt, deren Implementierung (momentan) nicht innerhalb einer Bank im Einsatz ist.
Die auftretenden Betrugsmuster in der Bankenbranche verändern sich im Laufe der Zeit [Reol07, S. 11]. Aus diesem Grund spiegelt der Ausgabewert des neuronalen Netzwerks lediglich die Wahrscheinlichkeit wieder, nach der es sich bei der untersuchten aktuellen Transaktion um einen Betrugsfall handelt bzw. handeln könnte.
Eine ursprüngliche Form dieser Architektur ist in [Widd08] beschrieben. In diesem Ansatz ist der Einsatz eines Entscheidungsbaums zur Vorselektion nicht diskutiert, da zum Zeitpunkt dieser Publikation noch keine nichtmetrischen Attribute für das Modell als relevant deklariert wurden. Dadurch war die Kombination von Diskriminanzanalyse und neuronalem Netzwerk als Betrugserkennungskomponente für dieses Vorgängermodell in [Widd08] ausreichend.
In diesem Kapitel wurde das Konzept der Betrugserkennungsanwendung, d.h. sowohl der Trainings- als auch der Laufzeitprozess, allgemein als Lösung dieser Arbeit ohne die detaillierten Parameterausprägungen beschrieben. Zur Sicherstellung der Qualität der Anwendung werden in den nachfolgenden Kapiteln die bereits erwähnten praktischen Experimente diskutiert. Begonnen wird in diesem Zusammenhang mit der experimentellen Umgebung um die Rahmenbedingungen der Versuche darzustellen.

8 Beschreibung der experimentellen Umgebung

Das in Abbildung 36 vorgestellte Hybrid-Modell zur Betrugserkennung setzt sich in seiner implementierten Form aus einer CEP *engine* als Datenversorgungskomponente und einer programmierten Betrugserkennungskomponente zusammen. Die Realisierung des Modells wird in den nachfolgenden Abschnitten erläutert.
Die Anforderungen an ein Softwaresystem werden in funktionale und nicht-funktionale Anforderungen unterteilt. Funktionale Anforderungen definieren die fachlichen Aufgaben, die ein Softwareprodukt erfüllen soll. Sie beschreiben, welche konkreten Ergebnisse die Software bei bestimmten Eingaben und einer definierten Verarbeitung liefern soll. Funktionale Anforderungen sind somit von der gegebenen Aufgabenstellung abhängig. Im Gegensatz dazu legen nicht-funktionale Anforderungen die Eigenschaften bezüglich der Qualität eines solchen Produkts fest. Diese Eigenschaften sind z.B. Anforderungen an die Erweiterbarkeit, Performance, Zuverlässigkeit, Flexibilität, Wartbarkeit oder Bedienfreundlichkeit des Systems und somit unabhängig von den konkreten fachlichen Funktionen des Softwareprodukts. [Andr03, S. 51 - 52; Böhm02, S. 139 - 140; Vers02, S. 19 - 21]

Die funktionalen Anforderungen des Modells dieser Arbeit ergeben sich aus dem Ziel der Identifikation von Betrugstransaktionen beim Online-Banking bei gleichzeitiger Minimierung von *false positive*-Quote. Die konkreten Verarbeitungsschritte und Funktionen zur Umsetzung dieser Anforderungen sind in den Abschnitten 7.1 und 7.2 erläutert.
Zur Erreichung des Ziels dieser Arbeit müssen von der implementierten experimentellen Umgebung nicht nur die funktionalen- sondern auch die folgenden nicht-funktionalen Anforderungen erfüllt werden:

- Schnittstelle von der CEP *engine* zur Betrugserkennungskomponente bzw. zu den verwendeten Analysealgorithmen.

- Kombinierbarkeit der verwendeten Algorithmen innerhalb der Betrugserkennungskomponente.

- Eventverarbeitung in Echtzeit um definierte Reaktionen in Echtzeit gewährleisten zu können.

- Einfach zu konfigurieren um keine teueren externen Beratungsleistungen in Anspruch nehmen zu müssen.

- Installation auf einem Client mit dem Betriebssystem Windows, da die Experimente auf einem handelsüblichen Windows-Rechner durchgeführt werden.

In den nachfolgenden Abschnitten werden die konkreten Implementierungen zur Erfüllung der funktionalen und nicht-funktionalen Anforderungen durch die Datenversorgungs- und Betrugserkennungskomponente des Modells beschrieben.

8.1 Verwendete Versuchsumgebung der Datenversorgungskomponente

Zur Implementierung der Datenversorgungskomponente standen die CEP *engines* verschiedener Hersteller (siehe Tabelle 5) zur Auswahl. Wie im Abschnitt 4.4 erwähnt, wurde für das Hybrid-Modell zur automatisierten Betrugserkennung im Online-Banking die CEP *engine* des Herstellers `StreamBase Inc`. ausgewählt und stand auch für die Durchführung dieser Experimente zur Verfügung.

Deren Produkt *StreamBase Studio* (siehe Tabelle 5), erfüllt alle oben genannten nicht-funktionalen Anforderungen, da:

- Die Betrugserkennungskomponente über eine Adapterschnittstelle angebunden werden kann.

- Der Eventdurchsatz nach Herstellerangaben bei 500.000 *events* pro Sekunde liegt.

- Die Installation auf einem handelsüblichen Windows-Betriebssystem durchgeführt und zur Laufzeit ausgeführt werden kann.

- Eine ausführliche Dokumentation für die Verwendung der Software zur Verfügung steht.

Die Lösung *StreamBase Studio* integriert die SQL-basierte Sprache *StreamSQL* als EPL, wobei für die Modellierung auch eine graphische Komponente verwendet werden kann. Die *StreamBase Platform* besteht aus folgenden Komponenten, die in Abbildung 39 dargestellt sind.

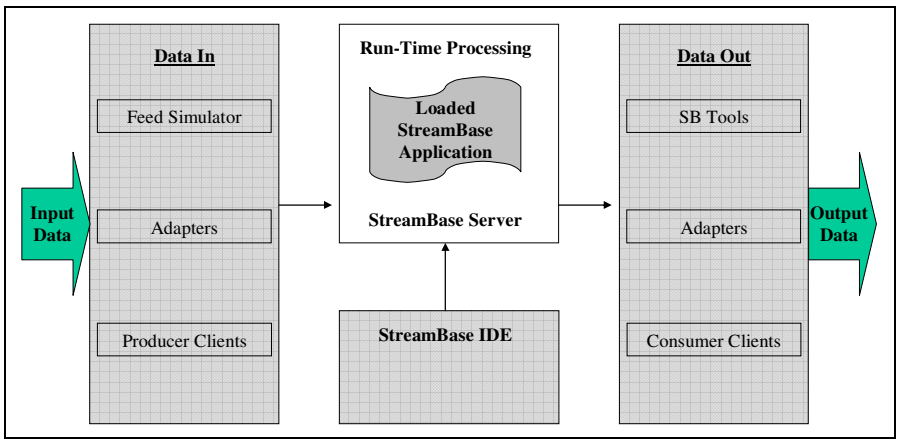

Abbildung 39: Aufbau der StreamBase Platform in Anlehnung an [Stre08, S. 4]

Als Eingabekomponenten bzw. *Data In*-Komponenten dienen in dieser Arbeit *Adapter* (*Inputadapter*), welche die Verbindung zu den Quellsystemen (im Bankenumfeld: Datenbanken der Transaktionen) darstellen. Es existieren verschiedene Adaptertypen für die unterschiedlichen Arten von Datenquellen, z.B. Datenbanken, CSV-Dateien oder Transaktionssysteme. Die *patterns* bzw. die CEP-Anwendung wird in der StreamBase IDE mittels *StreamSQL* programmiert oder graphisch konfiguriert.

Die fertig entwickelte Anwendung (in dieser Arbeit die Konfiguration in Abbildung 40) wird auf den StreamBase-Server geladen und dort ausgeführt. Zur Laufzeit arbeitet der StreamBase-Server die ankommenden *events* in Echtzeit ab. Als Ausgabekomponenten der CEP-Anwendung dienen wiederum *Adapter* (*Outputadapter*) zu den verschiedenen Zielsystemen. [Stre08, S. 4 - 7]
Der Ablauf des implementierten Hybrid-Modells dieser Arbeit ist in Abbildung 40 darstellt.

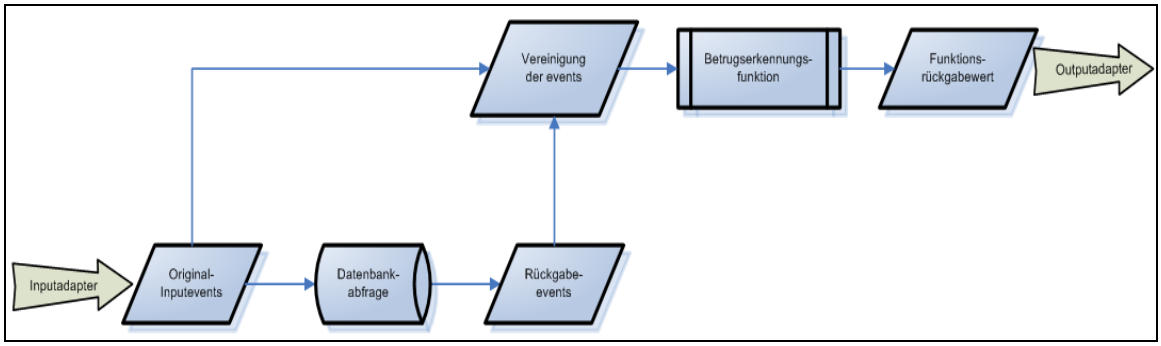

Abbildung 40: Allgemeine Architektur der Betrugserkennungsanwendung der CEP engine

Für das Modell in Abbildung 40 müssen die Daten bzw. *events* aus einer externen Quelle über die *Inputadapter* geladen und an das Format der CEP *engine* angepasst werden. Der geladene Eventstrom wird anschließend kopiert, wodurch ein zweiter Eventstrom entsteht. Dieser Kopiervorgang ist notwendig um die Originalevents mit weiteren *events* aus anderen externen Quellen zu korrelieren. Die Originalattribute der *events* sind – wie im Abschnitt 6.1 erwähnt – für eine exakte Betrugsanalyse mit dem Ansatz dieser Arbeit nicht ausreichend. Ein Eventstrom wird der externen Quelle als Inputdaten übergeben, der andere Eventstrom der Originalevents wird nach dem Beenden der externen Abfrage mit den zurückgelieferten *events* vereinigt. Anschließend werden die neu entstandenen, korrelierten *events* einer Funktion d.h. der Betrugserkennungskomponente übergeben, welche die Betrugsanalyse durchführt. Diese Funktion ist über eine *Java Operator*-Komponente mit der CEP *engine* verbunden. Der Rückgabewert der Funktion bzw. der Betrugserkennungskomponente wird abschließend an den *Outputadapter* gesendet. Die genaue Implementierung dieses Prozesses ist – einschließlich der verwendeten SQL-Abfragen und Ausschnitten aus *StreamBase Studio* – im Anhang 2 dokumentiert.

8.2 Verwendete Versuchsumgebung der Betrugserkennungskomponente

Für die Implementierung der in dieser Arbeit verwendeten Komponente zur Betrugserkennung wurde die Programmiersprache Java verwendet, weil eine JAR-Datei mittels einer *JavaOperator*-Komponente von *StreamBase Studio* problemlos in die CEP *engine* des

ausgewählten Herstellers `StreamBase Inc.` integriert werden kann (es kann allerdings kein *Third Party*-Produkt wie beispielsweise das *Data Mining*-Werkzeug SPSS Clementine des Anbieters `SPSS Inc.` integriert werden). Des Weiteren können mit Java die verwendeten Algorithmen problemlos miteinander kombiniert werden, was in dem Klassendiagramm in Abbildung 41 und dem externen Arbeitsbericht, der die Java-Klassen enthält, zu erkennen ist. *StreamSQL* als integrierte EPL ist bezüglich der Sprachelemente nicht so flexibel wie Java und somit weniger für die Implementierung der Betrugserkennungskomponente geeignet.
Zur Realisierung der funktionalen Anforderungen der Betrugserkennungskomponente in der, in den Abschnitten 7.1 und 7.2 beschriebenen Form, sind verschiedene Java-Hauptklassen notwendig:

- Eine Klasse, welche die *events* von der CEP *engine* übergeben bekommt und nach der Analyse die entsprechenden Ergebnisse zurückliefert.

- Klassen, welche die verwendeten Algorithmen Entscheidungsbaum, Diskriminanzanalyse und neuronales Netzwerk implementieren.

- Eine Klasse, welche den Ablauf der Algorithmen in der entsprechenden Reihenfolge für die Trainingsprozesse koordiniert.

- Eine Klasse, welche den Ablauf der Algorithmen in der entsprechenden Reihenfolge für die Testläufe entsprechend koordiniert.

Für die Umsetzung dieser funktionalen Anforderungen liegt der Betrugserkennungskomponente folgendes Klassendiagramm zugrunde, das in Abbildung 41 dargestellt ist. Die Java-Klassen wurden unter Verwendung der Entwicklungsumgebung Eclipse programmiert. Aus Übersichtsgründen wurde in Abbildung 41 auf die Darstellung und Erklärung der einzelnen Attribute und Methoden der Klassen verzichtet. Der komplette Javacode der einzelnen implementierten Klassen der JAR-Datei ist als externer Arbeitsbericht einsehbar.

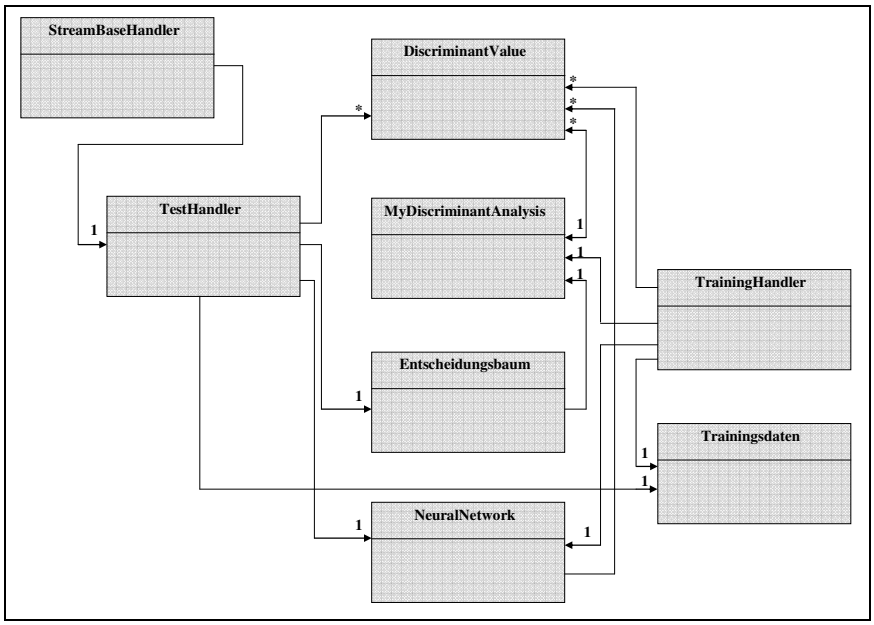

Abbildung 41: Klassendiagramm der experimentellen Umgebung

Für den Prozess der Testeventauswertung bzw. der Laufzeitphase der Betrugserkennungskomponente bildet die Klasse StreamBaseHandler die Verbindung mit der *Java-Operator*-Komponente der CEP *engine.* Die Klasse StreamBaseHandler nimmt in diesem Zusammenhang die Testevents entgegen und leitet sie an die Klasse TestHandler weiter. Diese Klasse übergibt wiederum die Testevents an die Klasse Entscheidungsbaum. Dort werden die *events* anhand der Struktur des Entscheidungsbaums klassifiziert. Das Ergebnis wird an die Klasse TestHandler zurückgegeben. Anschließend erfolgt eine weitere Übergabe der *events* an die Klasse MyDiscriminantAnalysis über das DicriminantValue-Objekt. In dieser Klasse wird der Diskriminanzwert berechnet und für die weitere Verarbeitung in einem Objekt der Klasse DiscriminantValue gespeichert.

Anhand des Attributs Gruppenzugehörigkeit des DiscriminantValue-Objekts, das von der Klasse MyDiscriminantAnalysis gesetzt wird, entscheidet die Klasse TestHandler, ob das aktuell analysierte *event* sofort an die Klasse StreamBaseHandler zur Rückgabe an die CEP *engine* übergeben wird (bei Vorklassifizierung als Nicht-Betrugsevent durch Diskriminanzanalyse und Entscheidungsbaum) oder an die Klasse NeuralNetwork zur weiteren Analyse weitergereicht wird (bei Vorklassifizierung als Betrugsevent durch Diskriminanzanalyse oder Entscheidungsbaum). Falls das *event* weiter analysiert wird, lädt die Klasse TestHandler die entsprechenden Diskriminanzwerte der historischen Transaktionen des aktuellen Kunden aus einer externen Datenbank und übergibt diese zusammen mit dem DiscriminantValue-Objekt an die Klasse NeuralNetwork. Diese Klasse enthält die aktuell trainierten Gewichtseinstellungen der optimalen Netzwerktopologie. Mit diesen Attributen berechnet die Klasse NeuralNetwork aus der übergebenen Kombination von Diskriminanzwerten den Ausgabewert des neuronalen Netzwerks mittels einer implementierten

Funktion des *Feedforward*-Prozesses. Der berechnete Ergebniswert wird der Klasse TestHandler zurückgeliefert, die wiederum das Resultat an die Klasse StreamBaseHandler als Schnittstelle zur CEP *engine* übergibt.
Für den Trainingsprozess der Betrugserkennungskomponente lädt die Klasse Trainingsdaten die aktuellen Trainingsevents ebenfalls aus einer externen Datenbank und übergibt diese an die Klasse TrainingHandler. Diese Klasse ruft wiederum die Klasse MyDiscriminantAnalysis auf, welche die Diskriminanzfunktion, den kritischen Diskriminanzwert und die Diskriminanzwerte der Trainingsevents berechnet. Auch hier erfolgt die Rückgabe eines Diskriminanzwerts an die Klasse TrainingHandler in Form eines DiscriminantValue-Objekts. Anschließend wird das neuronale Netzwerk mit Mustern aus aktuellen und historischen Diskriminanzwerten unter Anwendung des Backpropagationsverfahrens trainiert, wobei in den Trainingsevents als zusätzliches Attribut mit angegeben ist, ob es sich um einen Betrugsfall handelt (Flag=1) oder nicht (Flag=0). Nach Abschluss der Trainingsphase werden die neu berechneten Gewichte für die Analyse unbekannter *events* bzw. Testevents zur Laufzeit verwendet. Für weiterführende Literatur zu Klassendiagrammen sei auf [Kech06], [Andr03] und [Böhm02] verwiesen.
In diesem Kapitel wurde die Umgebung beschrieben, in der die Experimente ausgeführt werden. Es zeigt sich, dass sowohl die StreamBase-Entwicklungsumgebung für die Datenversorgungskomponente (in Form der CEP *engine StreamBase Studio*) als auch die programmierte Betrugserkennungskomponente (in Form einer mit Eclipse entwickelten JAR-Datei) zur Ausführung der Experimente und zur Erfüllung der funktionalen und nichtfunktionalen Anforderungen notwendig und geeignet sind. Im nächsten Kapitel werden darauf aufbauend die Durchführung der Versuche und deren Resultate beschrieben sowie entsprechende Schlussfolgerungen daraus gezogen.

9 Vorstellung der Experimente und deren Ergebnisse

Dieses Kapitel enthält das Vorgehen bei der Gewinnung der Transaktionsevents, die exakten Parameter der beschriebenen Betrugserkennungskomponente sowie die Ergebnisse nach dem Ausführen der Experimente.

9.1 Beschreibung der Simulationsbedingungen

In diesem Abschnitt wird der Aufbau der Simulation der den Experimenten zugrundeliegenden Daten bzw. *events* beschrieben. Vor der Diskussion der Simulationsbedingungen wird das Thema Simulation allgemein diskutiert.

9.1.1 Allgemeine Aspekte einer Simulation

Der Begriff Simulation entstammt dem lateinischen Wort *simulare* und bedeutet etwas vortäuschen [Eise04, S. 2]. In der Literatur existieren viele Definitionen einer Simulation, wobei diese oftmals vom jeweiligen Fachgebiet abhängig ist. Unabhängig von einem bestimmten Themengebiet dient eine Simulation dem Ziel der Nachstellung und/oder der Vorausberechnung eines bestimmten Szenarios [Bung08, S. 1; Eise04, S. 1 - 4; Boss04, S. 16]. Die Anwendungsgebiete von Simulationen sind vielfältig und reichen von der Raumforschung über den medizinischen Sektor bis zur Klimaforschung [Bung08, S. 2; Boss04, S. 19; Eise04, S. 4]. Zur Durchführung einer Simulation sind mehrere Schritte notwendig, die in [Bung08, S. 3] als Simulationspipeline bezeichnet werden. Dagegen verwendet [Eise04, S. 11] in diesem Zusammenhang den Begriff vom *Life Cycle* einer Simulation. Der Ablauf des Simulationsprozesses ist folgendermaßen gekennzeichnet, siehe [Eise04, S. 11 - 12; Bung08, S. 2 - 4]:

Schritt 1 – Modellierung:
In diesem Schritt erfolgt die formale Beschreibung der Parameter und der Ziele der Simulation.

Schritt 2 – Berechnung:
In diesem Schritt wird das Modell aufbereitet bzw. konzipiert, damit es im nächsten Schritt umgesetzt werden kann.

Schritt 3 – Implementierung:
In diesem Schritt werden die zuvor konzipierten Berechnungs- bzw. Erstellungsvorschriften auf Basis der festgelegten Zielarchitektur programmiert.

Schritt 4 – Visualisierung:
In diesem Schritt werden die entstandenen Simulations- bzw. die Ergebnisdaten interpretiert.

Schritt 5 – Validierung:
In diesem Schritt wird geprüft, wie plausibel und verlässlich die Ergebnisse der Simulation sind, d.h. es werden die Ergebnisdaten, der Generierungsalgorithmus sowie die Programmierung geprüft.

Schritt 6 – Einbettung:
In diesem Schritt erfolgt abschließend die Verwendung der Simulation in dem Kontext, für den sie generiert wurde.

Simulationen werden benötigt, da reale Daten oftmals nicht vorhanden oder verfügbar sind, z.B. in der Astrophysik bei der Erforschung des Lebenszyklus fremder Galaxien, weil die benötigten Dimensionen an Raum- und Zeitskalen nicht zur Verfügung stehen [Bung08, S. 2].
Im Rahmen der Anwendung von neuronalen Netzen werden Simulationen verwendet um entsprechende Netzwerktopologien für die Lösung von bestimmten Anwendungsproblemen zu konfigurieren. Allerdings besteht in diesem Zusammenhang die Problematik, dass (noch) keine konkreten Standardnetzstrukturen für bestimmte Problemstellungen bekannt sind. Daher muss ein System, welches neuronale Netzwerke zur Problemlösung einsetzt, in der Lage sein, sowohl die verschiedenen möglichen Parameterausprägungen des Netzwerks (z.B. Lernfaktor, Netzstruktur usw.) einzustellen als auch die benötigten Tests durchzuführen. [Haun98, S. 142 - 143]
Diese von [Haun98, S. 142 - 143] genannte Parametrisieranforderung wird von der Lösung im Rahmen dieser Arbeit berücksichtigt, da die Parameter der Betrugserkennungskomponente flexibel konfigurierbar sind (siehe Abschnitt 8.2).
Im Anwendungsgebiet der Betrugserkennung werden in Verbindung mit neuronalen Netzwerken ebenfalls Simulationen bei Bedarf verwendet. Ein konkretes Beispiel bildet die Forschung von [Lund03]. Bei dieser Arbeit wurden sowohl Betrugs- als auch Nicht-Betrugsdaten im Umfeld eines *Video on demand*-Dienstes simuliert um damit ein neuronales Netzwerk zu trainieren. Das Trainingsergebnis wurde anschließend mit realen und simulierten Daten getestet. Das Ziel von [Lund03] war zu beweisen, dass simulierte Daten ebenso effektiv zur Betrugserkennung verwendet werden können wie reale Daten. In dem Anwendungsgebiet *Video on demand* setzt sich die Art des Betrugs aus folgenden Formen zusammen, siehe [Lund03, S. 4]:

a) *Break-in fraud*: Hierbei stiehlt der Betrüger die Zugangsdaten eines Benutzers und kauft Videos in seinem Namen. In diesem Fall zeigt sich der Betrug durch eine überdurchschnittlich häufige Benutzung des Dienstes.

b) *Billing fraud*: Hierbei bricht der Betrüger in den Bezahlserver ein um die nötigen Zahlungen zu vermeiden. In diesem Fall zeigt sich der Betrug durch Lücken in der Protokolldatei der Zahlungseingänge.

c) *Illegal redistribution fraud*: Hierbei bezieht ein Kunde ein Video und verteilt es an andere Anwender, die für den Service nicht bezahlen bzw. den Betrüger bezahlen.

d) *Failed logins*: Hierbei erfolgt eine Anmeldung von Seiten des Betrügers mit „*dummy*“ Zugangsdaten in der Hoffung, als ein Benutzer akzeptiert zu werden.

Die zu analysierenden Daten setzten sich aus Elementen der Anwenderprofile zusammen, die beim Kauf eines Videos von einem neuronalen Netzwerk als alleinstehende Analysekomponente ausgewertet werden. Diese Profildaten wurden für das Training des neuronalen Netzwerks durch den Einsatz der folgenden Ablaufarchitektur aus Abbildung 42 generiert bzw. simuliert.

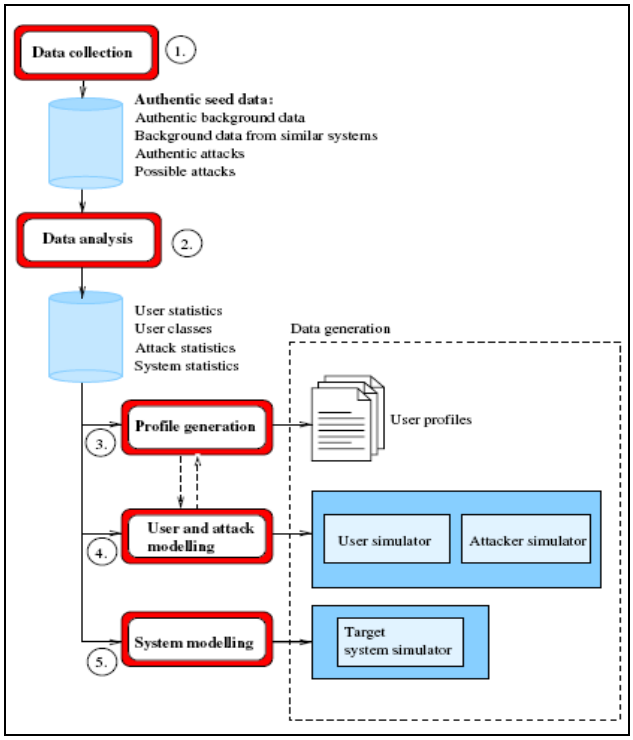

Abbildung 42: Ablaufarchitektur der Simulationsgenerierungsanwendung aus [Lund03, S. 3]

Im ersten Schritt wurden Echtdaten gesammelt, die relevant für den entsprechenden Anwendungsfall sind. Im zweiten Schritt erfolgte die Identifikation der betrugsrelevanten Attribute der Transaktionsprofile der *Video on demand*-Kunden. Diese Identifikation wurde manuell durch die Autoren durchgeführt und brachte folgende Resultate:

- Summe der erfolgreichen Loginversuche
- Summe der fehlgeschlagenen Loginversuche
- Summe der erfolgreichen Videobestellungen

- Summe der fehlgeschlagenen Videobestellungen
- Summe der Benachrichtigungen von Videolieferungen
- Summe der Benachrichtigungen von Zahlungen
- Verhältnis zwischen Upload- und Downloadbytes

Im dritten Schritt wurden die betrugsverdächtigen und nicht-betrugsverdächtigen Ausprägungen der relevanten Attribute identifiziert und daraus Benutzerprofile erstellt. Anschließend erfolgte im vierten Schritt die Modellierung der konkreten Betrugsmuster der aus den Benutzerprofilen. Im letzten Schritt wurde das Zielsystem für die Simulationsdaten konzipiert. Diese Ablaufarchitektur wurde im Rahmen einer früheren Arbeit der Autorengruppe erstellt und dort beschrieben, siehe dazu [Lund02].
Mit den generierten Simulationsdaten wurde ein neuronales Netzwerk individuell für jeden oben genannten Betrugstyp trainiert. Die Netzwerktopologie besteht aus sieben Eingabeknoten (jeder der Knoten nimmt eines der oben beschriebenen betrugsrelevanten Attribute entgegen), einer Hiddenschicht mit sieben Knoten und einem Ausgabeknoten, der den Betrugsstatus des analysierten Eingabemusters wiedergibt. Getestet wurde das Netzwerk jeweils mit Echtdaten, wobei die Autoren gezeigt haben, dass Simulationsdaten für das Training eines neuronalen Netzwerks erfolgreich eingesetzt werden können. Für die exakten Ergebnisse der einzelnen Betrugsarten sei auf [Lund03, S. 6 - 8] verwiesen. Diese Resultate zeigen, dass bereits „kleinere“ neuronale Netzwerke mit nur einer Hiddenschicht in der Lage sind, komplexe Muster zu erlernen und erfolgreich wiederzuerkennen.
Für das Training eines neuronalen Netzwerks ist nach Aussage der Autoren von [Lund03, S. 2] wichtig, dass die simulierten Betrugsfälle den realen Betrugsfällen entsprechen, gleiches gilt für Nicht-Betrugsfälle. Das Verhältnis von Betrugsdaten und Nicht-Betrugsdaten sollte lt. [Lund03, S. 2] ausgeglichen sein, damit beide Fälle entsprechend trainiert und nicht eine Fallart überrepräsentiert ist.
Die Autoren nennen als entscheidende Vorteile in der Verwendung einer Simulation, dass in simulierten Daten bekannte Betrugskonstellationen mit integriert werden, die möglicherweise in einer Menge von Echtdaten nicht enthalten sind. Darüber hinaus kann nach Aussage von [Lund03, S. 2] die Situation auftreten, dass die benötigte Menge an Betrugsdaten innerhalb der Echtdaten nicht vorhanden ist. Dagegen benötigen bestimmte Algorithmen wie z.B. neuronale Netzwerke einen hohen Anteil von Betrugsdaten (z.B. 50%) um optimal trainiert werden zu können.

Allgemein sehen die Autoren von [Lund03] drei konkrete Anwendungsszenarien für welche die Verwendung von Simulationsdaten prädestiniert ist. Diese sind nachfolgend aufgezählt [Lund03, S. 1]:

a) Anpassung und Training des Analysesystems, da Echtdaten oftmals nicht in der benötigten Form ausreichend vorhanden sind.

b) Erzeugung von möglichen Variationen der Ausprägungen der Datenmenge (in diesem Fall bei Betrugsdaten) um die Erkennungsgenauigkeit des Analysesystems zu erhöhen und alle bekannten Situationen abzudecken.

c) Erzeugung von weiteren Daten um Belastungstests (engl.: *Benchmarks*) mit dem Analysesystem durchzuführen.

Eine Diskussion der in diesem Abschnitt vorgestellten allgemeinen Literatur zu Simulationen mit den konkreten Gegebenheiten dieser Arbeit erfolgt im folgenden Unterabschnitt.

9.1.2 Simulation im Kontext dieser Arbeit

Die für die Experimente benötigten Transaktionsevents werden im Rahmen dieser Arbeit sowohl für Trainings- als auch für Testzwecke simuliert, da kein Bankpartner aus den verschiedensten Gründen (in der Regel Sicherheitsaspekte und/oder Bedenken aufgrund einer möglichen negativen Außendarstellung) bereit ist, für die Durchführung dieser Experimente reale Daten bzw. *events* zur Verfügung zu stellen. Die formale Beschreibung dieser Simulation (entspricht Schritt 1 beim Erstellen einer Simulation, siehe Unterabschnitt 9.1.1) sowie die verwendeten Betrugsmuster werden auf Basis der Interviews mit Experten aus den Bereichen Betrugsmanagement und Compliance verschiedenster Kreditinstitute und Verbände erörtert (siehe Anhang 1). Der Grund für die Verwendung der Angaben aus Experteninterviews ist, dass in der Literatur keine konzeptionellen Grundlagen zur exakten Modellierung einer Simulation im Kontext dieser Arbeit beschrieben sind. Das Simulationsmodell ist auf Basis der formalen Beschreibung der Interviewpartner für die Realisierung im Rahmen dieser Arbeit folgendermaßen konzipiert (entspricht Schritt 2 beim Erstellen einer Simulation, siehe Unterabschnitt 9.1.1):
Es werden im Rahmen der Simulation nur Konten von natürlichen Personen bzw. Privatpersonen als Transaktionsbasis herangezogen, weil diese Gruppe nach Meinung der Betrugsexperten häufiger Opfer von Identitätsdiebstahl beim Online-Banking ist als die Gruppe der Firmenkunden. Als Ergebnis der Simulation werden insgesamt 60.000 Tran-

saktionsevents erzeugt, wobei 40.000 *events* als Trainingsmenge und die restlichen 20.000 *events* als Testmenge fungieren. Das Verhältnis von einem Drittel Testmenge zu zwei Drittel Trainingsmenge wird in [Kara01, S. 275] vorgeschlagen, weil diese Aufteilung lt. [Kara01, S. 275] eine optimale Auswertung des Trainingserfolgs zulässt und gleichzeitig alle repräsentativen Muster in der Trainingsmenge enthalten bleiben. Bei der vollständigen Simulation handelt es sich um eine Eventmenge für insgesamt 12.000 Bankkunden. Die Aufteilung, ob ein bestimmter Kunde zur Trainings- oder zur Testmenge gehört erfolgt nach dem Zufallsprinzip. Für jeden Kunden sind insgesamt fünf Online-Überweisungen simuliert (eine aktuelle und vier historische Transaktionen). Nach Aussage der interviewten Experten sind wenigstens vier historische Transaktionen notwendig um die entsprechende Historie abbilden zu können. Aus diesem Grund werden keine Neukunden mit weniger als vier historischen Kunden generiert, da diese Kunden kein geeignetes Trainingsmuster aufweisen. Diese Zahl von vier historischen Transaktionen wurde im Laufe der Experimente nicht erhöht um den Einfluss der aktuellen Transaktion auf das Analyseergebnis des neuronalen Netzwerks nicht weiter zu verringern.

Die Menge der am Tag insgesamt zu verarbeitenden Transaktionen ist abhängig von der Größe und der Kundenanzahl des Kreditinstituts. Bei 2.476,3 Mio. Überweisungen in Deutschland im Jahr 2007 via Online-Banking durch Nichtbanken [Deut08b] ergibt das ca. 6,78 Mio. Online-Überweisungen täglich. Die simulierte Menge von 4.000 aktuell zu analysierenden Testtransaktionen am Tag entspricht daher dem Bild einer mittleren Geschäftsbank oder Sparkasse, für die ein Anbieter für Bankrechenzentrumsdienstleistungen (wie z.B. `Fiducia IT AG` oder `Finanz Informatik GmbH & Co. KG`) die Betrugsanalyse durchführt. Die Echtzeitfähigkeit bei der Betrugserkennung ist in diesem Zusammenhang für ein Kreditinstitut mit 4.000 zur untersuchenden Transaktionen am Tag ebenfalls relevant, da lt. der interviewten Experten die Online-Überweisungen unmittelbar nach der Freigabe durch den Kunden verrechnet werden. Das simulierte Kreditinstitut besitzt ca. 40.000 Privatkunden mit Girokonto, von denen ca. 45,0% die angebotene Möglichkeit des Online-Bankings unterschiedlich oft nutzen. Diese Zahlen spiegeln eine Genossenschaftsbank oder Sparkasse mit dem Einzugsgebiet einer mittelgroßen deutschen Stadt mit ca. 200.000 bis 250.000 Einwohnern und einem entsprechenden Flächenlandkreis im Hintergrund wieder. Das simulierte Kreditinstitut weist einen Marktanteil von knapp 15,0% in dieser beschriebenen Region auf und erreicht eine Bilanzsumme von knapp vier Mrd. Euro.

Da keine genauen Statistiken über die Tagestransaktionsmenge eines Kreditinstituts frei verfügbar sind, wird diesbezüglich für die Simulation exemplarisch auf die Aussage der Experten aus den Interviews zurückgegriffen, die zum Teil in einem ähnlichen Kreditinstitut wie oben angegeben tätig sind. Der exakte reale Anteil der Betrugstransaktionen an

der Gesamtheit aller Online-Transaktionen eines Kreditinstituts durfte oder konnte von den befragten Experten nicht weitergegeben werden, liegt aber lt. [Agge06, S. 2] unter 1,0%. Für das Training wird allerdings eine Gleichverteilung der Trainingsmenge zugrunde gelegt, d.h. 4.000 Kunden mit einer Betrugstransaktion und 4.000 Kunden mit einer Nicht-Betrugstransaktion als aktuelle Überweisung. Diese Gleichverteilung ist sowohl nach Meinung der Experten als auch nach Aussage von [Lund03, S. 2] die optimale Verteilung um zu verhindern, dass das neuronale Netzwerk einseitig zu stark mit einer bestimmten Transaktionsart trainiert wird. Dies soll gewährleisten, dass das Identifizieren von Betrugsfällen ebenfalls fehlerfrei möglich ist, da die Auftretungswahrscheinlichkeit von Betrugstransaktionen lt. [Agge06, S. 2] generell geringer ist als von Nicht-Betrugstransaktionen.

Die Trainings- und Testmenge weisen die gleichen Simulationsparameter auf, weil beide einen Teil der simulierten Gesamtmenge von 60.000 *events* (40.000 Trainings- und 20.000 Testevents) bilden. Innerhalb der 20.000 Testevents von 4.000 Kunden existieren 2.000 Kunden mit einem Betrugsfall und 2.000 Kunden mit einem Nicht-Betrugsfall als aktuelle Transaktion. Diese Gleichverteilung erlaubt ein aussagekräftiges Bild des Trainingserfolgs der Betrugserkennungsanwendung. Die restlichen 16.000 historischen Transaktionen der Testmenge bestehen – genau wie die historischen Transaktionen der Trainingsmenge – komplett aus Nicht-Betrugsfällen, da anhand der historischen Transaktionen gewöhnliche Verhaltensmuster der Kunden gebildet werden.

Die Nicht-Betrugstransaktionen sind in der Höhe des Transaktionsbetrags zufällig verteilt, wobei die Wahrscheinlichkeit für einen Betrag über 5.000 Euro geringer ist, da die Girokontostände seltener diese Höhe aufweisen (siehe unten). Bei der Generierung der Transaktionsbeträge ist das Simulationsprogramm so konzipiert, dass der maximal verfügbare Betrag (Summe aus Dispositionskreditlimit und Kontostand) nicht überschritten wird. Es wird hierbei davon ausgegangen, dass dieser maximal verfügbare Betrag auch den maximalen Transferbetrag darstellt. Somit hat das simulierte Kreditinstitut kein Standardtransferlimit festgelegt und auch für die simulierten Kunden ist kein maximal übertragbarer Betrag definiert. Der Grund dafür ist, dass die befragten Experten keine Angaben über die Verteilung von Transferlimits (Anteil und Höhe) innerhalb ihres Kundenstamms gemacht haben und diesbezüglich auch allgemein keine Statistiken verfügbar sind. Des Weiteren sind Kunden mit einem Transferlimit beispielsweise in Höhe von 500 oder 1.000 Euro für Betrüger und somit auch für die Betrugsanalyse weniger interessant, da die Identitätsdiebe bei einem maximalen Transferbetrag unter 1.500 Euro in der Regel keinen Betrugsfall und die damit verbundene mögliche Aufmerksamkeit riskieren (siehe Abschnitt 2.2).

Die Betrugstransaktionen sind nach den Mustern simuliert, die im Abschnitt 2.2 beschrieben wurden.
Der durchschnittliche Nettogehaltseingang ergibt nach den Angaben aus den Interviews folgendes Bild, das auf Basis der Interviews für Kunden gültig ist, welche die Möglichkeit des Online-Bankings nutzen:

- 10,0% der Online-Banking-Kunden besitzen keinen regelmäßigen Gehaltseingang.
- 12,0% der Online-Banking-Kunden besitzen ein monatliches Nettogehalt zwischen 1 und 900 Euro.
- 12,0% der Online-Banking-Kunden besitzen ein monatliches Nettogehalt zwischen 900 und 1.500 Euro.
- 15,0% der Online-Banking-Kunden besitzen ein monatliches Nettogehalt zwischen 1.500 und 2.000 Euro.
- 25,0% der Online-Banking-Kunden besitzen ein monatliches Nettogehalt zwischen 2.000 und 3.000 Euro.
- 26,0% der Online-Banking-Kunden besitzen ein monatliches Nettogehalt von mehr als 3.000 Euro.

Diese Angaben eines Interviewpartners dienen in dieser Form als Grundlage für die Berechnung des Dispositionskredits der simulierten Bankkunden, da hierfür ebenfalls keine frei zugängliche Statistik speziell über die Gruppe der Nutzer von Online-Banking verfügbar ist.
Der Dispositionskredit ist der maximale negative Betrag, den ein Girokonto aufweisen kann. Überweisungen, welche den Kontostand auf einen Betrag unterhalb des verfügbaren Dispositionskredits setzen würden, werden von den Kreditinstituten nicht akzeptiert. Die Höhe des Dispositionskredits kann vom Bankkunden individuell ausgehandelt werden. In den meisten Fällen orientiert sich der Dispositionskredit aber an der Höhe des regelmäßigen Gehaltseingangs. Je nach Kreditinstitut kann dieser zwischen dem einfachen und vierfachen des monatlichen Nettogehalts liegen. [Prät07, S. 140]
Die Kreditinstitute der interviewten Experten gewähren einen Dispositionskredit in Höhe des dreifachen Gehaltseingangs, daher wird dieser Wert ebenfalls für diese Simulation zu Grunde gelegt. Die Summe aus Dispositionskredit und Kontostand ergibt – wie oben erwähnt – den maximal verfügbaren Transaktionsbetrag, der als relevantes Attribut für die Betrugserkennung ausgewertet wird, siehe dazu Abschnitt 9.2.
Die erzeugten Kontostände der simulierten Kunden orientieren sich am Nettoeinkommen. Für die Kontostände der jeweiligen Online-Girokonten wurde von einem Interviewpartner

folgende Verteilung als repräsentativ für die Nutzer von Online-Banking angesehen (auch hier ist keine frei verfügbare Statistik speziell für die Benutzergruppe von Online-Banking vorhanden):

- Ca. 15,0% der Online-Banking-Kunden weisen einen negativen Girokontostand auf.
- Ca. 20,0% der Online-Banking-Kunden weisen einen Girokontostand zwischen 0 und 2.000 Euro auf.
- Ca. 50,0% der Online-Banking-Kunden weisen einen Girokontostand zwischen 2.000 und 5.000 Euro auf.
- Ca. 15,0% der Online-Banking-Kunden weisen einen Girokontostand von über 5.000 Euro auf.

Die Simulation ist so aufgebaut, dass die Wahrscheinlichkeit eines höheren Kontostands bei einem höheren Einkommen ebenfalls höher ist und umgekehrt. Die Kontostände verändern sich – ausgehend vom ersten simulierten Kontostand vor der ältesten simulierten Transaktion des Kunden – innerhalb der zeitlichen Transaktionsreihenfolge um den jeweiligen Transaktionsbetrag. Hierbei werden die monatlichen Gehaltszahlungen bei Monatswechsel zusätzlich als Kontozugang der Online-Banking-Kunden berücksichtigt und als Folge davon der Kontostand angepasst (die Zahlungseingänge auf Basis von Gehaltszahlungen sind nicht als historische Transaktionen simuliert, da für die Betrugsanalyse nur die Zahlungsausgänge der Kunden von Bedeutung sind). Der durchschnittliche Transaktionsbetrag wird auf Basis des ältesten Kontostands und der Höhe der Beträge der historischen Transaktionen ermittelt, da die historischen Überweisungen keine Betrugsfälle darstellen.
Die durchschnittliche Zahl der Online-Überweisungen im Monat ist von Kunde zu Kunde verschieden. Das Intervall reicht von keiner Aktivität bis zu ca. 15 Transaktionen bzw. Zugriffe auf das Online-Konto, in Ausnahmefällen können es bei Privatkunden sogar etwas mehr sein und ist lt. Aussage der Experten in der Regel gleich verteilt. Im Jahr 2008 nutzten lt. [Bank08] ca. 48,0% der Onlinekonto-Kunden mehrmals in der Woche das Angebot des Online-Bankings, z.B. zur Kontrolle der Kontenbewegungen oder zum Ausführen von Überweisungen. Daher wird in der Simulation zufällig ein monatlicher Wert von 1 bis 15 für die Trainings- und Testkunden vergeben, wobei in der Realität dieser Wert von Bank zu Bank je nach Kundenstruktur variieren kann.
Auf Basis dieser genannten Parameter wurden die analysierten Transaktionen mit Hilfe eines Simulationsprogramms auf Basis einer Oracle 10g Datenbank und der Program-

miersprache PL/SQL implementiert (entspricht Schritt 3 beim Erstellen einer Simulation, siehe Unterabschnitt 9.1.1).
Eine visuelle Überprüfung der Ergebnisdaten der Simulation (entspricht Schritt 4 beim Erstellen einer Simulation, siehe Unterabschnitt 9.1.1) sowie des Simulationsalgorithmus und des Programmcodes (entspricht Schritt 5 beim Erstellen einer Simulation, siehe Unterabschnitt 9.1.1) ergab, dass die Simulation fehlerfrei und wie gewünscht nach den oben beschriebenen Kriterien erstellt wurde. Zur Sicherstellung der Qualität wurden mittels Kontrollabfragen auf den generierten Simulationsdatenbestand die Verteilung der einzelnen Attributausprägungen sowie die Anzahl der erzeugten Transaktionsmenge für Betrugs- und Nicht-Betrugsfälle getestet. Ebenso wurde der Code des Simulationsprogramms manuell nochmals auf logische Korrektheit analysiert. Aufgrund der Interviews von Experten im Bereich der Betrugserkennung verschiedener Kreditinstitute ist diese Simulation für die oben beschriebene Bank zum Zeitpunkt ihrer Erstellung plausibel. Bei der Abbildung der Simulation auf die Realität muss berücksichtigt werden, dass sowohl manche Bankkunden Transferbetragslimits für ihr Online-Konto definieren (siehe Unterabschnitt 2.4.2) als auch manche Kreditinstitute Standard-Tagestransfer- oder Transferbetragslimits für das Online-Banking festsetzen (z.B. 5.000 Euro bei der `DAB bank AG` [Dab09]). In diesen Fällen würde für die Betrugserkennung der maximale Transferbetrag nicht durch die Summe aus Kontostand und Dispositionskredit sondern durch das vorgebende Transferbetragslimit festgelegt, falls das Transferlimit kleiner ist als die Summe aus Kontostand und Dispositionskredit.
Zusätzlich besteht in der Realität ein geringer Anteil an Kunden mit weniger als vier historischen Transaktionen. Zur Vermeidung von Fehlklassifikationen sollte für diese Kundengruppen jeweils ein eigenes neuronales Netzwerk mit weniger Inputknoten trainiert und für die Analyse verwendet werden. Im Rahmen dieser Arbeit wurde auf die Entwicklung neuronaler Netzwerke mit weniger als fünf Eingabeknoten verzichtet, aufgrund eines – lt. Experteninterviews – geringen Anteils von Kunden mit weniger als vier historischen Überweisungen.
Darüber hinaus besitzen die verschiedenen Banken jeweils eine abweichende Kundenstruktur mit variierender Vermögens- und Transferbetragsverteilung etc.
Ebenso ist in der Realität keine Gleichverteilung von Betrugsfällen zu Nicht-Betrugsfällen zu beobachten. Diese Verteilung wurde – wie oben erwähnt – für die Simulation dennoch in dieser Form festgelegt um zum einen die Diskriminanzfunktion mit zwei gleich großen Gruppen erzeugen zu können (siehe Abschnitt 3.2) und zum anderen das neuronale Netzwerk gleichmäßig trainieren und anschließend den Trainingserfolg besser abprüfen zu können.

Darüber hinaus ist zu beachten, dass sich sowohl die Betrugsmuster als auch die angegebenen Verteilungen z.B. bei Kontoständen oder Transferbeträgen durch politische oder wirtschaftliche Veränderungen (z.B. Steuererhöhungen bzw. -senkungen oder Wirtschaftskrise usw.) im Laufe der Zeit verändern.
Aus diesen Gründen bleibt ein geringer Restfehler bei der Übertragung der Simulation in die Realität. Allerdings bringt die Verwendung von simulierten Daten bzw. *events* den von [Lund03, S. 2] genannten Vorteil, dass alle von den Interviewpartnern genannten Merkmale von Betrugs- und Nicht-Betrugstransaktionen berücksichtigt werden können. Dies kann bei einem Einsatz von Echtdaten nicht gewährleistet werden, da nicht sicher ist, ob diese reale Datenmenge alle möglichen Betrugsmuster beinhaltet (siehe Unterabschnitt 9.1.1).
Die Einbettung der Simulationsdaten in den Kontext dieser Arbeit (entspricht Schritt 6 beim Erstellen einer Simulation, siehe Unterabschnitt 9.1.1) wird im folgenden Abschnitt diskutiert.

9.2 Beschreibung des Ablaufs der Experimente

Wie in den Abschnitten 7.1 und 7.2 beschrieben, besteht das Hybrid-Modell dieser Arbeit aus CEP-Technologie sowie einer Kombination von Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk. Der Ablauf, die Parametereinstellungen und die Struktur der Experimente für diese drei Algorithmen sind in den nachfolgenden Unterabschnitten beschrieben.

9.2.1 Struktur des Entscheidungsbaums

Auf Basis der im Abschnitt 6.1 selektierten betrugsrelevanten Attribute und der im Abschnitt 2.2 beschriebenen Muster wurde zu Beginn der Experimente in einem ersten Schritt ein Entscheidungsbaum für die nichtmetrischen Attribute erzeugt. Dieser wurde von einem Teil der interviewten Experten qualitätsgesichert (siehe Anhang 1). Der Entscheidungsbaum ist in Abbildung 43 dargestellt.

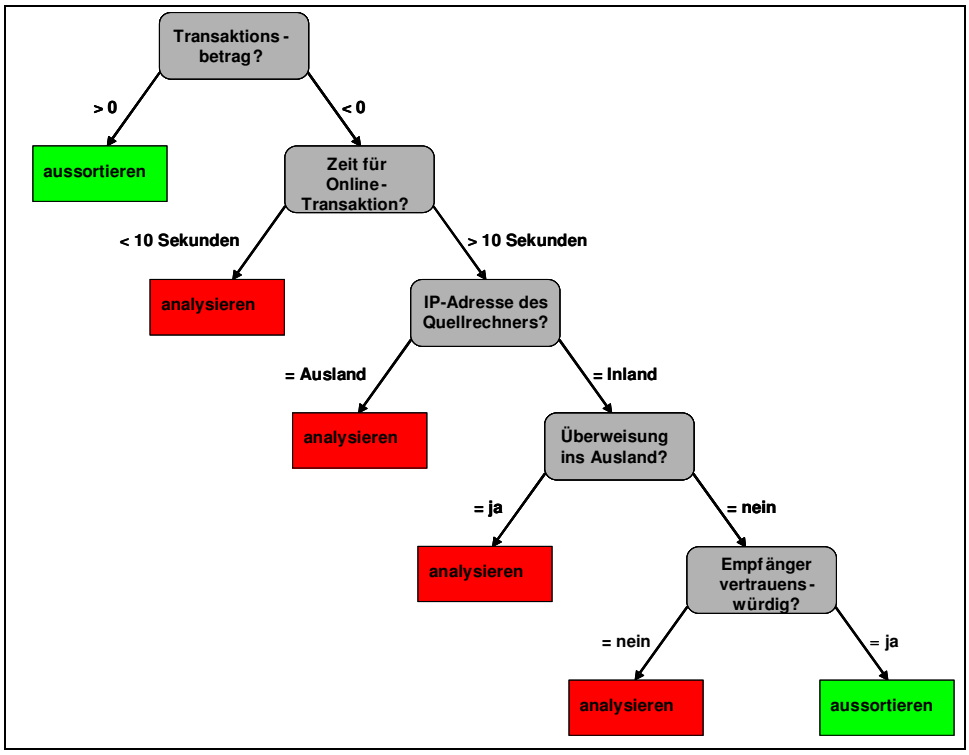

Abbildung 43: Aufbau des Entscheidungsbaums der Betrugserkennungskomponente

Auf der obersten Ebene des Entscheidungsbaums steht das Aussondern von Kontogutschriften (entspricht Transaktionsbetrag > 0), da diese Transaktionen Zugänge zum Konto bilden und somit keine betrügerische Abbuchung darstellen können. In diesem Fall könnten auftretende Transaktionen mit einem positiven Transaktionsbetrag nur fehlerhafte Buchungen darstellen, da die untersuchten Basisevents nur durchgeführte Online-Überweisungen abbilden, aber keine Kontogutschriften. Danach wird die Zeit, die vom Kunden für die Generierung und Abwicklung der Online-Transaktion benötigt wurde, ausgewertet. Unterschreitet diese Zeit einen Wert von zehn Sekunden, wird diese Transaktion vom neuronalen Netzwerk untersucht, da in diesem Fall der Verdacht besteht, dass ein Trojanisches Pferd die Transaktion ausgelöst hat. Anschließend wird die IP-Adresse des Quellcomputers der Überweisung geprüft, ob diese im inländischen IP-Adressraum liegt oder einem vertrauenswürdigen Provider zugeteilt ist. Ist dies nicht der Fall, wird die Transaktion ebenfalls vom neuronalen Netzwerk untersucht. Nicht bei allen Kreditinstituten liegen die Informationen über die Zeit der Online-Überweisung und der IP-Adresse vor. In diesem Fall müsste der Entscheidungsbaum aus Abbildung 43 um Ebene 2 und Ebene 3 gekürzt werden. Auf der nächsten Ebene wird die Empfängerbankleitzahl bzw. die Überweisungsart überprüft. Falls eine Auslandsüberweisung vorliegt, wird diese Transaktion bzw. der berechnete Diskriminanzwert dieser Transaktion vom neuronalen Netzwerk analysiert. Andernfalls wird abschließend auf der untersten Baumebene die Empfängerkontonummer untersucht. Wurde von diesem Kunden in der Vergangenheit

bereits eine Nicht-Betrugstransaktion auf dieses Zielkonto durchgeführt oder handelt es sich um die Kontonummer einer bekannten Organisation oder Behörde, gilt der Empfänger als vertrauenswürdig. Andernfalls wird die Transaktion ebenfalls vom neuronalen Netzwerk analysiert. Die drei Parameter „Zeit für die Online-Überweisung“, „IP-Adresse des Quellrechners“ und „Überweisung ins Ausland“ sind als KO-Kriterien für eine weitere Betrugsanalyse genannt.
Dieser Aufbau des Entscheidungsbaums sollte nicht statisch sein, da sich die Betrüger den Entscheidungsregeln anpassen könnten. Daher sollten bei einem realen Einsatz regelmäßig die Erkennungsquoten in Abhängigkeit von der Menge der zu analysierenden Transaktionen (z.B. wöchentlich oder monatlich) überprüft werden und bei einer Verschlechterung mit entsprechenden Anpassungen der Baumstruktur und Entscheidungsregeln reagiert werden. Gleiches gilt auch für die Attribute der Diskriminanzfunktion, die im nachfolgenden Unterabschnitt erläutert werden.

9.2.2 Struktur der Diskriminanzanalyse

Die in dieser Arbeit verwendete Diskriminanzfunktion besitzt drei metrische Attribute, die auf Basis der Interviews mit den Betrugsexperten als relevant zur Identifikation von Identitätsdiebstahl im Online-Banking angesehen werden. Diese sind:

a) Durchschnittliche Zahl an Online-Transaktionen (pro Monat) zum Zeitpunkt der Transaktion: Nach Aussage der Betrugsexperten ist eine Person, die wenig Online-Transaktionen durchführt tendenziell gefährdeter, Opfer von Identitätsdiebstahl wie z.B. Phishing zu werden. Dieser Wert enthält auch die Zugriffe auf die Online-Banking-Seite bzw. auf das Online-Konto ohne direkte Überweisungstätigkeit, beispielsweise Zugriffe zur Kontrolle des Kontostands oder der Kontobewegungen, da diese Tätigkeiten ebenfalls die Erfahrung der Kunden im Bereich Online-Banking erhöhen.

b) Verhältnis von Transaktionsbetrag zum verfügbaren Betrag: Nach Aussage der interviewten Betrugsexperten ist ein Transaktionsbetrag nahe an der maximalen Verfügbarkeitsgrenze sehr verdächtig, der Betrag einer Betrugstransaktion zu sein. In diesem Zusammenhang ist das Verhältnis wichtig, da ein Betrüger nicht mehr Geld vom Konto des Opfers transferieren kann, als maximal durch die Verfügbarkeitsgrenze möglich ist. Diese Verfügbarkeitsgrenze ist abhängig vom Kontostand, Dispolimit und eines evtl. gesetzten Transferlimits. Diese Parameter sind von Kunde zu Kunde verschieden und stärker schwankend als die berechneten Verhältniswerte, durch deren Nutzung eine bessere Vergleichbarkeit der Kunden innerhalb eines geringeren Wertebereichs entsteht. Aus diesem

Grund ist die Verwendung des Verhältniswerts für die Ermittlung der Diskriminanzfunktion günstiger als der exakte Transaktionsbetrag. Des Weiteren wird dieser Verhältniswert von [Bign06, S. 7] ebenfalls als relevantes Attribut zur Identifikation von Betrugstransaktionen genannt (siehe Unterabschnitt 5.2.2).

c) Verhältnis von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag: Nach Aussage der Betrugsexperten ist bei einem hohen durchschnittlichen Transaktionsbetrag die Wahrscheinlichkeit einer einzelnen Transaktion, eine Betrugsüberweisung darzustellen, geringer. Aufgrund des geringeren Wertebereichs innerhalb der Trainingsmenge wird hier ebenfalls der Verhältniswert anstelle des exakten durchschnittlichen Transaktionsbetrags verwendet.

Die Kennzahlen Transaktionsbetrag und durchschnittlicher Transaktionsbetrag werden in [Venk07, S. 5] ebenfalls als relevante Attribute einer Transaktion zur Betrugsanalyse genannt. In dem Ansatz dieses Artikels werden auf Basis eines regelbasierten Systems Betrugstransaktionen bei mobilen *ePayment*-Bezahlsystemen wie z.B. Mobiltelefonen identifiziert. In [Venk07] werden zu diesem Zweck Transaktionsprofile erzeugt, wobei die Autoren keine Aussagen über die Erkennungsgenauigkeit ihres Modells treffen.
Das Attribut „Verhältnis des Transferbetrags zum maximal verfügbaren Betrag“ sowie die Tatsache, dass Personen mit geringer Online-Banking-Aktivität anfälliger für Identitätsdiebstahl sind werden auch in der Arbeit von [Agge06, S. 1] als relevant genannt. In dem Überblicksartikel, der keine konkrete Problemlösung enthält, werden der Betrug beim Online-Banking und dessen Auswirkungen für ein Kreditinstitut erläutert, wie z.B. die Notwendigkeit der Beschäftigung von Mitarbeitern in diesem Bereich oder die Gefahr des Reputationsverlusts beim Kontakt mit Kunden von verdächtigen Transaktionen, siehe [Agge06, S. 2].
In [Wagn07, S. 7] werden für den Bereich Identitätsdiebstahl das Einkommen, das Alter und die Rasse der Opfer als relevante Attribute deklariert. In diesem Zusammenhang kam der Autor zu dem Ergebnis, dass gering verdienende Afrikaner und Lateinamerikaner im Alter von 18 bis 29 eine gefährdete Gruppe darstellen. Allerdings bezieht sich diese Studie auf Identitätsdiebstahl im Bereich der Kreditkartenzahlung und ist auf das Gebiet der USA beschränkt. Diese relevanten Attribute aus [Wagn07, S. 7] werden von den interviewten Experten für den Bereich Online-Banking nicht genannt.
Mit diesen oben genannten Attributen werden die Diskriminanzfunktion und der kritische Diskriminanzwert auf Basis der simulierten Trainingsmenge berechnet. Das genaue Verfahren ist im Abschnitt 3.2 beschrieben. Wenn der ermittelte Diskriminanzwert einer Online-Überweisung kleiner ist als dieser kritische Diskriminanzwert, gilt das *event* – wie im

Abschnitt 7.1 erwähnt – als betrugsverdächtig. Ist dagegen der berechnete Diskriminanzwert größer als der durch die Trainingsmenge berechnete kritische Diskriminanzwert, wird das *event* zunächst als nicht-betrugsverdächtig eingestuft. Eine endgültige Aussortierung eines durch den Vergleich mit dem kritischen Diskriminanzwert ermittelten nicht-betrugsverdächtigen Transaktionsevents erfolgt erst, wenn das *event* durch den Entscheidungsbaum ebenfalls als nicht-betrugsverdächtig klassifiziert wurde. Andernfalls wird es in Kombination mit historischen Transaktionsevents des Kunden vom neuronalen Netzwerk untersucht. Die Diskriminanzwerte der historischen Transaktionen werden mit der gleichen ermittelten Diskriminanzfunktion berechnet wie der Diskriminanzwert der aktuellen Transaktion. Die vier historischen Diskriminanzwerte werden zur Untersuchung durch das neuronale Netzwerk unter Verwendung der „KundenID“ des Auslösers (oder Opfers von Identitätsdiebstahls als vorgetäuschter Auslöser) der aktuellen Transaktion zeitlich absteigend sortiert aus der Datenbank geladen. Anschließend erfolgt eine Übergabe an die Eingabeknoten des neuronalen Netzwerks, das im folgenden Unterabschnitt beschrieben wird. Auch bei der Diskriminanzanalyse müssen – genau wie beim Entscheidungsbaum – die verwendeten Attribute angepasst und/oder die Diskriminanzfunktion und der kritische Diskriminanzwert neu berechnet werden, falls bei einem realen Einsatz die Erkennungsergebnisse dieses Verfahrens schlechter werden.

9.2.3 Struktur des neuronalen Netzwerks

Die Wahl der Topologie eines neuronalen Netzwerks ist entscheidend für die Qualität der Mustererkennung dieses Algorithmus. Der Aufbau der Struktur des neuronalen Netzwerks muss aufgabenspezifisch erfolgen. Bei zu kleinen Netzen werden oftmals nicht genügend Informationen gespeichert (engl.: *Underfitting*). Dagegen besteht bei zu großen Netzwerken wiederum die Gefahr, dass die Trainingsmenge zu exakt gelernt oder auswendig gelernt wird (engl.: *Overfitting*) und somit die Generalisierungsfähigkeit verloren geht bzw. der Generalisierungsfehler zu hoch wird. Diese Tatsache ist entscheidend für die Festlegung der Anzahl der Hiddenschichten und deren Knoten, dagegen ergibt sich die Anzahl der Knoten der Input- und Outputschicht aufgrund der praktischen Problemstellung. [Pete05, S. 228; Back06, S. 767; Lämm08, 247 - 248]
Allgemein gilt: Je geringer die Knotenzahl der Netzwerktopologie desto schneller ist der Trainingsvorgang [Pete05, S. 230]. Der Generalisierungsfehler eines neuronalen Netzwerks wird von folgenden Faktoren beeinflusst, siehe [Lämm08, S. 247]:

- Die Netzgröße, d.h. durch die Anzahl der Hiddenschichten und wiederum der Anzahl der Knoten der Hiddenschichten.

- Die Verbindungen zwischen den Knoten mit deren Gewichten.

- Das Lernverfahren und die dafür verwendeten Trainingsparameter.

Für das neuronale Netzwerk dieser Arbeit werden folgende Parametereinstellungen festgelegt, die teilweise in den Kapiteln 6 und 7 erwähnt wurden:
Es wird ein *Feedforward*-Netzwerk ohne zeitliche Rückkopplung verwendet, da jeder übergebene Diskriminanzwert bereits einen Zeitpunkt bzw. einen Wert der Transaktionsreihenfolge repräsentiert und alle zu einem Zeitpunkt übergebenen Diskriminanzwerte ein in sich geschlossenes zeitliches Muster bilden. Alle Knoten sind mit allen Knoten der Nachfolgeschicht verbunden, da die vollvernetzte Variante der *Feedforward*-Netzwerke in der Regel in der Praxis eingesetzt wird [Back06, S. 767 - 768; Pete05, S. 236].
Die Eingabemuster, die vor oder nach einem bestimmten Eingabemuster übergeben werden, sind von dem aktuellen Muster unabhängig, da jedes Muster einen anderen Kunden repräsentiert. Das Netzwerk besitzt fünf Inputknoten, da in der simulierten Trainings- und Testmenge aus jeweils fünf Transaktionen pro Kunde besteht. Jedem Inputknoten wird ein Diskriminanzwert einer Transaktion zeitlich sortiert übergeben. Das bedeutet, jeder Inputknoten erhält immer einen bestimmten Diskriminanzwert einer Überweisung der zeitlichen Transaktionsreihenfolge z.B. Inputknoten 1 nimmt immer den Diskriminanzwert der aktuellsten Transaktion entgegen, Inputknoten 5 dagegen analysiert immer den Diskriminanzwert der ältesten Überweisung usw.
Die Anzahl von fünf Eingabeknoten ergibt sich aufgrund der Aussagen der interviewten Experten, dass mindestens vier historische Transaktionen nötig sind um ein Transaktionsprofil zu erstellen, das zur Klassifizierung der aktuellen Transaktion in diesem Modell benötigt wird. Darüber hinaus besitzt das neuronale Netzwerk einen Outputknoten, der anhand seines Ausgabewerts zwischen 0,0 und 1,0 die Wahrscheinlichkeit angibt, nach der es sich bei der aktuellen Transaktion um einen Betrugsfall handelt. Je näher dieser Wert an 1,0 liegt umso größer ist die Gefahr eines Betrugsfalls. Zusätzlich wird für das neuronale Netzwerk mindestens eine Hiddenschicht verwendet, da nach [Rume86, S. 1 - 2; Dorf91, S. 102; Back06, S. 767] mit mindestens einer Hiddenschicht und maximal zwei Hiddenschichten jede beliebige Musterabbildung gelernt werden kann.
In diesem Zusammenhang ist für die Experimente zu beachten, dass die Gesamttopologie nicht „zu groß“ wird, da sonst die Generalisierungsfähigkeit des neuronalen Netzwerks verloren geht [Pete05, S. 228; Back06, S. 767; Lämm08, S. 247]. Nach [Dorf91, S. 102; Rume86, S. 1 - 2; Back06, S. 767] ist eine Hiddenschicht für den Zweck der Mustererkennung oft ausreichend, aber zwei Hiddenschichten erzielen lt. Aussage dieser Autoren tendenziell bessere Ergebnisse. Das Feststellen der optimalen Topologie zur Lösung eines

Problems ist – aufgrund dieser nicht exakt feststehenden Strukturparameter – ein Prozess, welcher auf der Erfahrung des Netzkonstrukteurs basiert. Die optimale Topologie muss oftmals schrittweise durch praktische Versuche herausgefunden werden [Back06, S. 767; Pete05, S. 228; Dorf91, S. 84]. Die gleiche Aussage trifft [Haun98, S. 141] in seiner Diskussion der Problematik, dass allgemein keine Standardnetzwerkeinstellungen für bereits definierte Problemstellungen bekannt sind. Bei entsprechenden Recherchen im Rahmen dieser Arbeit wurden ebenfalls keine Standardstrukturen bzw. Parametereinstellungen für die Thematik der Transaktionsanalyse zur Identifikation von Betrugsfällen beim Online-Banking gefunden. Aus diesem Grund wird im Zuge der Durchführung der Experimente schrittweise nach der optimalen Topologie geforscht und das verwendete neuronale Netzwerk punktuell um Knoten und Schichten erweitert, bis die optimale bzw. beste Topologie bezüglich der Erkennungsgenauigkeit identifiziert ist. Die realen Anforderungen und die zur Verfügung stehende Eventmenge sind von Kreditinstitut zu Kreditinstitut unterschiedlich. Daher soll im Rahmen dieser Arbeit exemplarisch für die vorhandenen, simulierten Trainings- und Testdaten nachgewiesen werden, dass die Kombination aus Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk grundsätzlich die auftretenden Betrugstransaktionen in Echtzeit bzw. nahe an der Echtzeit identifizieren kann. Die Struktur des neuronalen Netzwerks und die Gewichte können sich bei variierender Daten- bzw. Eventbasis ändern. Die verwendeten Algorithmen und Attribute bleiben gleich.

Die Werte der Initialgewichte sind zufällig festgelegt. Die Vorzeichen wurden später angepasst. Die Gründe für diese vorgenommene Anpassung sind im nachfolgenden Abschnitt erläutert. Als Lernverfahren wird der Backpropagationsalgorithmus ohne zeitliche Rückkopplungen verwendet, da es sich im Rahmen dieser Arbeit um ein überwachtes Verfahren handelt und mindestens eine Hiddenschicht verwendet wird. Hierfür ist die Backpropagationsmethode lt. [Dorf91, S. 39] prädestiniert.

Als Aktivierungsfunktion dient die nichtlineare Sigmoidfunktion (siehe Unterabschnitt 3.3.1). Nach [Dorf91, S. 102; Rume86, S. 1 - 2] kann durch die Anwendung dieser nichtlinearen Aktivierungsfunktion in Verbindung mit dem Einbau von Hiddenschichten jedes beliebige numerische Muster abgebildet werden. Durch die Anwendung der Sigmoidfunktion „feuert“ ein Knoten immer einen Wert im geschlossenen Intervall von 0 bis 1. Dadurch kann am Outputknoten sehr gut eine Wahrscheinlichkeit abgebildet bzw. ausgegeben werden, da die Betrugsfälle mit 1,0 und die Nicht-Betrugsfälle mit 0,0 als bekannten Ausgabewert trainiert werden. Als Lernkonstanten werden die Werte von 0,1 bis 0,9 in einem Schrittabstand von 0,2 verwendet, da durch diese Aufteilung die Veränderung des Lernerfolgs mit steigendem Lernfaktor beobachtet werden kann bzw. ob die Werte bei steigendem Lernfaktor zu oszillieren beginnen (siehe Unterabschnitt 3.3.2).

Die Anzahl der Lerndurchläufe des Backpropagationsverfahrens beginnt bei 100 Lernzyklen und wird sukzessive auf 1.000, 5.000, 10.000, 20.000, 30.000, 50.000 und 100.000 erhöht. Diese Werte sind daher so spezifiziert um stichprobenartig eine Steigerung darstellen zu können. Hierbei wird die Variante des Online-Trainings verwendet, da diese lt. [Schi93, S. 15 - 24] effektiver ist als das Offline-Training (siehe Unterabschnitt 3.3.2).
Für die Repräsentation der Klassifikationsergebnisse der verschiedenen überprüften Netzwerktopologien dient jeweils eine Matrix, bei der die Spalten den Lernfaktor und die Zeilen die Anzahl der Backpropagationsdurchläufe darstellen. In der jeweiligen Zelle steht ein Prozentsatz, der den Anteil der exakt klassifizierten *events* unter den 4.000 aktuellen Testevents unter Berücksichtigung der Parameter Zyklenzahl und Lernfaktor der aktuell analysierten Netzwerktopologie für die gesamte Betrugserkennungskomponente angibt. Als genau erkannt gilt nach Aussage der verantwortlichen interviewten Betrugsexperten für Betrugsfälle ein Ausgabewert des neuronalen Netzwerks im abgeschlossenen Intervall zwischen 0,9 und 1,0. Bei den Nicht-Betrugsfällen zählt hierbei sowohl ein Ausgabewert des neuronalen Netzwerks im abgeschlossenen Intervall von 0,0 bis 0,1 als auch eine Aussonderung des Testevents durch Diskriminanzanalyse oder Entscheidungsbaum als exakt klassifiziert. Die restlichen Transaktionsevents mit Netzwerkausgabewerten im offenen Intervall zwischen 0,1 und 0,9 müssten von der jeweiligen Revisionsabteilung der Kreditinstitute manuell untersucht werden und gelten für die Untersuchungen dieser Arbeit als nicht genau zuordenbar. Somit verringern solche Werte die Erkennungsgenauigkeit.
Bezüglich der Analysegeschwindigkeit der Anwendung soll lt. den befragten Betrugsexperten ein Wert nahe an der Echtzeit für ein *event* vorliegen, wobei im Rahmen des Prozesses am Wichtigsten ist, dass ein möglicher Betrugsfall identifiziert wird, bevor die Überweisung an das Zielkonto erfolgt. Zur Erreichung des Ziels, bei einem realen Einsatz des Modells in der Bankenbrache das neuronale Netzwerk aktuell zu halten, ist ein tägliches Nachtrainieren des neuronalen Netzwerks empfehlenswert. Das Training könnte hierbei parallel zu den auszuführenden Analysen erfolgen und somit müssten zur Laufzeit nur die Gewichtswerte ausgetauscht werden.

9.3 Experimentelle Ergebnisse und Interpretation

Zu Beginn des Verfahrens wird die Diskriminanzfunktion mit dem kritischen Diskriminanzwert ermittelt. Auf Basis der Trainingsmenge ergeben sich nach Anwendung der Schritte zur Berechnung der Diskriminanzfunktion für den zwei Gruppen - n Variablen-Fall (siehe Abschnitt 3.2) die folgenden Koeffizienten:

$$y = 0{,}0000216046516501255 5x_1 - 0{,}0000000000001755448407818092 9x_2 - 0{,}0000014456274134782 47x_3$$

Die Koeffizienten der Diskriminanzfunktion y stellen die, im Abschnitt 9.2 angegebenen, relevanten metrischen Attribute zur Betrugserkennung dar. In diesem Fall bildet x_1 den Wert „Verhältnis von Transaktionsbetrag zum maximal verfügbaren Betrag“ ab, x_2 repräsentiert den Parameter „Verhältnis von Transaktionsbetrag zum durchschnittlichen Transaktionsbetrag“ und x_3 stellt das Attribut „durchschnittliche Anzahl an Überweisungen im Monat zum Zeitpunkt der Transaktion“ dar.
Beim Einsetzen der Werte der vier Beispielfälle aus den Tabellen 1 bis 4 (siehe Abschnitt 2.2) ergeben sich folgende Funktionen und Diskriminanzwerte:

Fall 1 (verdächtiger Betrugsfall):
0,00002160465165012555 * 99,09 - 0,00000000000017554484078180929 * 5280,42 -
000000144562741347824 7 * 3,0 = -0,00216193108452867

Fall 2 (unverdächtiger Betrugsfall):
0,00002160465165012555 * 34,58 - 0,00000000000017554484078180929 * 776,18 -
0,000001445627413478247 * 8,0 = -0,000758710824994126

Fall 3 (unverdächtiger Nicht-Betrugsfall):
0,00002160465165012555 * 0,34 - 0,00000000000017554484078180929 * 19,58 -
0,000001445627413478247 * 10,0 = -0,00002.17609149338353

Fall 4 (verdächtiger Nicht-Betrugsfall):
0,00002160465165012555 * 25,94 - 0,00000000000017554484078180929 * 148,16 -
0,000001445627413478247 * 9,0 = -0,000573409309587571

Der berechnete kritische Diskriminanzwert, der die Gruppen Betrugsverdächtig und Nicht-Betrugsverdächtig bei der Vorklassifikation trennt, beträgt:

-0,0009736643644843095

Beim Vergleich der oben berechneten Diskriminanzwerte mit dem kritischen Diskriminanzwert ist zu erkennen, dass Fall 2 (unverdächtiger Betrugsfall) fälschlicherweise von der Diskriminanzanalyse als Nicht-Betrugsfall klassifiziert wird, da sein Diskriminanzwert größer ist als der kritische Diskriminanzwert (siehe Unterabschnitt 9.2.2). Dieser Betrugsfall wird aber dennoch dem neuronalen Netzwerk zur Analyse übergeben, weil die Auswertung des Entscheidungsbaums ergibt, dass es sich hierbei um einen nicht bekannten Kontoempfänger handelt und die benötigte Zeitdauer für das Ausfüllen der Online-Überweisung weniger als zehn Sekunden aufweist. Damit eine Transaktion im Vorfeld

aussortiert wird, muss sie sowohl von der Diskriminanzanalyse als auch vom Entscheidungsbaum als Nicht-Betrugsfall klassifiziert werden. Dieses Beispiel aus Fall 2 zeigt, dass die Diskriminanzanalyse alleine für die optimale Vorselektion der Transaktionen zur Weiterverarbeitung durch das neuronale Netzwerk nicht ausreicht. Somit ist der Entscheidungsbaum sowohl zur Auswertung der nichtmetrischen Attribute als auch als zusätzliches Verfahren zur Qualitätssicherung notwendig. Von den vier Beispielfällen aus Abschnitt 2.2 werden alle an das neuronale Netzwerk zur weiteren Analyse übergeben, da jede dieser Transaktionen einen nicht bekannten Empfänger – bezogen auf die Menge aller früheren Transaktionen der Kunden – zum Ziel aufweist. Die Ergebnisse der vier Beispielfälle nach der weiteren Analyse durch das neuronale Netzwerk werden am Ende dieses Abschnitts präsentiert und interpretiert.

Bezüglich der Topologie des neuronalen Netzwerks der Betrugserkennungskomponente starten die Experimente mit Strukturen bestehend aus einer Hiddenschicht und anschließend erfolgt eine Erweiterung auf zwei Hiddenschichten, da sich dies mit den Aussagen aus [Dorf91, S. 102; Rume86, S. 1 - 2; Back06, S. 767] deckt, wonach bei Verwendung von einer oder zwei Hiddenschichten jedes numerische Muster gelernt werden kann (siehe Unterabschnitt 9.2.3). Zusätzlich werden – darauf aufbauend – Analysen mit Netzwerktopologien, die drei Hiddenschichten beinhalten, durchgeführt.

Im Folgenden werden zunächst die Auswertungen der Gesamtmenge der Testevents mit den Prozentsätzen der Erkennungsgenauigkeit für die verschiedenen getesteten Netzwerktopologien der Betrugserkennungskomponente dargestellt. Im Anhang 3 sowie in den separaten Arbeitsberichten befindet sich zu jedem angegebenen Erkennungsgenauigkeitsprozentsatz eine genaue Auflistung der Anzahl der Ausprägungen der Ausgabewerte des neuronalen Netzwerks in Form von Intervallen. Diese Intervalle teilen den kompletten Ergebnisraum der Netzwerksausgabewerte von 0,0 bis 1,0 in zehn Einheiten im Abstand von jeweils 0,1 auf. Das bedeutet, es existieren insgesamt zehn Intervalle pro Auswertung für eine bestimmte Anzahl an Backpropagationsdurchläufen und einem bestimmten Lernfaktor. Jedes Intervall enthält in der Auswertung die Anzahl der jeweils eingruppierten Transaktionen. Ein Betrugsfall gilt dabei mit einem Ausgabewert des neuronalen Netzwerks im abgeschlossenen Intervall zwischen 0,9 und 1,0 als exakt klassifiziert, dagegen ist ein Nicht-Betrugsfall im abgeschlossenen Intervall zwischen 0,0 und 0,1 eindeutig identifiziert (siehe Unterabschnitt 9.2.3).

Nach dem Ausführen aller Experimente und Interpretation der Ergebnisse ergeben sich folgende Erkenntnisse:

Von der Diskriminanzanalyse und dem Entscheidungsbaum werden von den insgesamt 4.000 Testevents an aktuellen Transaktionen 983 Nicht-Betrugsevents vorab ausgesondert und müssen nicht mehr vom neuronalen Netzwerk analysiert werden. Dies entspricht

einem Anteil von 24,575% und bildet somit den minimalen Wert der möglichen Erkennungsgenauigkeit der gesamten Betrugserkennungskomponente. Ebenso bringt diese Vorselektion aufgrund der fehlenden Notwendigkeit einer Analyse durch das neuronale Netzwerk Verbesserungen in der Performance mit sich. Die Betrugsfälle dagegen werden alle vom neuronalen Netzwerk ausgewertet, d.h. von dieser Teilmenge wird kein Transaktionsevent durch Entscheidungsbaum oder Diskriminanzanalyse aussortiert.
Der Klassifikationsgenauigkeitswert von 24,575% in der folgenden Beispielauswertung der Betrugserkennungskomponente bedeutet, dass keine der analysierten Transaktionen des neuronalen Netzes eindeutig einer Klasse zugeordnet werden kann, d.h. der Outputwert fast aller vom neuronalen Netzwerk analysierten *events* liegt bei ca. 0,5. Dies wird nachfolgend in der ersten Beispielausgabe einer solchen Analyse deutlich (die hervorgehobenen Zeilen enthalten die Anzahl der jeweils korrekt klassifizierten Transaktionsevents).

```
Beispielauswertung 1:

Auswertung mit folgenden Parametern:
Lernfaktor: 0,1, Anzahl Backpropagationsdurchläufe: 1000

Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1
Betrugsfälle zwischen 0,9 und 1,0: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1016
Betrugsfälle zwischen 0,4 und 0,5: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________

Erkennungsgenauigkeit: 24,575%
```

Die ermittelte Erkennungsgenauigkeit der Betrugserkennungskomponente von 24,575% ergibt sich aus einer Dreisatzrechnung ((0 + 0 + 983) * 100 / 4.000).
Wird die Analyse der Klassifikationsgenauigkeit des neuronalen Netzwerks separat betrachtet, d.h. ohne Berücksichtigung der 983 von Diskriminanzanalyse und Entscheidungsbaum vorab aussortierten Nicht-Betrugstransaktionen, führt dies zu einem Resultat von 0,0% bei insgesamt 3.017 (= 4.000 - 983) verbliebenen analysierten Transaktionen (keine Transaktion ist exakt klassifiziert => (0 + 0) * 100 / 3.017).
Ein Grund für dieses Resultat ist, dass bei einem solchen Ergebnis das Training noch nicht abgeschlossen ist, d.h. die Anzahl der durchgeführten Lernzyklen waren für den jeweiligen Lernfaktor und der gewählten Topologie zu wenig. Die andere Möglichkeit für die Einstellung dieses Ergebnisses ist, dass die untersuchte Topologie aufgrund ihrer Struktur nicht in der Lage ist, die Eingabemuster richtig zu erlernen. In diesem Fall ist die Netzwerktopologie nicht für einen praktischen Einsatz geeignet. Beispiele dazu werden weiter unten in diesem Abschnitt diskutiert.
Bei einer geringen Anzahl von Lerndurchläufen können alle, vom neuronalen Netzwerk untersuchten, Testevents auch einen Netzwerkoutputwert nahe 1,0 aufweisen (siehe Anhang 3 bzw. die separaten Arbeitsberichte für die Auswertung der Topologien). Das bedeutet, jedes Nicht-Betrugsevent wird hierbei *false positive* klassifiziert (ausgenommen die 983 Nicht-Betrugsevents, welche in der Voranalyse durch Entscheidungsbaum und Diskriminanzanalyse ausgesondert wurden). Dadurch ergibt sich bei diesen Parametereinstellungen eine Erkennungsgenauigkeit von 74,575%, wie in der nachfolgenden zweiten Beispielauswertung zu erkennen ist (die hervorgehobenen Zeilen enthalten die Anzahl der jeweils korrekt klassifizierten Transaktionsevents).

```
Beispielauswertung 2:

Auswertung mit folgenden Parametern:
Lernfaktor: 0,1, Anzahl Backpropagationsdurchläufe: 100

Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
```

```
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________

Erkennungsgenauigkeit: 74,575%
```

Die Erkennungsgenauigkeit der kompletten Betrugserkennungskomponente von 74,575% ergibt sich in diesem Beispiel ebenfalls aus einer Dreisatzrechnung ((2.000 + 0 + 983) * 100 / 4.000). Das neuronale Netzwerk selbst liefert – alleinstehend betrachtet – eine Erkennungsgenauigkeit der analysierten Transaktionen von 66,29% (2.000 Betrugsfälle sind richtig klassifiziert => (2.000 + 0) * 100 / 3.017).

Dieser Wert ist nicht aussagekräftig, da mit Ausnahme der, in der Voranalyse aussortieren *events*, jede analysierte Transaktion vom neuronalen Netzwerk als Betrugsfall interpretiert wird. Dadurch werden einerseits die Betrugsfälle richtig klassifiziert, andererseits erzeugen 1.017 von 2.000 vorhandenen Nicht-Betrugsevents einen Fehlalarm, was eine zu hohe Quote ist. Die Begründung hierfür liegt ebenfalls in der Situation, dass bei einem solchen Ergebnis die Anzahl der absolvierten Lerndurchläufe für den jeweiligen Lernfaktor und der gewählten Topologie zu gering ist, d.h. das Training muss weiter fortgesetzt und/oder der Lernfaktor erhöht werden. Falls diese Maßnahmen zu keiner Verbesserung der Resultate führen muss die Topologie angepasst werden.

Im Laufe der Experimente ist bezüglich der Erkennungsgenauigkeit zu beobachten, dass bei steigender Anzahl von Lerndurchläufen zunächst einen Wert von 74,575% annimmt, danach auf 24,575% absinkt und sich schließlich oft in einem Bereich um 98% einpendelt (siehe Auswertungsmatrizen im unteren Teil dieses Abschnitts). Die folgende dritte Beispielauswertung zeigt eine Analyse, bei der sich nach längerer Trainingsdauer bzw. einer erhöhten Zahl von Lerndurchläufen ein solches stabiles Ergebnis einstellt (die hervorgehobenen Zeilen enthalten die Anzahl der jeweils korrekt klassifizierten Transaktionsevents).

```
Beispielauswertung 3:

Auswertung mit folgenden Parametern:
Lernfaktor: 0,9, Anzahl Backpropagationsdurchläufe: 50000

Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 6
Betrugsfälle zwischen 0,9 und 1,0: 1986
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 3
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 1
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 3
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 2
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 3
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 10
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 987
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________

Erkennungsgenauigkeit: 98,90%
```

Die Erkennungsgenauigkeit der Betrugserkennungskomponente von 98,90% ergibt sich in dieser Beispielausgabe – genau wie bei allen anderen durchgeführten Auswertungen – aus einer Dreisatzrechnung ((1.986 + 987 + 983) * 100 / 4.000). Eine gesonderte Betrachtung der Klassifikationsgenauigkeit des neuronalen Netzwerks ergibt hier ein Resultat von 98,54% (1.986 Betrugsfälle und 987 Nicht-Betrugsfälle sind richtig klassifiziert => (1.986 + 987) * 100 / 3.017). Aus diesen Beispielausgaben ist ersichtlich, dass sich die Erkennungsgenauigkeitsresultate der gesamten Betrugserkennungskomponente und des neuronalen Netzwerks dahingehend unterscheiden, dass bei der Analyse des neuronalen Netzwerks insgesamt 983 richtig klassifizierte Nicht-Betrugsevents nicht mit berücksichtigt werden und sich somit für diesen Algorithmus gesondert die Menge der ausgewerteten Transaktionen von 4.000 auf 3.017 verringert. Dadurch ist die Klassifikationsgenauigkeit des neuronalen Netzwerks allein insgesamt geringer als die Erkennungsgenauigkeit der gesamten Betrugserkennungskomponente (oder höchstens genau so hoch, falls das neu-

ronale Netzwerk 100% der Transaktionen richtig klassifiziert), da 983 Nicht-Betrugstransaktionen bereits durch Entscheidungsbaum und Diskriminanzanalyse exakt identifiziert wurden. Diese Tatsache in Verbindung mit dem geringeren Gesamtanalyseaufwand des neuronalen Netzwerks zeigt die Vorteile der Verwendung von Diskriminanzanalyse und Entscheidungsbaum in Kombination mit einem neuronalen Netzwerk in der beschriebenen Ablaufarchitektur (siehe Abschnitt 7.2).

Dieses stabile Klassifikationsergebnis von über 98% ist die Folge einer Veränderung der Outputwerte der Transaktionen bei steigendem Lernfortschritt des neuronalen Netzwerks. Zu Beginn des Trainings sind die Outputwerte aller Transaktionen (sowohl Betrugs- als auch Nicht-Betrugstransaktionen) nahezu gleich (zunächst zwischen 0,9 und 1,0 und später zwischen 0,5 und 0,6, siehe Beispielauswertungen 1 und 2). Mit zunehmender Zahl an Backpropagationsdurchläufen werden die Transaktionen gemäß ihrem Betrugsstatus immer exakter klassifiziert, falls die Struktur der Topologie passend ist (die geeigneten Strukturen werden in diesem Abschnitt weiter unten erläutert). In den positiven Fällen stabilisieren sich die Erkennungsquoten auf einem hohen Niveau und die noch auftretenden Schwankungen erweisen sich als gering. Mit zunehmendem Lernfaktor wird diese Stabilisierung der Erkennungsquote schneller erreicht, da insgesamt weniger Lerndurchläufe benötigt werden. Das bedeutet, der Graph der Gradientenfunktion des Backpropagationsverfahrens (siehe Unterabschnitt 3.3.2) beginnt bei diesen Topologien und erlernten Eingabemustern nicht zu oszillieren. Diese Regelmäßigkeit der Ergebnisse tritt erst bei zwei und Hiddenschichten auf. Bei weniger Knoten mit einer Hiddenschicht (z.B. 5-4-1 oder 5-2-1 Topologie) sind die Erkennungsergebnisse aufgrund der geringen Knotenmenge unregelmäßig (siehe dazu die separaten Arbeitsberichte mit den kompletten Auswertungen der entsprechenden Topologien).

Die nachfolgenden Matrizen mit den Erkennungsgenauigkeiten der Betrugserkennungskomponente variieren mit den verschiedenen möglichen Netzwerktopologien. Jede Matrix enthält in ihren Zellen die Erkennungsgenauigkeit der gesamten Betrugserkennungskomponente unter Berücksichtigung der jeweiligen Trainingsparameter (Anzahl Lerndurchläufe, Lernfaktor) für eine bestimmte Topologie des neuronalen Netzwerks. Das heißt, der Anteil von 24,575% der, durch Entscheidungsbaum und Diskriminanzanalyse aussortierten Nicht-Betrugstransaktionen, ist bei der Berechnung der Erkennungsgenauigkeit als exakt klassifiziert mit berücksichtigt, weil sich die jeweiligen Experimente immer auf das gesamte Hybrid-Modell beziehen. Ebenso setzt sich die Trainings- und Testmenge bezüglich der eingruppierten *events* für alle Auswertungen gleich zusammen um eine bessere Vergleichbarkeit der Klassifikationsergebnisse zu ermöglichen.

Die Matrizen sind nachfolgend so geordnet, dass mit der Netzwerktopologie, welche die wenigsten Hiddenschichten und Knoten aufweist (= 5-1-1 Topologie), begonnen wird und

danach die Knotenzahlen der Netzwerke für die jeweilige Schicht permanent erhöht werden. Das bedeutet, bei der Suche nach der optimalen Netzwerktopologie wird konstruktiv vorgegangen, da der Lösungsraum von möglichen Topologien theoretisch unendlich groß sein könnte, siehe [Pete05, S. 229]. Aus diesem Grund wird die Suche eingestellt, wenn die Erkennungsergebnisse der Topologien mit zunehmender Knotenzahl nachhaltig schlechter werden.

a) Topologien mit einer Hiddenschicht:

Matrix für Netzwerktopologie 5-1-1 (fünf Knoten der Inputschicht, ein Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 7: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-1 Netzwerktopologie

Bei dieser Topologie (Tabelle 7) wird unabhängig von Lernfaktor und Trainingszyklenzahl keine Transaktion vom neuronalen Netzwerk eindeutig klassifiziert. Fast alle Ausgabewerte des neuronalen Netzwerks liegen unabhängig vom trainierten Betrugsstatus im Bereich zwischen 0,4 und 0,5. Die 24,575% der erkannten Nicht-Betrugstransaktionen werden – wie oben erwähnt – von Diskriminanzanalyse und Entscheidungsbaum aus der gesamten Transaktionsmenge im Vorfeld herausgefiltert. Das bedeutet, dass eine Topologie mit diesen Resultaten für den praktischen Einsatz nicht geeignet ist.

Matrix für Netzwerktopologie 5-2-1 (fünf Knoten der Inputschicht, zwei Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	71,725%	73,800%	73,975%	74,075%
1.000	74,075%	74,775%	76,600%	79,475%	84,075%
5.000	76,575%	90,225%	94,325%	95,925%	96,325%
10.000	84,325%	95,200%	96,300%	96,900%	97,150%
20.000	93,125%	96,500%	97,150%	97,200%	97,325%
30.000	95,225%	97,100%	97,025%	97,350%	97,400%
50.000	96,300%	97,200%	97,425%	97,450%	97,450%
100.000	97,175%	97,450%	97,475%	97,475%	97,475%

Tabelle 8: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-3-1 (fünf Knoten der Inputschicht, drei Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	71,975%	73,825%	73,975%	74,075%
1.000	74,075%	74,825%	76,775%	80,550%	86,075%
5.000	76,450%	90,500%	94,700%	96,075%	96,475%
10.000	83,900%	95,175%	96,375%	97,100%	97,275%
20.000	93,050%	96,550%	97,175%	97,275%	97,425%
30.000	95,175%	97,125%	97,050%	97,400%	97,450%
50.000	96,250%	97,050%	97,400%	97,400%	97,450%
100.000	97,175%	97,450%	97,475%	97,450%	97,450%

Tabelle 9: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-1 (fünf Knoten der Inputschicht, vier Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,850%	73,750%	73,950%	74,075%
1.000	74,075%	74,825%	77,100%	81,750%	89,150%
5.000	76,325%	90,900%	95,000%	96,225%	96,775%
10.000	83,725%	95,250%	96,500%	97,150%	97,375%
20.000	92,975%	96,550%	97,200%	97,325%	97,450%
30.000	95,175%	97,125%	97,175%	97,425%	97,450%
50.000	96,250%	97,025%	97,425%	97,475%	97,475%
100.000	97,150%	97,450%	97,450%	97,450%	97,475%

Tabelle 10: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-1 (fünf Knoten der Inputschicht, fünf Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	73,875%	74,850%	78,350%	87,675%
5.000	74,475%	89,075%	94,925%	96,350%	96,825%
10.000	79,675%	95,125%	96,525%	97,150%	97,475%
20.000	92,075%	96,600%	97,275%	97,425%	97,575%
30.000	94,600%	97,125%	97,300%	97,450%	97,475%
50.000	96,175%	97,100%	97,425%	97,475%	97,475%
100.000	97,150%	97,425%	97,450%	97,475%	97,475%

Tabelle 11: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-1 (fünf Knoten der Inputschicht, sechs Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	83,225%	95,625%	96,725%
10.000	74,575%	89,375%	96,200%	97,025%	97,500%
20.000	74,450%	96,250%	97,200%	97,600%	97,750%
30.000	87,625%	97,075%	97,425%	97,500%	97,575%
50.000	95,400%	97,250%	97,450%	97,475%	97,500%
100.000	97,000%	97,400%	97,450%	97,475%	97,500%

Tabelle 12: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-1 (fünf Knoten der Inputschicht, sieben Knoten der Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 13: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-1 Netzwerktopologie

Anhand von Tabelle 8 ist zu erkennen, dass sich bereits bei einer Topologie mit zwei versteckten Knoten stabile Erkennungsergebnisse bis in den Bereich von 97% einstellen. Allerdings sind die Ergebnisse unregelmäßig, z.B. mit Erkennungsquoten im Bereich von 85% (Tabellen 9 bis 12). Bei einer Platzierung von sieben Knoten in der Hiddenschicht (Tabelle 13) tritt der oben beschriebene Effekt ein, dass alle Testtransaktionen, die vom neuronalen Netzwerk untersucht werden, einen Ausgabewert zwischen 0,9 und 1,0 annehmen (Erkennungsgenauigkeit 74,575%). Daraus folgt, dass diese Topologie aufgrund ihrer Knotenstruktur nicht in der Lage ist, die Trainingsmuster richtig zu erlernen, da die-

ses Ergebnis unabhängig von Trainingszyklenzahl und Lernfaktor bei jeder Auswertung eintritt. Bei Topologien mit zwei Hiddenschichten werden die Ergebnisse der verschiedenen Topologien ähnlicher, regelmäßiger und sind weniger gestreut.

b) Topologien mit zwei Hiddenschichten:

Matrix für Netzwerktopologie 5-1-1-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 14: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-1-1 Netzwerktopologie

Diese Auswertungsergebnisse von 24,575% (Tabelle 14) treten regelmäßig für alle Trainingsparameter ein, wenn die zweite Hiddenschicht nur einen Knoten aufweist. Topologien mit diesen Ergebnissen sind aus diesem Grund für den praktischen Einsatz ungeeignet.

Matrix für Netzwerktopologie 5-1-2-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	93,575%	95,650%	24,575%	24,575%	24,575%
10.000	92,500%	96,200%	96,625%	24,575%	24,575%
20.000	94,900%	97,525%	97,600%	24,575%	24,575%
30.000	96,575%	98,200%	98,350%	24,575%	24,575%
50.000	97,725%	98,550%	98,800%	97,275%	97,600%
100.000	98,475%	98,950%	98,950%	98,875%	98,350%

Tabelle 15: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-1-3-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	24,575%
20.000	74,575%	96,400%	97,450%	24,575%	24,575%
30.000	74,575%	97,500%	98,075%	24,575%	24,575%
50.000	93,775%	98,450%	98,750%	98,025%	98,450%
100.000	98,175%	98,975%	99,050%	98,450%	98,975%

Tabelle 16: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-3-1 Netzwerktopologie

In dieser getesteten Topologie (Tabelle 16) ist z.B. bei einem Lernfaktor von 0,5 der oben beschriebene Effekt des Abfalls der Klassifikationsgenauigkeit von 74,575% bei 5.000 Zyklen auf 24,575% bei 10.000 Zyklen und des erneuten Anstiegs auf 97,45% bei 20.000 Zyklen gut zu erkennen. Ab dieser Anzahl von Lerndurchläufen bleibt die Erkennungsgenauigkeit bei dieser Topologie konstant auf einem hohen Wert bzw. stabilisiert sich. Dieses Resultat lässt sich in den nachfolgenden Topologien noch öfter beobachten, wobei

bei manchen Auswertungen einer der beiden Werte (74,575% oder 24,575%) mehrmals oder niemals im Laufe des Lernfortschritts auftritt.

Matrix für Netzwerktopologie 5-1-4-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 17: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-4-1 Netzwerktopologie

Bei dieser Topologie (Tabelle 17) stellt sich ebenfalls der Effekt ein, dass alle Ausgabewerte der vom neuronalen Netzwerk analysierten Transaktionen unabhängig vom trainierten Betrugsstatus im Ausgabeintervall von 0,9 bis 1,0 liegen. Da hier – unabhängig von Lernfaktor und Trainingszyklenzahl – immer der Wert von 74,575% als Resultat eintritt und somit jede vom neuronalen Netzwerk untersuchte Nicht-Betrugstransaktion als Betrugsfall klassifiziert wird, sind Topologien mit diesen Ergebnissen nicht für den praktischen Einsatz geeignet. Dieser Effekt ist nicht von der Anzahl der Hiddenschichten abhängig, da er z.B. auch bei der 5-7-1 Topologie zu beobachten ist.

Matrix für Netzwerktopologie 5-1-5-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 18: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-1-6-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 19: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-1-7-1 (fünf Knoten der Inputschicht, ein Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 20: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-1-7-1 Netzwerktopologie

Hier erfolgt ein Übergang der Netzwerkstruktur in der Weise, dass die erste Hiddenschicht nunmehr zwei statt einen Knoten beinhaltet. Bei der Knotenzahl der zweiten Hiddenschicht wird nachfolgend mit den Auswertungen wieder bei einem Knoten begonnen.

Matrix für Netzwerktopologie 5-2-1-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 21: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-1-1 Netzwerktopologie

Auch bei dieser Topologie (Tabelle 21) weist die zweite Hiddenschicht nur einen Knoten auf, somit wird kein Transaktionsevent vom neuronalen Netzwerk exakt klassifiziert.

Matrix für Netzwerktopologie 5-2-2-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,700%
1.000	96,525%	98,850%	98,925%	98,850%	98,975%
5.000	98,650%	98,425%	98,625%	98,525%	98,575%
10.000	98,400%	98,500%	98,475%	98,275%	98,100%
20.000	98,050%	98,675%	97,600%	97,475%	97,575%
30.000	97,875%	98,075%	97,325%	97,350%	97,600%
50.000	97,825%	97,300%	97,175%	97,425%	97,650%
100.000	97,775%	97,100%	97,375%	97,550%	97,700%

Tabelle 22: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-2-3-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	98,625%	98,875%	99,000%	99,000%
5.000	98,725%	98,425%	98,750%	98,700%	98,800%
10.000	98,425%	98,425%	98,500%	98,600%	98,600%
20.000	98,100%	98,325%	97,975%	98,025%	98,200%
30.000	97,950%	98,175%	97,850%	97,800%	98,200%
50.000	97,875%	97,775%	97,825%	97,900%	98,275%
100.000	97,725%	97,925%	97,200%	97,325%	97,750%

Tabelle 23: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-3-1 Netzwerktopologie

Bei einem Lernfaktor von 0,1 tritt bei dieser Topologie (Tabelle 23) der beobachtete Abfall- und anschließende Stabilisierungseffekt ebenfalls ein. Bei einem Lernfaktor von 0,3 fehlt der Wert von 24,575%. Das bedeutet, hier wird wegen des höheren Lernfaktors schneller gelernt und somit der Wert von 24,575% bei einer Zyklenzahl über 100, aber unter 1.000 erreicht. Beim Test des Lernfaktors 0,7 stellt sich der Wert von 24,575% aufgrund des noch schnelleren Lernfortschritts bereits bei 100 Trainingszyklen ein.

Die Stabilisierung erfolgt in diesem Fall bei 1.000 Backpropagationsdurchläufen. Somit sind diese und andere Topologien mit ähnlichen Resultaten gut für einen praktischen Einsatz geeignet.

Matrix für Netzwerktopologie 5-2-4-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	98,675%	98,875%	99,000%	99,000%
5.000	98,750%	98,550%	98,600%	98,850%	98,900%
10.000	98,425%	98,450%	98,500%	98,625%	98,800%
20.000	98,200%	98,450%	98,475%	98,300%	98,900%
30.000	97,975%	98,400%	98,250%	97,975%	98,800%
50.000	97,850%	98,200%	97,900%	98,000%	98,725%
100.000	97,775%	97,875%	98,375%	98,650%	98,675%

Tabelle 24: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-2-5-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	98,450%	98,825%

Tabelle 25: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-2-6-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	98,775%	98,900%

Tabelle 26: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-2-7-1 (fünf Knoten der Inputschicht, zwei Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 27: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-2-7-1 Netzwerktopologie

Bei dieser Topologie (Tabelle 27) tritt bei einer Anzahl von sieben Knoten in der zweiten Hiddenschicht ein durchgängiges Auswertungsergebnis der Erkennungsgenauigkeit von 74,575% ein. Das bedeutet, es werden alle Transaktionen vom neuronalen Netzwerk als Betrugsfall interpretiert.

Matrix für Netzwerktopologie 5-3-1-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 28: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-1-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-3-2-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	98,550%	98,850%	98,900%	98,925%
5.000	98,825%	98,525%	98,600%	98,600%	98,525%
10.000	98,400%	98,525%	98,625%	98,500%	98,425%
20.000	98,250%	98,600%	97,550%	97,625%	97,725%
30.000	98,100%	97,700%	97,350%	97,475%	97,725%
50.000	98,175%	97,125%	97,275%	97,525%	97,750%
100.000	97,975%	97,075%	97,375%	97,600%	97,800%

Tabelle 29: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-3-3-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	97,975%	99,000%	99,050%	99,000%
5.000	98,900%	98,600%	98,675%	98,800%	98,800%
10.000	98,425%	98,625%	98,750%	98,625%	98,725%
20.000	98,225%	98,400%	97,875%	98,125%	98,450%
30.000	98,200%	98,025%	97,625%	98,150%	98,400%
50.000	98,275%	97,750%	97,575%	98,125%	98,425%
100.000	97,775%	96,850%	97,700%	98,150%	98,500%

Tabelle 30: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-3-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-3-4-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	97,975%	99,000%	99,050%	99,000%
5.000	98,900%	98,600%	98,675%	98,800%	98,800%
10.000	98,425%	98,625%	98,750%	98,625%	98,725%
20.000	98,225%	98,400%	97,875%	98,125%	98,450%
30.000	98,200%	98,025%	97,625%	98,050%	97,775%
50.000	98,275%	97,750%	97,550%	97,325%	97,575%
100.000	97,775%	96,850%	96,975%	97,325%	97,675%

Tabelle 31: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-3-5-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	98,800%	97,950%

Tabelle 32: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-3-6-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	98,750%	98,925%

Tabelle 33: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-6-1 Netzwerktopologie

Bei dieser Topologie (Tabelle 33) stellt sich der Lernerfolg erst bei einem hohen Lernfaktor (0,7 und 0,9) und einer hohen Trainingszyklenzahl (100.000) ein. Diese Topologie ist für einen praktischen Einsatz geeignet, allerdings muss eine höhere Zyklenzahl und somit eine längere Zeitdauer für das Training eingeplant werden.

Matrix für Netzwerktopologie 5-3-7-1 (fünf Knoten der Inputschicht, drei Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 34: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-3-7-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-1-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 35: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-1-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-2-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	97,050%	97,825%	98,450%
5.000	97,350%	98,625%	98,600%	98,650%	98,625%
10.000	98,625%	98,500%	98,450%	98,475%	98,600%
20.000	98,250%	98,500%	97,825%	97,775%	97,950%
30.000	98,350%	97,525%	97,475%	97,600%	97,875%
50.000	98,400%	97,150%	97,375%	97,625%	97,875%
100.000	98,050%	97,175%	97,400%	97,675%	97,900%

Tabelle 36: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-3-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	97,850%	98,475%
5.000	97,150%	98,725%	98,725%	98,850%	98,875%
10.000	98,725%	98,625%	98,550%	98,625%	98,850%
20.000	98,350%	98,625%	98,225%	98,350%	98,600%
30.000	98,375%	97,775%	97,800%	98,150%	98,525%
50.000	98,425%	97,400%	97,675%	98,150%	98,500%
100.000	98,125%	97,350%	97,725%	98,150%	98,525%

Tabelle 37: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-3-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-4-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	97,850%	98,475%
5.000	97,075%	98,725%	98,700%	98,850%	98,875%
10.000	98,700%	98,625%	98,550%	98,625%	98,850%
20.000	98,350%	98,625%	98,225%	98,350%	98,600%
30.000	98,375%	97,775%	97,800%	98,150%	98,525%
50.000	98,425%	97,400%	97,675%	98,150%	98,500%
100.000	98,125%	97,350%	97,725%	98,150%	97,650%

Tabelle 38: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-5-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	98,975%	99,000%
20.000	74,575%	98,850%	98,675%	98,825%	98,875%
30.000	74,575%	98,500%	98,450%	98,625%	98,900%
50.000	24,575%	97,800%	98,150%	98,575%	98,875%
100.000	98,425%	97,450%	98,150%	98,625%	97,925%

Tabelle 39: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-6-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	98,175%	98,975%	99,000%
20.000	74,575%	98,725%	98,675%	98,800%	98,875%
30.000	74,575%	98,525%	98,400%	98,625%	98,900%
50.000	98,900%	97,775%	98,150%	98,575%	98,875%
100.000	98,375%	97,450%	98,150%	98,625%	98,925%

Tabelle 40: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-7-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 41: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-7-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-4-8-1 (fünf Knoten der Inputschicht, vier Knoten der ersten Hiddenschicht, acht Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 42: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-4-8-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-1-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 43: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-1-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-2-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	97,000%	99,000%
10.000	24,575%	24,575%	98,825%	98,600%	98,675%
20.000	24,575%	98,425%	98,450%	98,250%	98,200%
30.000	24,575%	98,200%	97,675%	97,675%	98,150%
50.000	98,400%	97,425%	97,400%	97,700%	98,075%
100.000	98,375%	97,150%	97,475%	97,725%	98,100%

Tabelle 44: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-3-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	99,025%
10.000	24,575%	24,575%	98,925%	98,850%	98,875%
20.000	24,575%	98,525%	98,575%	98,725%	98,800%
30.000	24,575%	98,525%	98,125%	98,300%	98,600%
50.000	98,400%	97,500%	97,800%	98,200%	98,625%
100.000	98,400%	97,350%	97,825%	98,275%	98,625%

Tabelle 45: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-3-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-4-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	99,025%
10.000	24,575%	24,575%	98,925%	98,850%	98,875%
20.000	24,575%	98,525%	98,575%	98,725%	98,800%
30.000	24,575%	98,525%	98,125%	98,300%	98,600%
50.000	98,400%	97,500%	97,800%	98,200%	98,625%
100.000	98,400%	97,350%	97,825%	98,275%	98,625%

Tabelle 46: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-5-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	**0,9**
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	**99,075%**
20.000	74,575%	24,575%	98,925%	98,850%	98,975%
30.000	74,575%	98,925%	98,675%	98,800%	98,925%
50.000	24,575%	98,275%	98,125%	98,650%	98,900%
100.000	98,400%	97,475%	98,150%	98,650%	98,925%

Tabelle 47: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-5-1 Netzwerktopologie

Diese Topologie (Tabelle 47) enthält mit einer Erkennungsgenauigkeit der gesamten Betrugserkennungskomponente von 99,075% den höchsten Wert aller getesteten Topologien. Dieser Wert wird unter Verwendung von 10.000 Backpropagationsdurchläufen und einem Lernfaktor von 0,9 als Trainingsparameter erreicht. Die Diskussion dieses Ergebnisses folgt weiter unten in diesem Abschnitt.

Matrix für Netzwerktopologie 5-5-6-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	98,075%	98,950%
20.000	74,575%	24,575%	98,850%	98,825%	98,950%
30.000	74,575%	98,600%	98,650%	98,775%	98,925%
50.000	24,575%	97,975%	98,175%	98,625%	98,900%
100.000	98,625%	97,500%	98,150%	98,650%	98,925%

Tabelle 48: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-7-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 49: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-7-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-8-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, acht Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 50: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-8-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-1-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 51: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-1-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-2-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	97,900%	98,175%

Tabelle 52: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-3-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	98,525%	98,650%

Tabelle 53: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-3-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-4-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	98,500%	98,650%

Tabelle 54: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-5-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	24,575%
20.000	74,575%	24,575%	24,575%	24,575%	24,575%
30.000	74,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	98,850%	98,925%

Tabelle 55: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-6-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	24,575%
20.000	74,575%	24,575%	24,575%	24,575%	24,575%
30.000	74,575%	24,575%	24,575%	99,050%	98,975%
50.000	24,575%	24,575%	98,675%	98,850%	98,925%
100.000	24,575%	97,875%	98,200%	98,700%	98,925%

Tabelle 56: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-6-7-1 (fünf Knoten der Inputschicht, sechs Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 57: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-6-7-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-1-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, ein Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 58: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-1-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-2-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, zwei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	24,575%	24,575%	24,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 59: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-2-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-3-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, drei Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 60: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-3-1 Netzwerktopologie

Bei dieser Topologie (Tabelle 60) tritt der erwähnte Abfalleffekt von 74,575% auf 24,575% ein, allerdings bleibt – selbst bei hohem Lernfaktor und hoher Zyklenzahl – der Stabilisierungseffekt aus. Diese Resultate ergeben sich ebenfalls bei den folgenden Topologien mit sieben Knoten in der ersten Hiddenschicht. Aus diesem Grund sind diese Topologien mit dieser Ergebnisstruktur nicht für einen praktischen Einsatz geeignet.

Matrix für Netzwerktopologie 5-7-4-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, vier Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	24,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 61: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-5-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	24,575%
20.000	74,575%	24,575%	24,575%	24,575%	24,575%
30.000	74,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 62: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-6-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, sechs Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	24,575%
20.000	74,575%	24,575%	24,575%	24,575%	24,575%
30.000	74,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 63: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-6-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-7-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, sieben Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	24,575%	24,575%	24,575%
20.000	74,575%	24,575%	24,575%	24,575%	24,575%
30.000	74,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 64: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-7-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-7-8-1 (fünf Knoten der Inputschicht, sieben Knoten der ersten Hiddenschicht, acht Knoten der zweiten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	74,575%
30.000	74,575%	74,575%	74,575%	74,575%	74,575%
50.000	74,575%	74,575%	74,575%	74,575%	74,575%
100.000	74,575%	74,575%	74,575%	74,575%	74,575%

Tabelle 65: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-7-8-1 Netzwerktopologie

Bei den analysierten Topologien mit zwei Hiddenschichten ist zu beobachten, dass ab einschließlich sieben Knoten in einer Hiddenschicht die Eingabemuster nicht mehr richtig gelernt werden. Dies zeigt sich dadurch, dass alle Auswertungen bei solchen Topologien unabhängig von Zyklenzahl und Lernfaktor entweder einen Wert der Erkennungsgenauigkeit von 24,575% oder 74,575% aufweisen, der Stabilisierungseffekt aber nicht eintritt.
An dieser Stelle beginnt die Auswertung von Topologien mit drei Hiddenschichten, da eine weitere Erhöhung der Knotenmenge bei zwei Hiddenschichten keine höheren Werte der Erkennungsgenauigkeit als 74,575% mehr liefern würde (siehe Tabelle 65).

c) Topologien mit drei Hiddenschichten:

Bei den untersuchten Topologien mit drei Hiddenschichten wird von der besten Topologie mit zwei Hiddenschichten (= 5-5-5-1 Netzwerktopologie) ausgegangen und auf dieser Grundlage noch eine Schicht hinzugefügt.

Matrix für Netzwerktopologie 5-5-5-4-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, vier Knoten der dritten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	24,575%
1.000	24,575%	24,575%	24,575%	24,575%	24,575%
5.000	24,575%	24,575%	24,575%	24,575%	24,575%
10.000	24,575%	24,575%	24,575%	24,575%	24,575%
20.000	24,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 66: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-5-4-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-5-5-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, fünf Knoten der dritten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	24,575%
10.000	74,575%	74,575%	74,575%	24,575%	24,575%
20.000	74,575%	24,575%	24,575%	24,575%	24,575%
30.000	24,575%	24,575%	24,575%	24,575%	24,575%
50.000	24,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 67: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-5-5-1 Netzwerktopologie

Matrix für Netzwerktopologie 5-5-5-6-1 (fünf Knoten der Inputschicht, fünf Knoten der ersten Hiddenschicht, fünf Knoten der zweiten Hiddenschicht, sechs Knoten der dritten Hiddenschicht, ein Knoten der Outputschicht):

Lernfaktor Anzahl Lerndurchläufe	0,1	0,3	0,5	0,7	0,9
100	74,575%	74,575%	74,575%	74,575%	74,575%
1.000	74,575%	74,575%	74,575%	74,575%	74,575%
5.000	74,575%	74,575%	74,575%	74,575%	74,575%
10.000	74,575%	74,575%	74,575%	74,575%	74,575%
20.000	74,575%	74,575%	74,575%	74,575%	24,575%
30.000	74,575%	74,575%	74,575%	24,575%	24,575%
50.000	74,575%	24,575%	24,575%	24,575%	24,575%
100.000	24,575%	24,575%	24,575%	24,575%	24,575%

Tabelle 68: Klassifikationsergebnisse der Betrugserkennungskomponente mit 5-5-5-6-1 Netzwerktopologie

Nach Vollendung der Auswertungen der Betrugserkennungskomponente bei Verwendung von Netzwerktopologien mit drei Hiddenschichten ist insgesamt zu beobachten, dass sich die guten Erkennungsergebnisse von ca. 98% auf Topologien mit einer oder zwei Hiddenschichten beschränken. Dagegen werden beim Einfügen einer dritten Hiddenschicht die Muster nicht mehr richtig gelernt. Dies zeigt sich dadurch, dass die Outputwerte aller vom neuronalen Netzwerk untersuchten Testtransaktionen entweder einen Wert zwischen 0,4 und 0,5 (= Erkennungsgenauigkeit: 24,575%) oder zwischen 0,9 und 1,0 (= Erkennungsgenauigkeit: 74,575%) aufweisen (siehe dazu die separaten Arbeitsberichte der entsprechenden Topologien). Diese Resultate belegen die Aussagen aus [Dorf91, S. 102; Rume86, S. 1 - 2; Back06, S. 767], dass Netzwerktopologien mit einer oder maximal zwei Hiddenschichten für das Erlernen von numerischen Mustern am besten geeignet sind.
Im Rahmen der Experimente ist zu erkennen, dass Topologien mit einer geringen Knotenzahl (z.B. 5-1-2-1 oder 5-2-1 Topologie) ebenfalls gute bzw. stabile Erkennungsquoten (ca. 98%) erreichen. Zu diesem Resultat kamen z.B. auch die Autoren von [Lund03, S. 8] bei der Ermittlung der Topologie eines neuronalen Netzwerks zur Identifikation von Betrugsversuchen bei einem *Video on demand*-System (siehe Unterabschnitt 9.1.1). Dagegen werden von Topologien mit höheren Knotenzahlen (z.B. 5-6-7-1 oder 5-5-7-1 Topologie) die Muster oft nicht gelernt. Ab einer Knotenzahl von sieben Knoten in einer Hiddenschicht werden – wie oben erwähnt – die Muster nicht mehr gelernt (z.B. 5-7-5-1 oder 5-5-7-1 Topologie). Das Einfügen eines achten Knotens in die zweite Hiddenschicht verändert dieses Ergebnis nicht mehr. Ebenso stellt sich beim neuronalen Netzwerk kein Lernerfolg

ein, wenn die zweite Hiddenschicht nur aus einem Knoten besteht (z.B. 5-5-1-1 oder 5-4-1-1 Topologie).

Generell ist festzustellen, dass die Erkennungsquoten bei zunehmendem Lernfaktor höher werden. Bezüglich der Anzahl der Backpropagationsdurchläufe zeigen die Experimente, dass 100 Lernzyklen zu wenig sind um die vorliegenden Muster zu erlernen. Das bedeutet, zum Erreichen stabiler Ergebnisse mit einer Erkennungsgenauigkeit von ca. 98% sind hier mindestens 1.000 Lerndurchläufe notwendig (in Abhängigkeit von Lernfaktor und Topologie). Bei vielen Topologien tritt eine Stabilisierung des Lernerfolgs erst bei 100.000 Zyklen ein (z.B. 5-6-4-1 oder 5-3-6-1 Topologie).

Die Betrugserkennungskomponente mit einer Netzwerktopologie von 5-5-5-1 erfüllt in diesem Zusammenhang auf Basis der simulierten Transaktionsevents mit einer Erkennungsgenauigkeit von 99,075% bei 10.000 Backpropagationsdurchläufen und einem Lernfaktor von 0,9 die Anforderung der Betrugsexperten von einer Erkennungsgenauigkeit von 99% am besten, wie in Beispielauswertung 4 zu erkennen ist (die hervorgehobenen Zeilen enthalten die Anzahl der jeweils korrekt klassifizierten Transaktionsevents).

Beispielauswertung 4:

Auswertung mit folgenden Parametern:
Lernfaktor: 0,9, Anzahl Backpropagationsdurchläufe: 10000

Insgesamt: 4000 Testevents analysiert, davon:

Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1 (= false positive)
Betrugsfälle zwischen 0,9 und 1,0: 1981

Nicht-Betrugsfälle zwischen 0,8 und 0,9: 1 (= nicht zuordenbar)
Betrugsfälle zwischen 0,8 und 0,9: 2 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0 (= nicht zuordenbar)
Betrugsfälle zwischen 0,7 und 0,8: 10 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,6 und 0,7: 1 (= nicht zuordenbar)
Betrugsfälle zwischen 0,6 und 0,7: 5 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,5 und 0,6: 2 (= nicht zuordenbar)
Betrugsfälle zwischen 0,5 und 0,6: 0 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,4 und 0,5: 2 (= nicht zuordenbar)
Betrugsfälle zwischen 0,4 und 0,5: 0 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,3 und 0,4: 1 (= nicht zuordenbar)
Betrugsfälle zwischen 0,3 und 0,4: 0 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,2 und 0,3: 1 (= nicht zuordenbar)
Betrugsfälle zwischen 0,2 und 0,3: 0 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,1 und 0,2: 9 (= nicht zuordenbar)
Betrugsfälle zwischen 0,1 und 0,2: 0 (= nicht zuordenbar)

Nicht-Betrugsfälle zwischen 0,0 und 0,1: 999
Betrugsfälle zwischen 0,0 und 0,1: 2 (= false negative)

Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0

Erkennungsgenauigkeit: 99,075%

Die Ermittlung der Erkennungsgenauigkeit von 99,075% der Betrugserkennungskomponente erfolgt wiederum durch die Verwendung eines Dreisatzes ((1.981 + 999 + 983) * 100 / 4.000). Hierbei liegt die *false positive*-Quote bei 0,025% (1 * 100 / 4.000), die *false negative*-Quote bei 0,050% (2 * 100 / 4.000) und 0,850% (34 * 100 / 4.000) der Testevents können nach der oben genannten Definition nicht exakt zugeordnet werden. Wird für diese Beispielauswertung das Ergebnis des neuronalen Netzwerks separat berechnet, ergibt sich eine Erkennungsquote von 98,77% (1.981 Betrugsfälle und 999 Nicht-Betrugsfälle werden richtig klassifiziert => (1.981 + 999) * 100 / 3.017).
In Abbildung 44 ist der Verlauf der Erkennungsquoten dieser optimalen Topologie in Form eines dreidimensionalen Diagramms graphisch dargestellt. Im Anhang 3 befinden sich die Auswertungen aller Parametereinstellungen dieser Netzwerkstruktur.

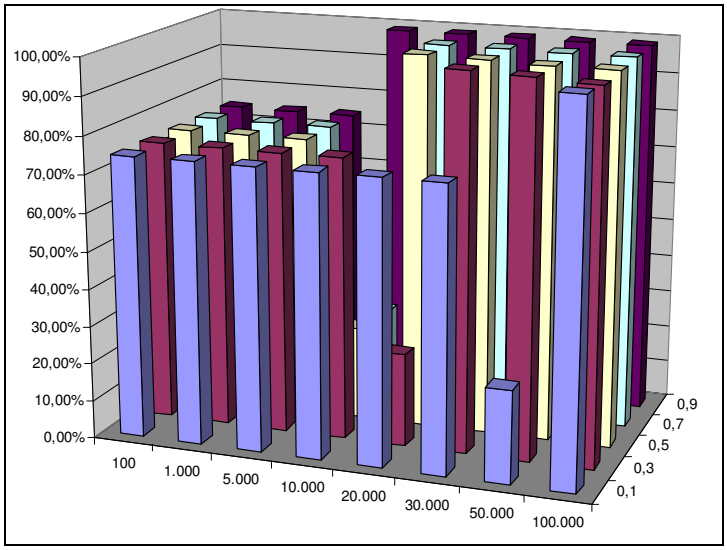

Abbildung 44: Dreidimensionales Balkendiagramm der Erkennungsgenauigkeiten der optimalen Netzwerktopologie

Die Länge des Diagramms gibt die Anzahl der Backpropagationsdurchläufe an, die Breite repräsentiert den Lernfaktor. Die Balkenhöhe wird vom Prozentsatz der Erkennungsgenauigkeit der jeweiligen Kombination der beiden Parameter bestimmt. Abbildung 44 zeigt in diesem Zusammenhang gut das Gefälle zwischen kleinen Lernfaktoren und geringer

Anzahl von Backpropagationsdurchläufen im Vergleich zu hohen Lernfaktoren und höheren Durchlaufzahlen.

Bei einer näheren Untersuchung dieser optimalen Topologie, z.B. mit einem Lernfaktor von 0,9 und 9.000 Lernzyklen liegt die Erkennungsgenauigkeit der Betrugserkennungskomponente noch bei 24,575%, dagegen wird sie im Vergleich zu 99,075% bei 10.000 Lernzyklen mit einer Quote von jeweils 99,000% bei 11.000 und 15.000 Lerndurchläufen wieder leicht schlechter. Das bedeutet, hier tritt der beobachtete Stagnierungseffekt ein (siehe dazu die separaten Arbeitsberichte der 5-5-5-1 Topologie mit 9.000, 11.000 und 15.000 Lernzyklen bei einem Lernfaktor von 0,9).

In Abbildung 45 ist das optimale neuronale Netzwerk der Betrugserkennungskomponente, das eine Erkennungsgenauigkeit von 99,075% bei 10.000 Backpropagationsdurchläufen und einem Lernfaktor von 0,9 aufweist, mit seinen antrainierten Zielgewichtswerten dargestellt.

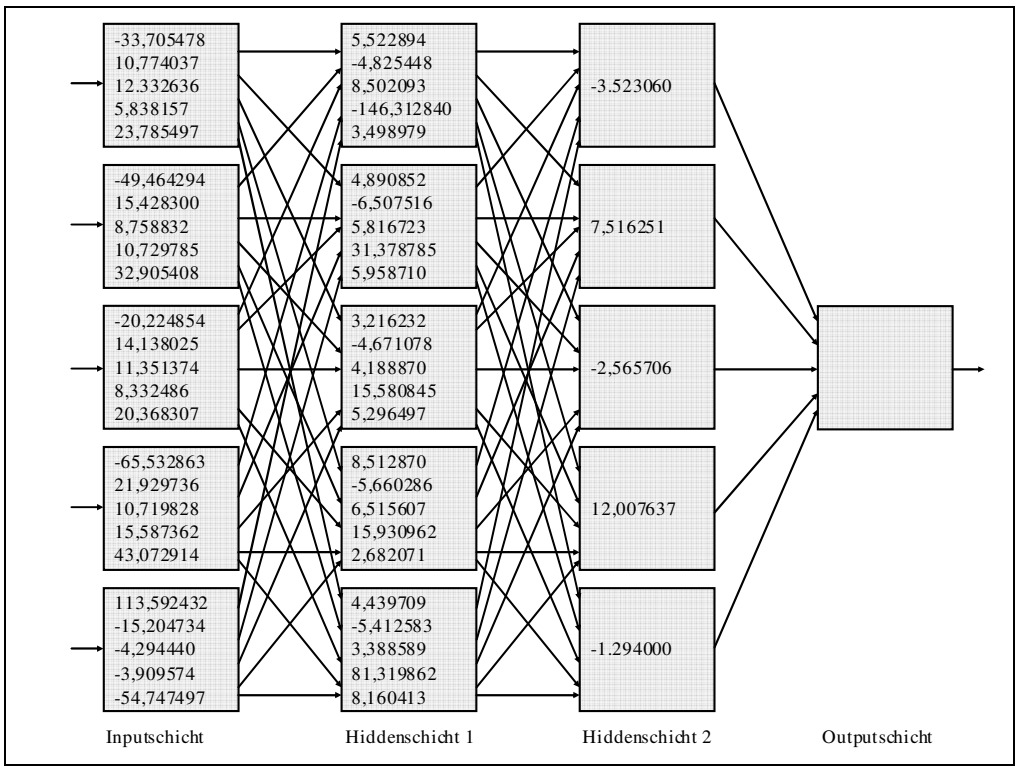

Abbildung 45: Struktur und Zielgewichtswerte des optimalen neuronalen Netzwerks

Aus Gründen der Übersichtlichkeit sind die Gewichtswerte in den Abbildungen 45 und 46 im Ausgangsknoten und nicht in den Übergangslinien eingetragen. Von oben nach unten gelesen stellen die Zielgewichtswerte die Übergänge zu den Knoten der nächsten Schicht, ebenfalls von oben nach unten dar. Der Inputknoten links oben nimmt den aktuellen Diskriminanzwert entgegen. Je weiter die Knoten in der Inputschicht in den Abbildun-

gen 45 und 46 in der Reihenfolge nach unten gehen, desto älter sind die Transaktionen, deren errechnete Diskriminanzwerte dem jeweiligen Knoten übergeben werden.
Dieses optimale neuronale Netzwerk besitzt folgende Initialgewichtswerte – dargestellt in Abbildung 46 – als Ausgangsbasis für das Training.

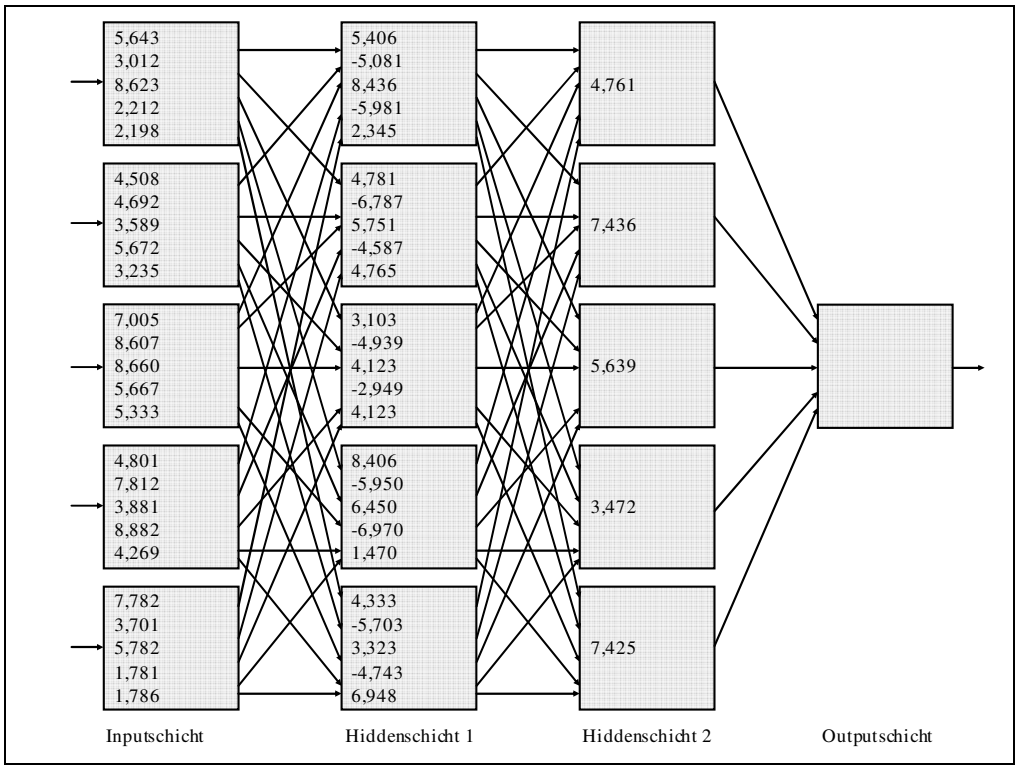

Abbildung 46: Struktur und Initialgewichtswerte des optimalen neuronalen Netzwerks

Anhand der beiden Netzwerkgraphiken aus den Abbildungen 45 und 46 ist die Veränderung der Gewichte nach den 10.000 Backpropagationsdurchläufen auf Basis des Lernfaktors von 0,9 gut ersichtlich.
Bei der Wahl der Initialgewichte ist es wichtig, dass die Vorzeichen der Werte der Gewichte der ersten Gewichtsübergangsschicht (Gewichte von Inputschicht zur ersten Hiddenschicht) das gegenteilige Vorzeichen der übergebenen Inputwerte (= Diskriminanzwerte) besitzen. Dies ist notwendig, weil ansonsten die Trainingsmenge vom neuronalen Netzwerk nicht gelernt wird, weil der Graph der Gradientenfunktion zu oszillieren beginnt. In diesem konkreten Fall weisen alle übergebenen aktuellen und historischen Diskriminanzwerte ein negatives Vorzeichen auf. Daher ist jedem Gewichtswert des Übergangs von Input- zur ersten Hiddenschicht ein positives Vorzeichen zugeteilt. Für die restlichen Verbindungen und deren Gewichte trifft dieser Umstand nicht zu, d.h. für diese Gewichte ist das Vorzeichen beliebig wählbar. Im Anhang 4 stehen als Ergebnis einer Voruntersuchung die Ausgabewerte des in Abbildung 46 aufgezeigten neuronalen Netzwerks bei 10.000 Backpropagationsdurchläufen und einem Lernfaktor von 0,9. Allerdings besitzt

hierbei die erste Gewichtsebene die gleichen Vorzeichen wie die übergebenen Diskriminanzwerte (d.h. negative Vorzeichen). In diesem Fall kann kein *event* exakt zugeordnet werden, da jede Transaktion nach der Analyse einen Outputwert des neuronalen Netzwerks zwischen 0,3 und 0,4 besitzt (siehe Anhang 4). Diese Erkenntnis ist bei der Durchführung der oben beschriebenen Experimente berücksichtigt.
Bezüglich der Performance wird der von `StreamBase Inc.` angegebene Durchsatz von 500.000 *events* pro Sekunde (siehe Tabelle 5) nicht erreicht, da ein handelsüblicher Laptop die Hardwarebasis dieser Experimente darstellt. Zur Erreichung von Echtzeitwerten müsste für den Ablauf der Experimente leistungsfähigere Hardware eingesetzt werden. Die analysierten *events* im Rahmen dieser Experimente bilden keine *low level events* der Netzwerkebene, auf die der angegebene Eventdurchsatz von *StreamBase Studio* basiert, sondern *business level events* in Form von Online-Überweisungen (siehe Einleitung zu Kapitel 7) mit den genannten Attributen. Ebenso mindert das Ausführen eines komplexen Analysealgorithmus sowie die Notwendigkeit der Datenbankzugriffe zum Einholen der zusätzlichen Attribute und der historischen Diskriminanzwerte die Ausführgeschwindigkeit des entwickelten Hybrid-Modells.
Bei einer Analyse der im Abschnitt 2.2 genannten Beispielfälle aus den Tabellen 1 bis 4 unter Verwendung der trainierten Zielgewichte der optimalen Netzwerktopologie 5-5-5-1 aus Abbildung 45 ergeben sich folgende Ergebnisse (Tabelle 69 beinhaltet die vier Fälle mit ihren Eingabemustern für die fünf Inputknoten und dem jeweiligen Ausgabewert des neuronalen Netzwerks).

Fall	Beschreibung	Aktueller Diskriminanzwert	Historischer Diskriminanzwert 1	Historischer Diskriminanzwert 2	Historischer Diskriminanzwert 3	Historischer Diskriminanzwert 4	Ausgabewert neuronales Netzwerk
1	Verdächtiger Betrugsfall	-0,0021619310 8452867	-0,0000057508 5591535033	-0,0000327012 594356374	-0,0000186190 206018643	-0,0000366906 681806747	0,990287117003944
2	Unverdächtiger Betrugsfall	-0,0007587108 24994126	-0,0000183113 291362225	-0,0000163498 906296789	-0,0000237097 24942845	-0,0000296512 583890115	0,981479227586163
3	Unverdächtiger Nicht-Betrugsfall	-0,0000217609 149338353	-0,0000208335 146641225	-0,0000182136 617356882	-0,0000546221 786269757	-0,0000241635 348265239	0,00945033319803732
4	Verdächtiger Nicht-Betrugsfall	-0,0005734093 09587571	-0,0005417409 38094025	-0,0002313862 24986025	-0,0002296809 42453345	-0,0001051533 6965943	0,0159037984140946

Tabelle 69: Analyseergebnisse der vier Beispieltransaktionen

Die Analyseergebnisse des neuronalen Netzwerks in Tabelle 69 zeigen, dass von den Beispieltransaktionen die Muster aus aktuellen und historischen Diskriminanzwerten sowohl bei den Betrugs- als auch bei den Nicht-Betrugsfällen eindeutig klassifiziert werden. Daraus ergibt sich, dass das Hybrid-Modell zur Betrugserkennung dieser Arbeit auch Transaktionen, bei denen der Betrugsstatus auf den ersten Blick nicht so eindeutig ist, korrekt klassifizieren kann, da die Betrugsfälle mit 1,0 und die Nicht-Betrugsfälle mit 0,0 als Ausgabewert trainiert wurden. In diesem Beispiel befindet sich der verdächtige Be-

trugsfall näher am jeweiligen trainierten Ausgabewert als der unverdächtige Betrugsfall (siehe Tabelle 69).

Die Analysen der gesamten Testmenge als auch die Auswertungen der vier Beispielfälle belegen, dass der Ansatz dieser Arbeit für die praktische Betrugserkennung im Online-Banking durchaus geeignet wäre. Momentan verfolgen die Kreditinstitute und Rechenzentrumsdienstleister allerdings stärker eine Betrugspräventionsstrategie. Die Gründe dafür liegen nach Aussage der interviewten Experten sowie [Agge06, S. 2] zum einen darin, dass *false positive*-Klassifikationen von den Banken zum Teil negativer bewertet werden als unerkannte Betrugsfälle. Zum anderen werden die Kosten und der Zeitaufwand für die Einführung eines solchen Systems gescheut. Des Weiteren werden Performanceeinbußen in der Transaktionsabwicklung beim Ausführen eines Betrugserkennungssystems befürchtet.

Bezüglich des konkreten Modells dieser Arbeit wäre die Implementierung in einem Kreditinstitut bzw. einem Bankrechenzentrum ebenfalls aufwändig, da neben der technischen Integration auch zusätzliche notwendige Berechnungen und Abfragen für die Arbeit des Modells im laufenden Betrieb durchgeführt werden müssten. Diese sind im Einzelnen:

- Die historischen Diskriminanzwerte jedes Kunden.
- Der maximale Transaktionsbetrag jedes Kunden.
- Der durchschnittliche Transaktionsbetrag jedes Kunden.
- Die durchschnittliche Anzahl an Online-Transaktionen und Kontozugriffen jedes Kunden.
- Die IP-Adresse des Quellrechners der Transaktion mit dem entsprechenden Providerbereich.
- Die benötigte Zeit des Kunden für die Durchführung der Online-Überweisung.
- Die Feststellung, ob auf das Zielkonto schon einmal eine Überweisung durchgeführt wurde bzw. ob der Empfänger vertrauenswürdig ist.
- Die Abfrage, ob die Zielbank im Ausland liegt.

Allerdings würden nach einer erfolgreichen Trainings- und Testphase ähnlich gute Erkennungsergebnisse auch in der Praxis zustande kommen, da die Daten- bzw. Eventsimulation auf Basis der Angaben von erfahrenen Betrugsexperten durchgeführt wurde. Ob das entwickelte Hybrid-Modell in Zukunft in die Strategie der Banken passt, wird sich zeigen. Ein Ausblick auf die Zukunft wird abschließend im folgenden Schlusskapitel gegeben.

10 Schlussbemerkung

In diesem Kapitel werden die Ergebnisse dieser Arbeit zusammengefasst und ein Ausblick auf weiterführende Tätigkeiten, die dieser Arbeit folgen könnten, gegeben.

10.1 Zusammenfassung der Ergebnisse dieser Arbeit

Das Ziel dieser Arbeit war es, die im Einleitungskapitel erwähnte wissenschaftliche Fragestellung zu beantworten, wie effektiv der Einsatz von *Complex Event Processing*-Technologie in Kombination mit maschinellen Lernverfahren ist, Betrugstransaktionen im Online-Banking aus der Gesamtmenge der Transaktionen zu identifizieren. In diesem Zusammenhang wurde anhand von einschlägigen Statistiken gezeigt, dass Identitätsdiebstahl beim Online-Banking ein wachsendes Problem darstellt und somit die gängigen Betrugspräventionsmaßnahmen keinen ausreichenden Schutz der Bankkunden gewährleisten. Zur Lösung des Problems wurde eine Architektur aus einer CEP *engine* in Verbindung mit einer Kombination der maschinellen Lernverfahren Entscheidungsbaum, Diskriminanzanalyse und neuronalem Netzwerk in Form eines Hybrid-Modells mit dem Ziel der Betrugserkennung in Echtzeit diskutiert. Es zeigt sich im Rahmen der durchgeführten Experimente auf Basis der simulierten, exemplarischen Trainings- und Testmenge, dass sich nach der Definition dieser Arbeit das VIRT-Problem für diesen Anwendungsfall mit 99,075% Erkennungsgenauigkeit lösen lässt. Zur Ermittlung dieses Ergebnisses wurde das Hybrid-Modell implementiert und die Betrugserkennungskomponente mit mehreren Netzstrukturen getestet, wobei nicht jede Topologie für die Praxis geeignete Ergebnisse erzeugte. Die optimale Netzwerktopologie besteht aus fünf Inputknoten, zwei Hiddenschichten mit jeweils fünf Knoten und einem Outputknoten, wobei diese Netzstruktur mit einem Lernfaktor von 0,9 und 10.000 Backpropagationsdurchläufen trainiert werden muss.

In diesem Zusammenhang ist davon auszugehen, dass sich bei einer anderen Datenbasis und –menge die optimale Netzwerktopologie sowie die Ausprägung der Trainingsparameter ebenfalls ändern könnten. In einem solchen Fall müsste wiederum nach der optimalen Topologie geforscht werden, wobei die Erkenntnisse von [Back06, S. 767], [Dorf91, S. 102] und [Rume86, S. 1 - 2] erneut als Ausgangsbasis herangezogen werden können (siehe Abschnitt 9.3). Bezüglich der Performance konnte im Rahmen der Experimente keine exakte Aussage getroffen werden, da bei einem realen Einsatz die Betrugserkennungsanwendung mit leistungsfähigerer Hardware ausgeführt werden würde. Dieses Modell wurde auf einem handelsüblichen Anwenderlaptop programmiert, der zur Durchfüh-

rung der Experimente verwendet wurde, da im Laufe der Arbeit kein Partner im Bankenumfeld gefunden werden konnte, der bereit war, das Hybrid-Modell innerhalb seiner technischen Infrastruktur zu implementieren und zu testen.

10.2 Ausblick in die Zukunft

Ein Rechenzentrum eines Kreditinstituts bzw. ein Anbieter für Bankrechenzentrumsdienstleistungen, welches das beschriebene Betrugserkennungsmodell in der Praxis verwenden möchte, muss einerseits – wie im Abschnitt 9.3 erwähnt – die entsprechenden zusätzlichen Kundendaten wie z.B. durchschnittlicher Transaktionsbetrag oder durchschnittliche Anzahl der Transaktionen ermitteln und bereitstellen. Hierfür könnten z.B. die Daten aus einem *Data Warehouse* – sofern vorhanden – verdichtet und in entsprechenden Tabellen hinterlegt werden, auf welche die Betrugserkennungsanwendung zugreifen kann. Falls das Kreditinstitut bzw. das Rechenzentrum keine CEP-Technologie im Einsatz hat, muss eine CEP *engine* installiert und konfiguriert werden.
Andererseits muss die Bank ein Konzept für das Neutrainieren des neuronalen Netzwerks und die Anpassung der Attribute entwickeln. Eine Anpassung des Modells sollte spätestens zu dem Zeitpunkt erfolgen, wenn die ersten Betrugsfälle nacheinander unerkannt bleiben bzw. sich die Anzahl der *false positive* klassifizierten Transaktionen erhöht. Die zeitlichen Abfolgen der Modellüberprüfung sind von der zu analysierenden Transaktionsmenge abhängig. Zur Aufrechterhaltung der Qualität sollten die Betrugsbeauftragten turnusmäßig mindestens einmal im Monat die Zuverlässigkeit der Modelleinstellungen überprüfen. Bei einem spontanen Anstieg der Fehlklassifikationen sollte das Modell so schnell wie möglich angepasst bzw. neu trainiert werden.
Allerdings kann der Ansatz dieser Arbeit unabhängig vom Anwendungsfall nur eine Betrugswahrscheinlichkeit ausgeben und auf einen Verdachtsfall hinweisen. Die endgültige Bestätigung bzw. Aufklärung muss durch einen Menschen erfolgen, wodurch auch in Zukunft der Bedarf an Betrugsexperten gegeben sein wird [Bose06, S. 5].
Im Moment laufen Gespräche, die Resultate dieser Arbeit interessierten Kreditinstituten vorzustellen um auf diese Weise das Hybrid-Modell zur Betrugserkennung im praktischen Betrieb der Banken bzw. der Bankrechenzentren zu integrieren. Eine Möglichkeit der Integration wäre z.B. die Einbindung als Dienst innerhalb einer Serviceorientierten Architektur (SOA) (SOA beschreibt eine Architektur, deren Softwarekomponenten als Dienste (engl.: *Services*) verfügbar sind und unabhängig von der technischen Basis beliebig verteilt sein können, für weiterführende Literatur siehe [Heut07]). In diesem Zusammenhang wäre es auch möglich, nur die Betrugserkennungskomponente für sich alleine ohne CEP-Technologie einzusetzen. Als Datenversorgungskomponente müsste zu diesem Zweck

eine andere Datenbasis gefunden werden, wie z.B. eine Schnittstelle zu den operativen Systemen. Allerdings ist in diesem Szenario die Echtzeitfähigkeit der Betrugserkennungsanwendung nicht gewährleistet.

Eine Alternative für die Praxis wäre auch ein Betrugserkennungsmodell bestehend aus einem regelbasierten System, wobei im Falle des Einsatzes eines solchen Modells permanent die Entwicklung der Betrugsfälle und deren Attributausprägungen beobachtet werden müsste. Entscheidend ist hierbei, wie schnell die Anpassung der Regeln an die Entwicklung der Betrugsfälle vorgenommen werden kann.

Da das Modell dieser Arbeit aufgrund der eingesetzten Methoden des maschinellen Lernens – wie im Abschnitt 6.2 erwähnt – flexibel gestaltet ist, kann es ebenso in anderen Branchen sowie für andere Anwendungsfälle eingesetzt werden. Der Grund dafür ist, dass neuronale Netzwerke auch andere numerische Musterstrukturen lernen können (siehe Unterabschnitt 3.3.2), die sich aus Diskriminanzwerten verschiedener metrischer Attribute zusammensetzen. Die gleiche Flexibilität gilt auch für den Aufbau von Entscheidungsbäumen (siehe Abschnitt 3.1) bei nicht-metrischen Attributen. Ein entsprechendes Projekt ist in diesem Zusammenhang für den Anwendungsfall der Betrugsbekämpfung in der Schadensfallabwicklung bei einer großen deutschen Versicherung bereits in der Initialisierungsphase. Die zugrundeliegenden betrugsrelevanten Attribute setzen sich in dieser Branche aufgrund eines abweichenden fachlichen Hintergrunds anders zusammen, wie z.B. Anzahl der Schäden in der Vergangenheit oder Laufzeit der Versicherungspolice des Kunden, siehe dazu [Widd09]. Dieses Projekt wird demnächst in die Realisierungsphase übergehen, auch aufgrund der Ergebnisse dieser Arbeit.

LITERATURVERZEICHNIS

[Aaai09] AAAI Spring Symposium. (2009). Intelligent Event Processing. <http://icep-aaai09.fzi.de/> (Aufgerufen am 26.03.2009).

[Abu07] Abu-Nimeh, S., Nappa, D., und Wang, X. (2007). A Comparison of Machine Learning Techniques for Phishing Detection. In: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit (Pittburgh), Band 269, S. 60 - 69, New York: ACM.

[Agen09] Agent Logic Inc. (2009). RulePoint® – Complex Event Processing Software. <http://www.agentlogic.com/products/rulepoint.html> (Aufgerufen am 05.08.2009).

[Agge06] Aggelis, V. (2006). Offline Internet Banking Fraud Detection. In: Proceedings of the 1st International Conference on Availability, Reliability and Security (Wien), Band 1, S. 904 - 905, Washington: IEEE Computer Society.

[Alba07] Albanese, J. (2007). Combating Piracy: Intellectual Property Theft and Fraud. New Brunswick: Transaction Publishers.

[Aler09] Aleri Inc. (2009). Aleri Streaming Platform. <http://www.aleri.com/products/aleri-cep/aleri-streaming-platform> (Aufgerufen am 05.08.2009).

[Alzo08] Alzomai, M., AlFayyadh, B., Josang, A., und McCullagh, A. (2008). An experimental investigation of the usability of transaction authorization in online bank security systems. In: Proceedings of the 6th Australasian Conference on Information security (Wollongong), Band 81, S. 65 - 73, Darlinghurst: Australian Computer Society.

[Ammo08] Ammon, R.v., Emmersberger, C., Springer, F., und Wolff, C. (2008). Event-Driven Business Process Management and its Practical Application Taking the Example of DHL. In: Proceedings of the 1st International workshop on Complex Event Processing for the Future Internet (Wien), <http://icep-fis08.fzi.de> (Aufgerufen am 29.05.2009).

[Ammo09a] Ammon, R.v., Emmersberger, C., Ertlmeier, T., Etzion, O., Paulus, T., und Springer, F. (2009). Existing and Future Standards for Event-Driven Business Process Management. In: Proceedings of the 3rd International Conference on Distributed Event-Based Systems (Nashville), Industrial Paper, New York: ACM.

[Ammo09b] Ammon, R.v., Ertlmaier, T., Etzion, O., Kofman, A., und Paulus, T. (2009). Integrating Complex Events for Collaborating and Dynamically Changing Business Processes. In: Proceedings of International Conference on Service-oriented computing, ICSOC/ServiceWave '09 (Stockholm), Band 1, S. 370 - 384, Berlin, Heidelberg: Springer Verlag.

[Andr03] Andresen, A. (2003). Komponentenbasierte Softwareentwicklung mit MDA, UML und XML. München, Wien: Hanser Verlag.

[Arbe07] Arbeitsgruppe Identitätsschutz im Internet (Hg). (2007). AG Neunkirchen: Urteil vom 20.06.2008, Az. 11, Ds 33, Js 1148/06 (27/07). <https://www.a-i3.org/images/ ag%20neunkirchen_urteil.pdf> (Aufgerufen am 23.12.2008).

[Arbe08] Arbeitsgruppe Identitätsschutz im Internet (Hg). (2008). Amtsgericht Wiesloch: Urteil vom 20.06.2008. <https://www.a-i3.org/images/ag%20 wiesloch_urteil.pdf> (Aufgerufen am 23.12.2008).

[Assi02] Assies, P. (2002). Anwalts-Taschenbuch Bankrecht. Köln: Verlag Dr. Otto Schmidt.

[Back06] Backhaus, K., Erichson, B., Plinke, W., und Weiber, R. (2006). Multivariate Analysemethoden: Eine anwendungsorientierte Einführung. 11. Auflage. Berlin, Heidelberg: Springer Verlag.

[Badr07] Badra, M., El-Sawda, S., und Hajjeh, I. (2007). Phishing Attacks and Solutions. In: Proceedings of the 3rd International Conference on Mobile multimedia communications (Nafpaktos), Band 329, Artikel Nr. 42, Brüssel: ICST.

[Bahr03] Bahrenberg, G., Giese, E., und Nipper, J. (2003). Statistische Methoden in der Geographie: Band 2 Multivariate Statistik. 2. Auflage. Berlin, Stuttgart: Gebrüder Borntraeger Verlagsbuchhandlung.

[Bank08] Bankenverband – Bundesverband deutscher Banken. (Hg). (2008). Online-Banking: Ergebnisse einer repräsentativen Umfrage des Bankenverbandes, April2008. <http://www.bankenverband.de/pic/artikelpic/ 072008/080707_Online-Banking-Charts.pdf> (Aufgerufen am 02.04.2009).

[Bass06] Bass, T. (2006). Fraud Detection and Event Processing for Predictive Business. TIBCO Whitepaper. <http://www.tibco.com/resources/mk/fraud_ detection_in_cep_wp.pdf> (Aufgerufen am 10.11.2008).

[Bate07] Bates, J. (2007). Algorithmic Trading. <http://complexevents.com/?p=182> (Aufgerufen am 23.05.2008).

[Beck03] Becker, J., und Meise, V. (2003). Strategie und Ordnungsrahmen. In: Becker, J., Kugeler, M., und Rosemann, M. (Hg). Prozessmanagement – Ein Leitfaden zur prozessorientierten Organisationsgestaltung. Berlin, Heidelberg, New York: Springer Verlag.

[Beie06] Beierle, C., und Kern-Isberner, G. (2006). Methoden wissensbasierter Systeme: Grundlagen Algorithmen Anwendungen. 3. Auflage. Wiesbaden: Vieweg Verlag.

[Bidg04] Bidgoli, H. (2004). The Internet Encyclopedia. 1. Auflage. Bakersfield: John Wiley & Sons.

[Bign06] Bignell, K. (2006). Authentication in an Internet Banking Environment: Towards Developing a Strategy for Fraud Detection. In: Proceedings of the International Conference on Internet Surveillance and Protection (Cap Esterel), Band 1, S. 23 - 30, Washington: IEEE Computer Society.

[Bitk08] Bitkom – Bundesverband Informationswirtschaft Telekommunikation und neue Medien e.V. (2008). Zahl der Phishing Opfer erreicht Höhepunkt. <http://www.bitkom.org/de/presse/8477_53846.aspx> (Aufgerufen am 06.09.2008).

[Blob98] Blobel, V., und Lohrmann, E. (1998). Statistische und numerische Methoden der Datenanalyse. Stuttgart, Leipzig: Teubner Verlag.

[Bode03] Bodendorf, F. (2003). Daten- und Wissensmanagement. 2. Auflage. Berlin, Heidelberg: Springer Verlag.

[Böhm02] Böhm, R., und Fuchs, E. (2002). System-Entwicklung in der Wirtschaftsinformatik. 5. Auflage. Zürich: Hochschul-Verlag an der ETH.

[Bolt02] Bolton, R., und Hand, D. (2002). Statistical Fraud Detection: A Review. In: Statistical Science, Band 12, Nr. 3, S. 235 - 255.

[Borg03] Borgelt, C., Klawonn, F., Kruse, R., und Nauck, D. (2003). Neuro-Fuzzy-Systeme: Von den Grundlagen künstlicher Neuronaler Netze zur Kopplung mit Fuzzy-Systemen. 3. Auflage. Wiesbaden: Vieweg Verlag.

[Bose06] Bose, R. (2006). Intelligent Technologies for Managing Fraud and Identity Theft Information Technology. In: Proceedings of the 3rd International Conference on Information Technology - New Generations (Las Vegas), Band 1, S. 446 - 451, Washington: IEEE Computer Society.

[Boss04] Bossel, H. (2004). Systeme Dynamik Simulation: Modellbildung, Analyse und Simulation komplexer Systeme. Norderstedt: Books on Demand GmbH.

[Brug04] Brugger, T. (2004). Data Mining Methods for Network Intrusion Detection. <http://www.bruggerink.com/~zow/papers/brugger_dmnid.pdf> (Aufgerufen am 24.05.2008).

[Bund08] Bundeskriminalamt. (2008). Polizeiliche Kriminalstatistik 2007. <http://www.bka.de/pks/pks2007/download/pks2007_imk_kurzbericht.pdf> (Aufgerufen am 06.09.2008).

[Bund09] Bundeskriminalamt. (2009). Polizeiliche Kriminalstatistik 2008. <http://www.bka.de/pks/pks2008/download/pks2008_imk_kurzbericht.pdf> (Aufgerufen am 15.06.2009).

[Bung08] Bungartz, H., Zimmer, S., Buchholz, M., und Pflüger, D. (2008). Modellbildung und Simulation: Eine anwendungsorientierte Einführung. Berlin, Heidelberg: Springer Verlag.

[Cain09] Cain, P., und Jevans, D. (2009). Extensions to the IODEF-Document Class for Reporting Phishing, Fraud, and Other Crimeware. <http://xml.coverpages.org/draft-cain-post-inch-phishingextns-06.txt> (Aufgerufen am 03.07.2009).

[Chan06] Chandrasekaran, M., Narayanan, K., und Upadhyaya, S. (2006). Phishing email detection based on structural properties. In: Proceedings of the 9th NYS Cyber Security Conference (Albany), Band 1, S. 2 - 8, New York: CSCIC.

[Chan07] Chandy, M., Ramo, S., und Schulte, R. (2007). What is Event Driven Architecture (EDA) and Why Does it Matter?. <http://complexevents.com/wpcontent/uploads/2007/07/EDA%20article%20long%20Chandy%20and%20Schulte%2015%20July%202007%20final.pdf> (Aufgerufen am 22.02.2009).

[Chan09] Chandy, M., und Schulte, R. (2009). Event Processing: Designing IT Systems for Agile Companies. New York, Chicago, San Francisco u.a.: Mc Graw Hill.

[Chen97] Chen, K., Yu, X., und Chi, H. (1997). Combining Linear Discriminant Functions with Neural Networks for Supervised Learning. In: Neural Computing & Applications, Band 6, Nr. 1, S. 19 - 41.

[Chip06] Chipman, H., George, E., und McCulloch, R. (2006). BART: Bayesian Additive Regression Trees. <http://www-stat.wharton.upenn.edu/~edgeorge/Research_papers/BART%206--06.pdf> (Aufgerufen am 18.02.2009).

[Chri97] Christensen, R. (1997). Log-Linear Models and Logistic Regression. 2. Auflage. New York: Springer Verlag.

[Comp08] complexevents.com (2008). Complex Event Processing. <http://complexevents.com> (Aufgerufen am 02.05.2008).

[Conz07] Conz, N., und Rodier, M. (2007). Predictive Analytics and Complex Event Processing Technology Move to Cutting Edge of Financial Services Industry: As data volumes continue to rise, bankers, insurers and traders are leveraging predictive models to anticipate future behavior and events. <http://www.insurancetech.com/showArticle.jhtml?articleID=202805132> (Aufgerufen am 17.02.2009).

[Cora07] Coral8 Inc. (2007). Guide to Evaluating Complex Event Processing Engines <http://members.ep-ts.com/images/7/73/GuideToEvaluatingCEPEngines.pdf> (Aufgerufen am 16.11.2009).

[Cora09] Coral8 Inc. (2009). The power of Continuous Intelligence™. <http://www.coral8.com/solutions/continuous-intelligence.html> (Aufgerufen am 05.08.2009).

[Cove08] Cover Pages. (2008). Technology Reports: Incident Object Description and Exchange Format (IODEF). <http://xml.coverpages.org/iodef.html> (Aufgerufen am 03.07.2009).

[Cris00] Cristianini, N., und Shaw-Taylor, J. (2000). An Introduction to Support Vector Machines and other kernel based learning methods. Cambridge, New York, Oakleigh u.a.: Cambridge University Press.

[Cybe08] Cybernetics Complex Event Processing Blog. (2008). Keyloggers: Why Banks Need Two-Factor Authentication. Eintrag von: Tim Bass, 14.01.2008. <http://www.thecepblog.com/2008/01/14/keyloggers-why-banks-need-two-factor-authentication/> (Aufgerufen am 27.09.2008).

[Dab09] DAB bank. (2009). Online-Überweisungslimit. <http://www.dab-bank.de/hilfe-service/faq/online-ueberweisungslimit.html> (Aufgerufen am 12.03.2010).

[Dand07a] Dandash, O., Le, P., und Srinivasan, B. (2007). Security Analysis for Internet Banking Models. In: Proceedings of the 8th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (Qingdao), Band 3, S. 1.141 - 1.146, Washington: IEEE Computer Society.

[Dand07b] Dandash, O., Le, P., und Srinivasan, B. (2007). Internet Banking Payment Protocol with Fraud Prevention. In: Proceedings of the 22th International Symposium on Computer and Information Sciences (Ankara), Band 1, S. 1 - 6, Berlin, Heidelberg, New York: Springer Verlag.

[Deut08a] Deutsche Bundesbank (Hg). (2008). Statistiken über den Zahlungsverkehr in Deutschland 2002 – 2006: Stand: Januar 2008. <http://www.bundebank.de/download/zahlungsverkehr/zv_statistik.pdf> (Aufgerufen am 23.12.2008).

[Deut08b] Deutsche Bundesbank (Hg). (2008). Zahlungsverkehrs- und Wertpapierabwicklungsstatistiken in Deutschland 2007: Stand: Dezember 2008. <http://www.bundesbank.de/download/statistik/zahlungsverkehr/zvs_daten_2007_in_2008.pdf> (Aufgerufen am 04.04.2009).

[Deza09] Deza, M., und Deza, E. (2009). Encyclopedia of Distances. Berlin, Heidelberg: Springer Verlag.

[Dham05] Dhamija, R., und Tygar, J. (2005). The Battle against Phishing: Dynamic Security Skins. In: Proceedings of the Symposium on Usable Privacy and Security (Pittsburgh), Band 93, S. 77 - 88, New York: ACM.

[Dham06] Dhamija, R., Tygar, J., und Hearst, M. (2006). Why Phishing Works. In: Proceedings of the SIGCHI Conference on Human Factors in computing systems (Montreal), Band 1, S. 581 - 590, New York: ACM.

[Dorf91] Dorffner, G. (1991). Konnektionismus: Von neuronalen Netzen zu einer „natürlichen“ KI. Stuttgart: B.G. Teubner.

[Down06] Downs, J., Holbrook, M., und Cranor, L. (2006). Decision Strategies and Susceptibility to Phishing. In: Proceedings of the 2nd Symposium on Usable Privacy and Security (Pittsburgh), Band 149, S. 79 - 90, New York: ACM.

[Ecke02] Eckey, H., Kosfeld, R., und Rengers, M. (2002). Multivariate Statistik: Grundlagen – Methoden – Beispiele. 1. Auflage. Wiesbaden: Gabler Verlag.

[Edge07] Edge, K., Raines, R., Bennington, R., und Reuter, C. (2007). The Use of Attack and Protection Trees to Analyze Security for an Online Banking System. In: Proceedings of the 40th Hawaii International Conference on System Sciences (Waikoloa), Band 1, S. 144 – 151, Washington: IEEE Computer Society.

[Eise04] Eiselt, P. (2004). Simulation und Modellierung. 1. Auflage. Norderstedt: GRIN Verlag.

[Erb90] Erb, W. (1990). Anwendungsmöglichkeiten der linearen Diskriminanzanalyse in Geographie und Regionalwissenschaft. In: Schriften des Zentrums für internationale Entwicklungsforschung der Justus Liebig Universität Gießen, Band 29, Hamburg: Weltarchiv.

[Even08] Event Processing Blog. (2008). Sibos 2008 - the event processing angle. Eintrag von: Giles Nelson, 19.08.2008, 11:33 Uhr. <http://apama.typead.com/my_weblog/2008/09/sibos-2008.html> (Aufgerufen am 01.10.2008).

[Even09] Event Processing Technical Society. (2009). Welcome to the Event Processing Technical Society. <http://www.ep-ts.com> (Aufgerufen am 25.09.2009).

[Fais06] Faison, T. (2006). Event-Based Programming: Taking Events to the Limit. New York: Springer Verlag.

[Fett07] Fette, I., Sadeh, N., und Tomasic, A. (2007). Learning to detect phishing emails. In: Proceedings of the 16th International Conference on World Wide Web (Banff), Band 1, S. 649 - 656, New York: ACM.

[Fico09a] FICO™. (2009). FICO™ Falcon Fraud Manager: Falcon Fraud Manager 6 Product Sheet. <http://www.fico.com/en/FIResourcesLibrary/Facon_Fraud_Manager_1247PS_EN.pdf> (Aufgerufen am 08.07.2009).

[Fico09b] FICO™. (2009). FICO™ Falcon ID: Falcon ID Product Sheet. <http://www.fico.com/en/FIResourcesLibrary/FICO_Falcon_ID_1722PS_EN.pdf> (Aufgerufen am 08.07.2009).

[Fidu08] Fiducia IT AG (Hg). (2008). Initi@tive D21: Eine Sonderstudie im Rahmen des (N)Onliner Atlas 2008. <http://www.old.initiatived21.de/fileadmin/files/08_NOA/Sonderstudie_Online-Banking_FINAL_72dpi.pdf> (Aufgerufen am 21.12.2008).

[Fina08] Finanzlexikon.de (Hg). (2008). Transaktionsüberwachung. <http://www.finanz-lexikon.de/transaktionsueberwachung_1324.html> (Aufgerufen am 31.12.2008).

[Fisc08] Fischer, T. (2008). Beck'sche Kurz Kommentare: Strafgesetzbuch und Nebengesetze. 55. Auflage. München: Verlag C.H. Beck.

[Fisc09] Fischer, T. (2009). Beck'sche Kurz Kommentare: Strafgesetzbuch und Nebengesetze. 56. Auflage. München: Verlag C.H. Beck.

[Frau06] Fraunhofer Institut Sichere Informationstechnologie. (2006). Was ist Phishing &Co. <http://www.sit.fraunhofer.de/fhg/Images/phishing_tcm105-98162.pdf> (Aufgerufen am 09.09.2008).

[Fu06] Fu, A., Wenyin, L., und Deng, X. (2006). Detecting Phishing Web Pages with Visual Similarity Assessment Based on Earth Mover's Distance (EMD). In: IEEE Transactions on Dependable and Secure Computing, Band 3, Nr. 4, S. 301 - 311.

[Gild09] Gilde, H. (2009). Complex Event Processing. <http://knol.google.com/k/hans-gilde/complex-event-processing/3ly41kwtb15i2/2#> (Aufgerufen am 22.02.2009).

[Görz03] Görz, G., Rollinger, C., und Schneeberger, J. (2003). Handbuch der künstlichen Intelligenz. 4. Auflage. München: Oldenbourg Wissenschaftsverlag.

[Grif08] Griffin, S., und Rackley, C. (2008). Vishing. In: Proceedings of the 5th annual Conference on Information security curriculum development (Kennesaw), Band 1, S. 33 - 35, New York: ACM.

[Grun08] Grundmann, W., und Körner-Delfs, R. (2008). Fallorientierte Bankbetriebswirtschaft: Mittels bankpraktischer Aufgabenstellungen BBWL verstehen und umsetzen. 1. Auflage. Wiesbaden: Gabler Verlag.

[Gual09] Gualtieri, M., und Rymer, J. (2009). The Forrester Wave™: Complex Event Processing (CEP) Platforms, Q3 2009. <http://cep1.assureroi.com/email/cep1/Forrester_Wave_complex_event_processing_cep_platforms.pdf> (Aufgerufen am 01.09.2009).

[Haas02] Haase, K. (2002). Java Messaging Service API tutorial. <http://java.sun.com/products/jms/tutorial/1_3_1-fcs/doc/jms_tutorialTOC.html> (Aufgerufen am 04.05.2008).

[Hamb08] Hamburger Abendblatt (Hg). (2008). Online-Banking: Sicherheit ist nicht garantiert. <http://www.abendblatt.de/daten/2008/12/21/995289.html> (Aufgerufen am 23.12.2008).

[Haro04] Harold, E., und Scott, W. (2004). XML in a Nutshell: A Desktop Quick Reference. 3. Auflage. Sebastopol: O'Reilly Media.

[Haub04] Haubner, K. (2004). FinTS V4.0 Kompendium Financial Transaction Services: Ein Einstieg in die neue Welt des Online-Banking. <http://www.hbci-zka.de/dokumente/diverse/fints40_kompendium.pdf> (Aufgerufen am 02.12.2009).

[Haun98] Haun, M. (1998). Simulation Neuronaler Netze: Eine praxisorientierte Einführung. Renningen-Malmsheim: expert Verlag.

[Haye06] Hayes-Roth, F. (2006). Model-Based Communication Networks and VIRT: Orders of Magnitude Better for Information Superiority. In: Proceedings of the Military Communications Conference (Washington), Band 1, S. 1 - 7, Piscataway: IEEE Press.

[Haye07] Hayes-Roth, F. (2007). Valued Information at the right time (VIRT): Why less volume is more volume in hastily formed networks. <http://www.nps.edu/cebrowski/docs/virtforhfns.pdf> (Aufgerufen am 31.12.2007).

[Hebb49] Hebb, D. (1949). The Organization of Behavior: A Neuropsychological Theory. New York: Wiley.

[Heck95] Heckerman, D., Breese, J., und Rommelse, K. (1995). Decision-theoretic troubleshooting. In: Communications of the ACM, Band 38, Nr. 3, S. 49 - 56.

[Henz04] Henze, N., und Last, G. (2004). Mathematik für Wirtschaftsingenieure und für naturwissenschaftlich-technische Studiengänge. Band 2. Wiesbaden: Vieweg Verlag.

[Herz08] Herzberg, A., und Jbara, A. (2008). Security and Identification Indicators for Browsers against Spoofing and Phishing Attacks. In: ACM Transactions on Internet Technology, Band 8, Nr. 4, Artikel Nr. 16.

[Heut07] Heutschi, R. (2007). Serviceorientierte Architektur: Architekturprinzipen und Umsetzung in die Praxis. Berlin, Heidelberg: Springer Verlag.

[Howa06] Howard, P. (2006). Event Processing: The market for event processing is growing and at some point, it will explode. <http://www.bloor-research.com/research/research_report/802/event_processing.html> (Aufgerufen am 22.12.2007).

[Hu03] Hu, Q., Liang, Y., und Fang, K. (2003). The Matrix Expression, Topological Index and Atomic Attribute of Molecular Topological Structure. In: Journal of Data Science, Band 1, Nr. 4, S. 361 - 389.

[Hu09] Hu, X., Zhao, G., und Xu, G. (2009). Security Scheme for Online Banking Based on Secret Key Encryption. In: Proceedings of the 2nd Workshop on Knowledge Discovery and Data Mining (Moskau), Band 1, S. 636 - 639, Washington: IEEE Computer Society.

[Ibm04] IBM (Hg). (2004). Developer Guide: CommonBaseEvent description. <http://www.ibm.com/developerworks/autonomic/books/fpy0mst.htm#Header_92> (Aufgerufen am 02.05.2008).

[Idth08] ID Theft Protect. (2008). Identity Fraud Statistics. <http://www.id-protect.co.uk/fraud_statistics.php> (Aufgerufen am 06.09.2008).

[Info08] Inform – Institut für Operation Research und Management GmbH. (2008). Maßnahmen gegen Betrug im Onlinebanking. <http://www.onlinebetrug.de> (Aufgerufen am 09.09.2008).

[Inte07] Intelligent Wave Inc. (Hg). (2007). Intelligent Wave Inc. and Future University Hakodate Announce Validity of Credit Card Fraud Detection Score System – ACE Plus Score System Joint Research. <http://www.iwi.co.jp/en/pdf/pdf_070423_1e.pdf> (Aufgerufen am 28.06.2008).

[Jain00] Jain, A., Duin, R., und Mao, J. (2000). Statistical Pattern Recognition: A Review. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, Band 22, Nr. 1, S. 4 - 37.

[Jako08] Jakobsson, M., Finn, P., und Johnson, N. (2008). Why and How to Perform Fraud Experiments. In: IEEE Security und Privacy, Band 6, Nr. 2, S. 66 - 68.

[Jano06] Janowicz, K. (2006). Sicherheit im Internet. Köln: O'Reilly Germany.

[Jens96] Jensen, F. (1996). An Introduction to Bayesian Networks. New York: Springer Verlag.

[Kara01] Karagiannis, D., und Telesko, R. (2001). Wissensmanagement: Konzepte der Künstlichen Intelligenz und des Softcomputing. München, Wien, Oldenbourg: Oldenbourg Wissenschaftsverlag.

[Kech06] Kecher, C. (2006). UML 2.0: Das umfassende Handbuch. 2. Auflage. Bonn: Galileo Computing.

[Klee72] Klee, V., und Minty, G. (1972). How Good is the Simplex Algorithm?. In: O. Shisha (ed.), Inequalities III, S. 159 - 175.

[Köni99] König, W., Rommelfanger, H., Ohse, D., Hofmann, M., Schäfer, K., Kuhnle, H., und Pfeifer, A. (1999). Taschenbuch der Wirtschaftsinformatik und Wirtschaftsmathematik. 1. Auflage. Frankfurt am Main: Verlag Harri Deutsch.

[Koho82] Kohonen, T. (1982). Self-Organized Formation of Topologically Correct Feature Maps. In: Biological Cybernetics, Band 43, Nr. 1, S. 59 - 69.

[Kou04] Kou, Y., Lu, C., Sirwongwattana, S., und Huang, Y. (2004). Survey of Fraud Detection Techniques. In: Proceedings of the International Conference on Networking, Sensing and Control (Taipei), Band 2, S. 749 - 754, Washington: IEEE Computer Society.

[Kraf07] Kraft, P. (2007). Anti Hackerz Book: Viren-, Trojaner- & Root-Kit - und die wirklich wirksamen Gegenspieler. 2. Auflage. Poing: Franzis Verlag.

[Krah98] Krahl, D., Windheuser, U., und Zick, F. (1998). Data Mining: Einsatz in der Praxis. 1. Auflage. Bonn, Reading, Menlo Park u.a.: Addison-Wesley.

[Krus93] Kruse, R., Gebhardt, J., und Klawonn, F. (1993). Fuzzy-Systeme. Stuttgart: Teubner Verlag.

[Kuro02] Kurose, J., und Ross, K. (2002). Computernetze: Ein Top-Down-Ansatz mit Schwerpunkt Internet. München: Pearson Studium.

[Lämm08] Lämmel, U., und Cleve, J. (2008). Künstliche Intelligenz. 3. Auflage. München: Carl Hanser Verlag.

[Lang03] Lange, C. (2003). Neuronale Netze in der wirtschaftswissenschaftlichen Prognose und Modellgenerierung: Eine theoretische und empirische Betrachtung mit Programmier-Beispielen. Heidelberg: Physica Verlag.

[Leve66] Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady, Band 10, Nr. 8, S. 707 - 710.

[Lini05] Lininger, R., und Vines, R. (2005). Phishing: Cutting the Identity Theft Line. Indianapolis: Wiley Publishing.

[Liu98] Liu, H., und Motoda, H. (1998). Feature Extraction, Construction and Selection: A Data Mining Perspective. Norwell: Kluwer Academic Publishers.

[Lobe04] Loberg, K. (2004). Identity Theft: How to Protect Your Name, Your Credit and Your Vital Information-And What to Do When Someone Hijacks. Los Angeles: Silver Lake Publishing.

[Logo08] Logofatu, D. (2008). Grundlegende Algorithmen mit Java. Wiesbaden: Vieweg Verlag.

[Luck02] Luckham, D. (2002). The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Boston, San Francisco, New York u.a.: Addison-Wesley Verlag.

[Luck04a] Luckham, D. (2004). As Companies try Business Activity Monitoring, and find it useful, they are going to begin conceiving new ways to use it. <http://complexevents.com/?p=10> (Aufgerufen am 23.05.2008).

[Luck04b] Luckham, D. (2004). Three Steps towards Information Technology Insight. <http://complexevents.com/?p=22> (Aufgerufen am 23.05.2008).

[Luck04c] Luckham, D. (2004). The Beginnings of IT Insight: Business Activity Monitoring. <http://complexevents.com/?p=19> (Aufgerufen am 23.05.2008).

[Luck04d] Luckham, D. (2004). BAM Providers as Online-Banking Fraud Preventers. <http://www.ebizq.net/topics/cep/features/4891.html> (Aufgerufen am 07.11.2008).

[Luck04e] Luckham, D. (2004). Avoiding Disasters Waiting to Happen. <http://www. ebizq.net/topics/eii/features/4257.html> (Aufgerufen am 07.11.2008).

[Luck05] Luckham, D. (2005). “Complex” or “Simple” Event Processing. <http://complexevents.com/?p=20> (Aufgerufen am 16.05.2008).

[Luck07] Luckham, D. (2007). The Future Event Driven World: Global Epidemic Warning Systems. <http://complexevents.com/?p=299> (Aufgerufen am 23.05.2008).

[Luck08] Luckham, D., und Schulte, R. (2008). Event Processing Glossary. <http://complexevents.com/?p=195> (Aufgerufen am 09.05.2009).

[Luck09] Luckham, D. (2009). Is There a Commercial Need for a Quantum Leap in CEP Technology?. <http://complexevents.com/wp-content/uploads/2009/01/ is-new-cep-technology-needed.pdf> (Aufgerufen am 16.02.2009).

[Lund02] Lundin, E., Kvarnström, H., und Jonsson, E. (2002). A Synthetic Fraud Data Generation Methodology. In: Proceedings of the 4th International Conference on Information and Communications Security, Lecture Notes In Computer Science (Singapur), Band 2513, S. 265 - 277, London: Springer Verlag.

[Lund03] Lundin, E., Kvarnström, H., und Jonsson, E. (2003). Synthesizing Test Data for Fraud Detection Systems. In: Proceedings of the 19th Annual Computer Security Applications Conference (Las Vegas), Band 1, S. 384 - 394, Washington: IEEE Computer Society.

[Lund06] Lundberg, A. (2006). Leverage Complex Event Processing to Improve Operational Performance. In: Business Intelligence Journal, Band 11, Nr. 1, S. 55 - 65.

[Lust02] Lusti, M. (2002). Data Warehousing und Data Mining: Eine Einführung in entscheidungsunterstützende Systeme. 2. Auflage. Berlin, Heidelberg, New York: Springer Verlag.

[Lync05] Lynch, J. (2005). Identity Theft in Cyberspace: Crime Control Methods and their Effectiveness in Combating Phishing Attacks. In: Berkeley Technology Law Journal. Band 20, Nr. 1, S. 256 - 300.

[Maur08] Maurice, F., Koch, M., Elrich, D., Schwabe, R., und Immler, C. (2008). Das große Franzis Computer Handbuch: Anschaffung, Anwendung, tägliche Praxis. Aschaffenburg: Franzis Verlag.

[May04] May, J. (2004). Johnny May's Guide to Preventing Identity Theft: How Criminals Steal Your Personal Information, how to Prevent It, and what to Do If You Become a Victim. Pontiac: Security Resources Unlimited.

[Mccu43] McCulloch, W., und Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activitiy. In: Bulletin of Mathematical Biology, Band 5, Nr. 4, S. 115 - 133.

[Mcdo05] McDonald, A. (2005). SpamAssassin: Leitfaden zu Konfiguration, Integration und Einsatz. München: Addison-Wesley.

[Medv08] Medved, E., Kirda, E., und Kruegel, C. (2008). Visual-Similarity-Based Phishing Detection. In: Proceedings of the 4th International Conference on Security and privacy in communication networks (Istanbul), Band 1, Artikel Nr. 22, New York: ACM.

[Miln93] Milner, P. (1993). The Mind and Donald O. Hebb. In: Scientific American, Band 268, Nr. 1, S. 124 - 129.

[Mish05] Mishra, D., Yadav, A., Ray, S., und Kalra, P. K. (2005). Levenberg-Marquardt Learning Algorithm for Integrate-and-Fire Neuron Model. In: Neural Information Processing - Letters and Reviews, Band 9, Nr. 2, S. 41 - 51.

[Moor07] Moore, T., und Clayton, R. (2007). Examining the Impact of Website Take-down on Phishing. In: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit (Pittsburgh), Band 1, S. 1 - 13, New York: ACM.

[Mühl06] Mühl, G., Fiege, L., und Pietzuch, P. (2006). Distributed Event-Based Systems. Berlin, Heidelberg: Springer Verlag.

[Müll08] Müller, K. (2008). IT-Sicherheit mit System: Sicherheitspyramide-Sicherheits-, Kontinuitäts- und Risikomanagement, Normen, Practices, SOA und Softwareentwicklung. 3. Auflage. Wiesbaden: Vieweg Verlag.

[Müll09] Müller, D. (2009). Unternehmen werden zum größten Sicherheitsrisiko ihrer eigenen Kunden. <http://www.silicon.de/sicherheit/management/0,39039020,3920202000/unternemen+werden+zum+groessten+sicherheitsrisko+ihrer+eigenen+kunden.htm> (Aufgerufen am 12.02.2009).

[Mukh08] Mukhanov, R. (2008). Using Bayesian Belief Networks for Credit Card Fraud Detection. In: Proceedings of the International Conference on Artificial Intelligence and Applications (Innsbruck), Band 1, S. 120 - 164, Calgary: ACTA Press.

[Muna08] Munakata, T. (2008). Fundamentals of the New Artificial Intelligence: Neural, Evolutionary, Fuzzy and More. 2. Auflage. London: Springer Verlag.

[Murc04] Murcia-Soler, M., Perez-Gimenez, F., Garcia-March, F., Salabert-Salvador, M., Diaz-Villanueva, W., Castro-Bleda, M., und Villanueva-Pareja, A. (2004). Artificial Neural Networks and Linear Discriminant Analysis: A Valuable Combination in the Selection of New Antibacterial Compounds. In: Journal of Chemical Information and Computer Sciences, Band 44, Nr. 3, S. 1.031 - 1.041.

[Nech06] Nechushtai, G. (2006). Complex Event Processing (CEP). <http://domino.wason.ibm.com/comm/research.nsf/pages/r.datamgmt.innovation.cep.html> (Aufgerufen am 22.05.2008).

[Nguy05] Nguyen, T., Schiefer, J., und Tjoa, A. (2005). Sense & response service architecture (SARESA): an approach towards a real-time business intelligence solution and its use for a fraud detection application. In: Proceedings of the 8th ACM International workshop on Data warehousing and OLAP (Bremen), Band 1, S. 77 - 86, New York: ACM.

[Nguy07] Nguyen, T., Schiefer, J., und Tjoa, A. (2007). ZELESSA: an enabler for real-time sensing, analysing and acting on continuous event streams. In: International Journal of Business Intelligence and Data Mining, Band 2, Nr. 1, S. 105 - 141.

[Nie05] Nie, J., und Ma, F. (2005). Network Security Risks in Online Banking. In: Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (Mawii), Band 2, S. 1.229 - 1.234, Washington: IEEE Computer Society.

[Nork09] Norkom Technologies. (2009). Online Fraud. <http://www.norkom.com/solutions/online-fraud.html> (Aufgerufen am 09.07.2009).

[Ohay06] Ohaya, C. (2006). Managing Phishing Threats in an Organization. In: Proceedings of the 3rd annual Conference on Information security curriculum development (Kennesaw), Band 1, S. 159 - 161, New York: ACM.

[Pal04] Pal, S., und Mitra, P. (2004). Pattern Recognition Algorithms for Data Mining. Boca Raton, London, New York u.a.: Chapman & Hall/CRC.

[Pare00] Parekh, R., Yang, J., und Honavar, V. (2000). Constructive neural-network learning algorithms for pattern classification. In: Neural Networks, IEEE Transactions on, Band 11, Nr. 2, S. 436 - 451.

[Pate07] Patel, D., und Luo, X. (2007). Take a Close Look at Phishing. In: Proceedings of the 4th annual Conference on Information security curriculum development (Kennesaw), Band 1, Artikel Nr. 32, New York: ACM.

[Pavl00] Pavlov, Y. (2000). Random Forests. Utrecht: Brill Academic Pub.

[Pete05] Petersohn, H. (2005). Data Mining: Verfahren, Prozesse, Anwendungsarchitektur. München: Oldenbourg Wissenschaftsverlag.

[Phua04] Phua, C., Alahakoon, D., und Lee, V. (2004). Minority report in fraud detection: classification of skewed data. In: Special issue on learning from imbalanced datasets, Band 6, Nr. 1, S. 50 - 56.

[Phua05] Phua, C., Lee, V., Smith, K., und Gayler, R. (2005). A Comprehensive Survey of Data Mining-based Fraud Detection Research. In: Artificial Intelligence Review, Band 1, Nr. 1, S. 1 - 14.

[Prät07] Prätsch, J., Schikorra, U., und Ludwig, E. (2007). Finanzmanagement. 3. Auflage. Berlin: Springer Verlag.

[Prog09] Progress Software Corporation. (2009). APAMA. <http://web.progress.com/apama/index.html> (Aufgerufen am 05.08.2009).

[Puen05] Puente, F., Sandoval, J., Hernandez, P., und Molina, C. (2005). Improving Online Banking Security with Hardware Devices. In: Proceedings of the 39th Annual 2005 International Carnahan Conference on Security Technology (Las Palmas), Band 1, S. 174 - 177, Madrid: IEEE Press.

[Quin83] Quinlan, J. (1983). Learning efficient classification procedures and their application to chess end-games. In: Machine Learning: An Artificial Intelligence Approach, Kapitel 15. San Mateo: Morgan Kaufman.

[Quin93] Quinlan, J. (1993). C4.5: Programs for Machine Learning. San Mateo: Morgan Kaufman.

[Rana06] Ranadive, V. (2006). The Power to Predict: How Real-Time Businesses Anticipate Customers Needs, Create Opportunities and Beat the Competition. New York, Chicago, San Francisco u.a.: McGraw-Hill.

[Raud01] Raudys, S. (2001). Statistical and Neural Classifiers: An Integrated Approach to Design. London, Berlin, Heidelberg: Springer Verlag.

[Reol07] Reolon, R. (2007). Analytische Betrugserkennung im Bankenumfeld: Evaluation von Relationalem Data Mining für die Suche nach Betrugsmustern. Saarbrücken: Verlag Dr. Müller.

[Rey08] Rey, G., und Wender, K. (2008). Neuronale Netze: Eine Einführung in die Grundlagen, Anwendungen und Datenauswertung. 1. Auflage. Bern: Verlag Hans Huber.

[Rich03] Richter, F. (2003). Kombination Künstlicher Neuronaler Netze: Zur Prognose von Wechselkursen. Wiesbaden: Deutscher Universitäts-Verlag.

[Roja96] Rojas, R. (1996). Neural Networks: A Systematic Introduction. Berlin, Heidelberg, New York u.a: Springer Verlag.

[Rosi07] Rosiello, A., Kirda, E., Kruegel, C., und Ferrandi, F. (2007). A layout-similarity-based approach for detecting phishing pages. In: Proceedings of the 3rd International Conference on Security and Privacy in Communications Networks (Nizza), Band 1, S. 454 - 463, Washington: IEEE Computer Society.

[Rozs07] Rozsnyai, S., Schiefer, J., und Schatten, A. (2007). Solution for Detecting and Preventing Fraud in Real Time. In: Proceedings of the 2nd Conference on Digital Information Management (Lyon), Band 1, S. 152 - 158, Washington: IEEE Computer Society.

[Rume86] Rumelhart, D., Hinton, G., und Williams, R. (1986). Learning internal representations by error propagation. In: Parallel distributed processing: explorations in the microstructure of cognition, Band 1, S. 318 - 362.

[Rutk08] Rutkowski, L. (2008). Computational Intelligence: Methods and Techniques. Heidelberg: Springer Verlag.

[Sart06] Sartorius, C., Albers, S., Klapper, D., Konradt, U., Walter, A., und Wolf, J. (2006). Methodik der empirischen Forschung. 1. Auflage. Wiesbaden: Deutscher Universitätsverlag.

[Schi93] Schiffmann, W., Joost, M., und Werner, R. (1993). Comparison of Optimized Backpropagation Algorithms. In: Proceedings of the 1st European Symposium on Artificial Neural Networks (Brüssel), Band 1, S. 97 - 104, Brüssel: D-Facto public.

[Schn06] Schneider, F., Dreer, E., und Riegler, W. (2006). Geldwäsche: Formen, Akteure, Größenordnung – und warum die Politik machtlos ist. 1. Auflage. Wiesbaden: Gabler Verlag.

[Schü08] Schütz, A. (2008). Keylogger: ein Riesengeschäft für Cyberkriminelle. <http://www.silicon.de/sicherheit/antivirus/0,39039019,39200536,00/keylogger+ein+riesengeschaeft+fuer+cyberkriminelle.htm> (Aufgerufen am 23.12.2008).

[Schu06] Schulte, T. (2006). Kokain: Bankmitarbeiter plündert Konten. <http://www.anwaltzentrale.de/rechtsanwalt_fachartikel/fachartikel_detail.php?id=212&Fachgebiet_id=17> (Aufgerufen am 04.09.2008).

[Shan63] Shannon, C., und Weaver, W. (1963). The Mathematical Theory of Communication. Urbana, Chicago: University of Illinois Press.

[Slew04] Slewe, T., und Hoogenboom, M. (2004). Who will rob you on the digital highway. In: Communications of the ACM, Band 47, Nr. 5, S. 56 - 60.

[Spei07] Speichert, H. (2007). Praxis des IT-Rechts: Praktische Rechtsfragen der IT-Sicherheit und Internetnutzung. 2. Auflage. Wiesbaden: Vieweg Verlag.

[Sqls08] SQL Server Developer Center MSDN. (Hg). (2008). Feature Selection in Data Mining: SQL Server 2008 Books Online. <http://msdn.microsoft.com/en-us/library/ms175382(SQL.100).aspx> (Aufgerufen am 26.06.2008).

[Sriv08] Srivastava, A., Kundu, A., Sural, S., und Majumdar, A. (2008). Credit Card Fraud Detection Using Hidden Markov Model. In: IEEE Transactions on Dependable and Secure Computing, Band 5, Nr. 1, S. 37 - 48.

[Stan03] Stankovic, R., und Falkowski, B. (2003). The Haar wavelet transform: its status and achievements. In: Computers and Electrical Engineering, Band 29, Nr. 1, S. 25 - 44.

[Stat08] Statistika.org. (2008). Bei welchen Geldinstituten haben Sie persönlich ein Giro- bzw. Gehaltskonto?. <http://de.statista.org/statistik/diagramm/studie/87898/umfrage/geldinstitute-bei-denen-gehalts-bzw.-girokonto-gefuehrt-wird> (Aufgerufen am 10.11.2008).

[Stev88] Stevens, C. (1988). Die Nervenzelle. In: Gehirn und Nervensystem. 9. Auflage. Heidelberg: Spektrum der Wissenschaft Verlagsgesellschaft.

[Stre08] StreamBase Systems, Inc. (Hg). (2008). Application Design Concepts, StreamBase Studio, Introduction to StreamSQL. Stream SQL Workshop Unterlagen.

[Stre09] StreamBase Systems, Inc. (2009). StreamBase Solutions Home. <http://www.streambase.com/solutions-home.htm> (Aufgerufen am 05.08.2009).

[Sull04] Sullivan, B. (2004). Your Evil Twin: Behind the Identity Theft Epidemic. Hoboken: John Wiley & Sons.

[Sunt08] Suntinger, M., Schiefer, J., Obweger, H., und Gröller, M. E. (2008). The Event Tunnel: Interactive Visualization of Complex Event Streams for Business Process Pattern Analysis. In: Proceedings of the Visualization Symposium (Kyoto), Band 1, S. 111 - 118, Washington: IEEE Computer Society.

[Tark05] Tarkoma, S., und Raatikainen, K. (2005). State of the Art Review of Distributed Event Systems. <http://www.minema.di.fc.ul.pt/reports/Minema EventsReport_final.pdf> (Aufgerufen am 18.05.2008).

[Tecc06] Tecchannel (Hg). (2006). Das Netz der Phisher: Wie Online-Betrüger arbeiten. <http://www.tecchannel.de/sicherheit/spam/447964/das_netz_der_phisher_wie_online_betrueger_arbeiten> (Aufgerufen am 23.12.2008).

[Thec07a] The complex event processing blog. (2007). Clouds (Partially Order Sets) - Streams (Linearly Ordered Sets). Eintrag von: Tim Bass, 28.07.2007, 05:07 Uhr. <http://thecepblog.com> (Aufgerufen am 17.05.2008).

[Thec07b] The complex event processing blog. (2007). CEP Event Sources. Eintrag von: Tim Bass, 23.08.2007, 13:42 Uhr. <http://thecepblog.com> (Aufgerufen am 02.05.2008).

[Thec08] The complex event processing blog. (2008). The 9 Features that CEP *engine*s Should Support. Eintrag von: Tom Puzak, 18.04.2008, 17:47 Uhr. <http://thecepblog.com> (Aufgerufen am 19.05.2008).

[Thom01] Thompson, R. (2001). Das Gehirn: Von der Nervenzelle zur Verhaltenssteuerung. Heidelberg, Berlin: Spektrum Akademischer Verlag.

[Tibc09] Tibco Software Inc. (2009). TIBCO BusinessEvents. <http://www.tibco.com/software/complex-event-processing/businessevents/businessevents.jsp> (Aufgerufen am 05.08.2009).

[Trae02] Traeger, D., und Volk, A. (2002). LAN: Praxis lokaler Netze. 4. Auflage. Stuttgart, Leipzig, Wiesbaden: Teubner Verlag.

[Tsim06] Tsimelzon, M. (2006). Complex Event Processing: Ten Design Patterns. <http://complexevents.com/?p=188> (Aufgerufen am 23.05.2008).

[Tyna05] Tynan, D. (2005). Computer Privacy Annoyances: How to Avoid the Most Annoying Invasions of Your Personal and Online Privacy. Cambridge: O'Reilly.

[Tzaf97] Tzafestas, S. (1997). Knowledge Based Systems: Advanced Concepts Techniques and Applications. Singapur: World Scientific Publishing Company.

[Univ08] Universität Tübingen (Hg). (2008). Simulation of Neural Networks. In: University of Tübingen. <http://www.ra.cs.uni-tuebingen.de/software/snns> (Aufgerufen am 12.04.2008).

[Vacc03] Vacca, J. (2003). Identity Theft. Upper Saddle River: Prentice Hall PTR.

[Venk07] Venkataram, P., Sathish-Babu, B., Naveen, M., und Samyama-Gungal, G. (2007). A Method of Fraud & Intrusion Detection for E-payment Systems in Mobile e-Commerce. In: Proceedings of the 26th International Conference on Performance, Computing, and Communications (New Orleans), Band 1, S. 395 - 401, Washington: IEEE Computer Society.

[Vers02] Versteegen, G. (2002). Software-Management. Berlin, Heidelberg: Springer Verlag.

[Vikr04] Vikram, A., Chennuru, S., Rao, H., und Upadhyaya, S. (2004). A solution architecture for financial institutions to handle illegal activities: a neural networks approach. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences (Big Island), Band 1, S. 181 - 190, Washington: IEEE Computer Society.

[Wagn07] Wagner, N. (2007). Identity Fraud Profiles: Victims and Offenders. In: Proceedings of the 8th World Congress on the Management of eBusiness (Toronto), Band 1, S. 22 - 30, Washington: IEEE Computer Society.

[Wang05] Wang, L. (2005). Support Vector Machines: Theory and Applications. Heidelberg: Springer Verlag.

[Weny05] Wenyin, L., Huang, G., Xiaoyue, L., Min, Z., und Deng, X. (2005). Detection of Phishing Webpages based on Visual Similarity. In: Proceedings of the 14th International Conference on World Wide Web (Chiba), Band 1, S. 1.060 - 1.061, New York: ACM.

[Weny06] Wenyin, L., Deng, X., Huang, G., und Fu, A. (2006). An Antiphishing Strategy Based on Visual Similarity Assessment. In: Internet Computing, Band 10, Nr. 2, S. 58 - 65.

[Widd07] Widder, A., Ammon, R.v., Schaeffer, P., und Wolff, C. (2007). Identification of Suspicious, Unknown Event Patterns in an Event Cloud. In: Proceedings of the 2007 inaugural International Conference on Distributed event-based systems (Toronto), Band 233, S. 164 - 170, New York: ACM.

[Widd08] Widder, A., Ammon, R.v., Schaeffer, P., und Wolff, C. (2008). Combining Discriminant Analysis and Neural Networks for Fraud Detection on the Base of Complex Event Processing. In: Proceedings of the 2nd International Conference on Distributed Event-Based Systems (Rom), Fast Abstract Paper, New York: ACM.

[Widd09] Widder, A., Ammon, R.v., Hagemann, G., und Schönfeld, D. (2009). An Approach for Automatic Fraud Detection in the Insurance Domain. In: Proceedings of the 2009 AAAI Spring Symposium (Menlo Park), AAAI Technical Report, S. 98 - 100, Menlo Park: AAAI Press.

[Wied03] Wiedmann, K., und Buckler, F. (2003). Neuronale Netze im Marketing-Management: Praxisorientierte Einführung in modernes Data-mining. 2. Auflage. Wiesbaden: Gabler Verlag.

[Xu05] Xu, J., und Sung, A. (2005). Adaptive fraud detection based on user behavior mining. In: Proceedings of the 16th International Conference on Modeling and Simulation (Cancun), Band 1, S. 68 - 73, Calgary: ACTA Press.

[Xu06] Xu, J., Sung, A., und Liu, Q. (2006). Tree Based Behavior Monitoring for Adaptive Fraud Detection. In: Proceedings of the 18th International Conference on Pattern Recognition (Hong Kong), Band 1, S. 1.208 - 1.211, Washington: IEEE Computer Society.

[Yu08] Yu, W., Nargundkar, S., und Tiruthani, N. (2008). A Phishing Vulnerability Analysis of Web Based Systems. In: Proceedings of the Symposium on Computers and Communications (Marrakech), Band 1, S. 326 - 331, Washington: IEEE Computer Society.

[Zent07] Zentraler Kreditausschuss (2007). Anlage 3 der Schnittstellenspezifikation für die Datenfernübertragung zwischen Kunde und Kreditinstitut gemäß DFÜ Abkommen. <http://www.ebics-zka.de/dokument/pdf/Anlage%203-Spezifikation%20der%20Datenformate%2020Version%202.2%20Endfassung%20vom%2029.10.2007.pdf> (Aufgerufen am 12.04.2008).

[Zhan09] Zhang, Q. (2009). Study on Fraud Risk Prevention of Online Banks. In: Proceedings of the International Conference on Networks Security, Wireless Communications and Trusted Computing (Wuhan), Band 1, S. 181 - 184, Washington: IEEE Computer Society.

[Zöll07] Zöller-Greer, P. (2007). Künstliche Intelligenz – Grundlagen und Anwendungen. Wächtersbach: Composia Verlag.

ANHÄNGE

Anhang 1: Interviews mit Experten für Betrugserkennung im Bankbereich

Im Anhang 1 befinden sich die Interviews mit den Experten aus dem Bereich Betrugsmanagement verschiedener Kreditinstitute und Bankverbänden. Vor der Durchführung des jeweiligen Interviews wurde den beteiligten Personen eine kurze Zusammenfassung des Ziels der Arbeit, eine Beschreibung des Auswertungsalgorithmus sowie vorhandene Publikationen zur Verfügung gestellt. Die Interviewfragen sind ähnlich, wobei sie für den jeweiligen Empfänger im Vorfeld angepasst und im Zeitverlauf weiterentwickelt wurden. Einige Beteiligte wurden im Laufe der Arbeit auch zweimal befragt. Die Namen der beteiligten Personen und Kreditinstitute dürfen aus Gründen der Vertraulichkeit nicht genannt werden. Unter dieser Bedingung gaben die befragten Experten Auskünfte, soweit es Ihnen möglich war. Bei der nachfolgenden Darstellung wird bezüglich der zeitlichen Reihenfolge mit dem ältesten Interview begonnen.

1. Interviewpartner (Tätigkeitsbereich: Electronic Banking in einem Kreditinstitut) am 18.03.2008 und 09.04.2009:

- Würde Ihr Institut evtl. Echtdaten zur Validierung des Algorithmus bereitstellen?

  Antwort: Wir haben nicht so viele Betrugsfälle im Online-Banking, dass Ihnen das weiterhelfen würde.

- Sind generell Auslandsüberweisungen gefährdet oder überweisen Phisher auch ins Inland? Wie ist hierbei die Verteilung?

  Antwort: Auslandsüberweisungen sind verdächtig, ja.

- Wie läuft der Prozess der Online-Überweisung bei Ihrem Kreditinstitut intern ab?

  Antwort: Nach dem Abschluss der Online-Banking-Session werden die Transaktionen gespeichert und unmittelbar danach vom Rechenzentrum verrechnet.

- Wie viele Online-Transaktionen hat eine Geschäftsbank oder Sparkasse täglich zu analysieren?

  Antwort: Das ist sehr verschieden. Bei uns sind es so ca. 3.000 – 5.000 Transaktionen am Tag, wobei dies von Tag zu Tag schwankt. Das entspricht einer Gesamtmenge von ca. 40.000 Girokonten, von denen ca. 18.000 online geführt wer-

den können. Dies führt zu einem Marktanteil von ca. 15% und einer Bilanzsumme von knapp 4 Mrd. Euro.

- An welchen Attributen und Eigenschaften einer Online-Transaktion könnte ein Betrugsfall evtl. identifiziert werden?

  Antwort: Hierbei ist der Transaktionsbetrag, der Kontostand und das Empfängerkonto wichtig. Wobei hier erwähnt werden muss, dass eine einzige Transaktion des Kunden nicht ausreichend ist um einen Betrugsfall eindeutig zu identifizieren. Ein Betrüger bewegt sich nämlich im Web nach einem erfolgreichen Einloggen oftmals genau so wie der eigentliche Kunde. Sie müssen die aktuelle Transaktion mit historischen Transaktionen des Kunden vergleichen. An seinem „normalen" Transaktionsverhalten kann dann identifiziert werden, ob es sich bei einer aktuellen Transaktion um einen Betrug handelt. Ein Beispiel ist, wenn ein Kunde oft Transaktionen an seiner Verfügbarkeitsgrenze durchführt, ist dies bei diesem Kunden weniger verdächtig als bei einem Kunden, der solch hohe Transaktionen noch nie oder nur selten durchgeführt hat.

- Können Sie mir bekannte Betrugsmuster im Bereich „Identitätsdiebstahl" beim Online-Banking nennen?

  Antwort: Bei Auslandsüberweisungen sollte genauer hingeschaut werden und auch bei den eben erwähnten Transaktionsbeträgen, die nah an die Verfügbarkeitsgrenze gehen. Ansonsten kann auch der IP-Adressbereich des Quellrechners analysiert werden. Wenn dieser im Ausland steht, ist dies verdächtig, da die Kunden in der Regel immer den gleichen IP-Adressbereich im Inland verwenden. Wichtig ist, dass die Anti-Virussoftware aktuell ist und allgemein der Kunde einfach ein waches Auge hat.

- Woran können Nicht-Betrugstransaktionen beim Online-Banking sicher erkannt werden?

  Antwort: Wenn die Überweisung zu einem vertrauenswürdigen Empfänger geht, wie z.B. an eine Behörde oder wenn der Kunde schon oft eine Überweisung auf dieses Zielkonto durchgeführt hat.

- Wie schnell muss eine Betrugstransaktion beim Online-Banking identifiziert werden?

  Antwort: Je schneller desto besser um reagieren zu können, bevor die Überweisung durchgeführt ist.

- Setzt ihre Bank Complex Event Processing-Technologie ein?

  Antwort: Nein, ist mir nicht bekannt.

- Wie ist ihre generelle Strategie im Bereich „Identitätsdiebstahl“ beim Online-Banking?

  Antwort: Wir haben im Bereich Online-Banking in den letzten Jahren wenig Betrugsfälle gehabt. Wir setzen hier auf eine Strategie der Betrugsprävention, nachträgliche Untersuchungen der Überweisungen führen wir hierbei nur durch, wenn sich ein betrogener Kunde meldet.

2. Interviewpartner (Tätigkeitsbereich: Business Intelligence in einem Kreditinstitut) am 24.10.2008:

- Würde Ihr Institut evtl. Echtdaten zur Validierung des Algorithmus bereitstellen?

  Antwort: Solche Daten können leider nicht an Außenstehende weitergegeben werden.

- Sind generell Auslandsüberweisungen gefährdet oder überweisen Phisher auch ins Inland? Wie ist hierbei die Verteilung?

  Antwort: Eine genaue Verteilung kann ich nicht sagen, aber die Überweisungen gehen oft ins Ausland, das ist richtig.

- Wie viele Betrugsfälle im Online-Banking hat ihr Kreditinstitut bzw. wie ist das Verhältnis von Betrugstransaktionen zu Nicht-Betrugstransaktionen?

  Antwort: Das ist von Monat zu Monat unterschiedlich. Diese Information darf ich Ihnen leider nicht geben.

- Wie läuft der Prozess der Online-Überweisung bei Ihrem Kreditinstitut intern ab?

  Antwort: Nach dem Anstoßen der Online-Transaktion erfolgen unmittelbar danach die Buchungen auf den Konten, also Gutschrift und Lastschrift.

- Wie viele historische Online-Transaktionen müssten von einem Kunden untersucht werden um ein Transaktionsprofil des Kunden zu bekommen bzw. um abschätzen zu können, wie das „normale“ Transaktionsverhalten des Kunden ist?

  Antwort: Es sollten wenigstens vier oder fünf historische Transaktionen sein, wobei diese Auswertungen dann bei Neukunden ohne historische Transaktionen in dieser Form nicht durchgeführt werden können. Der Anteil solcher Kunden ist aber eher gering.

- Wie hoch ist das Dispolimit von Girokonten bei Ihrem Kreditinstitut?

  Antwort: Mindestens das Dreifache des regelmäßigen Gehaltseingangs, höhere Beträge können durch den Wert des Wertpapierdepots abgedeckt werden.

- Wie ist die Vermögensverteilung auf den Girokonten ihrer Online-Banking-Kunden in ihrem Kreditinstitut?

  Antwort: Darüber kann ich keine Angaben machen.

- Wie ist die Verteilung des monatlichen Gehaltseingangs ihrer Online-Banking-Kunden?

  Antwort: Darüber kann ich auch keine Angaben machen.

- Wie viele Online-Transaktionen hat eine Geschäftsbank oder Sparkasse täglich zu analysieren?

  Antwort: Dies ist von Tag zu Tag unterschiedlich, da bei uns dies zentral abgewickelt wird, können das über 100.000 Transaktionen aus dem Online-Banking am Tag sein.

- Wie hoch ist die durchschnittliche Zahl von Online-Transaktionen eines Online-Banking-Kunden im Monat?

  Antwort: Das ist ebenfalls unterschiedlich. Es reicht von 1 bis 15 Transaktionen im Monat im Privatkundensegment. Mehr sind eher selten.

- An welchen Attributen und Eigenschaften einer Online-Transaktion könnte ein Betrugsfall evtl. identifiziert werden?

  Antwort: Bei der Transaktion selbst sind der Transaktionsbetrag und Empfänger wichtig. Auch der Bereich, aus dem die verwendete IP-Adresse stammt kann ein guter Hinweis sein, da Betrüger häufig vom Ausland aus tätig sind. Hierbei sollte jedoch das Verhältnis von Transaktionsbetrag zum maximal verfügbaren Betrag ausgewertet werden. Ebenso das Verhältnis von Transferbetrag zum durchschnittlichen Transferbetrag zum Zeitpunkt der Transaktion um festzustellen, ob ein Kunde oft Überweisungen durchführt, die nahe an der Verfügbarkeitsgrenze sind. Dann ist eine einzelne solche Transaktion weniger verdächtig.

- Können Sie mir bekannte Betrugsmuster im Bereich „Identitätsdiebstahl“ beim Online-Banking nennen?

  Antwort: Verdächtig sind beim Online-Banking Transaktionsbeträge nahe an der Verfügbarkeitsgrenze und Auslandsüberweisungen. Auch Menschen mit weniger Erfahrung beim Online-Banking sind tendenziell gefährdeter in diesem Bereich.

- Woran können Nicht-Betrugstransaktionen beim Online-Banking sicher erkannt werden?

  Antwort: Regelmäßige Transaktionen an den gleichen Empfänger oder an vertrauenswürdige Empfänger wie bekannte Organisationen oder Behörden.

- Bei welchem Ausgabewert eines neuronalen Netzwerks könnte eine Betrugstransaktion bzw. Nicht-Betrugstransaktion sicher erkannt werden?

  Antwort: Der Ausgabewert sollte bei wenigstens 90 bis 95% liegen, am besten wäre natürlich 100%. Bei Nicht-Betrugsfällen nicht größer als 5 bis 10% Betrugswahrscheinlichkeit. Auch hier sollte es so klein wie möglich sein, wobei false positives fast noch kritischer zu betrachten sind aufgrund des Reputationsverlusts. Aber wichtig ist natürlich beides.

- Wie schnell muss eine Betrugstransaktion beim Online-Banking identifiziert werden?

  Antwort: So schnell wie möglich um Gegenmaßnahmen treffen zu können bevor es zu spät ist.

- Welche Erkennungsgenauigkeit müsste von einem Transaktionsanalysesystem erreicht werden?

  Antwort: Mindestens sollte ein solches System 98% erreichen, aber auch hier sind die 100% natürlich am besten, wobei ich mir auch vorstellen kann, dass 100% schwer zu erreichen sind.

- Was passiert, wenn eine Online-Transaktion nicht sicher zugeordnet werden kann?

  Antwort: Sie würde manuell geprüft oder die Transaktionsbuchung sofort ausgeführt.

- Setzt ihre Bank Complex Event Processing-Technologie ein?

  Antwort: Nein, wobei ich schon über diese Technologie gelesen habe und sie als durchaus interessant einschätze. Aber ich denke, für den Einsatz einer solchen

Technologie müsste die Middleware komplett neu aufgebaut werden, was bei unserer momentanen Strategie und auch Belastung nicht so ideal wäre.

- Wie ist ihre generelle Strategie im Bereich „Identitätsdiebstahl“ beim Online-Banking?

  Antwort: Im Bereich Online-Banking versuchen wir, das Übel bei der Wurzel zu bekämpfen, in dem wir bekannte Phishingseiten, die unser Kreditinstitut nachahmen aktiv suchen und bekämpfen. Gleichzeitig versuchen wir, unsere Kunden für dieses Thema zu sensibilisieren und raten ihnen, entsprechende Schutzsoftware zu verwenden. Wir setzen aber kein Instrument zur nachträglichen Erkennung ein, worauf Ihr Ansatz basiert. Wir sind zwar Anbieter von Girokonten und Tagesgeld, wobei wir aber größere Probleme im Bereich Wertpapierbetrug haben, z.B. werden Depots gehakt und dann im Namen des Opfers Wertpapiere überteuert aus dubiosen Quellen eingekauft.

3. Interviewpartner (Tätigkeitsbereich: Betrugsanalyse und Compliance in einem Kreditinstitut) am 13.11.2008 und 20.11.2008:

* Der Entscheidungsbaum sowie die metrischen Attribute der Diskriminanzfunktion im Abschnitt 9.2 wurden von diesem Interviewpartner am 10.12.2008 begutachtet und qualitätsgesichert.

- Würde Ihr Institut evtl. Echtdaten zur Validierung des Algorithmus bereitstellen?

  Antwort: Das sind heikle Daten und dürfen nicht weitergegeben werden.

- Sind generell Auslandsüberweisungen gefährdet oder überweisen Phisher auch ins Inland? Wie ist hierbei die Verteilung?

  Antwort: Auslandsüberweisungen sind generell verdächtiger, allerdings gehen beim Phishing momentan ca. 80% der Überweisungen ins Inland. Im Inland sitzen sog. Finanzagenten, die die Gelder weiterleiten. Diese Leute sind sich ihres kriminellen Tuns oft nicht mal bewusst. Sie werden oftmals seriös angeworben mit der Aussicht auf einen Zusatzverdienst und vor allem in wirtschaftlich schlechten Zeiten wird dies oft angenommen. Die Zielländer sind unterschiedlich, oftmals Südamerika, Afrika, Osteuropa oder Asien. Zurzeit ist auch England wieder sehr stark involviert.

- Wie viele Betrugsfälle im Online-Banking hat ihr Kreditinstitut bzw. wie ist das Verhältnis von Betrugstransaktionen zu Nicht-Betrugstransaktionen?

  Antwort: Das ist von Bank zu Bank verschieden. Wir führen hierzu Auswertungen durch, dürfen aber diese Ergebnisse nicht nach Außen geben. Für Ihre Anwendung

ist dies aber eigentlich nicht von Bedeutung, da Sie ihr neuronales Netzwerk ohnehin mit genau so vielen Betrugsfällen wie Nicht-Betrugsfällen trainieren sollten um nicht das Netzwerk zu stark mit Nicht-Betrugsfällen zu trainieren und dann einen auftretenden Betrugsfall nicht erkennen.

- Wie läuft der Prozess der Online-Überweisung bei Ihrem Kreditinstitut intern ab?

  Antwort: Nach dem Start der Online-Transaktion durch den Kunden werden PIN und TAN überprüft und in einer operativen Transaktionsdatenbank gespeichert. Wenn PIN und TAN ok sind, wird die Überweisung zum Zielkonto vom Rechenzentrum durchgeführt.

- Wie viele historische Online-Transaktionen müssten von einem Kunden untersucht werden um ein Transaktionsprofil des Kunden zu bekommen, bzw. um abschätzen zu können, wie das „normale" Transaktionsverhalten des Kunden ist?

  Antwort: Das ist schwierig, weil wir keine solchen Auswertungen durchführen. Aber es sollten wenigstens vier historische Transaktionen sein. Besser wären acht oder zehn, allerdings wird dadurch wohl die Performance Ihrer Anwendung beeinflusst.

- Wie hoch ist das Dispolimit für Girokonten bei Ihrem Kreditinstitut?

  Antwort: Das Dreifache des Gehaltseingangs, kann aber unter Umständen auch individuell bei der Kontoeröffnung ausgehandelt werden.

- Wie ist die Vermögensverteilung auf den Girokonten ihrer Online-Banking-Kunden in ihrem Kreditinstitut?

  Antwort: Bei uns nutzen ca. 35% aller Kunden regelmäßig unser Online-Banking-Angebot. Nach unserer jüngsten Auswertung ergab sich bei uns für diese Kundengruppe folgende Kontostandverteilung, wobei sich diese auch ständig ändert:

  - *Ca. 15% unserer Online-Banking-Kunden haben einen negativen Girokontostand.*
  - *Ca. 20% unserer Online-Banking-Kunden haben einen Girokontostand von 0€ bis 2.000€.*
  - *Ca. 50% unserer Online-Banking-Kunden haben einen Girokontostand von 2.000€ bis 5.000€.*
  - *Ca. 15% unserer Online-Banking-Kunden haben einen Girokontostand von über 5.000€.*

  Diese Zahlen bilden aber nur einen ungefähren Wert. Sie können diese aber trotz der Schwankungen durchaus als repräsentativ ansehen.

- Wie ist die Verteilung des monatlichen Gehaltseingangs ihrer Online-Banking-Kunden?

  Antwort: Auch hierfür haben wir eine Auswertung gemacht, wonach sich für die Gruppe unserer Online-Banking-Kunden folgende regelmäßige Gehaltseingänge ergaben:

  - *10% der Online-Banking-Kunden haben keinen regelmäßigen Gehaltseingang.*
  - *12% der Online-Banking-Kunden haben ein monatliches Nettogehalt von 1€ - 900€.*
  - *12% der Online-Banking-Kunden haben ein monatliches Nettogehalt von 900€ - 1.500€.*
  - *15% der Online-Banking-Kunden haben ein monatliches Nettogehalt von 1.500€ - 2.000€.*
  - *25% der Online-Banking-Kunden haben ein monatliches Nettogehalt von 2.000€ - 3.000€.*
  - *26% der Online-Banking-Kunden haben ein monatliches Nettogehalt von mehr als 3.000€.*

- Wie viele Online-Transaktionen hat eine Geschäftsbank oder Sparkasse täglich zu analysieren?

  Antwort: Das ist unterschiedlich und hängt von der Struktur und der Anzahl der Kunden der Bank ab und ob die Verarbeitung der Transaktionen zentral oder dezentral erfolgt. Bei uns sind es so ca. 4.000 Transaktionen am Tag, wobei dies auch täglich schwankt.

- Wie hoch ist die durchschnittliche Zahl von Online-Transaktionen eines Online-Banking-Kunden im Monat?

  Antwort: Das ist ebenfalls unterschiedlich: Es reicht von 0 bis 15 Transaktionen pro Monat einschließlich der Kontoabfragen, bei Privatpersonen können es in Ausnahmefällen auch schon mal mehr werden. Die Verteilung der Häufigkeiten ist dabei ziemlich gleich verteilt. Bei Firmenkunden sind es in der Regel noch mehr. Aber dieser Wert ist auch abhängig von der Kundenstruktur der Bank.

- An welchen Attributen und Eigenschaften einer Online-Transaktion könnte ein Betrugsfall evtl. identifiziert werden?

  Antwort: Bei der Transaktion selbst sind der Transaktionsbetrag, das Zielkonto und die Zielbankleitzahl interessant. Der Verwendungszweck ist evtl. auch wichtig, wobei hierfür mit Ansätzen des Text Minings gearbeitet werden müsste. Wobei der Verwendungszweck auch nicht wirklich aussagekräftig ist, daher würde ich auf die Analyse dieses Attributs verzichten. Interessant sind in diesem Bereich auch noch die Quell IP-Adresse des Absenders und die Zeit, die für die Durchführung der Online-Überweisung benötigt wird, da eine zu schnelle Durchführung auf einen

Trojaner hindeuten könnte. Diese Informationen liegen aber nicht allen Kreditinstituten vor.

- Können Sie mir bekannte Betrugsmuster im Bereich „Identitätsdiebstahl“ beim Online-Banking nennen?

  Antwort: Verdächtig sind bei Phishingfällen Transaktionen nahe an der Verfügbarkeitsgrenze, wobei in einigen Fällen die Betrüger auch geringere Beträge abbuchen um weniger aufzufallen. Generell sind Überweisungen ins Ausland verdächtiger, vor allem, wenn an dieses ausländische Konto zuvor noch nie eine Überweisung geflossen ist. Generell sind nach unserer Erfahrung Personen, die durchschnittlich weniger Online-Banking durchführen gefährdeter als Kunden, die mehrere Transaktionen im Monat durchführen. Der Betrag der Phishingfälle ist meist größer als 1.500€. Beträge darunter sind eher selten, weil es sich für Betrüger nicht lohnt. Nach einer Zahl des Bitkom-Instituts ist der durchschnittliche Betrag beim Phishing bei 3.200€. Privatpersonen sind allgemein gefährdeter, weil Firmenkunden im Normalfall über die nötige Securitysoftware verfügen.

- Woran können Nicht-Betrugstransaktionen beim Online-Banking sicher erkannt werden?

  Antwort: Nur wenn die Überweisungen regelmäßig an den gleichen Empfänger gehen. Allerdings sind auch unbekannte Empfänger nicht immer sofort verdächtig. Bei Neukunden z.B. ist der Empfänger anfangs oft unbekannt.

- Bei welchem Ausgabewert eines neuronalen Netzwerks könnte eine Betrugstransaktion bzw. Nicht-Betrugstransaktion sicher erkannt werden?

  Antwort: Mit einer Betrugswahrscheinlichkeit von größer als 90%, wobei dies auch von den Erfahrungswerten mit dem System abhängig ist. Solche Transaktionen würden dann von Revisionsmitarbeitern geprüft und wenn sich der Verdacht erhärtet würden die Konten des Kunden eingefroren, die Transaktion gestoppt und die Staatsanwaltschaft eingeschaltet. Bei Nicht-Betrugstransaktionen dürfte dagegen die Betrugswahrscheinlichkeit nicht größer als 10% sein.

- Wie schnell muss eine Betrugstransaktion beim Online-Banking identifiziert werden?

  Antwort: So schnell wie möglich, weil bei uns nach dem Anstoßen der Transaktion in der Online Session zwischengespeichert und danach die Abbuchung und die Gutschrift erfolgt. Am besten in Echtzeit oder nahe der Echtzeit.

- Welche Erkennungsgenauigkeit müsste von einem Transaktionsanalysesystem erreicht werden?

  Antwort: Ein solches System müsste für unsere Zwecke eine Erkennungsgenauigkeit von 99% erreichen, damit wir es einsetzen würden.

- Was passiert, wenn eine Online-Transaktion nicht sicher zugeordnet werden kann?

  Antwort: Sie würde von Revisionsmitarbeitern geprüft oder (in Abhängigkeit von der Häufigkeit und Betragshöhe) ohne Prüfung akzeptiert. Schlimmer als ein unerkannter Betrugsfall ist für ein Kreditinstitut eine falsche Verdächtigung bei einem Nicht-Betrugsfall. Von daher würde die Bank eher einen Betrugsfall akzeptieren und den Kunden den entstandenen Schaden ersetzen, da es problematisch wäre, jeden Kunden zu kontaktieren, ob eine von ihm durchgeführte, verdächtige Transaktion so in Ordnung ist. Falls es sich dennoch um einen Betrugsfall handelt, würde die Transaktion rückabgewickelt, falls dies noch möglich ist und die Staatsanwaltschaft eingeschaltet.

- Setzt ihre Bank Complex Event Processing-Technologie ein?

  Antwort: Nein, obwohl diese Technologie vielleicht interessant für diesen Bereich wäre.

- Wie ist ihre generelle Strategie im Bereich „Identitätsdiebstahl“ beim Online-Banking?

  Antwort: Wir setzen in diesem Bereich auf Betrugsprävention, d.h. z.B. aktive Suche nach Phishing Websites und Verbesserung der Sicherheitsmaßnahmen beim Online-Banking durch Verbesserung der bestehenden PIN/TAN und HBCI-Verfahren. Eine nachträgliche Untersuchung erfolgt nur bei einem konkreten Verdachtsfall, wenn ein Kunde selbst einen Betrugsfall gemeldet hat. Transaktionsüberwachung setzen wir konkret nicht ein, da die Implementierung sehr zeit- und kostenaufwändig wäre. Zusätzlich würde wohl eine Betrugserkennungsanwendung die Performance der Transaktionsabwicklung verringern.

4. Interviewpartner (Tätigkeitsbereich: Betrugsprävention in einem Bankenverband) am 01.12.2008:

- Würde Ihr Verband evtl. Echtdaten zur Validierung des Algorithmus bereitstellen?

  Antwort: Wir haben keine solchen Daten, die Sie benötigen würden.

- Sind generell Auslandsüberweisungen gefährdet oder überweisen Phisher auch ins Inland? Wie ist hierbei die Verteilung?

  Antwort: Die genaue Verteilung kann ich Ihnen nicht sagen, aber Auslandsüberweisungen sind hierfür prädestiniert, da die Betrüger oftmals Organisationen in Asien oder Osteuropa bilden.

- Wie viele historische Online-Transaktionen müssten von einem Kunden untersucht werden um ein Transaktionsprofil des Kunden zu bekommen, bzw. um abschätzen zu können, wie das „normale" Transaktionsverhalten des Kunden ist?

  Antwort: Wenigstens fünf bis zehn historische Transaktionen, aber je mehr desto besser.

- Wie viele Online-Transaktionen hat eine Geschäftsbank oder Sparkasse täglich zu analysieren?

  Antwort: Das ist abhängig vom Fokus der Bank, der Anzahl und Struktur der Kunden.

- Wie hoch ist die durchschnittliche Zahl von Online-Transaktionen eines Online-Banking-Kunden im Monat?

  Antwort: Das ist kundenindividuell verschieden. Das Intervall reicht vermutlich von ca. 1 bis 20 Transaktionen, einschließlich der Zugriffe z.B. zur Kontrolle der Kontobewegungen.

- An welchen Attributen und Eigenschaften einer Online-Transaktion könnte ein Betrugsfall evtl. identifiziert werden?

  Antwort: Transaktionsbetrag, Empfängerbank, Verwendungszweck, Uhrzeit der Überweisung, wobei diese wohl nicht sehr aussagekräftig ist und die Überweisungshäufigkeit.

- Können Sie mir bekannte Betrugsmuster im Bereich „Identitätsdiebstahl" beim Online-Banking nennen?

  Antwort: Verdächtig ist hierbei ein sehr hohes Verhältnis von Transferbetrag zum verfügbaren Betrag und auch das Verhältnis des Transferbetrags zum durchschnittlichen Transferbetrag sowie Überweisungen zu unbekannten Empfängern im Ausland, vor allem nach Asien oder Osteuropa. Generell sind hohe Transfersummen verdächtiger. Privatpersonen sind hierbei auch gefährdeter als Firmenkunden.

- Woran können Nicht-Betrugstransaktionen beim Online-Banking sicher erkannt werden?

  Antwort: Theoretisch könnte jede Überweisung ein Betrug sein. Unverdächtig sind eher kleine Beträge und Überweisungen an einen Empfänger, an den der Kunde schon einmal etwas überwiesen hat.

- Bei welchem Ausgabewert eines neuronalen Netzwerks könnte eine Betrugstransaktion bzw. Nicht-Betrugstransaktion sicher erkannt werden?

  Antwort: Hierzu kann ich keine genaue Auskunft geben, da mir die Erfahrung mit solchen Analyseverfahren fehlt.

- Wie schnell muss eine Betrugstransaktion beim Online-Banking identifiziert werden?

  Antwort: In den ersten Minuten nach dem Abschluss der Online-Überweisung.

- Welche Erkennungsgenauigkeit müsste von einem Transaktionsanalysesystem erreicht werden?

  Antwort: So gut wie möglich, 98% wäre ein wünschenswerter Wert.

- Setzt ihr Institut Complex Event Processing-Technologie ein?

  Antwort: Kann ich nicht sagen, weil ich diese Technologie nicht kenne.

- Wie ist ihre generelle Strategie im Bereich „Identitätsdiebstahl“ beim Online-Banking?

  Antwort: Hierzu kann ich keine Auskunft geben.

5. Interviewpartner (Tätigkeitsbereich: Betrugsanalyse in einem banknahen Institut) am 04.12.2008:

* Der Entscheidungsbaum und die metrischen Attribute der Diskriminanzfunktion im Abschnitt 9.2 wurden von diesem Interviewpartner am 15.12.2008 begutachtet und qualitätsgesichert.

- Würde Ihr Institut evtl. Echtdaten zur Validierung des Algorithmus bereitstellen?

  Antwort: Unser Institut besitzt hierzu keine Daten.

- Sind generell Auslandsüberweisungen gefährdet oder überweisen Phisher auch ins Inland? Wie ist hierbei die Verteilung?

  Antwort: Ja, ca. 20-25% der Phishing-Überweisungen gehen ins Ausland, wobei der Anteil der Auslandsüberweisungen an der Gesamttransaktionsmenge von Privatpersonen insgesamt geringer als 20% ist. Daher ist hier eine statistische Relevanz gegeben.

- Wie viele Betrugsfälle im Online-Banking hat ihr Kreditinstitut bzw. wie ist das Verhältnis von Betrugstransaktionen zu Nicht-Betrugstransaktionen?

  Antwort: Wir unterstützen Banken nur beim Analysieren von Betrugsfällen aller Art, selbst haben wir keine Betrugsfälle im Online-Banking.

- Wie viele historische Online-Transaktionen müssten von einem Kunden untersucht werden um ein Transaktionsprofil des Kunden zu bekommen bzw. um abschätzen zu können, wie das „normale“ Transaktionsverhalten des Kunden ist?

  Antwort: Hierfür kann meiner Meinung nach keine genaue Zahl genannt werden.

- Wie viele Online-Transaktionen hat eine Geschäftsbank oder Sparkasse täglich zu analysieren?

  Antwort: Das ist ganz unterschiedlich und hängt von der Größe der Bank ab.

- Wie hoch ist die durchschnittliche Zahl von Online-Transaktionen eines Online-Banking-Kunden im Monat?

  Antwort: Das ist abhängig davon, ob es sich um Firmen oder Privatpersonen handelt. Bei Privatpersonen würde ich so max. zehn Überweisungen im Monat schätzen, wobei dies auch von Bank zu Bank variieren kann. Bei Firmen sind es je nach Größe deutlich mehr

- An welchen Attributen und Eigenschaften einer Online-Transaktion könnte ein Betrugsfall evtl. identifiziert werden?

  Antwort: Am Transaktionsbetrag, an der Zeitdauer der Online Session und am IP-Adressbereich des Quellrechners. Aber ich glaube nicht, dass diese Informationen bei jedem Kreditinstitut ausgewertet werden. Der Empfänger ist auch ein Indiz bzw. die Zielbank.

- Können Sie mir bekannte Betrugsmuster im Bereich „Identitätsdiebstahl“ beim Online-Banking nennen?

  Antwort: Verdächtig sind Transaktionen nahe an der Verfügbarkeitsgrenze und generell höhere Beträge. Oftmals gehen die Überweisungen ins Ausland, wie ich schon erwähnt habe. Die Erfahrung der Personen, die Online-Banking betreiben spielt eine große Rolle, da erfahrene Online-Kunden oftmals auch die nötige Anti-Virus Software laufen haben. Wenn eine Session unter zehn Sekunden dauert, z.B. nur zwei oder drei Sekunden, dann ist dies verdächtig, da möglicherweise ein Trojaner die Überweisung durchgeführt hat. Verdächtig sind auch generell Überweisungen zu einem unbekannten Empfänger, an den vorher noch nie eine Überweisung durchgeführt wurde.

- Woran können Nicht-Betrugstransaktionen beim Online-Banking sicher erkannt werden?

  Antwort: Nur wenn die Überweisungen regelmäßig an den gleichen Empfänger erfolgen und wenn sie ein Kunde persönlich am Bankschalter abgibt, aber dann ist es ja ohnehin keine Online-Überweisung.

- Bei welchem Ausgabewert eines neuronalen Netzwerks könnte eine Betrugstransaktion bzw. Nicht-Betrugstransaktion sicher erkannt werden?

  Antwort: Die Betrugswahrscheinlichkeit müsste vielleicht größer als 88-90% sein, wobei dies dann ohnehin manuell geprüft werden müsste, bevor die Staatsanwaltschaft eingeschaltet werden kann. Wobei es problematisch ist, die Transaktion zurückzuhalten, da hierbei auch der Kunde kontaktiert werden muss, ob die Transaktion so gewollt ist. Für Nicht-Betrug müsste die Betrugswahrscheinlichkeit wohl so unter 10% liegen um ganz sicher zu gehen.

- Wie schnell muss eine Betrugstransaktion beim Online-Banking identifiziert werden?

  Antwort: Am besten so schnell wie möglich.

- Welche Erkennungsgenauigkeit müsste von einem Transaktionsanalysesystem erreicht werden?

  Antwort: So gut wie möglich, 99% wäre ein sehr guter Wert. Mehr wird wohl kaum machbar sein.

- Setzt ihr Institut Complex Event Processing-Technologie ein?

  Antwort: Nein, diese Technologie ist mir auch nicht bekannt.

- Wie ist ihre generelle Strategie im Bereich „Identitätsdiebstahl“ beim Online-Banking?

  Antwort: Wir unterstützen Kreditinstitute, die mit dieser Problematik an uns herantreten.

Anhang 2: Implementierung der Datenversorgungskomponente mit StreamBase Studio

Im Anhang 2 ist die technische Realisierung des Hybrid-Modells dargestellt. Der Bildschirmauszug in Abbildung 47 zeigt die Implementierung und den Ablauf der kompletten Betrugserkennungsanwendung mit Hilfe der Komponenten von *StreamBase Studio*.

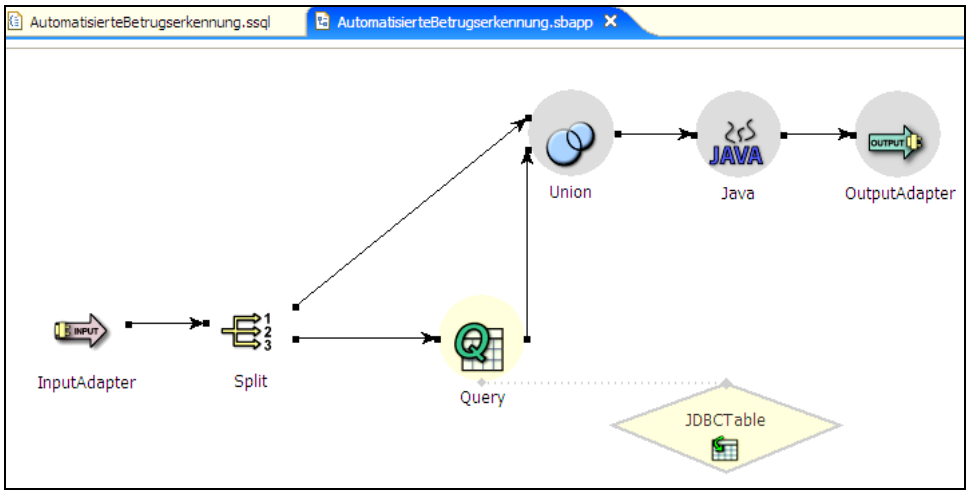

Abbildung 47: Implementierung der Betrugserkennungsanwendung mit Komponenten der StreamBase IDE

Die Transaktionsevents werden über die *Inputadapter*-Komponente aus dem Quellsystem in die CEP *engine* transferiert. In Tabelle 70 ist die Datenstruktur der Transaktionsevents aufgelistet, wie sie im Abschnitt 6.1 beschrieben und in dieser Form für die Experimente verwendet wird.

Attribut	Datentyp (StreamBase)
KundenID	Integer
TransaktionsID	Integer
IP-Adresse des Quellrechners	String
Zeit für Online-Überweisung	Time
Transaktionsdatum	Date
Transaktionsuhrzeit	Time
Transaktionsbetrag	Double
Senderkontonummer	Double
Empfängerkontonummer	Double
Empfängerbankleitzahl	Double

Tabelle 70: Attributstruktur der Transaktionsevents

Die Informationen „IP-Adresse des Quellrechners“ und „Zeit für die Online-Überweisung“ liegen lt. Angaben der interviewten Betrugsexperten nicht bei jedem Kreditinstitut vor. Diese Struktur wird in der *Inputadapter*-Komponente mit den exakten Datentypen angelegt. Der *Stream* der *Inputadapter*-Komponente wird anschließend in der *Split*-Komponente kopiert. Eine Kopie des Transaktionsevents wird der *Query*-Komponente für die, im Abschnitt 7.1 erwähnte, Anreicherung der *events* mit zusätzlichen Attributen verwendet. Zu diesem Zweck wird von der *Query*-Komponente einer externen Datenbanktabelle namens Betrugserkennung mittels einer JDBC-Verbindung, die über die *JDBCTable*-Komponente konfiguriert ist, folgender SQL-String übergeben:

„SELECT b.kontostand, b.dispolimit, b.durchschnTransferBetrag, b.inlandIpAdresse, b.auslandsTransaktion, b.bekanterEmpfanger, b.durchschnAnzahlTransaktionen
FROM Betrugserkennung b
WHERE b.kundenID = „+ inputstream.kundenID;

Der Grund für diese *Select*-Anweisung ist, dass – wie im Abschnitt 6.1 erwähnt – bestimmte benötigte Attribute nicht ursprünglich in der Struktur des Transaktionsevents enthalten sind, sie aber für die Erreichung der Qualitätsziele der Betrugsidentifikation gebraucht werden.
Aufgrund des einfacheren Ablaufs sind im Rahmen der Experimente bzw. der Simulation alle diese benötigten Felder schon vorher in einer einzigen Tabelle zusammengefasst. Das zurückgelieferte *event* der *Query*-Komponente enthält folgende Attributstruktur (in den Tabellen 70 und 71 sind aufgrund der besseren Identifizierbarkeit nicht die exakten technischen Feldnamen angegeben):

Attribut	Datentyp (StreamBase)
Dispolimit	Double
Kontostand	Double
Durchschnittliche Transferanzahl (pro Monat) zum Zeitpunkt der Transaktion	Double
Durchschnittlicher Transaktionsbetrag zum Zeitpunkt der Transaktion	Double
Inland IP-Adresse	Boolean
Auslandstransaktion	Boolean
Vertrauenswürdiger Empfänger	Boolean

Tabelle 71: Struktur der zusätzlich benötigten Attribute

Dieses Ergebnisevent wird im nächsten Schritt mit dem Original-Transaktionsevent korreliert und somit ein *complex event* generiert. Dieser Prozess erfolgt in der *Union*-Komponente. Das neu generierte *complex event*, das sowohl die ursprünglichen als auch die abgefragten Attribute enthält wird anschließend der *JavaOperator*-Komponente übergeben. In dieser Komponente ist eine .jar-Datei (*Java Archive*) eingebunden, welche den Javacode des entwickelten Betrugserkennungsalgorithmus bzw. der Betrugserkennungskomponente enthält und ausführt. Der Javacode der Anwendung ist in einem externen Arbeitsbericht einsehbar. Nach dem Durchführen der Betrugsidentifikation wird das Ergebnis dem *Outputadapter* übergeben. In diesem Fall bildet der *Adapter* eine Schnittstelle zu einer Textdatei, in welcher der Ausgabewert des neuronalen Netzwerks zur Auswertung abgelegt wird. Für weiterführende Informationen zu den verwendeten und weiteren *StreamBase Studio*-Komponenten, siehe [Stre08].

Anhang 3: Darstellung der Ausgabewerte des neuronalen Netzwerks

Im Anhang 3 werden die exakten Ausgaben der Erkennungsgenauigkeit der optimalen Netzwerktopologie (5-5-5-1 Netzwerk) angezeigt. Die Ausgaben für die weiteren analysierten Topologien befinden sich in identischer Struktur in einem separaten Arbeitsbericht.

100 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
```

```
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

1000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

5000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

10000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
```

```
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

20000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

30000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

50000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
```

```
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1
Betrugsfälle zwischen 0,9 und 1,0: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1016
Betrugsfälle zwischen 0,4 und 0,5: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 24,575%
```

100000 Backpropagationsdurchläufe; 0,1 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 4
Betrugsfälle zwischen 0,9 und 1,0: 1989
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 5
Betrugsfälle zwischen 0,7 und 0,8: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 6
Betrugsfälle zwischen 0,5 und 0,6: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 2
Betrugsfälle zwischen 0,4 und 0,5: 1
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 6
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 4
Betrugsfälle zwischen 0,2 und 0,3: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 22
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 964
Betrugsfälle zwischen 0,0 und 0,1: 6
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,4%
```

100 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 74,575%
```

1000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

5000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

10000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
```

```
Erkennungsgenauigkeit: 74,575%
```

20000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1
Betrugsfälle zwischen 0,9 und 1,0: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1016
Betrugsfälle zwischen 0,4 und 0,5: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 24,575%
```

30000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 2
Betrugsfälle zwischen 0,9 und 1,0: 1988
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 2
Betrugsfälle zwischen 0,7 und 0,8: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 3
Betrugsfälle zwischen 0,5 und 0,6: 3
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1
Betrugsfälle zwischen 0,4 und 0,5: 2
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 6
Betrugsfälle zwischen 0,3 und 0,4: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 5
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 8
Betrugsfälle zwischen 0,1 und 0,2: 2
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 986
Betrugsfälle zwischen 0,0 und 0,1: 2
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,925%
```

50000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 3
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 1
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 4
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 3
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 5
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 9
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 29
Betrugsfälle zwischen 0,1 und 0,2: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 961
Betrugsfälle zwischen 0,0 und 0,1: 12
_____________________________________________
```

```
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,275%
```

100000 Backpropagationsdurchläufe; 0,3 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 5
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 2
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 3
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 3
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 6
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 64
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 929
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 97,475%
```

100 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
```

```
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

1000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

5000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

10000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1
Betrugsfälle zwischen 0,9 und 1,0: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
```

```
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1016
Betrugsfälle zwischen 0,4 und 0,5: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 24,575%
```

20000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 3
Betrugsfälle zwischen 0,9 und 1,0: 1988
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 3
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 2
Betrugsfälle zwischen 0,5 und 0,6: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 6
Betrugsfälle zwischen 0,3 und 0,4: 3
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 4
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 8
Betrugsfälle zwischen 0,1 und 0,2: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 986
Betrugsfälle zwischen 0,0 und 0,1: 5
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,925%
```

30000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 3
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 1
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 2
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 4
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 5
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 7
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 16
Betrugsfälle zwischen 0,1 und 0,2: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 977
Betrugsfälle zwischen 0,0 und 0,1: 12
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,675%
```

50000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 5
Betrugsfälle zwischen 0,9 und 1,0: 1987
```

```
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 1
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 3
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 1
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 3
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 3
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 6
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 39
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 955
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,125%
```

100000 Backpropagationsdurchläufe; 0,5 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 6
Betrugsfälle zwischen 0,9 und 1,0: 1986
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 3
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 1
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 2
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 3
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 5
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 37
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 957
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,150%
```

100 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 74,575%
```

1000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
```

```
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 74,575%
```

5000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 74,575%
```

10000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1
Betrugsfälle zwischen 0,9 und 1,0: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 1016
Betrugsfälle zwischen 0,3 und 0,4: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 24,575%
```

20000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 3
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 2
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 2
Betrugsfälle zwischen 0,7 und 0,8: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 3
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 4
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 3
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 6
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 8
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 984
Betrugsfälle zwischen 0,0 und 0,1: 12
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,85%
```

30000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 5
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 1
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 1
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 2
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 5
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 1
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 17
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 982
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,800%
```

50000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 6
Betrugsfälle zwischen 0,9 und 1,0: 1986
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 3
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 3
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 2
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 5
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 18
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 977
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
```

```
Erkennungsgenauigkeit: 98,650%
```

100000 Backpropagationsdurchläufe; 0,7 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 8
Betrugsfälle zwischen 0,9 und 1,0: 1986
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 1
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 2
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 1
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 2
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 3
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 4
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 18
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 977
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,65%
```

100 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 74,575%
```

1000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
```

```
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 74,575%
```

5000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1017
Betrugsfälle zwischen 0,9 und 1,0: 2000
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 0
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 74,575%
```

10000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 1
Betrugsfälle zwischen 0,9 und 1,0: 1981
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 1
Betrugsfälle zwischen 0,8 und 0,9: 2
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 10
```

```
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 1
Betrugsfälle zwischen 0,6 und 0,7: 5
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 2
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 2
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 1
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 1
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 9
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 999
Betrugsfälle zwischen 0,0 und 0,1: 2
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 99,075%
```

20000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 3
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 3
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 6
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 2
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 10
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 989
```

```
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,975%
```

30000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 6
Betrugsfälle zwischen 0,9 und 1,0: 1987
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 1
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 2
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 1
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 1
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 3
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 3
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 12
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 987
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,925%
```

50000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 6
Betrugsfälle zwischen 0,9 und 1,0: 1986
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 3
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 1
```

```
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 2
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 3
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 2
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 3
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 10
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 987
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________
Erkennungsgenauigkeit: 98,90%
```

100000 Backpropagationsdurchläufe; 0,9 Lernfaktor:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 9
Betrugsfälle zwischen 0,9 und 1,0: 1986
_____________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 1
_____________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 3
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 2
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 1
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 2
Betrugsfälle zwischen 0,3 und 0,4: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 3
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 9
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________
```

```
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 988
Betrugsfälle zwischen 0,0 und 0,1: 13
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 98,925%
```

Anhang 4: Ausgabewerte bei veränderten Initialgewichten

Im Anhang 4 befinden sich die Klassifikationsergebnisse aus einer Voruntersuchung bei negativen Initialgewichten der ersten Gewichtsebene in einem neuronalen Netzwerk mit 5-5-5-1 Topologie bei 10.000 Lerndurchläufen und einem Lernfaktor von 0,9:

```
Insgesamt: 4000 Testevents analysiert, davon:
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,9 und 1,0: 0
Betrugsfälle zwischen 0,9 und 1,0: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,8 und 0,9: 0
Betrugsfälle zwischen 0,8 und 0,9: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,7 und 0,8: 0
Betrugsfälle zwischen 0,7 und 0,8: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,6 und 0,7: 0
Betrugsfälle zwischen 0,6 und 0,7: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,5 und 0,6: 0
Betrugsfälle zwischen 0,5 und 0,6: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,4 und 0,5: 0
Betrugsfälle zwischen 0,4 und 0,5: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,3 und 0,4: 1017
Betrugsfälle zwischen 0,3 und 0,4: 2000
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,2 und 0,3: 0
Betrugsfälle zwischen 0,2 und 0,3: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,1 und 0,2: 0
Betrugsfälle zwischen 0,1 und 0,2: 0
_____________________________________________________
Nicht-Betrugsfälle zwischen 0,0 und 0,1: 0
Betrugsfälle zwischen 0,0 und 0,1: 0
_____________________________________________________
Nicht-Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausge-
siebt: 983
Betrugsfälle von Diskriminanzanalyse oder Entscheidungsbaum ausgesiebt: 0
_____________________________________________________
Erkennungsgenauigkeit: 24,575%
```