Note Taking in the Digital Age – Towards a Ubiquitous Pen Interface

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

Vorgelegt von

Florin Schwappach

aus

Nürnberg

2021

Erstgutachter: Prof. Dr. phil. Christian Wolff

Zweitgutachter: Jun.-Prof. Dr. phil. Manuel Burghardt

Dedicated to my parents.

Acknowledgements

Before starting at the media informatics group, I was not at all set on writing a dissertation. The environment of freedom in research that Prof. Christian Wolff created and his positive, encouraging attitude were the deciding factors. I am especially thankful for his way of motivating and supporting me in the more dire straits. Throughout the years at his chair, I grew ever more appreciative of this kind place that is rich in knowledge. Without the important impulses that Jun.-Prof. Manuel Burghardt gave me, I would not have finished the dissertation just yet. To both, I owe a lot.

Completing this endeavor was not a solitary task. Without my parents' nurture of a curious mind and their rock solid, loving foundation, I would not have done it. To Carolin, with her unwavering interest even through the boring parts, her creative mind, huge heart, and endless energy and support, goes all my love.

I thank my family and friends who gave their support and opinion and just were there when I needed them. Joanna, Jeremias, Marco, Robert, thank you.

My colleagues were a great team and often the reason I looked forward to come in. Valentin led the way towards the finish line and I missed his office companionship ever since he handed his thesis in. Thomas's unique take on things was always very valuable to me. The good times with you both as well as with Florian, Andi, and Jürgen will leave me nostalgic probably way too soon. My thanks go to you, and Patricia and Victoria, for always having an open ear. I wish you all only the best in life.

Abstract

The cultural technique of writing helped humans to express, communicate, think, and memorize throughout history. With the advent of human-computer-interfaces, pens as command input for digital systems became popular. While current applications allow carrying out complex tasks with digital pens, they lack the ubiquity and directness of pen and paper. This dissertation models the note taking process in the context of scholarly work, motivated by an understanding of note taking that surpasses mere storage of knowledge. The results, together with qualitative empirical findings about contemporary scholarly workflows that alternate between the analog and the digital world, inspire a novel pen interface concept. This concept proposes the use of an ordinary pen and unmodified writing surfaces for interacting with digital systems. A technological investigation into how a camera-based system can connect physical ink strokes with digital handwriting processing delivers artificial neural network-based building blocks towards that goal. Using these components, the technological feasibility of in-air pen gestures for command input is explored. A proof-of-concept implementation of a prototype system reaches real-time performance and demonstrates distributed computing strategies for realizing the interface concept in an end-user setting.

Zusammenfassung

Die Kulturtechnik des Schreibens hat den Menschen im Laufe der Geschichte geholfen, sich auszudrücken, zu kommunizieren, zu denken und sich Dinge einzuprägen. Mit dem Aufkommen von Mensch-Maschine-Schnittstellen wurden Stifte als Befehlseingabe für digitale Systeme populär. Während aktuelle Anwendungen die Ausführung komplexer Aufgaben mit digitalen Stiften ermöglichen, fehlt ihnen die Allgegenwärtigkeit und Direktheit von Stift und Papier. Diese Dissertation modelliert den Prozess des Notierens im Kontext wissenschaftlichen Arbeitens, motiviert durch ein Verständnis des Notizprozesses, das über die reine Speicherung von Wissen hinausgeht. Die Ergebnisse, zusammen mit qualitativen, empirischen Erkenntnissen über zeitgenössische wissenschaftliche Arbeitsabläufe, die zwischen der analogen und der digitalen Welt wechseln, inspirieren ein neuartiges Stift-Interface-Konzept. Dieses Konzept beinhaltet die Verwendung eines gewöhnlichen Stifts und unveränderter Schreiboberflächen für die Interaktion mit digitalen Systemen. Eine technologische Untersuchung, wie ein kamerabasiertes System physische Tintenstriche mit digitaler Handschriftverarbeitung verbinden kann, liefert auf künstlichen neuronalen Netzen basierende Bausteine hin zu diesem Ziel. Mit diesen Komponenten wird die technologische Machbarkeit von In-Air-Stiftgesten zur Befehlseingabe erforscht. Eine Proof-of-Concept-Implementierung eines Prototyp-Systems erreicht Echtzeit-Performance und demonstriert verteilte Berechnungsstrategien zur Realisierung des Schnittstellenkonzepts in einer Endbenutzerumgebung.

Contents

List of Figures

List of Tables

AI	artificial intelligence
ANN	artificial neural network
AP	average precision
AR	augmented reality
BA	balanced accuracy
AUC	area under curve
CNN	convolutional neural network
CT	computed tomography
DIKW	data – information – knowledge – wisdom
DL	deep learning
DOF	degrees of freedom
DSR	design science research
FCNN	fully connected neural network
FFNN	feed forward neural network
FLOPS	floating point operations per second
fMRI	functional magnetic resonance imaging
FOV	field of view
FPR	false positive rate
FPS	frames per second
GAN	generative adversarial network

GMDH	group method of data handling
GPU	graphics processing unit
GRU	gated recurrent unit
HCI	human computer interaction
HD	high definition
ILSVRC	ImageNet large-scale visual recognition challenge
IOU	intersection over union
LSTM	long short-term memory
MSE	mean squared error
PDA	personal digital assistant
PR	precision/recall
ReLu	rectified linear unit
RNN	recurrent neural network
ROC	receiver operating characteristic
ROI	region of interest
SGD	stochastic gradient descent
TPR	true positive rate
WIMP	windows, icons, menus, pointer

1. Introduction

Since ancient times, humans used a rich variety of tools to make graphical marks. By simple technical means, pen-like instruments allow immediate, rich personal expression (Riche et al., 2017). Writing with them fosters learning as well as retention (Aiken, Thomas, & Shennum, 1975; Kiewra, 1989; Mueller & Oppenheimer, 2014; Mangen, Anda, Oxborough, & Brønnick, 2015; Park & Shin, 2015). As human inventions that aid cognition (Norman, 1993), they allow *thinking* on paper. Pens, pencils or markers are frequently used to jot down, scribble, or doodle on all kinds of paper scraps, the back of one's hand, calendars, or notebooks (Riche et al., 2017). Writing on larger vertical surfaces, like whiteboards or glass panes, enables development or research teams to collaborate freely (Socha, Frever, & Zhang, 2015). Pens vary in their weight, ergonomics, monetary value, and ink, allowing users to integrate them into their workflow according to their preference.

From the early days of computing on, their ubiquity, handling, and immediate expressiveness intrigued researchers to investigate pens as input devices for computers. Ivan Sutherland published *Sketchpad* before the advent of the computer mouse (I. E. Sutherland, 1963). Pen interfaces today have advanced considerably. Mature applications use them to provide powerful manipulation of digital content and have become an industry standard in specific areas, such as digital art creation and graphical content production. Convertible laptops and tablets allow for easy scribbling and note taking, and let users store data in the cloud, copy and paste, link, and structure it.

Still, currently available digital pen interfaces require a set of specialized surfaces and pens with sensors. On the digital end of the analog-digital continuum, there are tablet PCs or drawing tablets with screens that work with induction pens. Towards the other end, there are specialized paper notebooks, compatible with specific smartpens that track a dot pattern to reconstruct ink strokes (“Livescribe Smartpens,” 2020). Similar pens were used by Liao and Guimbretièere (2012) in their *PapierCraft* interface. They expressed interest in ”avoiding being tied to the Anoto pen and the dot-pattern paper“ (Liao & Guimbretièere, 2012, p. 22). Other solutions provide a smartpen and an electronic surface to which users can attach any paper (“Wacom Smart Pads,” 2020). Be it on the glass-like surface of a tablet or on dotted paper, the affordances of analog pens transfer only partially. People do not consider digital pens synonymous with analog ones. For example, they associate them with refining and sharing work products instead of considering them a drafting tool for ideation and thinking (Riche et al., 2017). Each feature of a digital note-taking application needs to be conceived, implemented and tested, thus coming at a high cost, limiting their flexibility. Furthermore, such applications are bound to specific hardware, so a lost or defective pen can not be replaced easily.

1.1. Research Agenda

This dissertation proposes and examines a novel approach to conceptualize pen computing. Rooted in a theoretical analysis of the note taking process, the requirements for digital support of *thinking on paper* are collected and substantiated through a qualitative study in the context of scientific note taking. Then, technological building blocks are investigated. They chart the course towards a ubiquitous pen interface which allows users to interact with digital systems using their favorite pen on any slip of paper. To verify the feasibility of the concept, a prototype implementation connects these building blocks, realizing some of the desired functionality.

1.1.1. Objectives

The scientific inquiry of the research project as outlined above can be formulated into the following objectives.

(1) **Analysis of the note taking process (chapters 2 and 3)** Review the history and current scholarly views on note taking in order to inform pen interface constraints and requirements. Study habits of scholars taking notes to support the findings.

(2) **Pen interface concept and technological consequences (chapter 4)** Compare existing solutions with the requirements identified and develop an interface concept based on the findings for common use cases where knowledge workers take notes while working, attending meetings, or while reviewing text. Discuss technological consequences and provide an operational model.

(3) **Technological building blocks (chapters 5–11)** Investigate and develop appropriate solutions matching the technological requirements. Assess their functionality by gathering data in realistic settings and test their performance with it.

(4) **Prototype (chapter 12)** Connect the building blocks in a proof-of-concept system that shows the feasibility of operating the proposed components in a user-centric application context.

1.1.2. Scope and Limitations

This work bases the requirements of note taking in the context of knowledge work, i.e. scientific research or engineering. It views note taking as more than mere information storage, but as a process that warrants reflection on a philosophical and cognitive level and argues this position in chapter 2. While it proposes an interface concept through a combination of building blocks, it does not aim to produce a system ready for daily use. Its technological investigation focuses on establishing components that connect purely physical notes with digital text recognition and processing systems. This connection is one of the *missing links* towards making a ubiquitous pen interface a reality.

1.2. Scholarly Context

This research project revolves around ways that *human-computer interaction* systems can be designed to bridge the gap between physical and digital media. Analyzing note taking processes in order to support them with technology touches on several disciplines of the humanities: *cognitive science, educational psychology*, and *media theory*. The organization and creation of knowledge through these processes raises fundamental questions of *philosophy* that *information science* picks up and discusses through the lens of the relationship between information and its users.

Media informatics connects these issues – of users, their interaction with machines, media, and the implications on information and knowledge as well as psychological aspects – with the technological advances in the area of *computer science* and *electronics* (Wolff, 2009; Herczeg, 2009). It is therefore an interdisciplinary field of research that draws from findings of scholars of numerous vocations. Relying on computer science and the humanities to chart a course towards a ubiquitous pen interface, this work includes theoretical analysis, empirical studies, and the creation of IT artifacts. It is a media informatics project at heart.

1.3. Research Paradigm

For media informatics projects, the *design science research paradigm* is highly relevant, since design science ”supports a pragmatic research paradigm that calls for the creation of innovative artifacts to solve real-world problems“ (Hevner & Chatterjee, 2010, p. 9).

The methodology for this dissertation was chosen from within the field design science research (DSR) approaches, which focus on an iterative process of construction and evaluation as a way towards scientific knowledge. The framework for selecting an appropriate DSR approach by Venable, Pries-Heje, and Baskerville (2017) offers a series of technological questions that guide this decision.

Based on the answers to the technological questions O/P-1 through O/P-3, either the *systems development research methodology* (SDRM) (Nunamaker Jr, Chen, & Purdin, 1990), the *design science research process model* (Vaishnavi & Kuechler, 2015), or the *design science research methodology* (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007) are deemed appropriate choices according to Venable et al. (2017).

This research project aims at creating an actual IT system as proof-of-concept (question O/P-1), which is an experimental prototype and as such, is not yet extensively adapted to daily use (O/P-2). The development of a design theory comes (question O/P-3) as the final result of several DSR cycles (Vaishnavi & Kuechler, 2015), which is out of scope for this research dealing with an emerging technology. Following Venable et al. (2017), SDRM is a suitable choice for this research project.

According to Nunamaker Jr et al. (1990, p. 97), the SDRM methodology aims to combine social/behavioral and engineering research processes that ”[b]oth have much to contribute to the font of information systems knowledge“. While there are differences between the DSR approaches, as Venable et al. (2017) make clear, the fundamental process is an iterative procedure starting from problem awareness that leads to various artifacts that are in themselves research contributions. In figure 1.1, a schematic overview of the process steps of SDRM is given. The steps can be revisited during the research process and serve to address research issues ranging from conceptual work, i.e. looking to other disciplines for ideas and collecting requirements, to evaluation in laboratory or field studies (Nunamaker Jr et al., 1990).

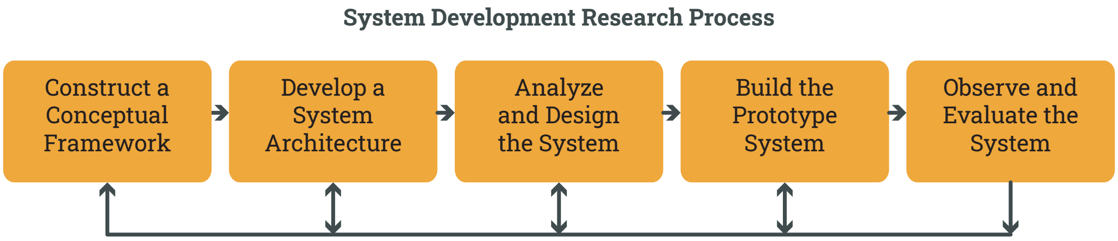

System Development Research Process
Construct a Conceptual Framework
Develop a System Architecture
Analyze and Design the System
Build the Prototype System
Observe and Evaluate the System

Figure 1.1. – Diagram of the system development research process (Nunamaker Jr, Chen, & Purdin, 1990). Each step can be part of several iteration cycles and addresses research issues ranging from the ”study of relevant disciplines for new approaches and ideas“ to ”learn[ing] about the concepts, framework, and design through the system building process“ (p. 98).

In particular, some the goals of *software engineering research* formulated by Nunamaker Jr et al. (1990, p. 97) are applicable to the process of finding suitable building blocks for the proposed pen interface: this dissertation aims to review and synthesize previous research by building a model of the note taking application domain, which is juxtaposed with qualitative empirical results. Based on an analysis of this model and the study results, requirements are extracted and several systems are built based on them. Those systems are observed and evaluated, and the findings consolidated and used to build a prototype that highlights the feasibility of the single components working together. This course of action represents applied design science research.

1.4. Outline of this Dissertation

This dissertation is structured into four conceptual parts that broadly align with the SDRM steps in figure 1.1. The relevant steps are given as captions separating the chapters.

Analysis and Construction of a Conceptual Framework

- **Chapter 1**: *Introduction*. The introduction provides the problem statement and research agenda.
- **Chapter 2**: *Note Taking*. The history of scientific note taking is the entry point for discussing the note taking process and the definitions of information and knowledge. The elements of the note taking process are delineated and their interplay is modeled after looking at them from an interdisciplinary perspective. Literature-based requirements for digital support of scientific note taking conclude this chapter.
- **Chapter 3**: *Qualitative Study of Scientific Note Taking*. To put the findings from chapter 2 into a present-day context, an interview study of contemporary scholarly workflows with a focus on note taking activities highlights practices and tools of participants at the Master's and PhD level. Conclusions are drawn regarding the place of note taking in physical and digital work spaces.

Development of a System Architecture/Analyze and Design the System

- **Chapter 4**: *Towards a Ubiquitous Pen Interface*. Chapter 4 reviews existing pen interface systems and consolidates findings from chapters 2 and 3 into a novel interface concept. It introduces the building block concept for investigating the missing link between physical ink and digital processing and explains which components are investigated in this dissertation.

- **Chapter 5**: *Machine Learning Models for Pen Interfaces*. After arguing the use of deep learning approaches, chapter 5 reviews the history of artificial neural networks and takes a closer look at the qualities of such systems. It concludes with a line-up of recent image processing architectures and their components, narrowing the search for suitable model types.

- **Chapter 6**: *Data Collection*. Deep learning models learn representations from data. For this dissertation, domain-specific datasets were collected in user studies with 60 participants. The collection of domain-specific datasets is detailed in this chapter, as well as annotation methods.

- **Chapter 7**: *Loss and Performance Metrics*. Optimization algorithms, like the training of artificial neural networks, rely on the minimization of target functions for finding beneficial parameter sets. This chapter explains the loss functions used throughout the technological part of this dissertation as well as the metrics used for evaluation.

Build/Observe/Evaluate

- **Chapter 8**: *A Baseline Model for Pen Tracking*. This chapter documents the investigation of a baseline for pen detection in camera streams using convolutional neural networks. The literature-based development of the model architecture template is followed by an analysis of 16 model variants using the data from chapter 7 to find the set of models most suited for the task of pen tracking.

- **Chapter 9**: *Handwritten Text Extraction*. Before providing a segmentation approach that allows pen interface applications to use the actual text contained in an image stream, this chapter reviews previous work for text detection and binarization.

- **Chapter 10**: *Sequence Analysis*. For the task of predicting the state of a pen from a video sequence, i.e. if it touches the paper at a given point in time or not, this chapter evaluates 33 neural network variants using specially annotated data from chapter 7.

- **Chapter 11**: *Pen Gestures*. A future interface needs a way to interact with it. This chapter discusses pen gestures as a mode of command entry in existing systems and proposes a novel in-air gesture concept for ordinary pens. With a user study collecting gesture data in a realistic scenario, the ability of the implemented building blocks to realize a pen gesture interface is assessed.

- **Chapter 12**: *Prototype*. This chapter elaborates on the approach to connect the investigated building blocks to a meaningful whole. The running prototype serves as a proof-of-concept that the pen interface principle is feasible, but does not provide a full production implementation.

Conclusion and Future Work

- **Chapter 13**: *Conclusion*. This chapter offers concluding statements and discusses the evaluation results in the context of user acceptance ratings and handwriting recognition error rates.

- **Chapter 14**: *Future Work*. Future work is proposed, and the research agenda for an augmented reality pen interface is broached as a concrete next step in the research process towards a ubiquitous pen interface.

1.4.1. Publications

The following publications are related to this dissertation:

- Schwappach, F. & Burghardt, M. (2019). Augmentierte Notizbücher und Natürliche Interaktion – Unterstützung der Kulturtechnik Handschrift in einer digitalen Forschungswelt. In: *DHd 2019 Digital Humanities: multimedial & multimodal. Konferenzabstracts*, Frankfurt am Main. pp. 258–260. `https://doi.org/10.5281/zenodo.2596095`.

- Achmann, M. & Schwappach, F. (2021). Grundlagenermittlung für die digitale Werkbank qualitativ-hermeneutisch arbeitender Geisteswissenschaftlerinnen: Exploration geisteswissenschaftlicher Forschung mit Fokus auf Exzerpten und Literaturverwaltung. In: T. Schmidt, C. Wolff (Eds.): *Information between Data and Knowledge. Information Science and its Neighbors from Data Science to Digital Humanities. Proceedings of the 16th International Symposium of Information Science (ISI 2021)*, Regensburg, Germany, 8th–10th March 2021. Glückstadt: Verlag Werner Hülsbusch, pp. 200–216. `https://doi.org/10.5283/epub.44945`.

2. Note Taking

This chapter provides an overview over the history of note taking, originating in the age of early modern science. The relationship of knowledge and information in the human-notebook context is explored by revisiting definitions and modeling the *notebook process* by identifying its elements and how they relate from the perspective of several scientific disciplines. This builds the foundation for establishing constraints and requirements that determine possible digital support of note taking as *thinking on paper*.

2.1. The History of Note Taking

Over the last decades, researchers investigating note taking from different perspectives found that note taking and writing processes have positive effects on retention, creativity, and learning (Aiken et al., 1975; Kiewra, 1989; Mueller & Oppenheimer, 2014; Mangen et al., 2015). But notebooks have been steady companions to scientists, writers, artists, and businessmen throughout much of younger history. The notes that have survived the times can deliver insights into the research processes of characters as notable as Isaac Newton or Paul Dirac (see McGuire and Tamny (1983) for the former, Galison (2000) for the latter). Figures 2.1 and 2.2 show examples of their notes and illustrate the diverse nature of scientific note taking. Although the ways of keeping records have changed considerably, a constant seems to be an inherent need for visualization and thus materialization of thought processes through writing.

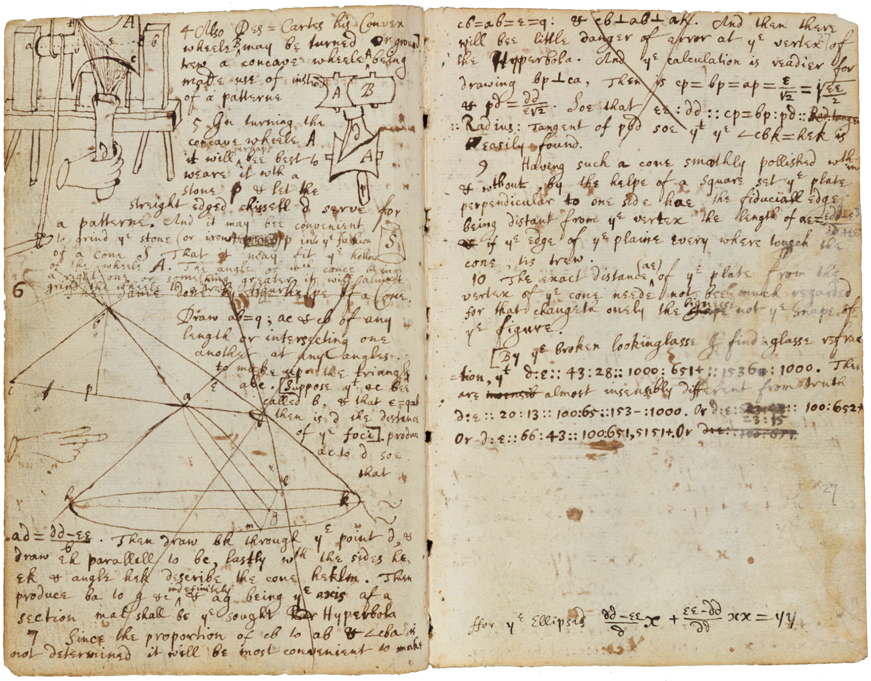


Figure 2.1. – Notes of Isaac Newton regarding convex wheels and glass refraction. (Image from the Cambridge digital library archive, MS Add. 4000: College Notebook of Isaac Newton, 26v and 27r. http:\\cudl.lib.cam.ac.uk\view\MS-ADD-04000\56.)

Various types of record keeping devices were around since ancient times. As paper mills spread throughout Europe between roughly 1100 and 1600, the production cost of paper declined (Hunter, 1978).

Towards the end of this period, so-called *paperbooks*, a bound collection of blank pages, became a common occurrence. Merchants developed well-defined bookkeeping methods using these. For short term memory support, specially prepared books offered erasable pages. With the advent of early modern science, practices and methods of taking notes emerged, accompanying a new, evolving scientific process. *Adversariae* and commonplace books provided space to gather knowledge to a budding profession of scientists (Yeo, 2014)[1].

[1] The given source is the origin of the ideas in the paragraph preceding it. This circumstance will be marked with * from here on

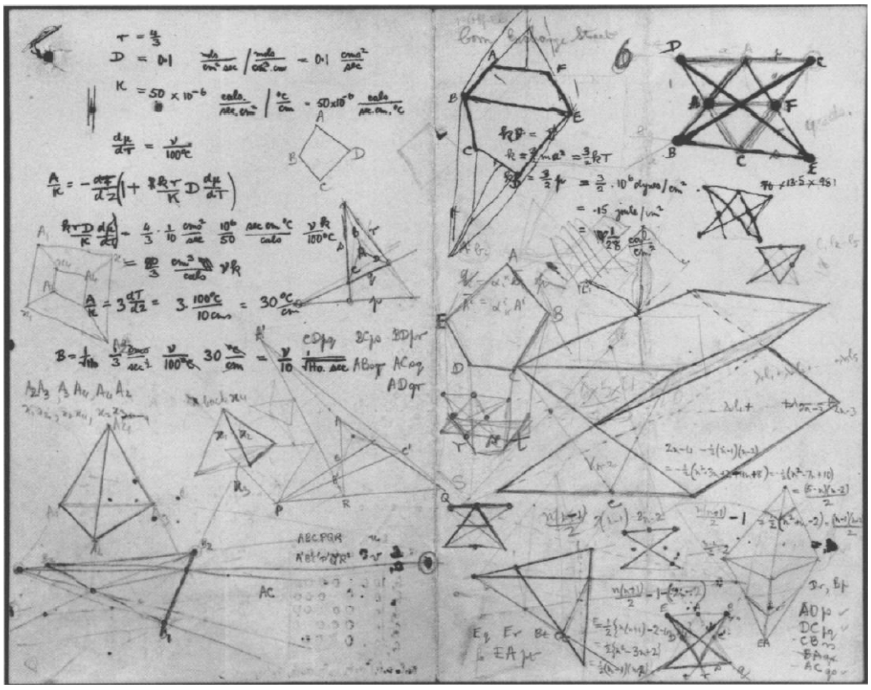


Figure 2.2. – Private notes of Paul Dirac: mostly geometric pondering, contrasting his graphically ascetic publications (Hoffmann, 2013).

These forms of notebooks were mainly used to collect and recombine established knowledge. Since the 18th century, the personal observations and experience of the writer gained importance. Note taking transformed into a research tool in itself, and, as a process, took up an elementary place in scientific endeavors. In the 19th and 20th century, the level of formalization dropped, as evermore idiosyncratic modes of note taking developed (Krauthausen & Nasim, 2010)*.

Throughout time, pen and paper remained common features of the process. Be it on ethnographic excursions or during nights of astronomic observation, researchers were jotting down, drawing, or meticulously noting information about their subject of inquiry.

2.2. The Notebook Process

Before taking a closer look at the process's inner workings, the meaning of the terms data, information, and knowledge will be discussed in the following section.

Then, for a systematic analysis of note taking in a scientific context, the elements involved and their roles will be investigated. Margin notes identify relevant disciplines or concepts which provide helpful insights into the aspect discussed.

2.2.1. Definitions

The work with information, the creation of knowledge, and the role of ideas are central concerns in the analysis of scientific note taking. It follows that it is necessary to explain the relationship between the core terms and clarify the characteristics separating them. Information and knowledge are particularly hard to define, as the numerous efforts of different disciplines show. This section thus does not aim to provide a complete discussion, but aims to establish some conceptual clarity by contrasting different schools of thought. With fundamental terms like these, popular one-sentence definitions are not conclusive explanations of a concept but rather entry points to a certain perspective on the notion discussed.

Data – Information – Knowledge – Wisdom

A widely used approach is to define the terms as parts of a hierarchical structure. A pervasive class of models present in information science research consists of various forms of the data – information – knowledge – wisdom (DIKW) hierarchy, which was published by Ackoff (1989), among others.

Rowley (2007) provides an overview of the use of the DIKW hierarchy in literature. While not all reviewed sources agree on all elements, the consensus according to her analysis is that even across differing definitions, ”higher elements in the hierarchy can be explained in terms of the lower elements by identifying an appropriate transformation process“ (p. 168). The challenge for researchers in cognitive science and philosophy, then, is to understand this transformation process.

Criticism of DIKW includes the argument that its usefulness for research is limited. Most discussions of it elaborate on the intuitive understanding of the terms in popular usage, wanting for refinement to advance theory (Bates, 2017)*.

This relation to the prevalent use of the terms could explain the lasting presence of the DIKW hierarchy. The concept offers succinct definitions, which are commonly cited in literature. Accessability, rather than exhaustive scholarly discussion, might play a role in their popularity.

Frické (2009) provides a critique of the DIKW hierarchy based on it being methodologically undesirable, since it encourages mindless data collection, hoping to promote it into information to answer questions. He identified a second, more fundamental misgiving coming from a logical error inherent to the hierarchy that pertains to the transformation from data into information: Fixing the scope of inferences allowed from data leads to a dilemma. A narrow range excludes statistical generalizations, while widening the scope includes invalid inferences. Either useful information is ignored, or the foundation of the pyramid hierarchy includes false information. Resolving ambiguities thus leads to logical inconsistency, which limits the usefulness of the model.

The definitions as they come from Ackoff (1989) will be included in the following, as they are still in use in some disciplines and provide an intuitive entry point to the discussion.

Data

In Ackoff’s (1989) definition, data consists of symbols that represent the property of an object, of an event or of their environment. It is the result of observation and differs from information in a functional, not a structural way. It is of no use until transformed into a usable form.

In a hierarchical structure, this limits all information and, thus, knowledge, to the observable (Frické, 2009). But, as Frické puts it, ”there is a huge domain of the unobservable for which no instruments of measurement exist“ (p. 134).

According to Rowley (2007), most reviewed definitions of data go by what it lacks: ”Data lacks meaning or value, is unorganized and unprocessed.“ (p. 171). In the DIKW hierarchy, this approach offers ways in which information can be defined in terms of data, i.e. which transformations are necessary.

Frické (2009) offers a formal approach using a subset of predicate logic. Each datum is interpreted as a logical atom and can be combined with other such atoms using logical operators, leaving out negation, implication, disjunction, and the universal qualifier. These facts or basic statements of existential-conjunctive logic are formed in a way that allows them to be recorded and entered into a relational database given an appropriate n-tuple. He proposes defining data – in the context of information science – somewhat trivially as ”anything recordable in a database in a semantically and pragmatically sound way“ (p. 139).

This definition is not built around deficiencies, but on requirements put on data points. Most importantly, by Frické’s words, depending on context, all data is also information, thereby complicating a distinction.

Information

Bates (2017) states that ”[d]efining information remains such a contested project that any claim to present a unified, singular vision of the topic would be disingenuous“. Still, it is a worthwhile undertaking to look at the debate and find out which aspects of the notion are relevant to this book.

Shannon

An objective entry point to the discussion is Shannon’s mathematical definition of information as the basis of information theory (Shannon, 1948). It is based on the probability of symbols transmitted by an information source, where lower probability means more information. It is a measure of the reduction of uncertainty upon seeing a symbol and has the unit *bit*. Messages can be coded using an alphabet, and the expected value of information in a message is called its entropy. With this, it is possible to determine the probabilities of symbols in an alphabet to attain optimal information transmission, i.e. to achieve the highest entropy.

While some of the terms used imply a linguistic level of transmission, all kinds of information can be transmitted this way. There just needs to be a coding rule.

This theory has had a wide-ranging impact in many fields and laid the foundation for the digital coding and compression of messages (Gallager, 2001).

In the social sciences, capturing the essence of information has proven more elusive. While certainly powerful, Shannon's definition does not take into account the role of the mind, culture, or, for example, deterioration. As Frické (2009) writes, it is not concerned with meaning and truth. It is a "purely syntactic property of something like a bit-string, or other structure that might be transmitted" (Sloman, 2011, pp. 397–398).

DIKW

Ackoff (1989) describes information as part of the DIKW hierarchy. It is inferred from data and contained in descriptions. It can be found in answers to questions that begin with 'who', 'what', 'when', or 'how many'. Frické (2009) notes the odd omission of one information seeking question: 'why', highlighting the restrictive set of information allowed. He argues against the transformative nature and posits that "information is irreducible to data" (Frické, 2009, p. 140).

Rowley (2007) finds the core element of information as presented in popular reference works regarding DIKW as some kind of organization, interpretation, or context. Structuring data gives it relevance for a specific purpose and turns it into information by making it useful, and giving it meaning and value. Note that here, Rowley's summary comprises terms that imply a fundamental human component.

Objective/
Subjective

Parker (1974) offers a broad definition of information as "the pattern of organization of matter and energy" (p. 10). Bates (2006) relies on this objective definition by Parker to differentiate separate, fundamental forms of information for the use in

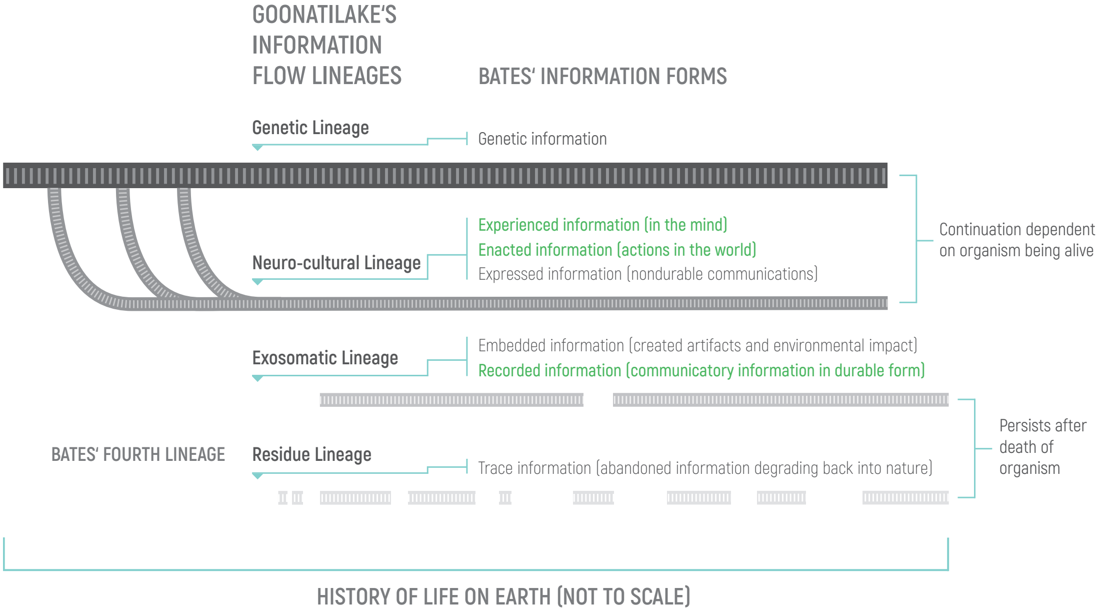

GOONATILAKE'S INFORMATION FLOW LINEAGES
BATES' INFORMATION FORMS
Genetic Lineage
Genetic information
Neuro-cultural Lineage
Experienced information (in the mind)
Enacted information (actions in the world)
Expressed information (nondurable communications)
Continuation dependent on organism being alive
Exosomatic Lineage
Embedded information (created artifacts and environmental impact)
Recorded information (communicatory information in durable form)
BATES' FOURTH LINEAGE
Residue Lineage
Trace information (abandoned information degrading back into nature)
Persists after death of organism
HISTORY OF LIFE ON EARTH (NOT TO SCALE)

Figure 2.3. – Graphic of Goonatilake's information flow lineages and Bates's information forms, with the forms relevant to the notebook scenario highlighted (based on Bates's (2015) visualization in the online version).

information science. She embeds them in the evolutionary framework of information flow lineages of Goonatilake (1991), as can be seen in figure 2.3. The information forms that would be relevant to the notebook scenario are highlighted in green.

There is an ongoing discussion on whether an objective definition is even useful. Hjørland (2011) states: ”I see no need for Bates's objective definition of information“ (p. 574). He relies on Bateson (1972) to define information as a ”difference that makes a difference“ (p. 321). By that, he argues, any differences (or forms of organization) are only information if they inform somebody about something.

Curiously, Bateson himself argues that information can be processed by ”any ongoing ensemble of events and objects which has the appropriate complexity of causal circuits and the appropriate energy relations“ (p. 321) and that it is measured in *bits*. He also embraces a technical definition later in the book, where it is defined as ”any difference which makes a difference in some later event.“ (p. 386). In its expanse this is similar to Parker’s definition, albeit with more emphasis on sequence.

In light of this, Hjørland’s argument might be interpreted to say: only a difference that is processed by an entity is relevant, and thus, is information, supporting his argument for subjectivity.

More recently, Bates (2015, p. 4) clarified her belief that:

> [I]nformation exists both subjectively and objectively: subjectively as our human experience of novelty, learning, emotion, perception, etc., and objectively, as the pattern of organization of matter and energy, the marks that take up the pages of books, or the electronic ones and zeroes that exist in digitised information stores.

Criticism Bateson

Sloman (2011) dismisses Bateson’s definition and its usage. He claims the definition is based on a misquote. Bateson defines a *bit* – the fundamental unit, not the notion – of information as a difference that makes a difference in several places throughout his book. Still, Bateson clearly provides a technical definition of information as quoted earlier.

However, Sloman also sees the definition of the unit as ”too simplistic“ (p. 399) and weighs the (speculated) influence on it by low level functioning of computers and brains as negative. His questioning of the definition is picked up by Kuhlen (2013).

Pragmatic Primacy

Kuhlen (2013) recognizes objective and evolutionary approaches as well as subjective attempts to define information. He agrees with Wersig’s (1971) comment that there are almost as many notions of information as there are authors writing about it. Kuhlen distances himself from hierarchical approaches and supports a functional distinction of formal-syntactic, semantic and pragmatic planes of information. He argues for a pragmatic rationale to take the central role in information science and does not believe in an ontological understanding of information.

Kuhlen explains that semantically, information always relates to knowledge and never stands by itself. Using relevant knowledge, we do information work, which is influenced by numerous contextual factors. Information is formed in this process. It can be used to inform actions and decisions. By learning it, the user can build new knowledge from information in a second transformation. Summarizing this process, Kuhlen has coined the phrase *information is knowledge in action and context* (Kuhlen, 2013).

From Information To Knowledge

Knowledge is even harder to carve out as a separate idea. It is ”an elusive concept which is difficult to define“ (Rowley, 2007, p. 173). Rowley summarizes it as an amalgamation of ”information, understanding, capability, experience, skills and values“ (p. 174).

Bates (2006) defines knowledge as ”Information given meaning and integrated with other contents of understanding“ (p. 1036). Some authors use information and knowledge interchangeably (Rowley & Hartley, 2008).

In its broad ambiguity, there seems to be little dissension about associating the terms *understanding*, *meaning*, and *experience* with knowledge. When looking into the philosophical discussion of the notion, a vast body of literature concerned with differing interpretations is revealed (Audi, 2010).

The justified-true-belief approach of traditional philosophy is represented in what fallibilists call strong knowledge, which they separate from weak knowledge by its justification. Another common distinction is between know-how and know-that, or related to that, between procedural and declarative or practical and theoretical knowledge. Following the idea that there is a noticeable degree of independence between these two concepts is also a philosophical decision (Ichikawa & Steup, 2018)*.

Knowledge management literature sometimes mentions tacit and explicit knowledge, referring to a similar general separation (Rowley & Hartley, 2008).

As explicit or declarative knowledge could be entered into a database, it fits into Frické's (2009) definition of data-information. Rowley and Hartley (2008) categorize explicit knowledge as information, too. It thus becomes apparent that the terms used largely rely on the context in which they are investigated – data can be information, and knowledge can be information.

Scientific method has, as a central tenet, the objective to somehow prove or justify statements by following a systematic, community-accepted argumentative strategy. The justification can be arrived at by a variety of means – be it hermeneutic cycles, the circumscription cycles of design science research, or the positivist empirical approach. In a research context, then, two apparent goals regarding knowledge can be identified:

- **Strong** declarative/theoretical/**know-that**-knowledge,
  to enrich one's **personal** knowledge base

- **Strong** declarative/theoretical/**know-that**-knowledge,
  made permanent as **recorded** information (publication)

Self-actualization of the researcher involves reaching those goals, as well as increasing their know-how or tacit knowledge while doing that.

Information and Knowledge in the Context of Scientific Note Taking

Figure 2.4 depicts the model generated from the literature-based analysis of the notebook process, putting information and knowledge in context. On the continuum between direct or personal and indirect or impersonal action, four planes of activity were identified. First and most important is the user as actor. Users interact with the notebook by writing or drawing. The notebook itself is interacted with in a more direct fashion than an information system, which is the element furthest from the user included in the model.

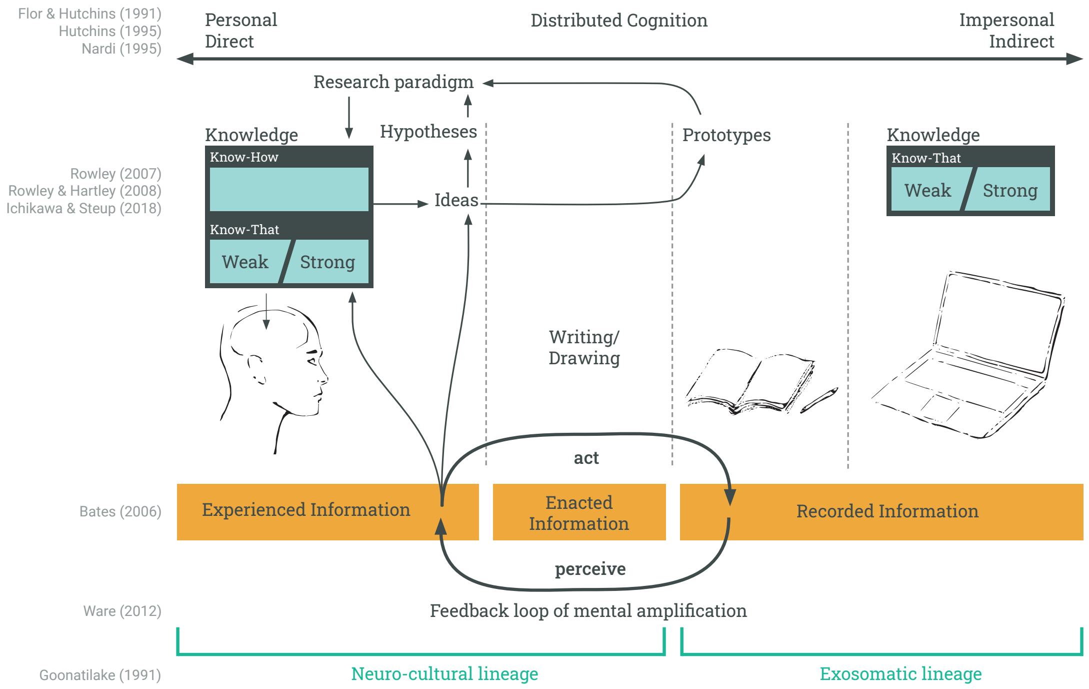


Figure 2.4. – Information and knowledge in the context of scientific note taking: A model of the notebook process.

This model utilizes a transformative, non-hierarchical approach towards information, following Kuhlen (2013). Additionally, it integrates objective forms of information by Bates (2006) as a way of talking about the role of elements in the distributed cognitive system. *Experienced information* in the mind is expressed as *enacted information* through writing and is stored as *recorded information* in the notebook. By perceiving written words or sketches and experiencing the writing process, new *experienced information* is transformed into knowledge or leads to ideas. Even the process of writing can modify the thought. As soon as the written word is on paper, it is taken in again and pushes new thought processes, connecting with other units of knowledge we have in mind. This cycle of represents a feedback loop amplifying cognition (Ware, 2012).

The knowledge in the researchers' mind can be divided according to the knowledge distinctions in section 2.2.1, as can the knowledge in information systems, with the notable exclusion of know-how.

Inherent to the ideas generated while working with information is a directive character. From them, hypotheses can be formulated, or prototypes can be built to investigate. Using various research paradigms, support for hypotheses can be gathered or prototypes can help answer research questions. All this work then ideally leads to an increased treasure of potentially relevant knowledge, available for further information work and ready for transformation.

2.2.2. Process Elements

The main elements of the note taking process investigated in this section correspond to the planes of activity identified in figure 2.4. The user or researcher is situated on the layer of *experienced information* and the mind. The layers of *enacted* and *recorded information* are concerned with the primary tools. They can be divided up into the pen as the expressive part for writing and drawing, and the notebook as a receptive element for recording traces of the note taking process.

These traces are the tangible product of the process and were treated separately in the assembly of elements. They represent the personal and direct *recorded information*. Their temporal sequence and spatial structure is of special interest.

A category of secondary objects was established to collect tools used in the vicinity of the notebook. This includes laptops, lab instruments, and books. These utensils are located at the impersonal end of the continuum in figure 2.4. They hold explicit knowledge, provide data processing services, or allow the observation of phenomena.

Employed as a research tool, note taking is more than a basic memory support technique. This section articulates aspects of it through the lenses of different disciplines to support this argument and to reflect the complex layers of note taking. As Hoffmann (2013) puts it: ”Writing must rather be considered an instrument of research itself, structuring practical tinkering, organizing the outcomes, and intervening in the more abstract work of reasoning and reflecting“ (p. 280).

In the following, the use of a notebook in a scientific context is referred to as *notebook process* for brevity. A table of requirements extracted from this analysis is provided at the end of this section.

User

Management Studies

The process hinges on the user of the notebook. In a research context, users can be considered knowledge workers in Drucker’s (1959) sense. In (1999) he picks up on this definition and argues that advancing the productivity of those workers is a challenge central to 21st century institutions. However, he also calls into question if variables like their efficiency and productivity can be accurately quantified.

The concept of a ”knowledge worker“, distinct from manual laborers, is helpful in an organizational context, since as business assets their unique needs need to be asserted and met. The structure of institutions can be evaluated and adapted to accommodate this category of employee (Drucker, 1999).

The system-oriented perspective of cognitive science, on the other hand, differs from an organizational approach by being mainly concerned with the inner workings of the researcher as opposed to a black box consideration tallying input and output of business value.

Cognitive Science

Scientists often start out intrinsically motivated with questions they refine during the process and arrive at answers to questions they never thought to ask at the beginning. Creativity and the ability to break out of confined ways of thinking are essential to discovering new avenues of inquiry. Of particular interest are mental processes that enable this kind of thinking. Cognitive science, stemming from psychology, delivers a system-oriented perspective onto what goes on inside the human mind.

In it, a constant reordering, restructuring and interconnection of knowledge takes place. The more relationships we build between new and present knowledge, the more complex this task becomes, as scopes widen and networks deepen.

To handle this sometimes overwhelming complexity, humans use materialization strategies, thereby creating a feedback loop utilizing their sensory pathways, which Ware (2012) describes in the context of information visualization. Cycling between idea, recording, observation, assessment, and refinement, the external tools – media – become part of the knowledge generation process.

Using a notebook to formulate thoughts and find connections between established arguments, as well as sketching graphs or contraptions, too, are examples of a feedback loop. Its visual nature allows for high-bandwidth processing of the information the user is working with (Ware, 2012).

Educational Psychology

In the fields of educational and cognitive psychology, effects of note-taking on learning, understanding and retention have been investigated. It is thought to facilitate and strengthen the internal connection between ideas that are processed (Kiewra et al., 1991, Friedman, 2014). Piolat, Olive, and Kellogg (2005) assert a high level of diversity in note taking practices, which develop during the various, constrained contexts a person is exposed to. Often times, they are unique or highly individual to the note taker.

Primary Objects

Cognitive Science

Norman (1993, p. 1) counts paper and pencils – among others – as ”physical artifacts that aid cognition“. He asserts the technology of mental and physical artifacts as essential for the growth of knowledge and mental capabilities.

Cognitive science delivers a perspective on the sensemaking cycle mentioned earlier through the distributed cognition framework, which includes the external world in the analysis of cognitive processes. It attributes culture with a meaningful role in cognition and is “moving the boundaries of the unit of cognitive analysis out beyond the skin" (E. Hutchins, 1995a, p. 355). This unit of analysis is termed a cognitive system, which is made up of individuals and artifacts they use (Flor and Hutchins, 1991, E. Hutchins, 1995b, Nardi, 1995). Giere (2006) discusses the roles of agency in distributed cognitive systems.

Applying the distributed cognition framework to the notebook process necessitates investigating the "physical processes that propagate representations across media" (E. Hutchins, 1995b, p. 266). These representations allow the handling of things that are purely imaginary, i.e. abstract concepts, and are thus an important tool for inquisitive minds. Humans construct artifacts as a support for representations in the real world (Norman, 1993).

Philosophy

The epistemological backing of distributed cognition, of expanding the unit of analysis outwards of the brain, is provided by embodied or extended cognition theory. It contests internalism, i.e. that the mind is contained in the brain, and that cognition happens inside it, in isolation (Clark and Chalmers, 1998). In the present case, it extends to the notebook, and the notebook represents an embodiment of the note taker's mind.

Expressive
Receptive

The primary tools can be differentiated by their role: the pen is the expressive component, augmenting the hand and lending permanence to gestures through ink. The paper is the complementary, receptive part, providing a two dimensional surface for articulation, that, besides spatial limitations, is uninhibited. Both impose minimal additional context, since they require no interaction abstractions to work, in contrast to a note taking application employing a digital pen interface.

Media
Theory

A recurring term in the scientific perspectives discussed so far is *media*. Media theory investigates the aforementioned artifacts. Kittler (1993) ascribes an amplifying effect to the improving tools for storing, processing and communicating data. He asserts a logic of escalation inherent to the development of media technology, dependent on this amplification. Most importantly, he argues media's autonomy from man – in his words, media are not pseudopods of the mind (Kittler, 1991). This may be an argument towards a clarification of his definition of media, not a denial of the existence of possibilities for cognition amplification. Still, it is opposed to the concept of the extended mind.

An important distinction between media discussed here and notebooks is the intended goal: while printed books, television or the internet are means of communication of information, the scientific notebooks discussed in this work do not serve a purpose other than a private research tool. Their benefit lies in the process afforded by them. They lead into a publication or a focus of research only sometimes, after the fact, like the notebooks of Paul Dirac (Galison, 2000).

The benefits of a media theoretical look at notebooks are thus limited. But, following the interpretation of a research effort as a cognitive system (see i.e. Giere, 2006), the agents still interface with a wide array of media. Books, memos, and of course the internet play a large role in the acquisition of new information. The researcher consumes a lot of input through these channels and uses his or her research tools to process them.

Secondary Objects

Classified as secondary in a sense of further removed or less personal and direct are information systems, lab instruments, and other sources of information. This is the most generic category in the breakdown of the notebook process, as it contains conceptually diverse elements functioning on different semantic levels.

Although E. Hutchins' (1995) example of a cockpit as a cognitive system is more hands-on, using a notebook in a research context also represents a cognition process utilizing mental artifacts – e.g. mnemonics, language, algorithms – as well as physical ones. Its goal is just more abstract than flying a plane. A person doing research forms a cognitive system with many elements, including literature retrieval systems, prototypes as well as other peers in science. This system's goal is knowledge generation.

The benefit of the distributed cognition perspective on the notebook process is a systemic view that guides attention to the underlying interlocking processes that make up a scientific endeavor. It allows analysis to move away from solely focusing on the individual's mind. Analyzing the roles of the agents and artifacts, we can arrive at propositions for improvements that incorporate additional aspects of the cognitive system under scrutiny.

Product

Products of the note taking process fall in two main categories: Tangible and intangible. The tangible output is written notes on paper. It is content – created, copied, or amalgamated. Its visual structure, as well as sequence on a higher structural level – i.e. pages – and the temporal fragmentation are all qualities relevant for analysis.

Despite the multiplicity of techniques, Piolat et al. have identified three levels of language affected by them:

- Word level. Abbreviating procedures such as end truncation or suffix contraction are used. Sometimes an individual note taker uses several different ways of shortening a word throughout their note.

- Syntax level. Transformed syntax that is adapted to constraints of time or volume occurs often. Statements can be shortened by using substitutive symbols, such as arrows, mathematical operators or others. A telegraphic style may be utilized, too.

- Format level. Physical formatting often times follows a non-linear make up. "...[T]he format of the notes [...] exploits all the physical space of a sheet in a non-linear way." (Piolat et al., 2005, p. 294)

All these methods are well suited for being used with pen and paper (Friedman, 2014). Regarding sequence, non-linear note taking strategies are deemed more effective for learning and retention and the notes themselves are looked upon as external memory (Piolat et al., 2005).

From this educational psychology perspective, a pragmatic view of notes and note taking appears. Putting aside epistemological discussions, the make up of actual written notes can be seen, which are simply viewed as external storage. This trivialization in regards to knowledge delineates the discipline, as discussions on knowledge or extension of mind fall outside the scope.

The intangible products of the notebook process are the various forms of knowledge discussed in section 2.2.1, as well as the ephemeral information as knowledge in action, as Kuhlen (2013) called it. In a more abstract sense, the psychological effects of the process can be considered an intangible product.

		Notebook Process	
Elements	Properties	Requirements	Argument Sources
User	creative intrinsically motivated	support should not limit non-linearity of process	Drucker (1959), Drucker (1999), Piolat, Olive, and Kellogg (2005)
	employs feedback loops	features should enable amplification by following information visualization principles	Ware (2012)
	employs mental artifacts, sometimes specific to discipline	no word-level support	Norman (1993)
	has habits regarding tool use has habits regarding note taking	non-invasive augmentation	Piolat, Olive, and Kellogg (2005) Friedman (2014)
Primary tools: pen	extension of mind, part of cognitive system	free choice necessary	E. Hutchins (1995a), E. Hutchins (1995b), Clark and Chalmers (1998)
	subject to emotional connection	no modifications	
	writing helpful for memorization writing important for expression	retain handwriting in all aspects	Piolat, Olive, and Kellogg (2005), Kiewra et al. (1991), Friedman (2014)
Primary tools: notebook	subject to preferences	free choice necessary	E. Hutchins (1995a), E. Hutchins (1995b), Clark and Chalmers (1998)
	private and sensitive	employ data privacy measures	Clark and Chalmers (1998)
	affords countless interactions	no modifications	Gibson (1977), Norman (1993)
Secondary tools	necessitate media discontinuities	features should reduce discontinuities allow links between analogue and digital media	Signer (2005)
Notes	highly idiosyncratic	no word-level support	Piolat, Olive, and Kellogg (2005), Krauthausen and Nasim (2010)
	highly fragmented in subject matter but sequential temporal layout	structural support through tagging structural support through search	

Table 2.1. – Requirements for technological support of scientific note taking categorized by element concerned.

2.3. Requirements for Digital Support of Note Taking

Table 2.1 categorizes the requirements for technological support of scientific note taking by the element concerned, based on the analysis in the previous section.

2.3.1. Technological Improvement of Note Taking

Amplification is a common theme when it comes to the effects of physical artifacts (Kittler, 1993, Norman, 1993, Ware, 2012). These artifacts advance in sophistication as technology improves. The ubiquitous devices of our time are the computer and its various forms. The interaction between users and these devices is the subject of a wide range of research (Ogunyemi, Lamas, Lárusdóttir, & Loizides, 2019). Brooks (1996, p. 64) declares creating “amplifiers for minds” as the goal of research in human computer interaction (HCI) and with this, fits well into the cognitive science approach. Hollan, Hutchins, and Kirsh (2000) discuss the distributed cognition framework in the HCI context.

Technological advances keep shaping our access to and our interaction with information. Data intensive research tasks now almost always involve digital collection and processing methods. A host of digital notebooks, touchpens and handwriting recognition programs provide note taking functionality that was not available to researchers even 30 years ago. It is possible to search, reference, save images, and collect excerpts, in interactive, cloud based applications.

Even though knowledge workers are supported by those increasingly elaborate tools, they still sometimes shun those means in favor of traditional paper. One reason is that pen an paper offer a large amount of possible interactions (Sellen & Harper, 2003). These interactions are immediate and not impaired by indirections through digital interfaces. They are reliable, predictable and still work when there is no power to charge device batteries. They grant the possibility of individual modes of interaction, because they are not bound by software development constraints and requirements engineering processes. Another difference to digital tools is that writing on paper creates a unique original copy.

Gibson's (1977) theory of affordances provides a scientific frame, where affordances describe actions made possible by an object in relation to the user. Although digital systems can supply and process a lot more data, they cannot yet match the affordances of paper.

Mackay (2003) argued against replacing all paper documents with digital ones, presenting case studies that combine both worlds. The role of writing with analog pens has changed and is now complementary to working with computing devices. Sellen and Harper reported in 2003 that digital alternatives to paper documents were not technically advanced enough to replace those in an office context (Sellen & Harper, 2003). More recent research supports the idea that available alternatives do not fill the same role in the mind of users (Riche et al., 2017). The personal information space of users still contains a considerable amount of physical artifacts, and "we still use paper to an extensive degree" (Trullemans & Signer, 2014, p. 95). People often use digital alongside physical documents and transfer information between them (Hayes, Pierce, & Abowd, 2003; Trullemans & Signer, 2014), which speaks for the need to integrate analog notes with digital devices in the same work space.

2.3.2. Summary

Collecting views from different angles of discipline on the notebook process and cognition in general yield several conclusions: The notebook process amplifies mental capabilities and shapes the cognition of the individual. As it takes up working as part of the individual's mind from a philosophical standpoint, it is highly sensitive and often times private. This individuality of notes is also supported by the advantage of working with one's own notes versus perusing notes of others, as cognitive psychologists Piolat et al. (2005) mention.

The process is messy and frequently imperfect, with incomplete ideas, as notebooks are rarely written with publication in mind. Impositions of technique need to be carefully weighed against the possibility of constricting the manifestation of the individual's mind.

To assess the notebook process in its entirety, relevant artifacts and agents can be considered part of a cognitive system. These parts all play a role in the cognition of the researcher and are to be acknowledged in any approach of process improvement.

A goal is only identifiable in the abstract sense: knowledge production. Ernst Mach, an influential scientist with several major discoveries in the field of physics, put forward the idea that the conception of new knowledge can be conceived as a guided process of finding, aimed at a fortunate mental coincidence (Krauthausen & Nasim, 2010). It is this practice that scientific note taking supports.

3. Qualitative Study of Scientific Note Taking

To gather qualitative evidence of note taking activities in contemporary workflows of scholars, semi-structured interviews with 12 participants (10 female, 2 male, ages 24–31) were conducted and transcribed in the context of the master's thesis by Achmann (2021), supervised by the author[1]. The scientific inquiry of the master's thesis focused on the scholarly workflow as a whole, capturing note taking aspects as a part of it. In this chapter, the results are interpreted regarding the prevalence and characteristics of the practice of note taking among participants as well as the roles physical and digital media play. In combination with the theoretical treatment of the matter in chapter 2, this qualitative evaluation of scientific note taking substantiates requirements for digital support of pen and paper tasks.

3.1. Participants and Recruitment

Participants were recruited through a convenience sampling process from the university campus in Regensburg using social media adverts and personal networks. Five interviewees were actively involved in research for their PhD. Seven recruits

[1] This chapter of the dissertation is based on the research studies conducted by **Michael Achmann** for his master's thesis, which was supervised by the author. Interviews were conducted and coded by Michael Achmann. Study parameters, illustrations, and interview codings are adapted from his thesis.

were studying their subject at the master's level. This level of academic education was set as minimal requirement to ensure a familiarity with scholarly work. One participant's lack of experience with research due to the particularities of educational studies emerged during interviews. For the subsequent analysis, this interview was excluded. Since the subject of scholarly workflows with some focus on note taking was made known to potential participants, there is a possible sample bias towards knowledge workers embracing such techniques.

Still, the diversity of the participants' disciplines helps to identify aspects of scientific note taking that emerge across the boundaries of fields: their PhD and master's fields of study range from history, Slavic linguistics, Eastern European studies, macroeconomics, work and social psychology, to political science, and chemistry.

3.2. Interviews

Both in-person and remote interviews were carried out in German, with each taking about one hour. The interviews were transcribed manually.

The interview guide was split into four parts exploring the participants' research practices in general, note taking habits, their approach to writing down their insights, and how they archive their material. Besides a variety of questions examining those aspects of their workflow, every interview part included questions about perceived possibilities and personal desire for improving the process. The questions regarding note taking are available in the appendix A.2.

The questions about note taking focused on the *how*, *when*, *what*, and *what for*. In this way, knowledge about the interviewees' preference regarding media, the place of note taking in the workflow, the concrete structure and content, as well as the motivation behind it could be gathered. Several participants brought or showed notes they took, both on a laptop screen and on paper. While asking subjects about their process of writing a scientific text, they were also asked if they produced mind maps, sketches, or similar to structure their thoughts. This question was included to cover note taking processes the participants might not have considered as relevant previously.

For analysis, the grounded theory approach by Corbin and Strauss (1990, 2008) was used to perform several coding iterations. As Charmaz (2006, p. 43) puts it: ”[q]ualitative coding, the process of defining what the data are about, is our first analytic step“. This helps to establish a semantic structure by assigning theoretical categories that are *created*, not *preconceived*, to pieces of data, which can then be used towards more general theoretical statements and contextual analyses (Charmaz, 2006).

3.3. Note Taking in Scholarly Workflows

Notably, every participant reported four *universal* activities that were taking place regardless of the workflow stage their research projects were in. These activities were *note taking*, *bibliographic management*, *file management*, and *annotation*. Many of the tasks that are part of these activities were carried out concurrently with other workflow tasks.

3.3.1. Note Types

The interviewees mentioned creating a variety of note types, like todo notes, timelines, and thinking notes during their scholarly activities. The nine reported types of notes listed in table 3.1 demonstrate the diversity of note taking methods in play during academic work. The presence of note types on the task, structure, and insight level show the versatility of note taking techniques in an academic context. The categories were annotated for this dissertation and were chosen based on the focus and origin of the notes’ content: communication, research tasks, structuring work, and insight.

Detailed interview references for the note types can be found in the appendix in table A.1.

3.3.2. Note Taking Process

The participants revealed strong tendencies to shape the process according to their individual needs. S1 mentions that their notes regarding their own work are not meant to be understood by others, but the distinctive nature does not hinder work when revisiting them (P64). For example, structuring elements – e.g. lists – do not consequently use the same notation (P68). S5 also doubts the understandability of their notes for others (P72).

S4 uses color markings in paper documents to identify and structure important parts. The colors chosen do not follow a system until very late in the process of distilling knowledge. Tacit understanding of how they marked the text supplements the work process (P22). Color is mentioned by several interviewees: S7 for example mostly creates notes organized in tables and uses colors to highlight subject areas (P36, P64). S10 identifies the structuring of notes by underlining and using color markings as an important practice (P79).

S10 describes their note taking process as not strictly systematic (P20) and elaborates on its nonlinearity, where bits of knowledge get added at a later date to some notes, connected by a variety of arrows (P22). They take notes with an ever-changing structure of the future scholarly text in mind (P75). S4 also reports that there is no strict scheme to their way of taking notes when working with literature of different kinds (P60).

S5 remarks on spur of the moment preferences and availability when choosing a pen (P94). This implies that quick decisions about the tools happen in the process for them and artificial limitation or interference could interrupt it. S1 highlighted their preference for graph paper because of the ease of structuring notes and dislike for lined and blank paper. The former is not suited for their bullet-point way of noting things, the latter does not lend itself to easy structuring (P74).

The personal character of notes, especially of the way how they are made and used, is emphasized by S2 when talking about the limitations of using a computer for it (P60). S2 distinguishes notes on lectures, which they consider records of ”intersubjective“ matter, comparable to lecture scripts of others. Evidently, there are different mental models for note taking and preparing material with others in mind.

Problems

Part of the interviews focused on problems with the note taking process. S1 describes forgetting their preferred writing pad and having to use blank paper as an issue, impacting structure and legibility for the notes taken under these circumstances (P74). Another difficulty reported by S5 relates to note organization, as they tend to forget where they put the notes on something, especially when having many notes. A unique issue among interviewees was S5's fear of throwing away the paper documents and the resulting archival burden (P108). The distribution of note taking among physical and digital devices lead to a degree of disarray for S8: they almost always create literature excerpts in a word document, but struggle with keeping order to the various files that accrue during different phases of a research project (P54). If a piece of paper is at hand, sometimes literature notes are done by hand (P98). This hints at coordination issues between different modes and media for notes taken. Annoyances with purely digital notes were brought up by S11, who described Word's correction function as too easily agitated and of nagging disposition (P64). S11 showed their notebook during the interview and mentioned that it is coming apart, but didn't seem distressed by that (P50). Another vulnerability in the form of liquid spills is pointed out by S1 (P88).

3.3.3. Reasons for Taking Notes

S4 describes *remembering* as the main reason for taking notes (P60). Similarly, S6 uses notes to self as *short-term reminders* (P28, P72). More permanent, content-related notes are written into books with a pencil most of the time (P28). Digital notes are taken in the form of code comments, regarding specific runs and parameters of experiments and analyses (P86).

When working with literature, S7 writes on a paper pad – Noting things by hand helps them *visualize* the ideas or aspects of interest and their place in the research project and is faster than using a word processor (P36, P38). For concrete tasks, the primary use is to track sources and distinguish own thoughts and literary references, providing helpful *support* when writing insights down formally (P56, P58).

S9 uses notes a lot to *organize* future tasks (P85) and calls their notebook the ”clever book“ and carries it everywhere. While taking conversational notes at meetings with professors and feedback from colleagues, a sequential order of resulting tasks designated by arrow elements is produced (P71). The same holds true when noting ideas and the following next steps for their dissertation, for field studies, and for transcription and annotation of texts (P71).

Even though, as S10 mentions, they type faster than they can write by hand, they enjoy writing things by hand because, in their words, there is a direct connection between hand and brain, helping not only *retention*, but also *refinding* (P20, P29).

S11 expresses that taking notes during lectures helps *focusing* on and *listening* to the talk (P54). The refer to writing by hand during literature work as helpful, even though they would like to improve the handwriting itself (P52).

3.3.4. Digital and Physical Media

Figure 3.1 illustrates the media used by the participants, differentiated by digital and physical means of note taking. Only a single participant (S8) reported solely using software to capture notes, while S11 used only their paper notebook. S9 and S11 exclusively used a paper pad for taking notes. The others mentioned several forms of taking notes, employing both digital and analog processes in their workflow. Single sheets of paper and paper pads were common among the writing surfaces.

Dedicated notebooks were present and S4 and S5 reported using the margins of printed documents as area to write in. S4 extended such documents by adding post-it notes with handwritten statements.

8 out of 10 of the people interviewed that used physical note taking utilized only a single type of paper-based writing product. This can either be down to convenience or preference of specific properties. Additionally, the category *paper* includes a lot of variety. While the type varied within the group that opted for a single type of paper note surface, the mode reported by the five participants that reported use of a single mode of digital note taking was always the word editor. This could imply that the word editor is either the closest relative to paper in the mental model of users, or that it simply is the software most adapted for producing text.

Several interview partners reported that they preferred taking notes with pen and paper in at least some situations. Reasons given for choosing to take notes by hand and not on the PC include freedom of expression, the fact that no suitable digital devices were available to them, and that it is impossible to forget to save when writing on it (S1, P88). S1 also includes the idea that *a paper pad can't crash*, which indicates a lack of trust in the seamless functioning of computer systems as a reason for preference. S2 specifies personal preference for pen and paper because it comes easy to them (P26). Here, the familiarity with the medium has a beneficial influence. Additionally, for them it is harder to forget things (P54) and there is a large perceived effort of reorganizing material should they decide to try out digital note taking. This and the lacking usability on digital devices prevents them from moving their note taking to a digital medium (P60). S5 mentions limited choices for annotations hindering their use of digital alternatives (P80, P87). S11 describes their preference for their paper notebook by the greater mobility and the freedom from "trivialities", e.g. where to plug in a laptop (P54).

The immediate positive effect of striking a todo-item from a physical list, described as a satisfying feeling by S3, is cited as something that can not be replicated on a computer (P80). The matter of better retention in comparison to computer-based notes pops up with S4, who points out that writing the notes and marking up documents by hand helps retention and has more value (P20, P62).

Another perspective on a different type of notes is given by S2, who considered physical mind maps superior because the offer a better bird's eye view, after having tried digital tools for it (P28). This ties in with the affordances of paper, which can be viewed from any perspective and easily manipulated and transported for immediate inspection.

S10 explains switching between media mostly by convenience and availability, leading to a mix of physical and digital notes on the same project (P20). They also mention printing the word document intermittently, to have it at hand while working. This proved inconvenient later on, and S10 struggled with integrating digital notes into their workflow (P43).

3.3.5. Improvement

When asked about possibilities for improvement, some interview partners were critical of their own discipline in organizing or structuring their material. On the subject of combining physical and digital workflows, S2 called the possibility of connecting their handwritten notes with the stored knowledge on their computer using their personal note taking process a dream (P60). S7 would like to search material that they accumulated both physically and digitally and mentions text processor word search as example, but wants to retain the process of writing on a paper pad by hand (P40). S7 struggles with available space on paper and wants to transfer the notes to a PC in the future. They stated that they would be thankful for a function interlinking keywords noted down and the relevant page in a publication (P67).

S1 emphasized that immediacy and no distractions from other software would be necessary requirements for moving their note taking to digital media (P88). A

Note Type	Category	Count of Subjects	Reported by Subject
Conversation notes	1	Communication	S9
Lecture notes	3	Communication	S1, S5, S11
Translation notes	2	Research Task	S1, S11
Literature Notes (Excerpts)	10	Research Task	S1, S2, S4, S5, S6, S7, S8, S9, S10, S11
To-Do lists	4	Structure	S2, S3, S6, S9
Mind Maps	3	Structure	S2, S5, S9
Timelines	1	Structure	S1
Thinking notes	4	Insight	S4, S5, S7, S10
Notes to Self	5	Insight	S1, S2, S3, S4, S6

Table 3.1. – Types of notes the participants reported making during their scholarly work.

low threshold of entry and a very flexible solution to keep up with a changing and personal workflow would be crucial for S2 to consider taking notes digitally (P60).

3.4. Summary

Even though the interviewees were all from a generation where digitalization processes were underway during their youth and the internet was well beyond its infancy, all except one reported using paper notes for their scholarly work. Many participants expressed criticism of their own discipline when it comes to structuring their notes, i.e. creating indices (S11, P52). The desire for structure across media boundaries, coupled with the perceived lack of energy to maintain it, highlights a starting point for technological support of note-taking practices.

While the tone of participants speaking of possibilities to combine physical and digital note taking workflows was enthusiastic, they had trouble imagining a concise way towards practical solutions. Considering the comments about the note taking process, preferences for physical media, and the actual workflow that always included both modalities of note taking, a pen interface needs to be unintrusive and flexible in connecting both worlds, retaining the immediacy of pen and paper interaction. The following conclusions result from the analysis of the interviews in the context of note taking practices, and are limited in that they only draw on the qualitative empirical sample collected for this study.

(1) In scholarly workflows, note taking is a universal activity, taking place during every step of scholarly work.

(2) Note taking is a personal and individually adapted practice the product of which is not meant to be understood by others.

(3) Note taking mostly happens across digital and physical work spaces, with users employing two or more media to deal with their needs.

(4) There is a desire for connecting physical and digital notes and information scraps among the interviewed scholars.

(5) Interviewees tried enhancing their process with digital tools but returned to their process because of a perceived lack of flexibility or mobility.

(6) Participants desire more structure, but are not willing or able to expend the energy to consistently manage it themselves.

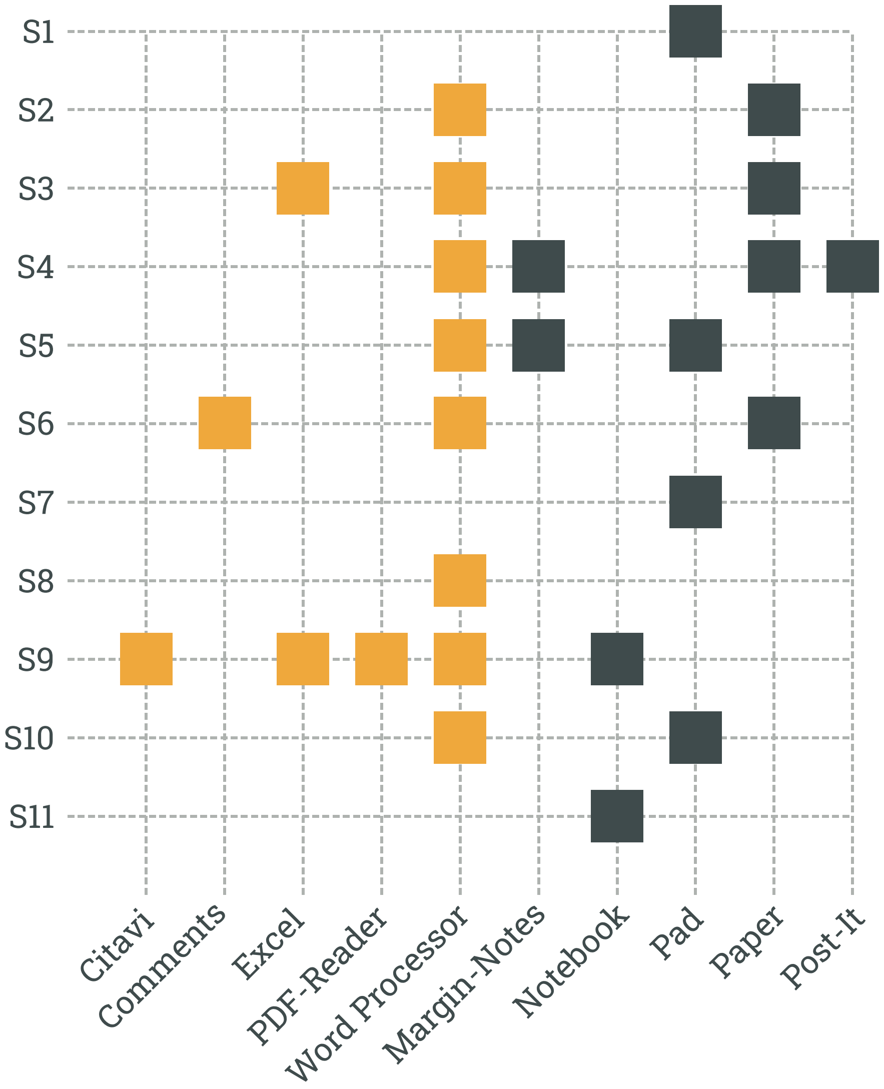


Figure 3.1. – Note taking media distribution among participants of the scholarly workflow study. Orange items describe digital media and dark gray items indicate the use of physical media. Illustration adapted from Achmann (2021).

4. Towards a Ubiquitous Pen Interface

This chapter reviews previous work on pen interfaces, consolidates the research of previous chapters, and proposes a ubiquitous pen interface concept and delineates the research concerns, setting out the exploration of building blocks in the following chapters.

4.1. Previous Pen-based Interfaces

In the literature that deals with integrating pen and paper workflows into digital environments, the focus of research projects sometimes lies on *pens*, and at other times lies on the writing surface, mostly *paper*. The modes and intricacies of interconnecting both physical and digital worlds also play a role. As both primary objects always matter in some capacity, in the following, both pen- and paper-focused systems are reviewed.

Early Systems

An early paper-based interface was the *DigitalDesk* system by Wellner (1993). The motivation as it was back then is still valid today in that ”choosing to interact with a document in one world means forgoing the advantages of the other“ (p. 87), albeit to a different extent. The *DigitalDesk* augmented a physical desk by projecting a computer-generated image on top of it, while still allowing interaction with physical pens and documents. In its infancy, paper-based computing clashed with the severe technological limitations, but nonetheless Wellner produced a prototype capable of tracking fingers, capturing images, and performing OCR. He concluded that some technological issues need addressing: image resolution, finger tracking accuracy, and adaptive thresholding, i.e. image segmentation. The spirit of Wellner's contribution motivates this dissertation as a design science research project iterating and producing prototypes for innovative solutions to known problems. The issues highlighted in his work still exist, but can now be addressed with a wholly different arsenal of technological and algorithmic tools. How these novel methods can help bridge the digital-physical gap is the concern of the technology-focused part of this dissertation.

Another system from the same year is *XAX*, which cited the affordances of paper as motivation for creating a *paper user interface* (Johnson, Jellinek, Klotz, Rao, & Card, 1993). Their system required users to scan documents that were prepared with additional marks and command items indicating actions to be performed on subsequent pages as a form of paper-driven batch processing. Notably, it already included a WYSIWYG editor for the paper command interfaces and implemented distributed processing with a server program that handled incoming fax documents. With *XAX*, the focus solely lay on expanding paper with a kind of symbol-based markup creating a landscape of command possibilities configured with an ordinary pen.

Using a type of modular paper notebook, Heiner, Hudson, and Tanaka (1999) created *PaperPDA*, focusing on tasks like e-mailing from paper entries and paper-to-paper linking that produces digital connections when the notebook is digitized. They used printed symbol-based command structures that were processed when the notebook was scanned into a digital system, reminiscent of *XAX*. Pages with a distinct layout related to an application – like e-mail – could be inserted and removed from the modular notebook at will.

This kind of *offline* operation in paper interfaces represents a line of research that started with *XAX*. Another vein of inquiry focused on more interactivity while 90s AR exploring the limits of technological feasibility. Building on the AR based concept of paper interaction advanced by Wellner (1993), Mackay et al. (1995) introduced *Ariel*, an augmented engineering drawing board, based on a large size graphics tablet and a projector. In user studies, they found that most engineers at that time referred to their personal, annotated hardcopies of engineering drawings, instead of the digital versions available to them. Mismatch between paper and digital versions was cited as a major issue, which informed the design of *Ariel*. Mackay et al. (1995) used barcodes on paper to identify individual plans and integrated a red light pointer, as well as LEDs on a paper clip for interaction.

Mackay, Pothier, Letondal, Bøegh, and Sørensen (2002) developed an augmented laboratory notebook prototype for research biologists. They explored several prototype proposals, arriving at a system that uses a personal digital assistant (PDA) as an "interaction lens" for paper notes and a graphics tablet to capture pen strokes on the paper. This *a-book* linked physical and digital documents in a more elaborate way than previous work, enabling immediate interactive possibilities like selecting text and linking external physical objects through the interaction lens PDA. Mackay (2003) looked back on several such research projects combining physical and digital affordances, summarizing that instead of transferring interaction into a virtual space, the aim is to "create systems that allow people to interact with the real world in natural ways and at the same time, benefit from enhanced capabilities from the computer" (p. 7).

Smart Pens

Following a different path, Guimbretière (2003) explored the *cohabitation* concept – that physical paper documents and digital documents are two different approaches of interacting with the information that can coexist. This cohabitation can be supported by implementing a way of transferring analog notes to digital documents. Their *PADD* employed a smart pen that collected strokes and implemented a system that matched annotations to digital documents. This approach is the pen counterpart to *offline* paper interaction as was presented in PaperPDA and XAX, in that commands are collected in the pen, not the paper. Both variants introduce the necessity to make the primary note taking object ”smart“ somehow.

Several systems were proposed in more recent years to combine the positive aspects of both physical and digital pen and paper interaction, but as Brandl, Richter, and Haller (2010, p. 600) put it, ”it is still a challenge for developers to reproduce or even exceed the simplicity of taking notes on paper with a digital system“. They conducted a study of note-taking habits and identified categories of notes to inform their digital-analog notebook design. Their prototype *NiCEBook* lets users take notes with a smart pen and categorize them according to pre-defined and custom topics.

Portability is one of the main arguments of systems using smart pens. Several researchers presented systems working with them – an early example is *PapierCraft*, first presented in 2005, and subsequently developed and evaluated across a number of years (Liao, Guimbretière, & Hinckley, 2005; Liao, Guimbretière, & Loeckenhoff, 2006; Liao, Guimbretière, Hinckley, & Hollan, 2008; Liao & Guimbretièere, 2012). They integrated a gesture-based command system focusing on hyperlinking, copying/pasting, and creating collages from physical documents. The system operated by collecting strokes offline with the pen that were later performed when synchronizing with a daemon server to edit the digital document. Their later studies with improved feedback mechanisms showed that after some training time, users achieved similar task performance as tablet pc users.

Other smart pen applications, like *CoScribe*, support collaboration and linking physical and digital documents, as well as tagging both types with an *Anoto* pen (Steimle, Brdiczka, & Muhlhauser, 2009). Garcia, Tsandilas, Agon, and Mackay (2014) developed *PaperComposer*, which gives users with smartpens the opportunity to personalize paper interfaces for music composition. Klamka and Dachselt (2017) presented *IllumiPaper*, offering visual feedback directly on modified paper. Here, both the paper and the pen are technological artifacts. Users with a digital pen could interact by making marks and the system responded by illuminating parts of the document.

There are several smart pens currently in and out of production. Since a smart pen is only compatible with the kind of dot paper that the producing company decides (“Livescribe Dot Paper,” 2020), company support is paramount. The models are expensive and are usually priced above 100$.

Recent Studies

There is still research interest regarding the foundations of physical-digital pen interfaces. Riche et al. (2017) presented a comprehensive survey of analog and digital pen use, establishing a collection of affordances unique to pens, paper, as well as digital support. Talkad Sukumar, Liu, and Metoyer (2018) confirmed earlier findings in a study regarding commonly used text gestures when editing or annotating. They elicited gestures that resemble commonly used actions when annotating analog text. Tian et al. (2013) carried out interviews and implemented prototypes using a drawing tablet to evaluate pen tail gestures to switch interaction modes. Users preferred them over traditional mode-switching techniques like turning a pen on its head. Söderström, Hellgren, and Mejtoft (2019) conducted a comparative user study between electronic ink displays and tablets regarding their capabilities for drawing and note taking, concluding that a paper replacement by electronic ink is far away.

Mixed reality workspaces containing augmented paper documents and augmented whiteboards were investigated by Z. Li, Annett, Hinckley, Singh, and Wigdor (2019). Their prototype allowed participants to write text on paper with a smart pen and start a search in the document they were viewing. Users found this very useful and wished for more such functionality. The implementation of pen tracking was based on dedicated paper scraps that were marked up for processing, similar to early paper interface approaches.

4.2. Requirement Consolidation

While the qualitative empirical sample from chapter 3 alone is not able to support reasoning for general guidelines, the combination of literature-based modeling in chapter 2, the qualitative study, and literature review can support a concept for a ubiquitous pen interface.

Previous systems, designs and prototypes provided promising insights into combining analog and digital notes. They often either left computing-related traces on the writing surface, or enforced using a certain type of pen. Both requirements are in conflict with the conclusions from chapters 2 and 3 in the context of note taking. Only being able to write on dotted paper with a smart pen is not conducive to the note taking process. The findings of previous chapters show that combining the affordances of physical and digital systems when it comes to note taking requires respecting the idiosyncratic process of users (section 3.4 (2)). Limiting free choice of materials hinders this, as was deducted in theory (table 2.1, *primary tools*), and gathered from interviews.

A possible support system needs to be available in several workflow contexts, i.e. needs to be mobile and flexible, since note taking is a universal activity (section 3.4 (1), (5)). Ideally, the interface system is ubiquitous, in the same manner as smartphones are. The non-linearity of the process was mentioned by interviewees and the concept additionally came up in literature review (table 2.1, *user*). Not only the work with documents that have a physical and a digital representation should be taken into account. Scholars often work with freeform notes, annotate, and sketch. They use physical and digital notes alongside each other, often for different purposes (section 3.4 (3) and (4)). It should aim to enable structural support, i.e. digitize text and make it available for processing (table 2.1, *notes* and section 3.4 (6)).

4.3. Building Blocks: Towards a Ubiquitous Pen Interface

Johnson et al. (1993) distinguish between pen-based computing and paper-based computing with the former relating to tablet PCs and similar devices. These terms are sometimes used interchangeably in other works, in that paper computing involves pen-based interaction or pen-based interfaces require paper to function. For this dissertation, the term *ubiquitous pen interface* is meant to signify using ordinary pens on a variety of non-digital surfaces, i.e. surfaces that have not been specially adapted for the use with digital pens.

Pen and paper interface research looks back on a long history of developing prototypes of various stages of maturity. There seems to be a common thread of employing design science research methods, even if not explicitly mentioned as such. Every published prototype and technological system approach contributed its part. The systems development research process applied in this dissertation follows this path of inquiry and guides the creation of prototype systems based on the conceptual framework developed in the previous chapters.

4.3.1. Prototype

The development of a prototype instantiation of a pen interface founded on the model developed in chapter 2 and the consolidated requirements poses several challenges. The support of pen and paper users by providing seamlessly integrated access to helpful information systems calls for a natural interaction metaphor, as argued in chapter 2.

The degrees of freedom of the input tools operated by the user far surpass those of traditional input devices, like computer mice or keyboards. For an ordinary pen to become an input device to a computer system, it needs to be tracked and its movement translated into discrete actions, which, once identified, trigger a system response.

Preconditions compiled from the application context, e.g. not modifying the pen itself, constrain the problem space. This translation of actions in the user's surroundings into computer-readable information is a key challenge. The primary objects – pen and surface – represent passive input devices that need to be tracked in a contactless way. The paper content constitutes the tangible product of the notebook process, which can be inspected for information gathering purposes and to supplement tracking.

The proposed interface ties into existing research as it imports analog pen strokes for digital processing. The novelty lies in using an ordinary pen as well as paper or other writing surfaces without any markup. Interaction is enabled through gesture-command recognition that could enable features like search of written text, annotation, as well as cross-referencing resources using any pen. The components of such an interface implementation are schematically drawn in figure 4.2. In this illustration, the abstract demands of untethered interaction are translated into concrete technology-based research tasks.

4.3.2. Missing Links

This dissertation can not deliver all components necessary for a fully integrated ubiquitous pen interface. It opts to establish a baseline for the *missing links* that connect ordinary pens and writing surfaces to digital systems. Those lie in tracking the pen and strokes without any internal fixture in a mobile and flexible way. While offline handwriting recognition, i.e. recognition of handwritten text scanned from a document, still eludes end-user applications, online recognition is implemented in operating systems like *Windows 10* and note taking applications like *Microsoft OneNote* (Keysers, Deselaers, Rowley, Wang, & Carbune, 2017; Microsoft, 2020). Because the stroke movements and sequences of isolated samples are available for processing, this recognition approach is more accurate (Priya, Mishra, Raj, Mandal, & Datta, 2016). Text entry through standard *Windows Vista* online handwriting recognition was shown to be more fun, about as error-prone as using a software

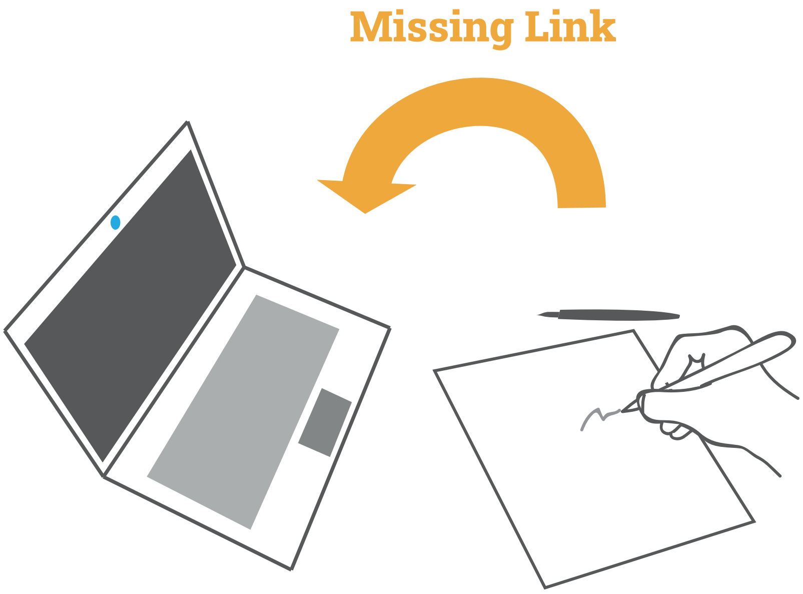


Figure 4.1. – The missing link addressed in the technological part of this dissertation concerns the translation of physical ink strokes with ordinary pens on non-enhanced surfaces into digital pen paths.

keyboard with an error rate of 1.4%, and just as fast in a study where participants wrote on tablet PC devices for hours by Kristensson and Denby (2009).

The research project aims to translate analog strokes into digital traces that can be input into the high-performance systems available for online recognition, illustrated in figure 4.1. Such a system enables more than the decoding of handwritten items – it potentially unlocks support of note taking activities through an interactive computer system. Still, more missing links exist, like the tracking of individual documents without modifying them, and the handling of occluded areas of writing.

The building blocks that are filled orange in figure 4.2 are the ones realized throughout this thesis. The dashed line around the cloud components indicates that not all aspects of it are implemented. Grayed out blocks are out of scope for this dissertation – the application-specific elements are distinct from the quest for missing link building blocks. The surface pose in the illustration is a basic computer vision problem of unskewing a flat surface from a known camera angle and is thus not considered further. The orange arrows signify that the connection of the components is also part of the technological focus. More explicitly, the following aspects are visited:

(1) Detection of keypoints

(2) Extraction of handwritten text segments

(3) Pen state classification

(4) Gesture recognition

(5) Component interface

(6) Parallelization

(7) Integration of remote computation

4.3.3. Camera-Based Operation

Tracking the primary objects means extracting samples of their pose information over time from whatever measurement device is employed. Using those samples, gesture and ink information can then be produced through applying the transformation methods mentioned later in this chapter to the data points. Inspecting the paper content and tracking its changes demands sampling the state of the ink on the writing surface.

Optical Tracking

The tracking concepts can be implemented using a wide variety of technologies. Following the preconditions set out in table 2.1, any tracking technology should be contactless to avoid modifying the primary objects of the note taking process. Optical methods allow capturing high frequency samples without integrating additional

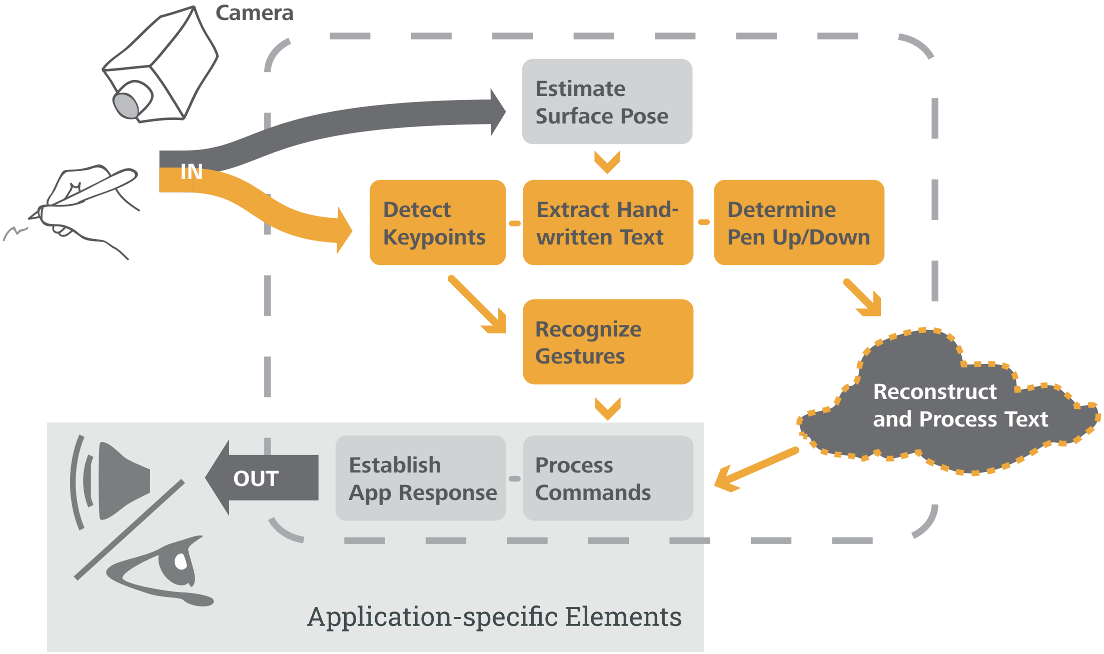


Figure 4.2. – The make-up of the proposed pen interface. The realization of the elements colored yellow is investigated in this dissertation.

sensors into those objects. A cheap and ubiquitous possibility are color cameras, with current consumer models delivering between 20 and 60 frames per second (FPS) at high definition (HD) resolution and above.

Mackay et al. (2002) rejected the idea of using a camera for their portable pen interface, citing environmental interference and lack of mobility. The technological preconditions have now changed – mobile cameras are everywhere, and current image processing methods are able to function under highly variable conditions for certain applications (Pouyanfar et al., 2018).

By employing cameras for optical tracking, the pen interface concept can be used for a range of applications from desktop situations where a laptop webcam captures the writer to AR systems providing fully a integrated digital-analog interface. Basing the pen interface on camera technology keeps it flexible and modular – suited for adaptation to other cases. It also means mainly employing image processing techniques to gather the information needed from the visual data.

4.3.4. Pen Tracking

Several approaches in recent years focused on tracking pens without relying on integrated electronics. P.-C. Wu et al. (2017) developed a system for marker-based 6-degrees of freedom (DOF) tracking using a single camera. They tackle the ROI detection problem by using several markers mounted on a passive fiducial visible from most perspectives. They estimate the pen pose with high accuracy by attaching the fiducial to a stylus and tracking it at 50 Hz. In contrast, the proposed system is based on the premise that users won't need to modify their writing tools to use it, foregoing the use of markers.

End-to-end video tracking and text recognition systems have evolved in accuracy and sophistication. Seok, Levasseur, Kim, and Kim (2008) track a pen tip of predefined shape and color on a printed document background to allow digital-analog annotations. They achieve good ink reconstruction accuracy in their tests, but caution of lighting dependencies. The system proposed by Kim, Chiu, and Oda (2017) requires manual pen tip marking in a video to track a single, dedicated pen using a convolutional neural network. They employ a recurrent neural network for pen up/down classification. The system reproduces ink paths with high accuracy using a high speed camera. They demonstrate reliable handwriting recognition of their reconstructed pen paths using a commercial on-line handwriting solution. However, the system tracked a unique digital pen from a top-down perspective. The approach taken in this research project, while not offering handwriting recognition, traces paths of many different pens from a wide range of camera angles. Additionally, the goal is to use consumer-level cameras, which can be found in popular smartphones.

In summary, regarding technological implementation, the question is: ”How far can one get using imperfect hardware and state-of-the-art software towards the goal of creating a ubiquitous pen interface? “

5. Machine Learning Models for Pen Interfaces

For some tasks that humans perform well, ”the introspection concerning how [they] do them is not sufficiently elaborate to extract a well defined program“ (Shalev-Shwartz & Ben-David, 2014, pp. 21–22). Shalev-Shwartz and Ben-David argue that in those cases, an extracted program would be too rigid to adapt under variable conditions – machine learning approaches are better suited. Very early in the history of computing, Minsky already stated that ”Pattern-Recognition, together with Learning, can be used to exploit generalizations based on accumulated experience“ (Minsky, 1961, p. 8).

Goodfellow, Bengio, and Courville (2016) list several problem categories best approached with *deep learning* models that are capable of learning from data: object recognition, speech recognition, and machine translation, among others. Since tracking pens and ink relies on object recognition at a base level and classification of pen states is dependent on complex sequence analysis, this dissertation focuses on investigating deep learning models for these tasks.

deep learning (DL) systems are able to automatically extract useful abstract feature representations, distinguishing them from other machine learning approaches like rule-based systems and classic machine learning, as is shown in fig. 5.1 (Goodfellow et al., 2016, p. 4). DL systems use distributed parallel processing methods called artificial neural networks, where ”[t]he central idea is to extract linear combinations of the inputs as derived features, and then model the target as a nonlinear function

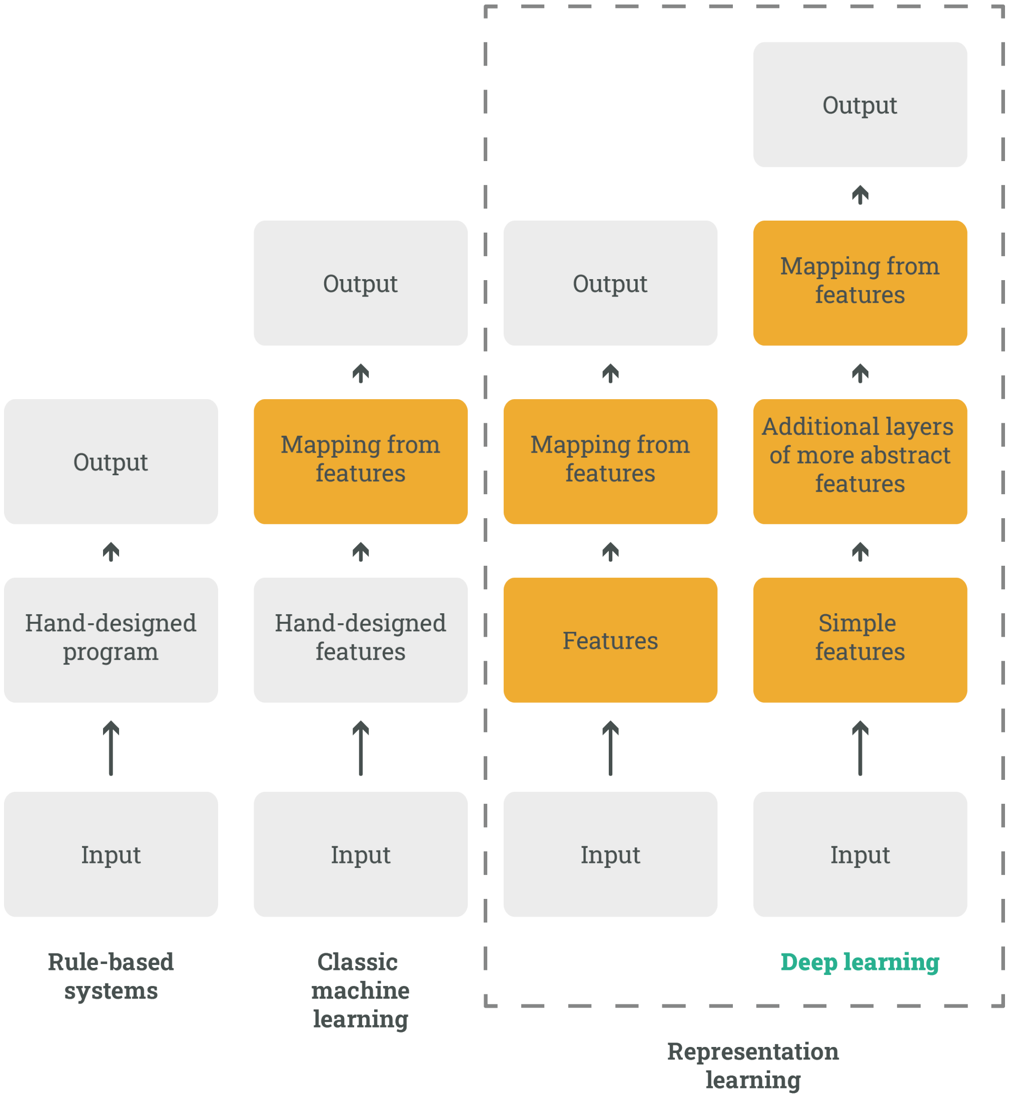


Figure 5.1. – Different types of AI systems can be distinguished by how much of the system can learn from data – here, colored boxes designate parts that are learning-based (Illustration after Goodfellow, Bengio, and Courville, 2016, p. 4). This dissertation is concerned with representation learning systems performing *deep learning* of abstract feature representations.

of these features“ (Hastie, Tibshirani, & Friedman, 2009, p. 389). Deeper networks are beneficial when complex functions need to be modeled, since ”functions that can be compactly represented by a depth k architecture might require an exponential number of computational elements to be represented by a depth $k-1$ architecture“ (Bengio, 2009, p. 14).

Learning-based systems have been around for a long time, but have been mostly limited to shallow architectures before advances made training deeper architectures possible (Bengio, 2009). Current neural network machine learning methods for image processing allow high throughput and low latency (J. Huang et al., 2017). Their capabilities have improved tremendously in the last decade, which, according to Szegedy et al. (2015), is not only owed to more powerful hardware, larger datasets and bigger models, but to new ideas and algorithms as well.

5.1. Learning

In general, machine learning techniques play an important role in classification, regression, denoising, transcription, translation, synthesis, imputation of missing values, anomaly detection, and other tasks (Goodfellow et al., 2016, pp. 98–101). Deep neural networks have produced state-of-the-art results with applications in natural language processing, image processing, and speech and audio processing using a variety of training paradigms (Pouyanfar et al., 2018).

Shalev-Shwartz and Ben-David (2014) describe learning as a process of ”using experience to gain expertise“, characterized by the nature of interaction between learner and environment. This interaction is determined by how a task like e.g. object recognition is formulated as a learning problem for an algorithm to solve.

There are three basic categories of learning approaches, which Shalev-Shwartz and Ben-David delineate as follows:

Supervised learning specifies an approach where the desired output – the experience – is presented as a training example containing significant information, like class labels. With accumulated experience, the learning system should then predict missing information for unseen data. Examples include object recognition (He, Zhang, Ren, & Sun, 2016) and video action recognition (Feichtenhofer, Fan, Malik, & He, 2019).

Unsupervised learning describes the case where ”[t]he learner processes input data with the goal of coming up with some summary, or compressed version of that data“, a typical example being the clustering of a data set (Shalev-Shwartz & Ben-David, 2014, p. 23). Here, there is no distinction between training and unseen or test data. Goodfellow et al. (2016, p. 103) expand on this in the case of deep learning, where the goal is to ”learn the entire probability distribution that generated a dataset“ and is often used for tasks like synthesis.

Reinforcement learning algorithms do not just experience a fixed dataset, but interact with an environment, i.e. they work with a feedback loop between system and experiences (Goodfellow et al., 2016, p. 104).

In their recent survey, Pouyanfar et al. (2018) stated that while there is a growing body of research in unsupervised methods, the majority of existing deep learning approaches are supervised in nature, especially most CNNs (Khan, Sohail, Zahoora, & Saeed Qureshi, 2020). The development of neural networks for this dissertation will focus on supervised models for image processing.

A neural network trained with supervised learning performs either regression analysis, predicting real-valued numbers, or classification, which is a special case of regression where the target variable is categorical (Hastie et al., 2009, p. 392). Using the example of object recognition, a system reads the pixel values of an image as input variables to arrive at a certain value identifying the object category. If the system is tasked with object localization, i.e. by finding appropriate bounding box coordinates, it needs to perform regression for those coordinates.

5.2. A Brief History of Artificial Neural Networks

This section briefly outlines the genesis of artificial neural networks. For an extensive treatment of neural network research history, refer to Schmidhuber (2015). In the following, the quest for ever better neural network models is treated isolated from other machine learning developments. Several periods of time were deemed an ”AI winter“ (Crevier, 1993, p. 203), describing years where funds were cut and machine learning research programs ended (Crevier, 1993). These *winters* do not necessarily coincide with relevant scientific publications regarding neural networks, which represent only one of many fields in AI research. Documenting the development of general artificial intelligence (AI) is out of scope for this dissertation.

The first historical wave of artificial neural nets research in the 1940s–1960s can be broadly subsumed under the umbrella of *cybernetics* research (Goodfellow et al., 2016, pp. 12–13). Beginning in the 1940s, researchers from several disciplines developed models of how neurons operate with increasing detail. McCulloch and Pitts (1943) linked neural activity with turing computability for the first time and proposed a thresholded neural cell model capable of producing binary signals. Hebb (1949) formulated the first unsupervised learning rule regarding self-amplification of neuron ”weights“, which proved fundamental in neurocomputing research (Schmidhuber, 2015). 1940s

With the *perceptron*, Rosenblatt (1958) was first to present a neuron model for supervised learning, i.e. for the case where desired output values corresponding to certain input configurations are used to train a system (Haykin, 2009). His model had synaptic weights and bias, and was built around the McCulloch-Pitts model (Rosenblatt, 1958). An example illustration of this model can be seen in figure 5.3. Its goal was to separate an m-dimensional input space into two linearly separable classes (Haykin, 2009). While the bias served to shift the decision boundary away from the origin, the weights were used to linearly combine the m input variables (Haykin, 2009). The sum of both bias and linear combination then were put through 1950s

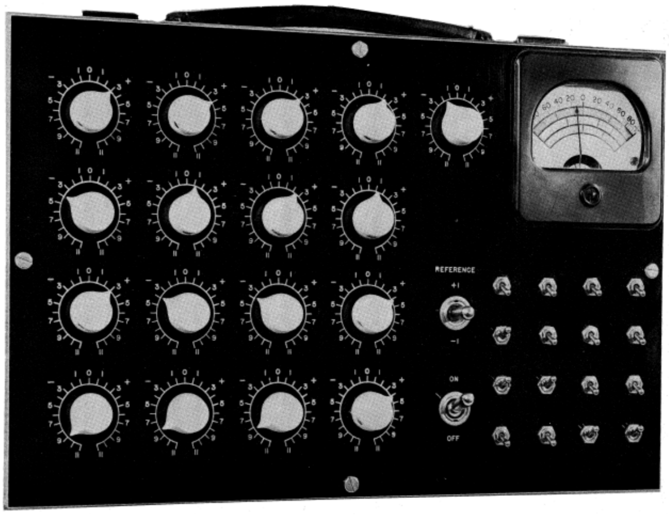


Figure 5.2. – ADALINE: One of the early artificial neural network machines (Widrow, 1960).

a hard limiter performing the signum function (Rosenblatt, 1958; Haykin, 2009). The weights for a specific problem could be found by iterative adaption with an error-correction algorithm, for which the *perceptron convergence theorem* offers proof of convergence in the case of linearly separable classes (Haykin, 2009).

Early in the 1960s, supervised learning systems were created in hardware. An adaptive pattern classification machine called *ADALINE* (see fig. 5.2) was built by Widrow and Hoff (1960) to illustrate adaptive behavior and could be used to distinguish between original and noisy versions of input patterns. Like the input for the perceptron, the classes needed to be linearly separable, but it used a novel least-mean-squares learning algorithm for supervised training (Widrow & Hoff, 1960). A deeper variant of this system – *MADALINE* – where neurons were organized in several layers, could perform practical tasks like echo reduction (Widrow, 1962). According to Schmidhuber (2015, p. 10), this was the decade that saw ”the first example of open-ended, hierarchical representation learning in NNs“. In his survey, he explains that those networks, which were trained with the group method of data handling (GMDH) (Ivakhnenko & Lapa, 1965; Ivakhnenko, Lapa, & McDonough, 1960s

1967; Ivakhnenko, 1968), were perhaps the first deeper feed-forward multilayer perceptron-type systems.

Minsky wrote dismissively of perceptrons in 1961, stating that ”these nets, with their simple, randomly generated, connections can probably never achieve recognition of such patterns as ’the class of figures having two separated parts’“ (Minsky, 1961, p. 15). Minsky and Papert later proved the severe limitations of perceptrons in their 1969 book ”*Perceptrons*“ (Minsky & Papert, 1969). In a later edition, looking back, they contrasted a widely held opinion that their book had interrupted research on learning in network machines for years – they argued that the field lacked basic theories and analyses were looking in the wrong direction when researchers couldn’t explain why perceptrons could recognize certain kinds of patterns and not others (Minsky & Papert, 1988). This interpretation was contested by e.g. Block (1970) and Macukow (2016), who stated that Minsky and Papert purposefully chose limited perceptron variants to argue against, when more powerful concepts were already available.

Rosenblatt died in 1971. Crevier (1993, p. 107) noted that ”[h]aving lost its most convincing promoter, neural-network research entered an eclipse that lasted fifteen years“. In this eclipse fell the discovery of the *backpropagation* algorithm by Werbos (Hinton, 1992), which allowed efficient training of multi-layer perceptrons with hidden units that could do what Minsky and Papert had proven ordinary perceptrons could not (Crevier, 1993, pp. 214–215). Crevier suspected the lack of impact had to do with the general disinterest in neural network research. Another exception was the *Cognitron*, a self-organizing cascaded multi-layer design that was claimed to be similar to an animal brain in many points (Fukushima, 1975). 1970s

The term *connectionism* or *neural networks* can be used for the second wave of research in the 1980s–1990s (Goodfellow et al., 2016, pp. 12–13). Towards the 1980s scientific interest was rejuvenated when new ideas and algorithms appeared and computational power increased, according to Wasserman and Schwartz (1987). 1980s

The *Neocognitron* was one of the first deep artificial neural networks (ANNs) and introduced networks with convolutional units, which today are often called CNNs or ConvNets (Fukushima & Miyake, 1982; Schmidhuber, 2015). A neurobiology-based type of neural network, the *Hopfield* network, was conceived during this time. Hopfield (1982) devised a network topology containing feedback paths to produce a content-addressable memory. Other new developments according to Crevier (1993) were neural networks for speech synthesis (Sejnowski & Rosenberg, 1987) and systems that learned to play backgammon (Tesauro & Sejnowski, 1989). Jordan (1986) published a theory on recurrent neural networks and investigated them in the context of coarticulation in speech production. One particularly important contribution during those years was the popularization of the *backpropagation* algorithm by Rumelhart, Hinton, and Williams (1986). Werbos (1990) later provided a holistic perspective and notable prior work on the algorithm which had been applied by him since 1974 and had been discovered several times before and since. In contrast to the Rosenblatt perceptron, which used a signum output function, backpropagation requires the output function to be differentiable. Rumelhart et al. (1986) used a sigmoid activation function instead. *Backpropagation* was a major step towards deep neural networks, because the ability to extract new feature representations in hidden layers with relative computational ease lent artificial neural networks new expressive capabilities. Applied to problems like handwritten digit recognition, new concepts like the CNN coupled with backpropagation learning delivered impressive results (LeCun et al., 1989). Schmidhuber (2015, p. 12) highlighted a problem that emerged applying backpropagation at the time, stating that ”although [backpropagation] allows for deep problems in principle, it seemed to work only for shallow problems“. The difficulties with training deep neural nets stayed a recurring theme across the decades, with the exception of CNNs, the reasons not being formally clear (Bengio, 2009).

Early on in the 1990s, Hochreiter (1991) investigated those problems occurring while training recurrent networks, i.e. networks with cyclical paths, with backpropagation. In his diploma thesis, he formally identified the problems of *vanishing* or *exploding gradients* in backpropagation, which ”represented a milestone of explicit DL[deep learning] research“ (Schmidhuber, 2015, p. 16). In this decade, the conquest of new areas of application followed – neural networks gained attention in diverse fields, from manufacturing (Burke & Rangwala, 1991), finance and investing (Trippi & Turban, 1992), to forest resource management (Peng & Wen, 1999) and numerous others. Hochreiter and Schmidhuber (1997) developed a neuron cell type with memory, the LSTM cell, that is in use for sequence modeling tasks like natural language processing to this day (Goodfellow et al., 2016, p. 17). Broad research impetus paired with commercially deployed systems, e.g. one by LeCun, Bottou, Bengio, and Haffner (1998) reading several million bank cheques a day, created a second boom of neural network technology in the 1990s (Crevier, 1993, p. 216). However, the problems identified by Hochreiter remained unsolved, and many tasks that would require deeper networks stayed out of reach (Schmidhuber, 2015). 1990s

Around 2000, non-neural machine learning methods like *Support Vector Machines* were still dominating many practical and commercial pattern recognition applications, although some good results of not-so-deep neural nets were reported (Schmidhuber, 2015, p. 21). Training deeper, more capable networks was still wrought with difficulties (Schmidhuber, 2015). The end of this lull in enthusiasm was marked by several 2006 publications, with the expression *Deep Learning* being coined around that time (Schmidhuber, 2015, p. 21). Hinton, Osindero, and Teh (2006) proposed an architecture called *Deep Belief Network* and a learning algorithm for deep nets that combined sequential training of simpler models, both unsupervised and supervised. Hinton et al. (2006) used their algorithm to train a deep neural network that outperformed the best discriminative learning algorithms in handwritten digit recognition at that time. A new record in handwritten digit recognition was reported 2000s in the same year by Ranzato, Poultney, Chopra, and LeCun (2006) using a CNN trained with backpropagation and an augmented training set. Ranzato et al. (2006) combated the difficulties when performing gradient descent during backpropagation by using a method inspired by Hinton et al. to pre-train lower network layers. The computing power of graphics processing units (GPUs) was increasingly used to train feed forward neural networks (FFNNs) and CNNs around the middle of the 2000s as well (Schmidhuber, 2015). Large datasets were accessible online and with *ImageNet* came the object recognition challenge that would soon be solely dominated by neural networks (Deng et al., 2009).

In the past years, advances in *deep learning* were brought upon by a combination of new ideas, computing power, ever deeper neural networks, and huge, readily available datasets (Szegedy et al., 2015). Currently, ”Machine-learning technology powers many aspects of modern society“ (LeCun, Bengio, & Hinton, 2015, p. 436). Krizhevsky, Sutskever, and Hinton (2012) created the first modern deep CNN using several techniques to improve generalization and reduce overfitting and harnessed the computational power of GPUs to successfully train millions of parameters. Neuroscience has taken a back-seat role in deep learning research, because scientists realized ”that we simply do not have enough information about the brain to use it as a guide“ (Goodfellow et al., 2016, p. 15). 2010s

Cho et al. (2014) developed a novel hidden unit – the GRU cell – that was simpler to compute and implement than the LSTM unit and used it in a recurrent neural network (RNN) model for statistical machine translation. A documentation of the current state-of-art in neural networks follows in the next sections.

5.3. Types of Deep ANN Architectures

The expressive capability of deep ANN architectures, i.e. their ability to learn complex nonlinear relationships between input and output, is based on their multi-layer

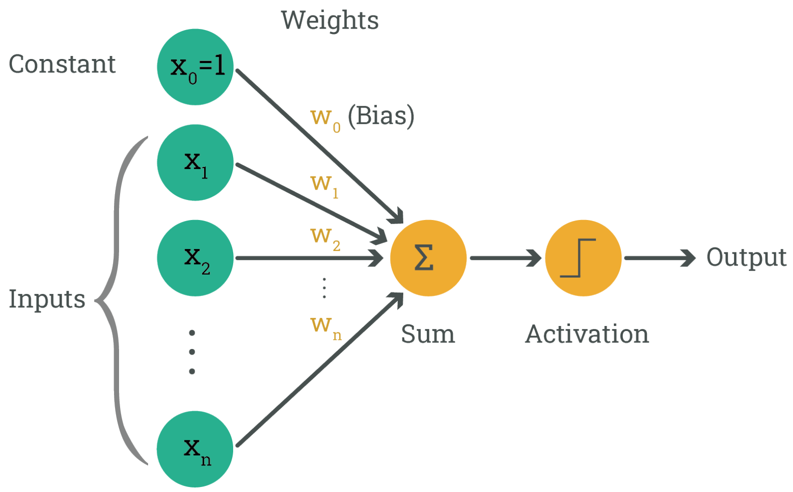


Figure 5.3. – The *perceptron* neural cell model with inputs $x_0 \ldots x_n$ and weights $w_0 \ldots w_n$. Depicted here is the form with bias as a weighted constant input and a step activation function.

approach of learning increasingly abstract feature representations from data (Goodfellow et al., 2016, p. 4). While all *deep learning* approaches have this property in common, how they learn features for different tasks varies. Artificial neural networks can be distinguished by their topology, training approaches, cell types, activation functions, and by the problem class they solve. Today's architectures often combine elements from several areas of neural network research. Additionally, on some tasks, considerable energy is spent beyond neural network inference, e.g. Z. Cao, Simon, Wei, and Sheikh (2017) use post-processing and bipartite graph matching to eventually estimate human poses. This section provides an overview of the main strands of neural network approaches, documenting currently relevant archetypes, which inform hybrid real-world designs.

5.3.1. Feed-Forward Neural Networks

Basic distinguishing factors between neural network topologies are the connection patterns between cells and if outputs at some stage are routed back (Altenberger & Lenz, 2018). Feed-forward neural networks are directed acyclical graphs.

Fully-Connected Neural Networks

fully connected neural networks (FCNNs) are one of the earliest multi-layer topologies. Each neuron in a hidden layer has weighted connections to all neurons in the previous and all neurons in the following layer. Cybenko (1989) showed that FCNNs with a single hidden layer can approximate any continuous function using sigmoidal activation functions. This introduction of nonlinearity thus makes such networks more powerful than thresholding perceptron networks. The author cautioned that this does not answer questions of feasibility, since their theorem only holds if there are no constraints on node count. Later, Barron (1993) provided proof that the approximation error in such networks is bounded in a way that makes them useful in moderately-high dimensional problems. There are functions that have a compact representation in a deep architecture, i.e. have few degrees of freedom that need to be tuned by learning, that might require an exponential number of computational elements when represented by an insufficiently deep architecture (Bengio, 2009). But deep FCNNs have proven difficult to train from scratch, since the ”credit or blame for the output error is distributed too widely and thinly“ (Bengio, 2009, p. 44).

An advantage of FCNNs is that they are structure agnostic as their topology presumes no specific distributions in their inputs (Ramsundar & Zadeh, 2018). In contrast, other, *sparse* connection schemes like CNNs commonly assume spatial relationships between inputs.

Today, architectures have diverged from FCNNs to other connection schemes, but often include some layers utilizing any-to-any connections. CNNs use fully connected layers for classification in the final stages of the model for the class prediction output (see section 5.7). Image processing with networks solely consisting of fully-connected layers is not competitive because of the higher training effort and much larger memory footprint.

Autoencoders

For Goodfellow et al. (2016, p. 4), ”[t]he quintessential example of a representation learning algorithm is the autoencoder“. They elaborate that an autoencoder is a combination of an encoder that converts input into a different representation, and a decoder which converts new representations back into the original format. These models are trained in an unsupervised fashion, resulting in representations which have various nice properties tuned for specific aims (Goodfellow et al., 2016, p. 4).

Autoencoders – sometimes alternatively called auto-associators or Diabolo networks – have been used as building blocks for deep network training, where at each level, an autoencoder can be trained in an unsupervised fashion, leading to a better weight initialization (Bengio, 2009, p. 45). Vincent et al. (2010) created a deep neural network of stacked autoencoders capable of representing ”several levels of nonlinearity“ (p. 3371) for denoising discrete input signals. Masci, Meier, Ciresan, and Schmidhuber (2011) and Kallenberg et al. (2016) have employed a variant called convolutional stacked autoencoders more recently to pre-train CNNs for image processing. Variational autoencoders, introduced by Kingma and Welling (2013), have a structure similar to compression or denoising autoencoders, but use a probabilistic approach for the forward pass (Patterson & Gibson, 2017). Kingma and Welling trained generative models to produce handwritten digit images as well as faces.

Closely related to the autoencoder is the encode-decode concept of a variety of deep nets for keypoint regression (Z. Cao et al., 2017) and semantic segmentation (He, Gkioxari, Dollár, & Girshick, 2017). Since this principle is highly relevant to this dissertation, it is treated separately in section 8.2.

Convolutional Neural Networks

Current artificial neural networks are able to beat highly skilled players of complex games. Silver et al. (2018) developed a system that taught itself to play Go, chess, and a Japanese chess version called shogi by reinforcement learning from self-play. Their *AlphaZero* system is able to beat previous approaches as well as human players and is based on a convolutional neural network (CNN) architecture. CNNs receive their input in the form of a regular grid, which means they can process game boards as well as images. These inputs are convolved layer-wise with a large number of kernels that are small in relation to the input dimensions. The kernels are made up of weights that are trained in an iterative backpropagation approach. The restricted connection design of CNNs lets them extract local and repeating features, making them advantageous for shape recognition (LeCun et al., 1990). This *sparse connectivity* (Goodfellow et al., 2016, p. 330) also means that CNNs need a lot less trainable weights for the same input size than a fully connected network. In deeper layers, nodes have a large receptive field with respect to the input and can learn to combine features isolated in previous layers, in an optimization process guided by an error function (LeCun et al., 2015). This fully automatic feature extraction now often outperforms manual feature engineering in complex tasks (LeCun et al., 2015). Convolutional neural network-based algorithms have become the state-of-the-art in many image processing tasks (Krizhevsky et al., 2012; Alom et al., 2018). Silver et al. (2018) and Silver et al. (2016) exploited the fact that the translationally invariant rules of Go can be represented well by the translational invariance emerging from the weight-sharing property of CNNs to create algorithms that play the game with superhuman performance.

5.3.2. Recurrent Neural Networks

Recurrent neural networks possess feedback paths that lets the model reuse previous outputs when processing the current input and are used extensively for tasks ”where the embedded structure in the data sequence conveys useful knowledge“ (Pouyanfar et al., 2018, p. 5). In RNNs, learning was difficult using conventional *backpropagation through time* (e.g. Werbos (1990)), as gradients tended to explode or vanish (Hochreiter, 1991; Hochreiter & Schmidhuber, 1997). The variations in gradient magnitude and the fact that the ”effect of long-term dependencies is hidden (being exponentially smaller with respect to sequence length) by the effect of short-term dependencies“ posed a serious problem to gradient-based optimization (Chung, Gulcehre, Cho, & Bengio, 2014).

Hochreiter and Schmidhuber (1997) developed a recurrent network architecture using a *long short-term memory* concept that enforced constant error flow through specially designed cells that can decide to keep a ”memory“ or replace it. When a LSTM cell detects an important feature early on in the input sequence, it can carry information about the existence of the feature over a long time and captures potential long-term dependencies (Chung et al., 2014). This major advance allowed for efficient sequence analysis and the LSTM approach is still used today, with minor modifications (e.g. Graves (2013)).

A simpler cell design that was easier to compute and implement was introduced by Cho et al. (2014). It makes each recurrent unit adaptively capture dependencies on different time scales, still regulating information flow inside of the unit, but without separate memory cells (Chung et al., 2014). Chung et al. (2014) demonstrated the advantage of RNNs with gating units like LSTM and GRU over traditional recurrent units. In terms of benchmark performance, both gated cell networks performed similarly.

5.3.3. Deep Belief Networks

Because of the difficulties training deep neural networks ”much of machine learning research has seen progress in shallow architectures“ (Vincent et al., 2010, p. 3372). Variants with undirected connections in top layers called *Deep Belief Networks* were the first to overcome those difficulties (Hinton et al., 2006). These methods used unsupervised training of parts of the network to arrive at a set of weight values for which backpropagation could be utilized successfully (Hinton et al., 2006; Schmidhuber, 2015). This variant is mentioned for its role in the evolution of neural networks, but has largely been superseded by CNN for image modeling (Patterson & Gibson, 2017).

5.3.4. Hybrid Networks

In the search for ever better task performance, researchers have developed numerous models that incorporate aspects of several branches mentioned above. Xingjian et al. (2015) built a convolutional net with LSTM cells for exploiting spatial relationships for precipitation nowcasting, i.e. accurately predicting rainfall intensity locally over a short period of time. Yu Zhang, Chan, and Jaitly (2017) use this convolutional LSTM approach in combination with skip connections based on He, Zhang, et al. (2016) to create a deep end-to-end speech recognition model. In this case, the authors used the *sparse connectivity* of convolutional layers to maintain the spectral structure in the learned representations.

5.3.5. Generative Adversarial Networks

Generative adversarial networks were presented by Goodfellow et al. (2014). They developed a training regime where two networks work in tandem – one produces e.g. synthesized images, and the other discriminates if the images come from training data or from the generator. Goodfellow et al. aimed to maximize the error probability of the discriminator. Berthelot, Schumm, and Metz (2017) improved training stability and robustness with their *BEGAN* design. The idea has been developed further, and it is now possible to synthesize photorealistic faces using generative adversarial networks (GANs) (Karras, Laine, & Aila, 2019). Other applications include speech enhancement (Pascual, Bonafonte, & Serrà, 2017), MRI scan reconstruction enhancement (Mardani et al., 2018), and generative models of audio data (van den Oord et al., 2016).

5.4. What makes a good Neural Network?

In their seminal work on deep learning, Goodfellow et al. (2016) posit that besides latency, and other computational performance factors, the ability to generalize well, i.e. to have low error rates for new input, is a key property of a well-performing neural network. The training and test sets, i.e. the empirical distribution, have to be somewhat representative of the data-generating distribution (p. 109). The neural network should achieve a sufficiently low error on the training set, meaning it should not *underfit* (p. 109–110). Also, the gap between the error on the training set and the error on a test set shouldn't be too large, which would indicate *overfitting* (p. 110). By adapting the capacity of the neural network, the tendency to over- or underfit can be controlled (p. 110). The informal definition of capacity describes the network's ability to fit a wide variety of functions (Goodfellow et al., 2016, p. 273).

Deep neural networks usually have many more parameters respectively weights than data points available for training, which means that according to classical machine learning metrics of model expressivity like the Vapnik-Chervonenkis dimension (Vapnik & Chervonenkis, 1991, 3), they should overfit the training data and generalize poorly (Jakubovitz, Giryes, & Rodrigues, 2019). In practice, this is often not the case (C. Zhang, Bengio, Hardt, Recht, & Vinyals, 2017; Novak, Bahri, Abolafia, Pennington, & Sohl-Dickstein, 2018; Jakubovitz et al., 2019). The training process was proven to be NP-complete in the case of linear activation functions (Blum & Rivest, 1988).

Jones (1997) later provided partial proof for the NP-Hardness of training sigmoidal networks. However, ”[d]espite this theoretical pessimism, in practice, modern-day neural networks are trained successfully in many learning problems“ (Livni, Shalev-Shwartz, & Shamir, 2014, p. 2), even though the error surface is high-dimensional, non-convex, has local minima, saddle points, and flat spots (Hao Li, Xu, Taylor, Studer, & Goldstein, 2018). Goodfellow, Vinyals, and Saxe looked at the stochastic gradient descent path of several well-performing neural network models qualitatively and concluded that ”it seems likely that very large neural networks are easier to fit to a particular task“ than smaller models (Goodfellow et al., 2015, p. 7). They reason that the stochastic gradient descent training algorithm seems to be able to bypass obstacles in the error surface. Rumelhart et al. (1986) already mentioned this behavior in their seminal work about backpropagation, suggesting that ”[a]dding a few more connections creates extra dimensions in weight-space and these dimensions provide paths around the barriers that create poor local minima in the lower dimensional subspaces“ (Rumelhart et al., 1986, p. 535). Using stochastic gradient descent and its variants is a notable factor in successfully training neural networks, as it performs implicit regularization and thus helps generalization (Chaudhari & Soatto, 2018). Livni et al. (2014) conclude that several tricks allow real-world neural network training to work as well as it does: *over-specification*, confirming the interpretation by Rumelhart et al. (1986) and Goodfellow et al. (2015); choosing the right *activation functions* for neuron activity; and *regularization strategies*. Further research (He, Zhang, et al., 2016) showed that over-specification alone only helps to improve performance to a certain point. If plain networks become too deep, the stochastic gradient descent struggles to find a solution. In recent years, a lot of work went into increasing model expressivity by exploiting spatial and channel-wise interdependencies in novel ways to overcome the limitations of plain networks (refer to section 5.7).

The application of a well-performing neural network is a product of several advanced procedures touching on statistical learning theory, numerical optimization, theoretical informatics, hard- and software engineering, as well as data collection. Results from the field of of neural networks have been met with considerable enthusiasm not only by scientists but by journalists and tech evangelists. Sensationalist claims about the capabilities of neural networks and a somewhat free interpretation of the wider implications sometimes cloud the fact that there still is ”no free lunch“ and that machine learning systems can accrue significant technical debts (Sculley et al., 2015). Furthermore, creating a good training set is not always a straightforward process and can be fraught by ethical concerns (Yang, Qinami, Fei-Fei, Deng, & Russakovsky, 2020).

5.5. Image Processing Architectures

Because of the vast amount of publications on the matter, the performance on benchmark datasets acts as a guide towards the most promising architectures for various tasks. Table 5.1 shows the popular challenges for the field of image processing that provide a common frame of reference. Image classification in particular ”serves as the foundation of multiple tasks such as object detection, image segmentation, object tracking, action recognition, and autonomous driving“ (Su et al., 2018, p. 1) and researchers compete in reaching ever higher accuracy scores in the “Image Classification on ImageNet” (2020) benchmark based on the *ImageNet* dataset (Deng et al., 2009) and others.

Goodfellow et al. (2016, p. 116) comment that ”[t]he no free lunch theorem implies that we must design our machine learning algorithms to perform well on a specific task“. A common method for state-of-the-art architectures is using the image classification networks mentioned above to extract reliable features from input images to use them for such a specific task (He et al., 2017; J. Huang et al., 2017; Jiao et al., 2019). Agrawal, Girshick, and Malik (2014) showed that fitting a CNN to the *ImageNet* dataset (Deng et al., 2009) in particular results in a general and portable feature set.

For convolutional architectures, depth, width, connection design, and resolution are the most important architectural parameters when it comes to find a trade off between latency, accuracy, and indirect measures like floating point operations per second (FLOPS) (Tan et al., 2019). Many of the best-in-class models are a product of automated machine learning techniques trying to optimize said aspects, either by automated parameter search (Thornton, Hutter, Hoos, & Leyton-Brown, 2013), automated architecture search (Real et al., 2017; C. Liu et al., 2018; Tan et al., 2019), or automated finding of data augmentation strategies for training (Cubuk, Zoph, Mane, Vasudevan, & Le, 2019; Wei et al., 2020). At the time of writing, variants of a product of neural architecture search – *EfficientNet* (Tan & Le, 2019) – perform best in the "Image Classification on ImageNet" (2020) benchmark.

The spontaneous nature of note taking dictates that a technological solution for pen interfaces needs to be viable for untethered operation. Recent work on lightweight image processing architectures for mobile devices shows that applications of neural networks are not limited to desktop computing environments (Howard et al., 2017; Tan et al., 2019). Another recent development, the growth of cloud computing platforms like "Microsoft Azure" (2020) and "Amazon Web Services" (2020), enables remote on-demand neural network inference on bespoke hardware setups. In short, utilizing neural networks has become practical on a wide range of devices. The capability for state-of-the-art performance and robustness in complex image processing tasks make neural networks the technological solution best suited for a camera-based analog-digital pen interface.

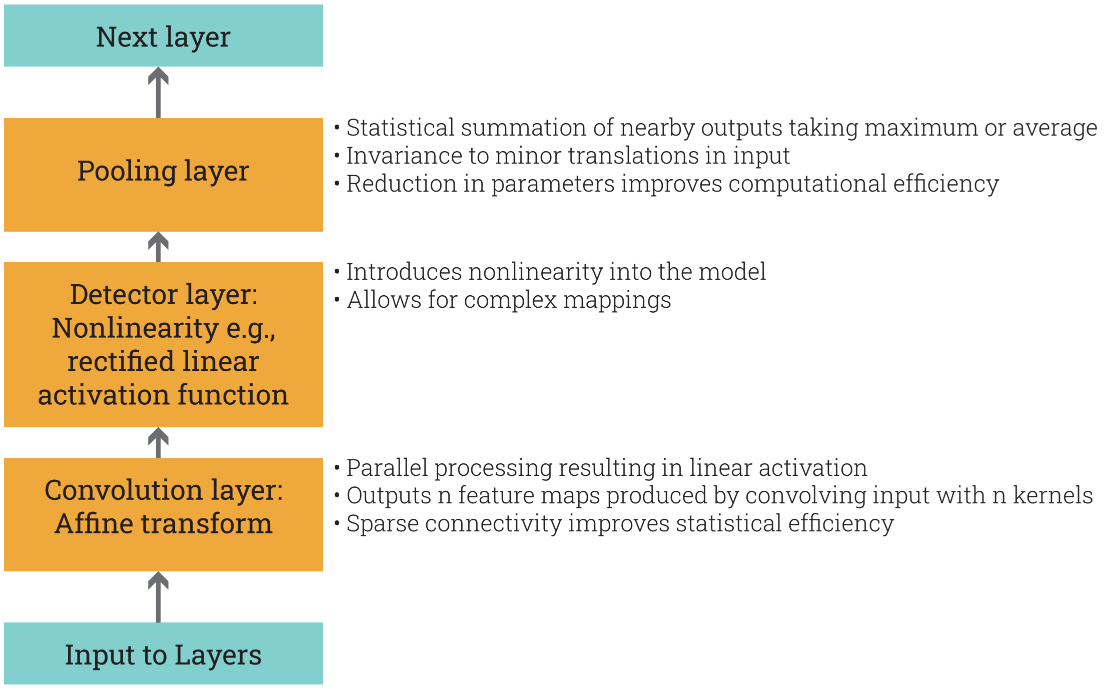

Next layer
Pooling layer
• Statistical summation of nearby outputs taking maximum or average
• Invariance to minor translations in input
• Reduction in parameters improves computational efficiency
Detector layer: Nonlinearity e.g., rectified linear activation function
• Introduces nonlinearity into the model
• Allows for complex mappings
Convolution layer: Affine transform
• Parallel processing resulting in linear activation
• Outputs n feature maps produced by convolving input with n kernels
• Sparse connectivity improves statistical efficiency
Input to Layers

Figure 5.4. – The basic elements of a convolutional neural network processing stage. This illustration is based on the CNN chapter in Goodfellow, Bengio, and Courville (2016, pp. 326–366).

5.6. Basic Elements of Convolutional Neural Networks

Most CNNs are built from the single layer building blocks in various configurations, with possible concatenations and other operations between them. In figure 5.4, the make up of a single stage in a convolutional neural network is illustrated and explained. This illustration is based on the CNN chapter in Goodfellow et al. (2016, pp. 326–366).

5.7. Popular Classification Architecture Families of the Last Decade

In this section, the defining aspects of some popular families of image processing architectures are presented, showcasing major contributions to the progress of CNNs. Even though the full *ImageNet* contained problematic categories, the challenge dataset defined a common ground and is an effective evaluation tool. Most of the architectures have been selected because of their performance in the ImageNet large-scale visual recognition challenge (ILSVRC), alongside contributions regarding efficient models for mobile devices and networks that were published when the competition was no longer held. Although the official ILSVRC ended, the challenge subset of *ImageNet* still remains an important benchmark for CNN classification performance. For a comprehensive in-depth survey of recent CNN history, refer to Khan et al. (2020). Properties of the CNN architectures that are listed in the following are sourced from the respective architecture publications if not referenced otherwise.

AlexNet (Krizhevsky et al., 2012) is seen as the first modern deep CNN and established a new state-of-the-art for image classification and recognition. With eight layers, it had more than *LeNet* (LeCun et al., 1998), which had five, and applied several techniques to improve generalization and reduce overfitting: (a) Krizhevsky et al. used data augmentation to increase training set size. (b) Dropout (Hinton, Srivastava, Krizhevsky, Sutskever, & Salakhutdinov, 2012) is used in the first two fully-connected layers, lengthening convergence time but improving generalization ability. (c) It used *ReLu* activation functions to alleviate the *vanishing gradient* problem and significantly increase training speed (Nair & Hinton, 2010).

ZFNet (Zeiler & Fergus, 2013) provided important insight into how deep convolutional networks function by developing a visualization technique for filter maps. The insights gained were used to design a model that won the ILSVRC in 2013.

Using a deconvolutional network (Zeiler, Taylor, & Fergus, 2011), Zeiler and Fergus reprojected feature activations back into the input pixel space. Their starting point for analysis was the *AlexNet* model, for which they identified several weaknesses in the first layers regarding filters for mid frequency information and aliasing resulting from the stride set for convolution. Adapting filter size and stride guided by this new knowledge was demonstrated to improve recognition capabilities.

Inception (Szegedy et al., 2015) increased the depth and width in comparison to previous convolutional networks but kept the computational budget nearly constant. The incarnation dubbed *GoogLeNet* as a nod to its heritage – *LeNet* – won the ILSVRC (classification) in 2014. Increased network size can be helpful for improving classification performance, but larger and expensive datasets are necessary to avoid overfitting. They also demand more computational effort, since ”any uniform increase in the number of their filters results in a quadratic increase of computation“ (Szegedy et al., 2015, p. 2). Szegedy et al. developed the *inception module* as a way to approximate the sparsity and clustering in neuron activation with available dense components. In this module, filters of different sizes are applied in parallel and concatenated afterwards, making the network able to detect local and global features in every layer. They showed that quality can be improved significantly this way in comparison with shallower and narrower architectures while keeping a similar computational effort. To prevent vanishing gradients during training, they introduced auxiliary outputs at several steps inbetween.

VGGNet Simonyan and Zisserman (2015) demonstrated the beneficial effects on image recognition performance when increasing depth to up to 19 weight layers and reducing kernel sizes of a conventional *ConvNet* architecture like *AlexNet*. This approach improved object localization performance and won the ILSVRC in this subcategory in 2014.

ResNet He, Zhang, et al. (2016) showed that even though increasing the depth of neural networks improves performance, there is a turning point where accuracy gets saturated and a degradation sets in that is not caused by overfitting. This degradation leads to a *higher* training error when adding depth to a suitably deep model. They successfully trained a 152-layer-deep network and addressed the vanishing gradient problem (Bengio, Simard, & Frasconi, 1994) by using residual or skip connections between earlier and later layers (He, Zhang, et al., 2016). With these connections added to a *VGGNet*-like architecture, the network can be successfully trained using stochastic gradient descent to find more expressive representational mappings because of its increased depth. He, Zhang, et al. used batch normalization (Ioffe & Szegedy, 2015) to allow for higher learning rates and improved regularization. *ResNet* architectures and their variants are still widely used for comparison and baselines (Xiao, Wu, & Wei, 2018; Tan & Le, 2019).

DenseNet G. Huang, Liu, Van Der Maaten, and Weinberger (2017) built on the findings regarding skip connections (He, Zhang, et al., 2016; Srivastava, Greff, & Schmidhuber, 2015) and proposed a simple connectivity pattern connecting all layers directly with each other through concatenation. Each layer passes its feature maps on to all deeper layers. This scheme has more connections than *ResNet* or other traditional CNN architectures, hence the name. An important distinction to other models is *DenseNet*'s narrow construction, where single layers have a lot less filters and add only a small amount to the "collective knowledge“ of the network. G. Huang et al. (2017) argue that because of the network connectivity, fewer parameters than traditional CNNs are required because redundant feature maps do not need to be relearned further down the model. *DenseNets* also required comparably fewer FLOPS than previous approaches with similar performance.

Dual Path Networks Yunpeng Chen et al. (2017) showed that *ResNets* resemble a subclass of *DenseNets* using shared connections. With this insight, they created a model combining the advantages of both to further increase accuracy and computational performance while reducing the memory footprint. As Yunpeng Chen et al. argued, *DenseNets* are good at exploring new features during training but having feature extractors relearn feature maps from all previous steps in parallel can lead to high redundancy, as some features are extracted more than once. In contrast, *ResNets* implicitly reuse feature maps because of identity skip connections where only the input is propagated along the layers, but are not as good at finding new ones since there appear less new combinations of previous features. Yunpeng Chen et al. integrated both the residual and the densely connected paths into a network that provides a slightly better top-1-accuracy than a *DenseNet*-family model while reducing computational effort and training speed.

SENet (Hu, Shen, Albanie, Sun, & Wu, 2020) The winner of the 2017 ILSVRC in the classification subcategory improved the top-5-accuracy of the winning entry of 2017 by ~25% through explicitly modeling interdependencies between channels to create helpful feature maps. Previous work often focused solely on exploiting spatial relationships between features. Hu et al. created an architectural block that allows the network to perform feature recalibration that helps to emphasize informative features. A squeeze-and-excitation block first reduces spatial elements to a single channel descriptor for every channel to create statistics representative of the whole image. An excitation operation aids to capture the channel interdependencies outside the receptive field of filter kernels. This SE block can be inserted into existing network architectures instead of their original building blocks to improve accuracy while keeping FLOPS approximately the same.

MobileNet (Howard et al., 2017) While the results were not competitive in the ILSVRC, Howard et al. contributed a class of efficient networks that can be easily tuned towards different hardware platforms. Instead of compressing networks or directly training small models, they reduce computational cost by using depthwise separable convolutions (Sifre & Mallat, 2014).They introduce width and resolution multipliers enabling trade off decisions between latency and accuracy. Their approach yields a class of models that they show to be flexible and capable of performing a variety of image processing tasks.

NASNet Published after the ILSVRC ended, *NASNet* achieved the same accuracy as *SENet* while almost halving the computational effort. Zoph, Vasudevan, Shlens, and Le (2018) employed neural architecture search by first finding a well-performing building block for a small dataset and applying an extended architecture using this block to a larger dataset. Zoph et al. (2018) also contributed a new regularization strategy – *ScheduledDropPath* – which they argue improves generalization of their model family. The search process employs a RNN controller that provides sample architectures, which are then evaluated. The resulting accuracy is used to adapt the weights of the controller using a policy gradient. The controller tries varying combinations of CNN building blocks, i.e. different pooling and convolution operations. Zoph et al. predetermine the overall model architecture manually, while the structures of the cells are determined by the search algorithm.

EfficientNet In their publication for *EfficientNet*, Tan and Le (2019) adapt several architectures using automated architecture search using a novel method for scaling architectural components. *EfficientNet* improves on FLOPS, parameter count, as well as accuracy, by ”carefully balancing network width, depth, and resolution“ (p. 9).

5.8. Summary

The above list shows that over the last decade, researchers used the existing knowledge about training neural networks to come up with new ideas to circumvent their shortcomings. *AlexNet* in particular only combined already existing approaches into a powerful new architecture but resulted in a significant improvement over the

Dataset	Instances	Classes	Annotations
ImageNet (Deng et al., 2009)	14M	1000	BBox and Classes
CIFAR10/100 (Krizhevsky et al., 2009)	60K	10/100	Classes
Pascal VOC (Everingham et al., 2010)	46K	20	BBox, Classes and Masks
MS COCO (T.-Y. Lin et al., 2014)	2M	80	Keypoints, BBox, Classes and Masks
MNIST (LeCun et al., 1998)	70K	10	Classes

Table 5.1. – Table of image processing dataset challenges. Based on table from Pouyanfar et al. (2018).

existing state-of-the-art. Improvements became more gradual as the challenge went along, but new subcategories were introduced to pose harder problems. In the last year of the ILSVRC (2017), the winners in several categories, i.e. the *dual path networks* and the *squeeze-and-excite networks*, represented recombinations of existing model architectures offering only slight improvements in the top-1-accuracy and most competition entries achieved a very high classification accuracy.

Harder tasks, like human pose estimation or pan-optic scene analysis, continue to be challenging for state-of-the-art approaches. In this dissertation, relevant literature for the concrete tasks investigated is reviewed in the respective chapters: keypoint detection is reviewed in chapter 8, handwritten text extraction in chapter 9, and sequence analysis in chapter 10.

6. Data Collection: *UbiPenTrack*

For the supervised training of the neural networks investigated in this dissertation to track pens and text, a data collection study was conducted to create the *UbiPenTrack* dataset. The focus with data collection lay on collecting a large amount of high quality samples, suitable as a general baseline for automated feature extraction of writing processes. To the knowledge of the author, no other dataset exists which contains comparable features and annotations.

When developing appropriate deep learning models for a pen interface, the prerequisite for any learning-based algorithm is a dataset exhibiting a distribution that is close to the application setting. In the case of this dissertation, data from 60 individuals was collected in a process divided into four parts:

(1) a **prestudy** with 5 participants to assess the study parameters

(2) a **large collection study** with 39 participants creating the main body of data

(3) collection of a **simple validation set** with 8 participants

(4) collection of an **in-the-wild validation set** with challenging aspects including variations in devices, resolution and positioning with 8 participants

The annotated *UbiPenTrack* repository consists of six subsets for training and evaluating pen detection and tracking algorithms. In the remainder of this dissertation, they will be designated as given in table 6.1.

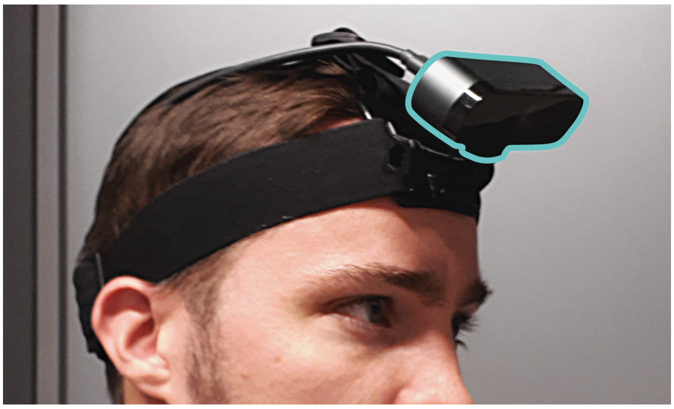


Figure 6.1. – The head mounted camera setup, using a Logitech BRIO Webcam.

Subset	Subjects	Samples
POV-Keypoint	39	18098
POV-Sequence	39	14040
Simple-Keypoint	8	1876
Wild-Keypoint	8	1859

Table 6.1. – *UbiPenTrack* subsets and their designations.

The validation sets *Simple* and *Wild* both depict writing utensils being used by individuals writing sentences, just as the *POV* set does. They differ in other aspects, such as camera positioning, table and background texture, recording devices, and overall scene setup. Robustness to shift in input data distribution is an important aspect for neural network systems that are developed for real-world use, and has been in focus of recent research (Ovadia et al., 2019; Nado et al., 2020). Both datasets were collected to assess model performance under slight (*Simple*) and strong covariate shift (*Wild*), meaning that the underlying concepts do not change, but the features that represent them do (A. Zhang, Lipton, Li, & Smola, 2020, p. 175).

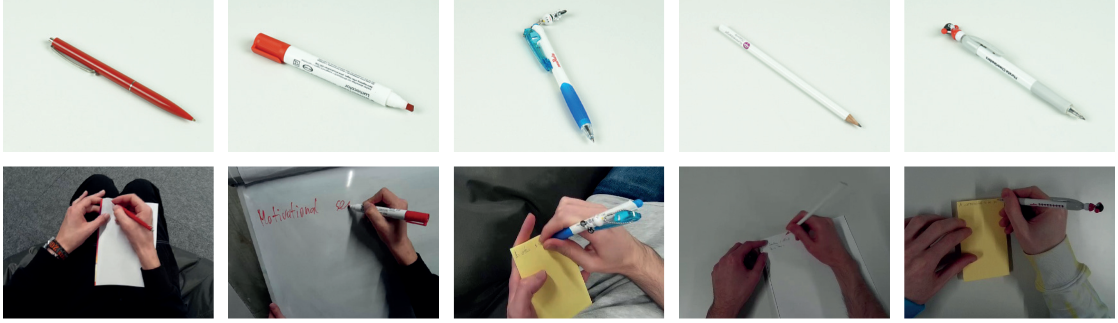


Figure 6.2. – Image data points created during the prestudy. *Top row*: clear, high quality pictures of the pens used during the experiment. *Bottom row*: The same pens in use during a randomized writing task.

6.1. Prestudy

6.1.1. Study Setup

A prestudy was conducted in a laboratory setting under artificial lighting with the author as supervisor. To achieve a high variance in the video material produced, writing tasks were generated in a randomized fashion. Variables included the pen, notebook or paper pad, the situation, and the text. Participants wrote down randomly chosen text fragments from the widely used phrase set by MacKenzie and Soukoreff (2003) for text input tasks. This phrase set remains a recommended choice after evaluation and comparison to other, newer sets by Kristensson and Vertanen (2012).

6.1.2. Limitations

The test subjects were all native German speakers, while the phrase set consisted of English phrases. Sufficient knowledge of the English language was assumed to be present. Subjects were advised that misspelled or misunderstood words would not hinder the experiment and that they were free to write as they see fit.

<table>
<tr><th></th><th>Prestudy</th><th>UbiPenTrack</th></tr>
<tr><td><b>Participants</b></td><td>5</td><td>10 female and 29 male</td></tr>
<tr><td><b>Handedness</b></td><td>5 right</td><td>37 right and 2 left</td></tr>
<tr><td><b>Writing Tasks per Person</b></td><td>10</td><td>5</td></tr>
<tr><td><b>Phrases per Task</b></td><td>6</td><td>3</td></tr>
<tr><td><b>Sum of Usable Tasks</b></td><td>–</td><td>186</td></tr>
<tr><td><b>Average Length of Task</b></td><td>–</td><td>51.5s</td></tr>
<tr><td><b>Number of Pens</b></td><td>25 + 5</td><td>25 + 5</td></tr>
<tr><td><b>Writing Surfaces</b></td><td>6</td><td>6</td></tr>
<tr><td><b>Situations/Positions</b></td><td colspan="2">Seated at table</td></tr>
<tr><td></td><td colspan="2">Standing at table</td></tr>
<tr><td></td><td>Seated on beanbag</td><td>Seated on sofa</td></tr>
<tr><td></td><td>Standing at flip chart</td><td>Standing at whiteboard</td></tr>
<tr><td><b>Annotated Samples</b></td><td>2267</td><td>18098</td></tr>
<tr><td><b>Training set size</b></td><td>–</td><td>14324 (79.15%)</td></tr>
<tr><td><b>Test set size</b></td><td>–</td><td>3774 (20.85%)</td></tr>
</table>

Table 6.2. – Data collection study parameters for the prestudy and the *UbiPenTrack* dataset.

6.1.3. Implications for the Study Setup

Based on the experience gained during the prestudy, the study parameters were modified. Less video material was produced of each subject but the amount of subjects was increased substantially, with the intention to create a more diverse dataset. Due to organizational constraints, the bean bag was replaced with a couch. The flip chart was replaced with a mobile whiteboard as a surface for board markers, to gather data points more closely aligned with real working conditions. The scene backdrop was changed from the laboratory to an office setting and the lighting was improved, as videos came out overly dark.

The technical setup was improved, too. The resolution of images taken did not allow reading the text participants produced on paper in most cases. The image quality in general was lower than expected, informing the decision to use a higher quality camera for the main data collection study.

6.2. Main Dataset

Recruiting a much larger number of individuals was the key objective for the main study. With more participants, a larger set of first person views on the region of interest, pen poses, as well as a lot more individual features like bracelets, painted fingernails, and wrist watches could be captured. Otherwise, the dataset collection was informed by the lessons learned during the recording of the prestudy dataset[1].

6.2.1. Participants

37 right-handed and 2 left-handed participants were recruited, aged between 21 and 36, out of which 10 were female and 29 were male.

[1] The recruitment and execution of this study was substantially supported by Andrea Fischer.

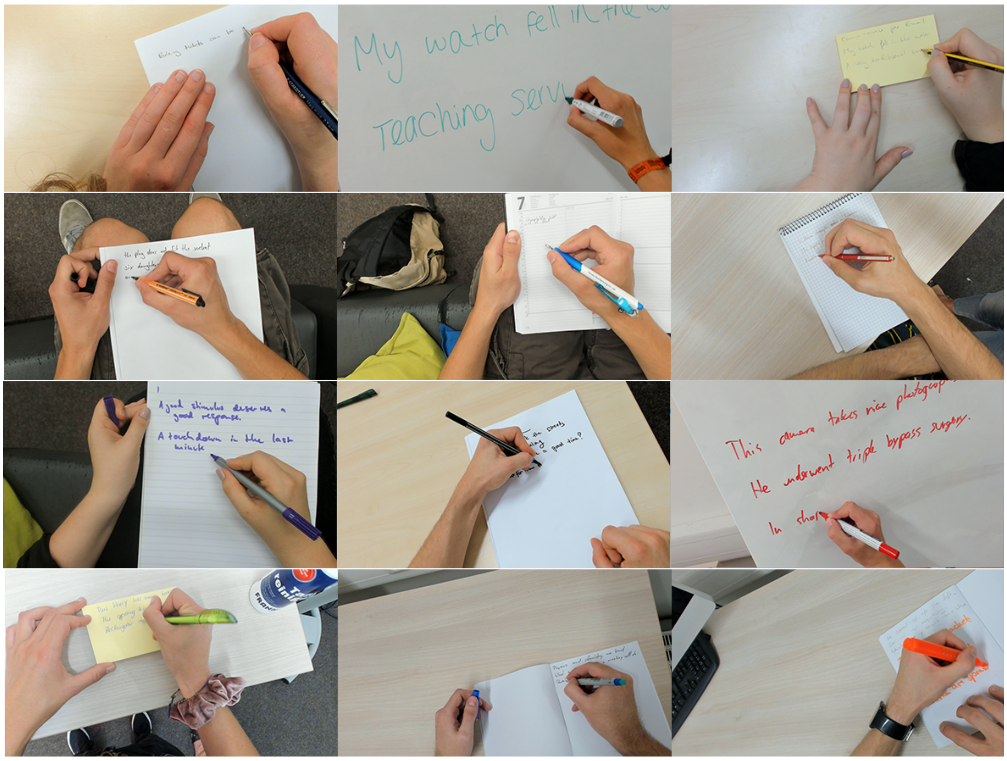

My watch fell in the w
Teaching serv
A good stimulus deserves a good response.
A touchdown in the last minute
This camera takes nice photograp
He underwent triple bypass surgery.
In sho

Figure 6.3. – Sample frames from the data collection study.

6.2.2. Study Setup

Participants were asked to write three phrases from the MacKenzie phrase set from dictation (MacKenzie & Soukoreff, 2003). The phrases were randomly chosen for five different scenario configurations, leading to 15 phrases being written by each participant. To introduce variance, all configuration parameters were picked at random beforehand. Writing scenarios for the tasks could either be *sitting at a desk*, *standing at table*, *standing in front of a whiteboard*, or *sitting on a couch*. The latter scenario was aimed at providing a less constrained setting to gather unexpected writing poses. The writing surface was selected from a set of lined and graph paper pads, a blank paper book, post-it notes, a calendar, and a whiteboard. A random generator also chose a pen for each situation from a set of 25 pens and a set of 5 different whiteboard markers for the whiteboard case. Table 6.2 provides an overview over the study parameters.

To record the writing tasks from a first person view, participants wore a consumer-level camera attached to a head band (see figure 6.1). In contrast to a static camera installed at a certain angle, occlusion of the writing area by participants' body positioning was minimized in this way. By choosing an appropriate field of view (FOV), the head-mounted camera could capture what participants saw at every point in time – if something blocked the camera's view, it most likely blocked the test person's view, also. In this way, subjects did not have to take care to write inside a designated area dictated by camera parameters, making the writing process less constrained. Many angles of text at various scales as well as body poses are present in the collected samples, since head orientation, hand placement habits, as well as handwriting differ for each individual.

The videos were recorded in 3840x2160 (4K) resolution at 30 FPS, the maximum possible for the *Logitech BRIO* camera used. High resolution was chosen over high frequency, since the aim was to collect a future-proof dataset with a focus on single high quality samples for analysis of writing processes.

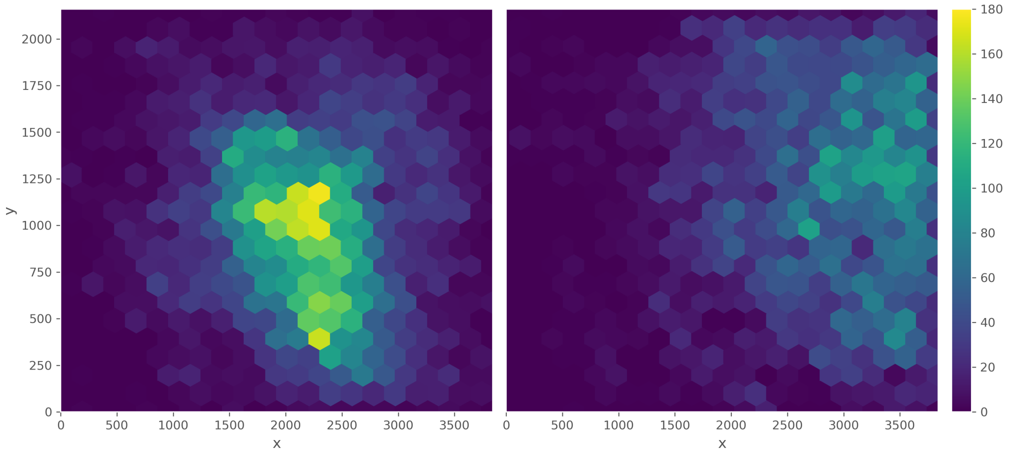


Figure 6.4. – The distributions of annotated pen tips (left) and pen tails (right) across all annotated images in the data set. Pen tips are mostly located in the center region because of first person view recording. The distributions imply bias towards right handed participants.

6.2.3. Keypoint Annotation

Kim et al. (2017) used an automated annotation system because their setup with just one pen and one view made it possible to employ optical flow approaches through priming. In contrast to Kim et al. (2017), no automated annotation was used for *POV-Keypoint*, due to changing views frame to frame and a much higher variance in image content.

18098 frames were uniformly sampled of the video material gathered, on a per task basis. On every chosen frame, the tip and tail of the pen were marked if they were visible[2]. Occluded keypoints were marked $(-1, -1)$. Frames with excessive blur were marked as not containing relevant information. To improve annotation accuracy, the author reviewed a random 50% subset of the annotations and corrected

[2] Annotators for this dataset were Miriam Schlindwein, Viet Dung Le, Tassilo Schwarze, and the author.

them where necessary. Figure 6.4 displays the distribution of pen tips and tails in the annotated frames.

6.2.4. Limitations

The dataset is not balanced regarding the location of annotations, as can be seen in figure 6.4. Two major contributors to bias are the first person view setup as well as the mostly right handed participants.

The phrases the participants wrote down were not in their native language. Some reported insecurities about their ability to spell English words correctly when writing from dictation. This could have led to less natural or slower writing movements.

6.3. Sequence Subset: *UbiPen-Sequence*

For the purpose of investigating time-based aspects of pen movement, a training dataset consisting of 234 sequences was created from the videos created for *UbiPen*. In contrast to the main dataset, where the annotation focused on collecting a diverse set of stills, this dataset contains only sequences of consecutive frames. Three sequences representing 60 frames or two seconds of video were randomly taken from two tasks of each subject. In total, 14040 frames were annotated with pen tip location and pen up/down state. In an effort to create a diverse sequence dataset, frames from each subject of the main data collection were included. The train and test data were split between subjects the same way as they were for *UbiPen*. Because of operational limitations, not the whole video material could be annotated.

For keypoint annotation, a neural network trained on the *UbiPen* dataset using the baseline detail architecture described in chapter 8 supported the annotator. In a custom annotation interface developed for this project, the annotator selected the image patch containing the region of interest, where the neural network then detected the pen tip. Figure 6.5 shows an example frame from a sequence, where

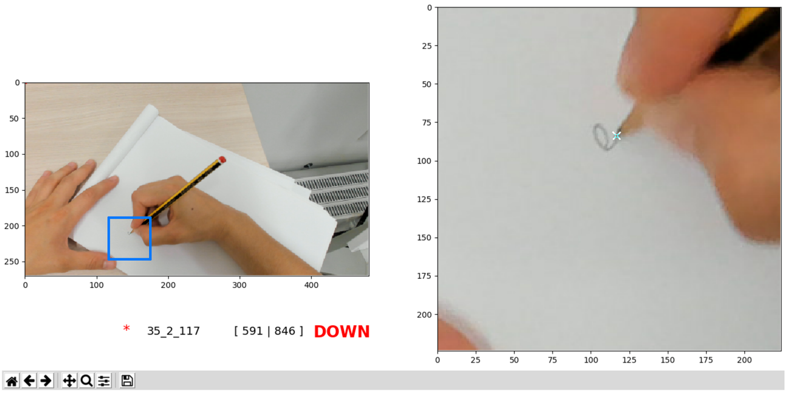


Figure 6.5. – The sequence annotation tool, based on the interactive frontend of *MatPlotLib*. The left image represents a downsampled video still, where the annotator selects the patch for automated keypoint annotation. The right side shows this patch with the generated keypoint. It allows the user to edit potentially misplaced keypoints. Using the keyboard, up/down status can be changed. This image was edited for emphasis.

the tip keypoint was provided by the neural network, and the *pen down* annotation was added manually.

6.4. Binarization Subset: *UbiPen-Binarize*

Applications involving handwritten text sometimes revolve around analyzing and manipulating the actual text pixels. To create a base for investigations focusing on the text pixels themselves, videos from each participant in *UbiPen* were sampled uniformly. Figure 6.6 (right) shows an example of a binarized patch ground truth. Automated binarization with adaptive thresholds did not perform well due to changing light conditions, light ink colors, and low local contrast. To create high quality ground truth images despite the challenging makeup of the dataset, *Photoshop CS6* was used to manually binarize the 372 images.

Subset	**Sequences** **(#Frames)**	**Images**	**Subjects**	**Positives** **(pen down)**	**Negatives** **(pen up)**
Train	186 (60)	11 160	31	4780	6380
Test	48 (60)	2880	8	1338	1542
All	234	14040	39	6 118	7922

Table 6.3. – *UbiPen-Sequence* dataset parameters. The dataset is not fully balanced, which can be mitigated by class-weighting.

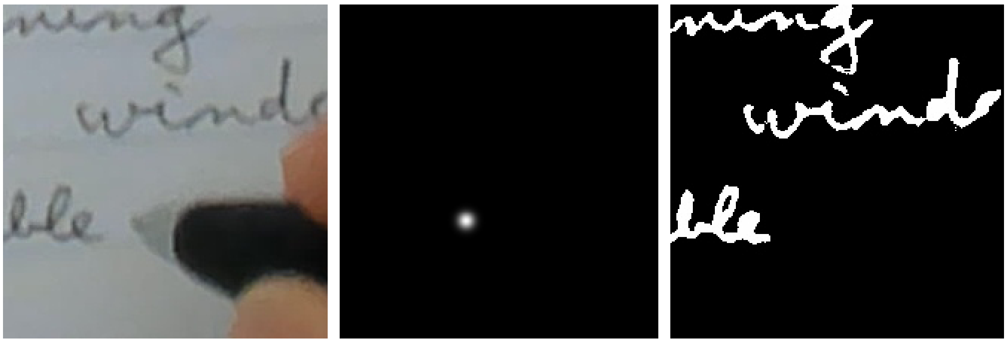


Figure 6.6. – An image patch (left) and its ground truth annotations: pen tip (middle) and text (right).

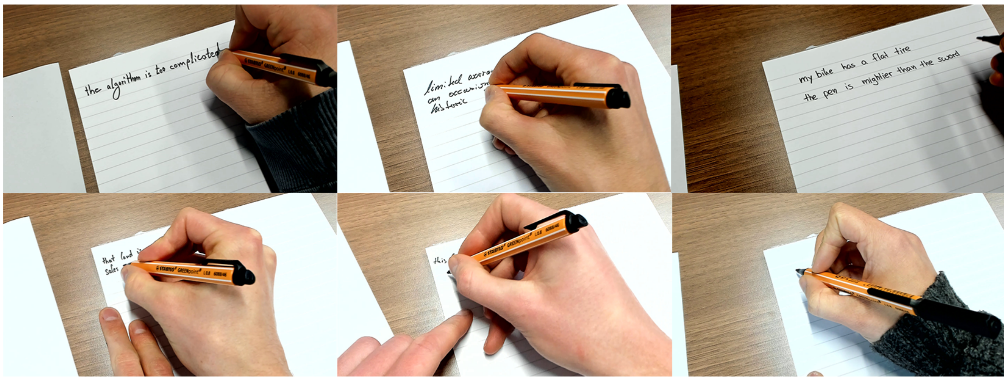


Figure 6.7. – Video stills from several subjects from the first validation study. Setup for each participant was identical. This study was focused on different ways people hold and move pens.

6.5. Validation Set A (*Simple-Keypoint*)

To validate the training approach and test set results, a validation study was conducted with participants who did not take part in data collection. This study was similar in setup to the collection. It was focused on the impact of differences between users and their writing poses on system performance in a single desktop scenario.

6.5.1. Participants

8 right-handed participants were recruited, of which 1 was female and 7 were male.

6.5.2. Study Setup

The same camera (Logitech BRIO) we used for data collection was mounted on a tripod. All study subjects were handed the same paper pad and pen. They were asked them to copy three phrases from the MacKenzie phrase set (MacKenzie & Soukoreff, 2003) from a list shown to them. Figure 6.7 shows example video stills from the setup.

The table texture and changing lighting conditions were the only factors introducing variance besides the users.

6.5.3. Annotation

Equidistant samples were annotated from all participant recordings to reach a sample size of 1876, which corresponds to 10.4% of *UbiPenTrack*. Several recordings included stretches of occluded pen tips. Only visible keypoints were annotated.

6.6. Validation Set B (*Wild-Keypoint*)

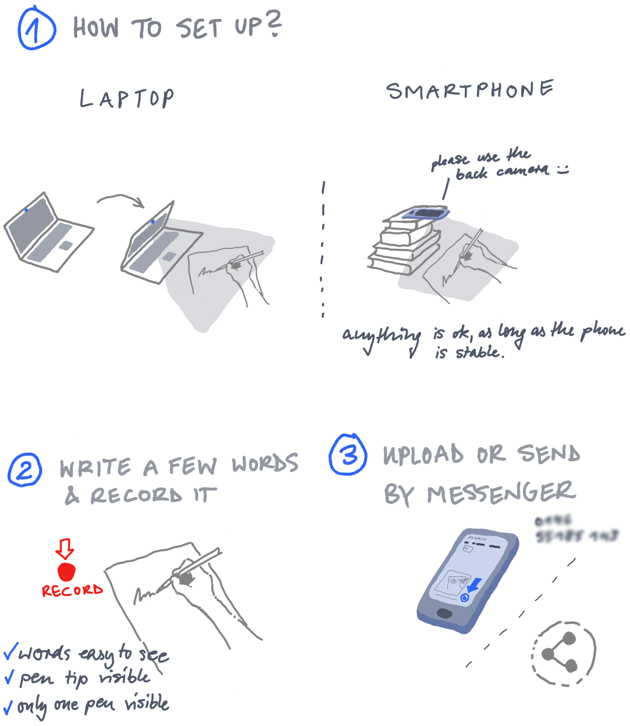

1 HOW TO SET UP?
LAPTOP
SMARTPHONE
please use the back camera :)
anything is ok, as long as the phone is stable.
2 WRITE A FEW WORDS & RECORD IT
RECORD
✓ words easy to see
✓ pen tip visible
✓ only one pen visible
3 UPLOAD OR SEND BY MESSENGER
FLORIN
Hi
[redacted]

Figure 6.8. – The instructions as they were sent to participants of the second validation study. They suggest two setups and specify basic requirements for the recordings.

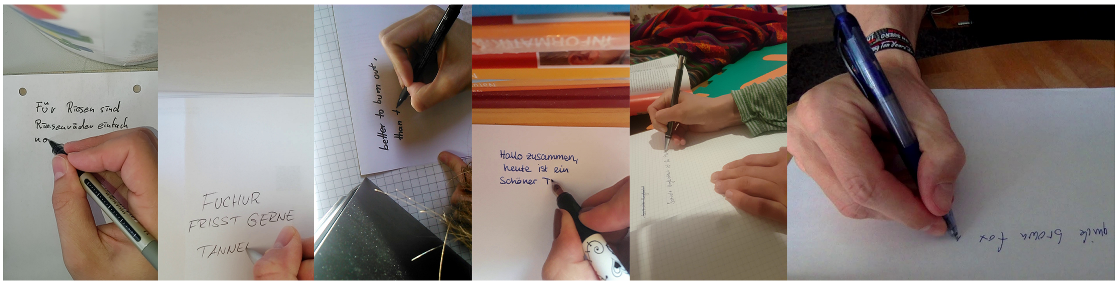


Figure 6.9. – Images from the second validation study, remotely done, had a much higher variance in setting, lighting and perspective than either the training set or the first validation set.

The second validation study was designed to assess actual in-the-wild performance, including scenarios and pens the models were not explicitly trained for. For this reason and because of the Corona Virus lockdown taking place, a remote study approach was chosen, leaving many degrees of freedom to the individuals volunteering as test subjects. In contrast to previous data collection, the phrases users could write were not limited. The technological setup was only constrained by the study requirements to record a writing process. Participants were instructed to set up a smartphone or webcam to record themselves writing 5–10 words. The type of pen, camera orientation, or lighting situation requirements were deliberately not mentioned, to produce a dataset of maximum diversity.

6.6.1. Participants

For the collection of this validation set, 11 right-handed participants (2 female, 9 male) aged 22–36 who did not take part in the previous studies were recruited. In two cases, couples living together (S4 and S5, S6 and S7) agreed to participate and used similar setups in their recordings.

6.6.2. Submissions and Annotation

Participants were asked to either upload their video to our file server or use a text messenger app to submit it. One participant (S1) used a builtin laptop camera to record the video, all other participants opted to use their smartphones for recording. Three submissions did not meet the minimum resolution requirement, which were set to at least half of the training setup (960x540). This resulted in the evaluation excluding 3 subjects and leaving the final sample size of 8 subjects.

Videos with 4 different orientation angles were obtained from 6 kinds of devices. In the case of one couple (S4 and S5), orientation was flipped between recordings. Video resolutions for submissions included in the second validation set were 1080x720 (S1), 1280x720 (S3), and 1920x1080 (S2, S4, S5, S6, S7).

Frames from each video were taken with equidistant sampling and annotated, to reach a sample size of 1859, which represents 10.3% of *POV-Keypoint*. Frames with heavily blurred pens were marked as containing no tip.

7. Loss and Performance Metrics

Neural network based learning algorithms stochastically minimize loss functions. In practice, they iteratively adjust the weights in the network units according to the gradients of mini-batches taken from a training set such as *UbiPenTrack* in order to better approximate the real data distribution (Goodfellow et al., 2016, p. 148).

The models investigated in this dissertation either give predictions about the location of features in input images or they predict them belonging to one of two categories. For both tasks, a strong literature base exists for choosing an appropriate loss function. The relevant functions are discussed in the first section of this chapter. However, evaluating model performance is not as clear cut. In the second section, metrics, their difficulties, and solutions are discussed.

7.1. Loss Functions

For the goal of minimizing the error on the real data distribution, a loss function that allows backpropagating changes throughout the network by differentiation is necessary. While reducing the training error is critical, a training process that does not also reduce the generalization error has failed to produce a viable model (A. Zhang et al., 2020, p. 142). Numerical optimization algorithms like stochastic gradient descent (SGD) or Adam (Kingma & Ba, 2014) require numerically friendly formulations of objective functions to perform well.

7.1.1. Maximum Likelihood Estimation

Maximum likelihood estimation ”is the concept that when working with a probabilistic model with unknown parameters, the parameters which make the data have the highest probability are the most likely ones“ (A. Zhang et al., 2020, p. 865). It is favored because it is consistent and statistically efficient, with the property that when ”the number of training examples approaches infinity, the maximum likelihood estimate of a parameter converges to the true value of the parameter“ (Goodfellow et al., 2016, p. 132). The conditional maximum likelihood formulation for optimizing parameters of a model is commonly used when describing the estimation principle that e.g. supervised learning is based on (Goodfellow et al., 2016, p. 131).

The following formulation and explanation is based on the one given in Goodfellow et al. (2016, 129ff.).

(7.1)

Where θ stands for the model parameters, with X describing the inputs to the model and Y representing all observed targets. P is the conditional probability that Y are predicted given the input X. With m samples $\mathbf{x^{(1)}}, \dots, \mathbf{x^{(m)}}$ drawn independently from the unknown data-generating distribution $p_{data}(\mathbf{x})$ and assumed to be identically distributed, equation 7.1 can be transformed:

$$\theta_{ML} = \arg\max_{\theta} \prod_{i=1}^{m} P(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}; \theta) \tag{7.2}$$

$$\theta_{ML} = \arg\max_{\theta} \sum_{i=1}^{m} \log P(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}; \theta) \tag{7.3}$$

With $\mathbf{y^{(i)}}$ as outputs for the examples. Note that the i.i.d. assumption about the samples often only partially holds with e.g. sequence data. The product in equation 7.2 is numerically problematic for large sets of samples. Since logarithmic transformation maintains monotonic behavior it is equivalent to express it as a sum of logarithms.

The supervised learning task of training a network to assign a probability to an input image belonging to one of two classes is known as logistic regression, which can be optimized by minimizing the *negative log likelihood* (Goodfellow et al., 2016, p. 130). Minimizing the *binary cross-entropy* is equivalent to this (Goodfellow et al., 2016, p. 130).

7.1.2. Mean Squared Error

Heatmap regression is a common method for finding salient features in input images, which will be discussed in the context of pen interfaces in chapter 8. For this task, the *mean squared error* loss or *Brier* score, another proper scoring rule (Brier, 1950), is most often used (Tompson, Jain, LeCun, & Bregler, 2014; Newell, Yang, & Deng, 2016; Xiao et al., 2018; F. Zhang, Zhu, Dai, Ye, & Zhu, 2020). It ”optimizes the pixel-wise similarity between the output and a synthetic heatmap generated from ground truth locations“, and has the drawback that some more accurate results can be penalized more (Nibali, He, Morgan, & Prendergast, 2018, p. 5). A coordinate regression approach would allow for a direct euclidean distance error metric between extracted and ground-truth coordinates, but presents a challenge to learning in visually ambiguous scenarios (F. Zhang et al., 2020). In facial landmark recognition, a modified loss function named *adaptive wing loss* has been introduced that penalizes small errors for large probabilities more than the MSE (X. Wang, Bo, & Fuxin, 2019). X. Wang et al. (2019) also proposed a weighted loss map to focus optimization on important parts in the heatmap. The *adaptive wing loss* introduces three interdependent additional parameters for which the authors have published optimal values for one facial recognition data set. Using it for pen keypoint detection would require extensive additional evaluation since the application context is different and the loss function has not been widely adapted yet.

7.2. Evaluation of Predictions

As Goodfellow et al. (2016, p. 101) put it, ”it is often difficult to choose a performance measure that corresponds well to the desired behavior“ of a system.

Classification systems highlight how this choice is fraught with subtle difficulties. Neural networks used for classification usually output a probability vector indicating class membership for the available categories – for example, all the architectures that took part in the ILSVRC did so. The actual class decision is taken after the statistical procedure has concluded. In the case of the image classification networks discussed in section 5.7, the probabilities are ranked according to their value. The highest ranking category is chosen for calculating the *Top-1 accuracy*. If an image is classified as ”airplane“ with probability 0.33 and as ”cat“ with probability 0.34, the decision for ”cat“ influences the classification accuracy score just as much as if the system had predicted the probabilities to be 0 and 1. Harrell (2017b) argues that decisions like this can lead to the wrong model being favored, since classification accuracy is a discontinuous improper scoring rule, where very small changes in threshold can lead to large changes in the metric. The *Top-5 accuracy*, indicating how often the model lists the correct output in the top five results, is a softer metric that enriches the impression of the model performance, but is still discontinuous and improper.

However, in automated pattern recognition a forced decision can be necessary. There is often no time for a decision maker with knowledge of the domain to weigh cost and risk, in contrast to other fields, like medicine (Harrell, 2017a). It is still helpful to throw away as little information as possible (Harrell, 2017a). In the case of a neural network detecting and localizing pens in images, this could mean delivering probabilities to subsequent parts of the pen interface application, such as stroke reconstruction, to supply them with as much information as possible.

An example of a pragmatic performance evaluation is the *Deepfake Detection Challenge*, where the authors used the negative log-likelihood to rank submissions (Dolhansky et al., 2020). The negative log-likelihood is the formulation of the maximum likelihood optimality criterion in terms of loss, i.e. for minimization, and is a strictly proper scoring rule. For the binary classification task of detecting deepfakes, Dolhansky et al. reported precision at various recall levels, as well as the area under precision/recall curves and ROC curves. Notably, some lower ranked submissions exhibited higher precision at the recall levels given. In this way, the authors produced a pragmatic picture of correctly ranked models.

7.2.1. Accuracy

The *accuracy* of a classifier is often also called *proportion correct* metric. It is sensitive to class imbalances and is not a proper scoring rule.

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.4}$$

7.2.2. Precision

The *precision* or *positive predictive value* of a classifier can be expressed as the relation of true positives to the sum of all predicted positives. A high precision indicates that a classifier is reliably providing only true positives. It does not tell anything about its ability to find all true positives.

$$P = \frac{TP}{TP + FP} \tag{7.5}$$

7.2.3. Recall

The *recall* or *sensitivity* of a classifier is given by the ratio of true positives detected to all positives in the dataset. A classifier with high recall finds most positives in a dataset. It is possible for a classifier to exhibit high recall if it classifies all samples as positives, but its precision would suffer heavily.

$$R = \frac{TP}{TP + FN} \tag{7.6}$$

7.2.4. Specificity

The *specificity* or *true negative rate* is analogous to the sensitivity for negatives.

$$S = \frac{TN}{TN + FP} \tag{7.7}$$

7.2.5. Balanced Accuracy

The balanced accuracy aims to alleviate some problems of the classification accuracy measure by averaging the recall and specificity of a predictor, resulting in a metric that is more robust to class imbalance (Brodersen, Ong, Stephan, & Buhmann, 2010).

$$BA = \frac{\text{Sensitivity} + \text{Specificity}}{2} = \frac{1}{2} \cdot (\frac{TP}{TP + FN} + \frac{TN}{TN + FP}) \tag{7.8}$$

7.2.6. Other Confusion Matrix Metrics

There is a host of metrics derived from the confusion matrix. Of those, the measures described above are the ones most relevant to the neural network evaluation approach in this research project. For an overview of standard statistical measures, refer to e.g. Tharwat (2020).

7.2.7. Precision/Recall Curves

Plotting the precision and recall values for the interval of thresholds applied to the output of a class predictor is a common way to get an impression of the classifier performance that is robust against class imbalance. For example, it was used in evaluating entries for the large-scale *Deepfake Detection Challenge*. The AUC represents the average precision (AP) of the classifier (Boyd, Eng, & Page, 2013). For multi-class problems, this AP is often averaged, resulting in the mean AP.

7.2.8. Receiver Operating Characteristic Curves

ROC curves show the relationship between the false positive rate (FPR) and the true positive rate (TPR) in binary classification systems and are employed as a technique to determine classification performance in signal detection theory, analysis of medical diagnoses, and machine learning (Fawcett, 2004). Fawcett argues that their insensitivity to unbalanced classes makes them useful tools for analysis in real world domains. This makes them well suited for evaluating classifier performance on the imbalanced validation sets, as well as the test set.

The decision of keypoint presence is made based on the output heatmap of a model configuration. The maximum value can give information about the level of certainty the model possesses regarding a specific input containing a pen tip. The ROC curves for a classifier in this assessment were created by plotting the FPR and TPR value pairs for different maximum threshold values following the algorithm by Fawcett (2004).

The AUC of the ROC can be used as a metric to compare the ROC of different classifiers, given the thresholds for classification fall into the same interval (Fawcett, 2004).

Pen tracking relies on a sequence of correctly detected coordinates in order to make ink reconstruction possible. While missing keypoints can be detrimental, they can be interpolated. However, automated outlier detection algorithms dealing with false positives require optimization of additional parameters. False positives thus introduce a higher cost, since they distort stroke paths in ways that are hard to robustly predict, especially in a real-time scenario. Typical cases where a false positive introduces problematic outliers are the temporary occlusion of pen tips by the writing hand while writing and pen movements between words that put the pen tip out of frame. ROC and AUC lay importance on the FPR and thus provide an accurate evaluation of the classification performance of the assessed model configurations.

Euclidean Distance

The euclidean distance metric delivers an intuitive measurement of the shortest distance between two points in space and is well suited for low-dimensional distance measurements, and was chosen over other metrics like the city block or chessboard metrics for those reasons. Since the system works with images, distances are all calculated in pixel space. Ground truth images were always annotated with integer coordinates for pen keypoints. Various scaling operations are performed throughout pre- and postprocessing steps and use floating point coordinates to avoid lossy operations. The measurements given in the following are pixel distances between detected and ground truth coordinates on HD resolution images for comparison.

Face keypoint detection algorithms use the bounding box width (Sun, Wang, & Tang, 2013), or the popular inter-ocular distance (Y. Wu & Ji, 2019) in annotated ground truth images to normalize the landmark distance error. For the pen interface scenario, similar concepts are frustrated by the large variations in size and thickness, as well as the orthogonal relationship between pen length and tip path. Therefore, this dissertation settles on reporting the euclidean distances as absolute pixel values in relation to a HD resolution image.

8. A Baseline Model for Pen Tracking

The pen tracking framework developed in this dissertation does not rely on integrated sensors or visible markers and does not need priming. Instead, it depends on tracking two basic invariants: the pen tip and tail. Other features, like length, thickness, shape, color, texture, ink, and print vary wildly across commercially available pens. The two tip and tail keypoints represent the pen pose across the whole object class. Relying on them for tracking builds generalization into the core of this pen interface architecture and supports the overarching goal of ubiquity.

According to Gao et al. (2020, p. 2), ”visual tracking is still a challenging problem due to multiple negative scenarios such as occlusions, fast motions, scale variations, and background clutters“. This holds especially true for real-time applications. The focus on isolated points of interest has an advantage over e.g. shape detection in terms of computational cost, and an adequate training set can help a neural network handle scale variations and noise. Still, missing data resulting from sequences where a hand covers the writing region of interest, depending on the recorded frame, can potentially undo any tracking success by corrupting detected ink sequences. This baseline model thus focuses on the elementary task of finding pen keypoints in an image and represents a building block for a future pen interface. It also serves as a reference point for further investigation regarding the compromise between latency, accuracy, and robustness. For a real-time interface application, all three aspects need to be considered when mapping the solution space.

Current state-of-the-art feature extraction backbones often work with deep neural network structures for expressive capacity (Jiao et al., 2019). There are several powerful CNN architectures available for fast feature extraction from input images, as discussed in section 5.7. This chapter compares the performance of a set of CNN models as feature extractors. Additionally, decoding the extracted features into meaningful output is investigated through the technical studies performed.

8.1. Keypoint Regression

Detecting a pen tip in an image using a deep learning approach means performing keypoint regression if no previous location is known. Any model used needs to gather relevant feature sets from input images and transform those into keypoint locations.

Keypoint regression is the task of detecting the location of a point of interest in an image. Many problems in image processing, like human pose estimation (F. Zhang et al., 2020; Z. Cao et al., 2017; Xiao et al., 2018; Fang, Xie, Tai, & Lu, 2017), object detection (Law & Deng, 2020; Duan et al., 2019; Zhou, Zhuo, & Krahenbuhl, 2019), and facial feature detection (Sun et al., 2013; Yu Chen et al., 2019) have been formulated as keypoint regression tasks (Tensmeyer & Martinez, 2019).

F. Zhang et al. (2020) discuss two model designs for keypoint regression in the context of human pose estimation. One is *coordinate regression*, where the keypoint coordinates are taken directly as model output. This formulation is less common than *heatmap regression*, and, as F. Zhang et al. (2020, p. 2) argue, ”lacks the spatial and contextual information“ necessary for human pose estimation, which Nibali et al. (2018) support for general landmark localization. *Heatmap regression*, on the other hand, operates with a heatmap as the model output target, allowing for easier training of the inherently spatial relationships between features in images. In this context, a heatmap can be interpreted as a two-dimensional representation of how certain the model is that a keypoint is in a certain location. A large median value would indicate that the neural network was not able to determine a distinct position and is unsure how to interpret the input. Tompson et al. (2014) were first to use heatmap regression in neural network training.

For generating heatmaps for keypoint regression, the standard label representation is a 2-dimensional Gaussian with a small variance and mean centered at the coordinate – instead of simply marking on pixel, a spatial support is provided around the ground-truth location, incorporating the inherent target position ambiguity (Tompson et al., 2014; F. Zhang et al., 2020).

8.2. Encoder-Decoder Networks

In a general sense, a CNN encodes features into latent space by virtue of convolving and pooling. In the case of classification, image classes can be attained as output by using fully-connected layers trained to model the relationships between compressed spatial features and object categories. Numerical coordinate regression would entail similar modeling. As mentioned in section 8.1, in the case of keypoint regression, the formulation as a regression problem in euclidean space has proven helpful – to train the developed models for keypoint regression, heatmaps are used as output targets. This is beneficial for training convergence. When operating, a pen tracking framework built in this manner will translate a camera stream into such heatmaps indicating the certainty with which it locates the pen keypoints at each pixel coordinate. Simple maximum detection can then deliver the most likely position according to the machine learning software. Several procedures can be used to improve accuracy when decoding coordinates from heatmaps (F. Zhang et al., 2020).

To implement this formulation, relationships of compressed feature representations encoded by the CNN with the target heatmap output need to be modeled, i.e. the features need to be decoded into the desired form. A commonly used network paradigm that can translate abstract feature representations of camera images into heatmaps is the encoder-decoder net using various upsampling strategies. In contrast to autoencoders, which also possess encoding and decoding segments, the models in this section are trained with supervision.

Encoder-decoder CNNs are some of the most widely used network architectures when tackling inverse problems (Ye & Sung, 2019). Variants thereof are able to perform instance segmentation (He et al., 2017; Badrinarayanan, Kendall, & Cipolla, 2017) and human pose estimation by tracking joints as keypoints (Z. Cao et al., 2017), and have found their way in remote sensing applications like aircraft recognition (Yuhang Zhang et al., 2018). *Hourglass* networks are related to encoder-decoder models, and are named after the shape of the architecture. Newell et al. (2016, p. 4) introduced an architecture that stacked several *hourglass* modules, which ”process spatial information at multiple scales for dense prediction“. Gao et al. (2020) proposed a hourglass siamese network that visually tracks an object marked with a bounding box at the start of a video. Pavlakos, Zhou, Chan, Derpanis, and Daniilidis (2017) adapted hourglass models from human pose estimation for detecting semantic keypoints on objects.

Heatmap regression is often performed using a combination of convolution layers for feature extraction and encoding and a number of deconvolutional layers for the decoder. In the latter part, a CNN can learn upsampling filters by supervision. In their approach to human joint keypoint tracking, Gkioxari, Toshev, and Jaitly (2016, p. 10) found that ”multi-scale deconvolutions lead to a better and very competitive baseline“, supporting the argument that to appropriately represent upsampling nonlinearities, more than one transposed convolution layer is needed.

Belagiannis and Zisserman (2017) on the other hand did not use an upsampling approach while processing their heatmaps, instead relying on the leftover resolution after two pooling passes between convolutional layers. While application-specific accuracy was sufficient, for the demands of pen path tracking, such low resolution heatmaps are not suited. Newell et al. (2016) did not use transposed convolution layers, but recombined features from different scales to recreate the original resolution.

Xiao et al. (2018) proposed a simple baseline for fast and reliable keypoint regression for human pose estimation with an encoder pretrained on *ImageNet* and transposed convolution layers for the decoder. The resulting feature maps are reduced to a heatmap through feature map pooling (M. Lin, Chen, & Yan, 2013). They demonstrated the model's ability to come within state-of-the-art performance ("COCO Challenge Keypoint Leaderboard," 2020) at that point in time with a relatively simple model layout and low inference time, demonstrating the capabilities of the architecture type to tackle keypoint regression problems. This philosophy of creating simple, interpretable, well-performing baselines inspired the approach taken in this chapter.

8.3. Architecture Evaluation Study Design

Goodfellow et al. (2016, p. 413) write that "[a]fter choosing performance metrics and goals, the next step in any practical application is to establish a reasonable end-to-end system as soon as possible“. Finding a suitably accurate keypoint detection model is the first step for this task.

In this section, several configurations of aforementioned encoder-decoder networks are proposed. Their architecture, their training, and considerations for their evaluation will be discussed in the following.

The configurations will then be evaluated regarding **(a)** predictive performance, **(b)** their ability to recognize if a pen keypoint is present, **(c)** the euclidean distance error between detected keypoint and annotated ground truth, **(d)** their distinction power regarding pen tips and tails, and **(e)** parameter count, determining computational and memory requirements.

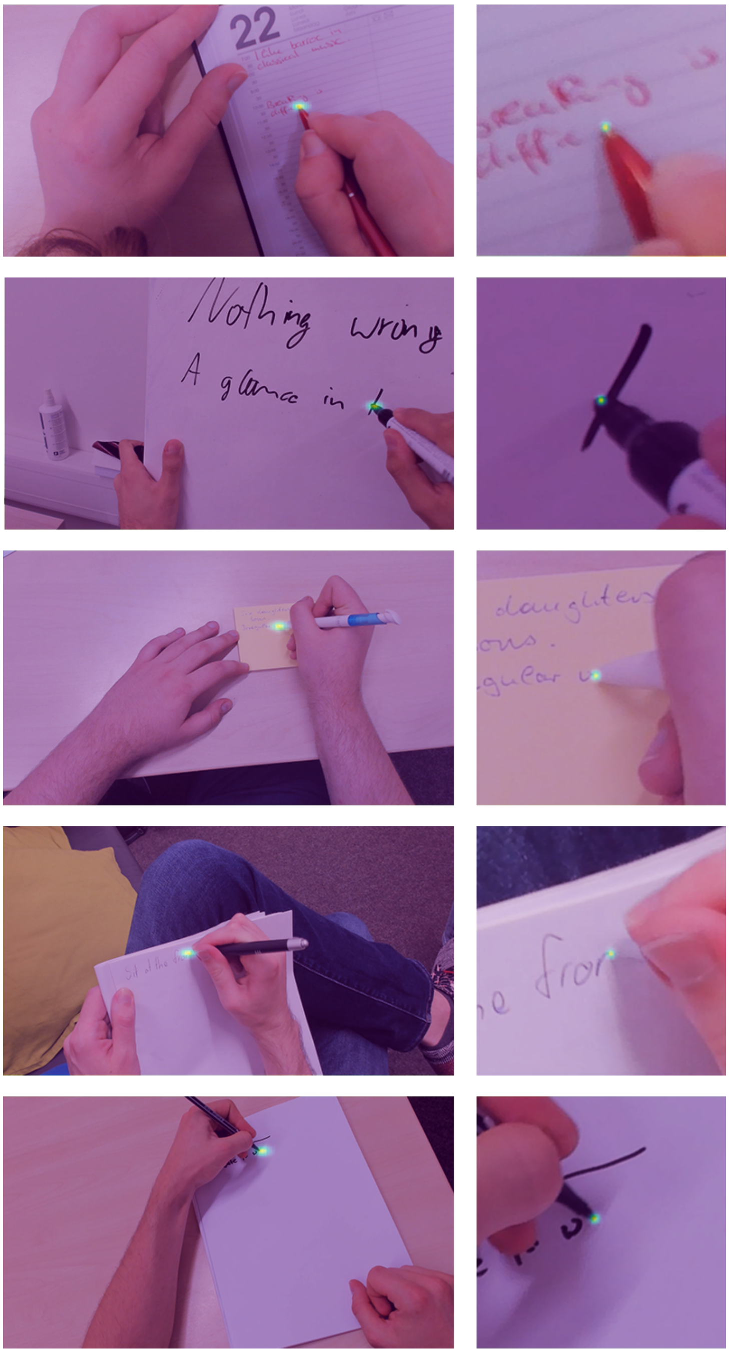


Figure 8.1. – Frames from the test set of *POV-Keypoint* and the resulting heatmaps produced by the *ResNet-50-L-NF* keypoint detection variant, where bright green means high certainty. The images on the left show the output of the ROI detection network overlaid on input frames. The images one the right show the image patch extracted using ROI peak detection with the detail tip detection laid over it.

8.3.1. Network Hyperparameters

The goal of this analysis is a broad overview of the performance of different kinds of architectures on the task at hand. There is a large amount of possible layer configurations, as well as hyperparameter sets for training each architecture configuration. An exhaustive analysis of each variant regarding all possible hyperparameters would lead to a combinatorial explosion that is impossible to handle with the resources of the author.

According to (Goodfellow et al., 2016, p. 423), ”[t]he primary goal of manual hyperparameter search is to adjust the effective capacity of the model to match the complexity of the task“. Hyperparameters like kernel sizes or keypoint regression targets were gathered from existing literature when possible. Other aspects, like the representational capacity, were systematically varied to find a local optimum. Sensible configurations were chosen to the best ability of the author to guide decision making to a reasonably good baseline.

8.3.2. Input Resolution

The resolution of input images supplied to the network is a defining network parameter and represents the amount of information the model can work with. Earlier model families used resolutions starting at 224x224, while current CNNs for object detection use resolutions up to 600x600 to classify images with state-of-the-art accuracy (Tan & Le, 2019). This is still far lower than the resolution that cameras of current smartphones produce.

Object detection tasks are not the only ones where input images are scaled down due to computational constraints. In heatmap regression, the computational cost ”is a quadratic function of the input resolution“ (F. Zhang et al., 2020, p. 7094). For the pen keypoint detection use case, an appropriate architecture ideally extracts keypoint coordinates at the original image resolution. Low resolution samples are not sufficient for reconstructing ink paths later on. F. Zhang et al. (2020) emphasize that the coordinate decoding and resolution recovery from heatmaps is an area of interest for improving network performance.

An estimate of the minimum image resolution necessary can be made dependent on the distance of the camera to the writing surface, the angle, and the field of view. In the *UbiPenTrack* data collection study, participants wore the camera on a headband. For that camera position, an average distance from camera to paper of 0.5 m can be assumed. At this distance, a camera with a FOV of 65° like the Logitech BRIO used for capturing *POV-Keypoint* approximately resolves to 30 pixels per cm horizontally at HD resolution[1]. This is only a rough approximation, since it does not incorporate lens distortion and other optical effects. Handwritten letter size varies, but standardized line heights on an A4 piece of paper according to the DIN 16552-1:2005-05 norm range from 9 to 10 mm except for elementary school use (DIN, 2005). The digit pixel resolution of the *MNIST* dataset, a widely used reference dataset for handwritten digit recognition, is 28x28 pixels (LeCun et al., 1998). Putting this reference baseline for letter resolution together with the standardized line height, this matches closely with the previously approximated 30 pixels/cm for the Logitech BRIO at HD resolution.

HD is far larger than the network input resolutions mentioned above. The encoding process, while reducing resolution, raises feature map dimensionality. Configuring a model with a state-of-the-art CNN feature extractor for resolution such as HD is prohibitive in terms of memory use and FLOPS. Real-time stream processing applications always drive the demand for optimal use of available computing power. Efficiency is paramount, so the objective is to find a solution that satisfies the requirements with reasonably low computational cost.

[1] This follows from the approximation of the view cone as two right triangles, where one side length is 0.5m and the angle at the viewer's end is half of $65^\circ = 32.5^\circ$. The half view width at 0.5m is then $x = 0.5 * \tan(32.5^\circ) \approx 0.32m \Rightarrow 1920px/64cm = 30px/cm$

8.3.3. Pen Keypoint Regression Architecture

The limitations mentioned above match with what Gao et al. (2020) identify as ongoing problems in the visual tracking community. Citing real-world performance requirements, they state that architectures capable of adequate feature representations are often too heavy-weight and are pre-trained on large datasets that are not fit for the specific task at hand. Attention guiding mechanisms, commonly in the form of *regional proposal networks*, are computationally expensive and bring a lot of additional hyperparameters. In contrast to their solution, this baseline approach does not perform general visual tracking but is fine-tuned for pen interface use. This fine-tuning requires training with a domain-specific dataset. In turn, it does not require an exemplar that needs to be matched by the network and is focused on two essential keypoints and their precise location instead of a bounding box.

A two-stage architecture for pen keypoint regression was chosen instead of resolution recovery approaches – the sub-pixel localization problem (see F. Zhang et al. (2020, p. 3)) is circumvented and complex attention guiding approaches are not needed.

In the first encoder-decoder stage, the downsized input image is mapped to an output probability map containing a two-dimensional gaussian with the peak at the locations of pen tip and tail keypoints following standard heatmap regression supervision procedure (Tompson et al., 2014). The resulting accuracy of output probability maps based on downscaled inputs is not sufficient for pen tip tracking, but the general region can be identified – this network determines the region of interest for a detailed keypoint extraction, thus fulfilling the role of an attention guiding process.

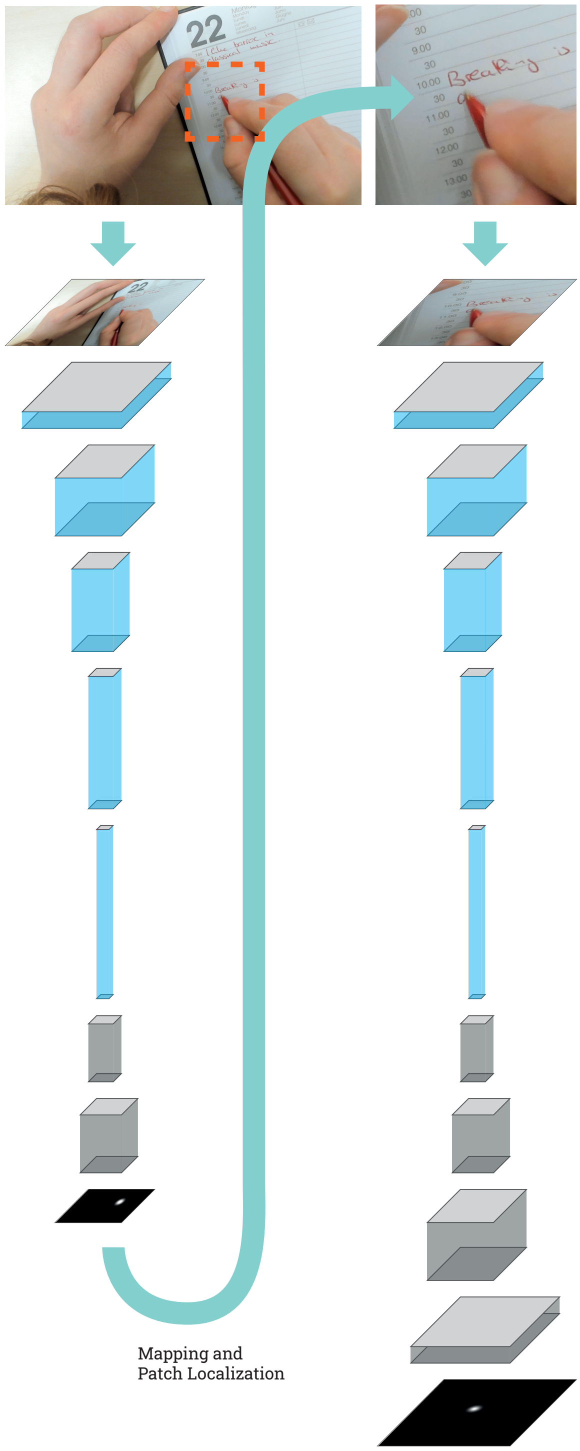
The second model, another instance of an encoder-decoder network, performs detail keypoint detection, working with small image patches extracted from the camera stream at the original image resolution around the ROI. On one hand, in a configuration like this, detail detection is dependent on a successful determination of the ROI. On the other hand, network parameter count can be kept lower than scaling the input dimensions up to HD or even 4K for the full resolution alternative. An example configuration of the whole architecture is explained in figure 8.2.

Furthermore, with this approach, the keypoint regression task can be split and conducted in a distributed fashion, e.g. letting a mobile device detect the ROI and only sending small patches of the full resolution image to a remote server for path reconstruction. By enabling a design that allows mobile or embedded devices to function as interface endpoints, this aspect is essential to possible use cases in a ubiquitous computing context. Load reduction is made possible by deciding early if an adequate ROI was found and only then doing further processing.

8.3.4. Decoder

To decode the abstract features received as output from the feature extraction backbone, transposed convolutional layers were used, following the work of Xiao et al. regarding convolutional decoders, using their upsampling parameters (Xiao et al., 2018). One part of the study's focus is the performance impact of decoder capacity. The model configurations were thus evaluated with either a *small* (2 layers) or a *large* (4 layers) decoder placed after the feature extractor. After each transposed convolution layer, batch normalization (Ioffe & Szegedy, 2015) and *ReLu* (Nair & Hinton, 2010) activation takes place. The very last layer that produces the heatmap uses feature map pooling (M. Lin et al., 2013).

The model output consists of one or more probability maps reflecting the confidence of the network regarding the presence of relevant features in the input image. The outputs depicted in fig. 8.2 show typical outputs for a confident detection. In figure 8.1, several of these output images were resized and overlaid on input images from the *UbiPen-Keypoint* test set to demonstrate an example model's capabilities. Noise, high standard deviation, or several peaks are usually indicators of a model struggling to make sense of its input.



Input image.

Camera frames with HD resolution arrive for keypoint detection.

For ROI detection, an image is downsampled and reshaped into neural network input.

The input is encoded mostly through convolutional and pooling layers.

The abstract features are then mapped to an output heatmap by transposed convolution operations.

The output locates the ROI on the input image. The maximum value of the heatmap indicates, after upsampling and reshaping, where to extract the patch for high resolution regression.

Detail Patch.

The patch extracted from the camera frame allows a low resolution network to operate at full image resolution.

Like the ROI network, the model encodes the input through convolutional and pooling layers.

It is trained on images cropped around annotated keypoints from the main dataset, learning features beneficial for accurate tip detection.

The pen tip location found through heatmap analysis is translated to full image coordinates and can be used for accurate pen tracking.

Figure 8.2. – An example layout of the encoder-decoder architecture developed to detect pen keypoints in this chapter.

8.3.5. Selected Model Families

The selected model families were chosen as representatives for a wider field of neural network approaches. Comparing older and recent models, this study aims at delivering a solid overview on how different network designs affect keypoint regression performance in terms of their power to recognize the presence of pen keypoints and keypoint accuracy.

ResNet *ResNet-50* achieves a 77.15% top-1-accuracy on the *ImageNet* classification task (He, Zhang, et al., 2016). This is 8.65% less than the leading architecture when training with *ImageNet* data only, according to Wei et al. (2020). Even if it is not state-of-the-art anymore, it is still widely used as a reference point, e.g. by the creators of *EfficientNet* (Tan & Le, 2019). A *ResNet-50* pretrained on the *ImageNet* dataset (Deng et al., 2009) was used by (Xiao et al., 2018) as a feature extraction backbone. Based on this, the *ResNet-50* model was chosen as the high parameter count reference. Larger *ResNet* versions were excluded from evaluation as their parameter count and thus the necessary computational expense go contrary to the real-time inference target of this project. Additionally, current approaches, like *EfficientNet*, produced similar scores with much less parameters.

NASNet A representative of popular adaptive lightweight architectures, *NASNet Mobile* was included to demonstrate feasibility of the tracking concept for mobile devices, which usually have less parallel computing power than consumer-level GPUs.

EfficientNet The decision to include the two variants B0 and B5 of the current state-of-the-art is based on the referential value of an elaborate (B5) feature extraction backbone. A smaller version (B0) represents a trade-off between accuracy and computational effort and provides direct competition to the *NASNet Mobile* model, showcasing potential advantages through better architecture.

8.3.6. Dataset

The *POV-Keypoint* dataset was split into a training set consisting of 79.15% of samples from 31 subjects and a test set with 20.85% of samples consisting of data from the other 8 subjects. One left handed participant was assigned to each set, thus weighting left handed detection higher in the test set score.

For the ROI stage of the architecture, the samples from the dataset were resized to HD resolution.

8.3.7. Training

Previous work on image processing networks designed for specific tasks like human pose estimation often used pre-trained feature extraction backbones, consisting of the convolutional parts of classification and object detection models (Lifshitz, Fetaya, & Ullman, 2016; Xiao et al., 2018). The training process relies on the fact that ”supervised pre-training and fine-tuning are effective when training data is scarce“ (Girshick, Donahue, Darrell, & Malik, 2014; Agrawal et al., 2014, p. 2). Transfer learning is a popular approach using the weights learned while training architectures on large datasets as basis for solving new learning tasks with the same architectures. Pan and Yang (2010) provided an early survey on the subject.

To create a baseline for camera-based pen keypoint regression that is easy to compare and optimize, all models were initialized with weights pre-trained on the *ImageNet* dataset and trained using the Adam (Kingma & Ba, 2014) optimizer with MSE as target metric on the *UbiPenTrack* dataset. At each point in training where a new minimum test error was reached, the model was saved. The patience-based *early stopping* meta-algorithm was employed to determine the weights with the highest generalization potential, avoid overfitting, and improve regularization (Goodfellow et al., 2016, p. 244). C. Zhang et al. support the idea that ”early stopping could potentially improve the generalization performance“ (C. Zhang et al., 2017, p. 7).

The training batch sizes were chosen small (32 and less) according to memory limits of the hardware used. Since larger batch sizes shorten training time but may negatively affect accuracy (Hoffer, Hubara, & Soudry, 2017), any adverse impact was isolated to GPU hours needed.

Learning rate decay was chosen to be step-wise according to recommendations by F.-F. Li, Krishna, and Xu (2020). They propose to use a slower rate to make sure that the best minimum is reached. The decay schedule was set following the baseline given by Xiao et al. (2018), who used a similar encoder-decoder setup for keypoint regression with a *ResNet-50* feature extraction backbone.

Since every model tested was pre-trained on *ImageNet*, it could be assumed that the lower layers contain general edge, color, and other filters that are beneficial for higher level feature extraction. For each model configuration evaluated, one was trained with the lower layer weights frozen to *ImageNet* values, and one where all weights stayed adjustable by the backpropagation process to investigate this assumption.

Patch Training

The patch training dataset for the detail stage was extracted from *POV-Keypoint*, selecting patches at keypoint locations from all samples, as well as randomly chosen patches from images that did not contain any keypoints for the background class. The dataset was divided up between subjects the same way as for ROI detection training. Patch sizes were chosen according to the native resolution of the respective feature extraction models. The patches were taken from dataset images that were resized to the same resolution used for the ROI stage, to produce consistent models that work with a camera device at a single resolution.

For training the detail stage of the architecture, exploratory training runs using a *ResNet-50*-based encoder-decoder model have shown that it is not sufficient to use ground truth patches centered around the pen keypoint. This led the network to predict a centered gaussian if it detects certain features, while it failed to learn the spatial relationship between input keypoint location and the ground truth heatmap.

For the model assessment, the patches were thus selected with a uniformly random offset from the keypoint location, to provide a clear indicator of spatial association between input and ground truth heatmap. The random offset excluded a border of 5% of patch dimensions at the edges, to prevent subsequent image augmentation operations pushing too many borderline keypoints out of the frame.

Image Augmentation

To counteract the bias introduced by gathering samples from a first person view of mostly right-handed writing recorded at a similar distance, and to increase virtual training set size, the images in the dataset were augmented. In a survey of image augmentation techniques, Shorten and Khoshgoftaar (2019) documented that even in comparison with advanced augmentation approaches involving e.g. generative adversarial networks, traditional image operations boost neural network performance well. Methods like *AutoAugment* (Cubuk et al., 2019), *Fast AutoAugment* (Lim, Kim, Kim, Kim, & Kim, 2019), and *Population Based Augmentation* (Ho, Liang, Chen, Stoica, & Abbeel, 2019) employ computationally expensive searches for augmentation policies. A randomized sequence of modifications, with the only two parameters being the number of augmentations and their magnitude, was more recently shown to achieve nearly identical accuracy on major image classification datasets by Cubuk, Zoph, Shlens, and Le (2020). The decision what magnitude means in the context of a specific image operation still leaves a lot of sub-parameters. The sub-parameters were taken from various existing *RandAugment* implementations, and the image augmentation algorithm was reimplemented for the *Tensorflow 2* dataset pipeline by the author.

The set of image operations used by Cubuk et al. (2020) and others consists of the following items:

- identity
- autoContrast
- equalize
- rotate
- solarize *
- color
- posterize *
- contrast
- brightness
- sharpness
- shear-x
- shear-y
- translate-x
- translate-y

Items with a star were found to have no positive effect in further experiments by Cubuk et al. (2020). They were left out of the training regimen for the model assessment. For training with large image classification datasets like *ImageNet*, Cubuk et al. (2020), as well as Cubuk et al. (2019), used a randomized left-right flip and a random crop operation before applying their image augmentation. The flip operation was used in the following evaluation, too. In contrast to the image classification task, training for keypoint regression depends on the availability of specific locations in the input data. Random cropping was not implemented for the baseline model assessment to avoid valuable ground truth annotations being skipped. Future work includes finding ideal keypoint augmentation strategies, but is out of scope for this evaluation.

Hardware and Training Time

For training the models in this comparative study, a linux system running *Debian* with an Intel i7-8700 CPU, 16 GB of RAM, and a NVIDIA GeForce GTX 1080 GPU with 8 GB of RAM was used. The time it takes to train the different baseline models for keypoint regression depends on this hardware setup, the image augmentation, the model, and other necessary preprocessing steps. Training time ranged from a few hours for the smaller networks to several weeks for the *EfficientNet B5*-based variants. The batch size influences training speed and is limited by available GPU memory.

Image resizing in particular is a resource-heavy operation that is part of the data preparation. The images in the datasets are stored at HD resolution and are rescaled for the input dimensions of the neural networks. Chapter 12 provides a comparison of several resize algorithms, inference times, and a network transmission study.

8.3.8. Evaluation Approach: Prediction and Decision

The assessment of model performance requires distinguishing between prediction and decision aspects of evaluation. Model selection should be based on proper scoring, using e.g. the Brier score or a cross entropy loss. For the keypoint detection, the MSE loss is used, as previous work has shown it to be effective for heatmap regression (see section 7.1.2).

Decisions like calculating a landmark location based on the values in a heatmap or thresholding a probability value for a classification problem come after the statistical part (Kolassa, 2019). For insights into the practical application of a neural network, the performance according to decision metrics is helpful.

The following aspects are evaluated by a number of metrics concerned with prediction and decision.

- **a)** Predictive Performance: MSE
- **b)** Recognition of keypoints present in input: ROC and PR AUC (threshold-based)
- **c)** Keypoint ROI error: failure rate (threshold-based)
- **d)** Keypoint localization error: Euclidean pixel distance between predicted and ground truth annotations (maximum detection)
- **e)** Distinction ability regarding pen tips and tails (maximum detection)

8.4. Evaluation Results

ROI detection

In the first part of the results section, the ROI stages are evaluated and compared to each other in terms of the metrics given in subsection 8.3.8.

The first stage of the baseline model architecture detects the ROI and thus needs to reliably and rapidly deliver a prediction regarding the pen keypoint location in a camera image. As long as the distance error to the true position does not exceed the offset afforded by the resolution of the detail stage, it is not detrimental to overall system performance. When a keypoint is localized further than half the minimum detail stage resolution away, this is regarded as a detection failure, because patch extraction happens centered on the keypoint location. The threshold is chosen the same for all evaluated models at $224/2 = 112$ for consistency, as the smallest detail stage network input is 224x224.

Keypoint coordinates

The detail stage provides pen keypoint coordinates for further processing and thus needs to produce high precision coordinates with little deviation. Even with a reliable ROI system, classification performance of the detail stage is relevant, as challenging scenarios can introduce some ambiguity regarding object classes. Constraints on processing speed are the same as with the ROI network. The detail stage additionally needs to deliver keypoint locations with minimal pixel error. For this, larger decoders are included, even though they have more parameters and potentially higher inference time.

8.4.1. Region Of Interest Configurations

For the ROI assessment, eight different encoder-decoder architectures were trained. The four feature extraction backbones used stem from three architectural families. All models were trained on the full image set *POV-Keypoint*, resized to the model input shape. For each of *ResNet-50, NasNet Mobile, EfficientNet B0, and B5*, a configuration with a large decoder and one with a small decoder was trained. Each of those configurations was once trained with lower weights frozen and once without freezing any layers, leading to 16 trained configurations in total. The variants are denominated **S-F** (small decoder/frozen lower layers), **S-NF** (small decoder/no frozen lower layers), and correspondingly **L-F** and **L-NF** for the larger decoder models.

The datasets used for calculating the following metrics are the test set part of *POV-Keypoint*, the *Simple-Keypoint* set, and the *Wild-Keypoint* set. In the text and figures, they will be referred to as *test*, *val1* or *validation 1*, and *val2* or *validation 2*, respectively.

The evaluation results in this section are presented mostly in graphical form. Five kinds of plots visualize important aspects of model performance, and their significance is explained in the following paragraphs.

Loss (prediction metric *a)*) The mean squared error loss plot shows the minimum loss the training process reached on the test set and the two validation sets for each configuration. It uses a logarithmic axis to show the differences between models on an appropriate scale.

ROC and PR AUC (decision metric *b)*) The areas under curve of the ROC and PR curves of tip and tail keypoints are presented using a matrix plot, with the cells colored according to value to communicate trends clearly. Notably, the color scale starts at 0.5. This value was chosen because it represents a random classifier, i.e. the worst possible classification. Values lower than that theoretically mean an inverse classification is better than random, leading to a pessimistic visualization.

Distance Failure Rate (decision metric *c)*) While an exact localization of keypoints is not as important as it is for the detail stage of the architecture, there is an upper bound to acceptable distance errors. This results from the working principle of the ROI-detail system. If the detected keypoint is placed so far away from the actual location that it would result in the extraction of a patch from the original input image that excluded the true location, the ROI model failed. The threshold for counting these failures was chosen to be half the input patch size. Some pixels at the edges of the patch would theoretically be excluded by this metric, making it pessimistic for fringe cases.

Median Distances (decision metric c_2)) For brevity, only the median tip and tail distances are reported for ROI networks using a classic line plot. For this measurement, only samples with a ground truth annotation representing a visible pen keypoint were considered. Distances that were above the failure threshold, i.e. distances of false positives, were also included to prevent an overly optimistic evaluation. The probability threshold used to determine if the network output means *no keypoint present* was chosen as 10^{-3} to account for a noise floor. For distance metrics in both ROI and detail stage evaluations, when comparing ground truth and network output, true negatives and false positives in terms of determining if *no keypoint is present* were excluded, because a distance to a non-existent point has no meaning. This decision power is instead analyzed by providing and assessing ROC and PR AUC.

Distinction Failure Rate (decision metric *d*)) This visualization shows the ability of a model to distinguish between tip and tail successfully. It only incorporates samples where one of two keypoints was annotated. Each ROI model always produces a heatmap for each possible keypoint. If only the pen tail is present in the input image, the model is expected to produce a maximum peak at its location on the corresponding output heatmap, and close to zero values on the other heatmap. If the maximum value of the corresponding heatmap the model produced is lower than that of the one for which no keypoint is present in the input, it failed to correctly distinguish between keypoints. Even though the resulting keypoint can still be useful, this failure provides insight into the lack of confidence in the class affiliation.

Family	Backbone Only	S-F	S-NF	L-F	L-NF
ResNet-50	23.52 M	32.73 M	32.96 M	33.39 M	33.61 M
NasNet Mobile	4.23 M	9.55 M	9.61 M	10.21 M	10.26 M
EfficientNet B0	4.00 M	10.28 M	10.30 M	10.94 M	10.96 M
EfficientNet B5	28.34 M	37.65 M	37.78 M	38.31 M	38.44 M

Table 8.1. – Trainable parameter count for the architecture variants. The ROI and detail stages only differ in the output channel count, with the former having two for tip and tail and the latter only having one, for tip. As a result, the trainable parameter counts are close to identical, so this table does not differentiate between ROI and detail stage. The largest and smallest models are colored orange and blue, respectively. The *backbone only* counts are for backbones with the classification head cut off and no decoder, and thus may differ from published counts.

ResNet-50 Backbone

The variants were tested with a backbone based on the improved residual blocks model published shortly after the original *ResNet* paper (He, Zhang, Ren, & Sun, 2016). The dense layers were cut off and the output of the last rectified linear unit (ReLu) layer with spatial structure was used as input for the first transposed convolution layer. For the *freezing* experiment, the `conv1` and `conv2` blocks, i.e. the first two of five blocks, were frozen. Both small and large decoder variants were evaluated with lower layers frozen and unfrozen. The full trainable parameter count for the baseline architecture variants are listed in table 8.1.

As can be seen in figure 8.3 (top left), the MSE for the test and validation 1 sets steadily declines with increasing model size. The MSE for the validation 2 set is closely grouped for all variants. The lowest was delivered by the largest model, L-NF. The second lowest val2 error in figure 8.3 (top left) was produced by the smallest model, S-F.

S-F trumps all other models with its ability to distinguish between tips and tails for validation 2, with a failure rate of 20.6%, against 27.4% for the next best variant, L-NF (see figure 8.3, top right). For the other sets, all variants perform at 10% and lower.

Highest decision metric performance of S-F continues with ROC and PR AUC. While the results vary on the datasets exhibiting little shift from the training set, S-F has the highest AUC for both ROC and PR when it comes to tip detection in validation set 2. The only exception is average tail detection precision, where L-NF landed ahead. Worst average precision on validation 2 was measured on the smaller variant with no frozen layers.

The distance assessment in figure 8.4 enhances the analysis of the variants of *ResNet-50*-based keypoint detectors. The larger decoder models expectedly reached better median pixel distance for all sets, as their output heatmap resolution was higher. The most important aspect of distance evaluation for ROI detection, the distance failure rate, complements the raw measured distances. For both tips and tails in validation 2, S-F again reaches the best – i.e. lowest – outcome with failure rates at 15.7% and 12.0%, respectively. While not the most accurate, it proved the most reliable ROI patch extractor when it comes to datasets with a large shift.

Discussion The results the variants reached on *Wild-Keypoint* are the linchpin of performance assessment. L-NF, the largest variant with the most flexible backbone due to all parameters being trainable, reached the lowest MSE and achieved the best AUC scores for test and val1 sets. On val2, it was consistently outperformed by S-F, the smallest network with the least trainable parameters. This indicates that generalization disability and thus tendency to overfit on the training data is most prominent on the larger *ResNet*-based keypoint detection variants. This conclusion is dependent on the training data used, and the large amount of parameters in *ResNet-50* seem to negatively affect performance when the training set is limited, as is the case with *POV-Keypoint*.

The best variant based on a *ResNet-50* backbone, S-F, reliably delivered detail patches of validation 2 containing a keypoint with a success rate of 84.3% for tips and 88.0% for tails in the face of large dataset shift. The average precision of keypoint recognition in that case was 99.0% for tips and 85.0% for tails. This high performance offsets the cost in terms of a higher distance median 26.1px for tips and 33.9px for tails.

NasNet Mobile Backbone

The evaluated *NasNet* variant is ”NasNet-A (4 @ 1056)“, proposed by Zoph et al. (2018). The feature extraction backbone has 4.23 M parameters and outperformed similar capacity models like *MobileNet-224, ShuffleNet 2x,* and *Inception V1* in experiments by Zoph et al. The notation denotes the layout of the last convolutional layer, with larger *NasNet* variants being larger in terms of filters (here: 1056) and

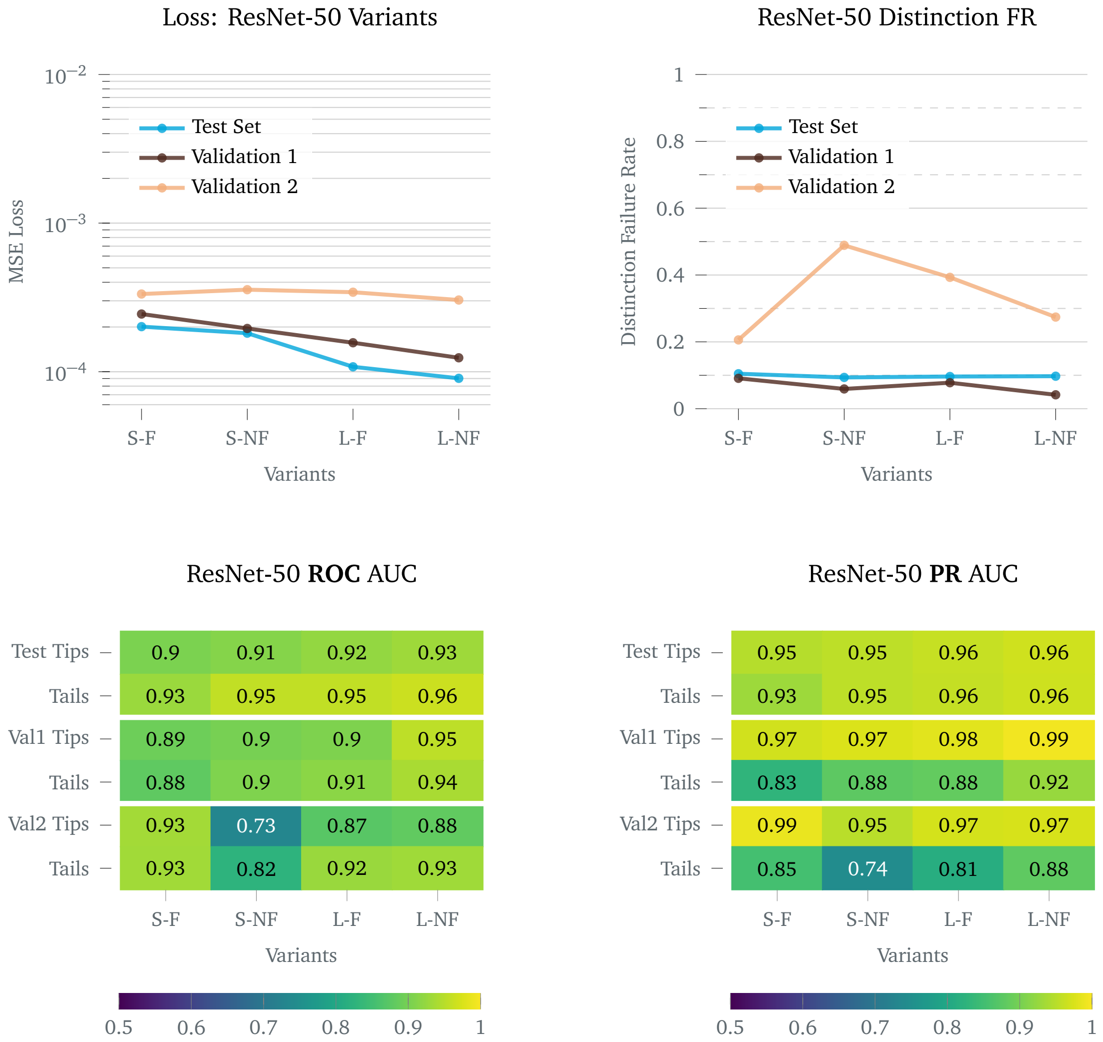


Figure 8.3. – MSE and detection results of *ResNet-50* variants – Top left: MSE loss. Top right: distinction failure rate. Bottom left: areas under ROC curve. Bottom right: areas under PR curve. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

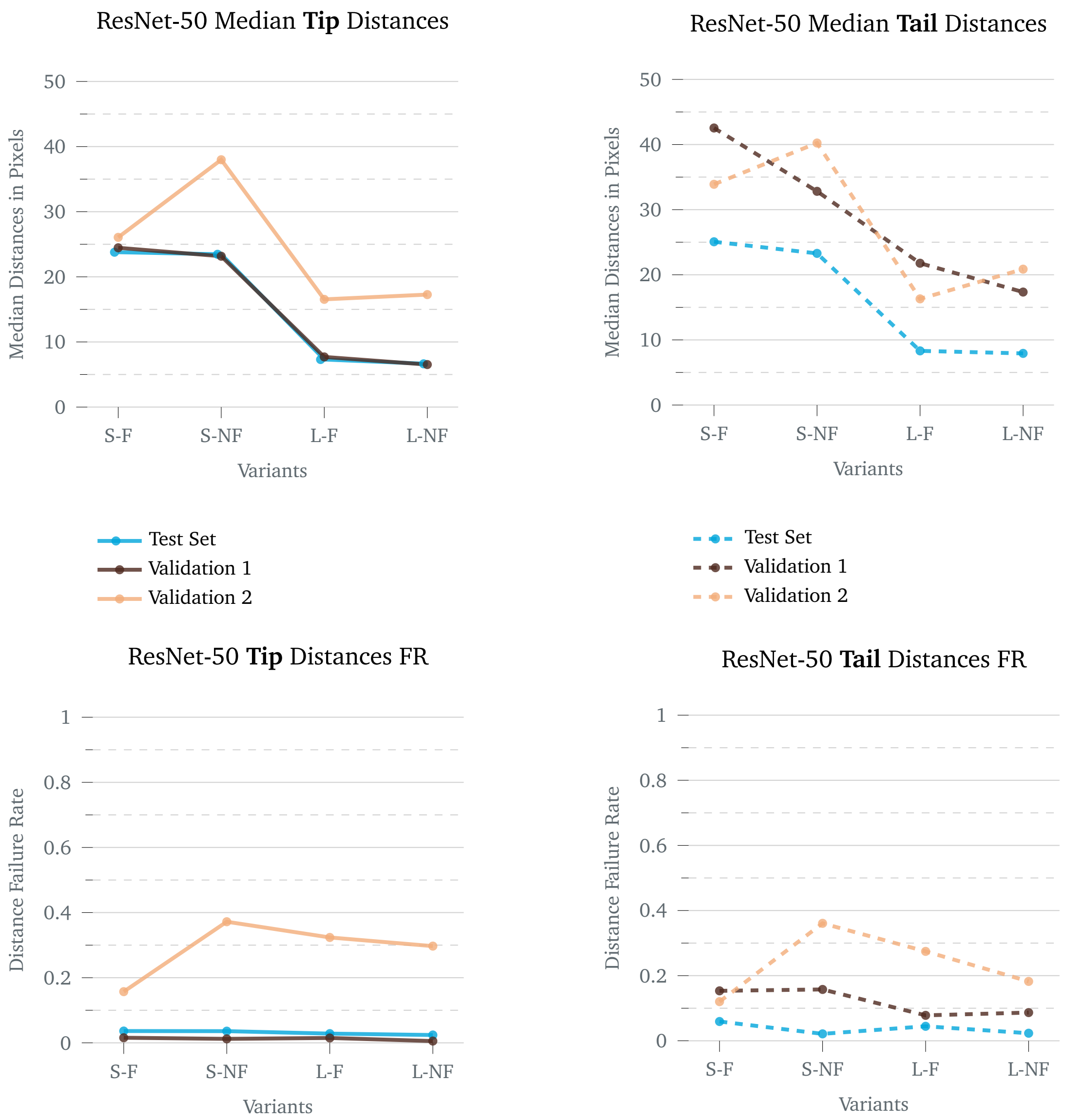


Figure 8.4. – Distance results of *ResNet-50* variants – Top left: median tip distances in pixels. Top right: median tail distances in pixels. Bottom left: distance failure rate among detected tips. Bottom right: distance failure rate among detected tails. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

cell repeats (here: 4) (Zoph et al., 2018). For freezing lower layers of this *NasNet*, the weights of 20% of them were fixed during training. The full keypoint detection variants using *NasNet Mobile* have the lowest parameter count out of all tested models (see table 8.1).

Between the small and large configurations with either frozen or unfrozen lower layers, the large decoder variant with frozen layers reached the best MSE on all sets (see figure 8.5, top left). The other models' performance on validation 2 was very close to each other, while the large decoder variants reached a better MSE on both test and validation 1.

The failure rate of distinguishing between tips and tails in validation 2 is largest for S-NF (figure 8.5, top right), while the smallest and largest variants share the lowest rates at 17.8% and 18.7%, respectively.

The average precision scores for the models in figure 8.5 (bottom right) lie close together with the exception of the tails case for the validation 2 set. Here, a clear advantage for the model with the lowest MSE on the set, L-F, emerged. The same picture is painted by the ROC AUC values for validation 2 tails. Here, the probability that the model rates a random positive sample higher than a random negative one is highest for L-F, too. It outperforms the other models for validation 1 tips and is on par with L-NF on test tips, while curiously falling behind on validation 2 tips for both ROC and PR AUC.

For distance error, L-F also reached the lowest median on all sets for tips and tails, as can be seen in figure 8.6 (top row). The model achieved a median euclidean pixel error of 12.6px for validation 2 tips and 15.8px for tails. The failure rate of detection of L-F, indicated by distances larger than half the potential detail patch size, is only lowest for validation 2 tails. For tips, the small decoder with frozen lower weights in the backbone S-F bested L-F by 2.4%.

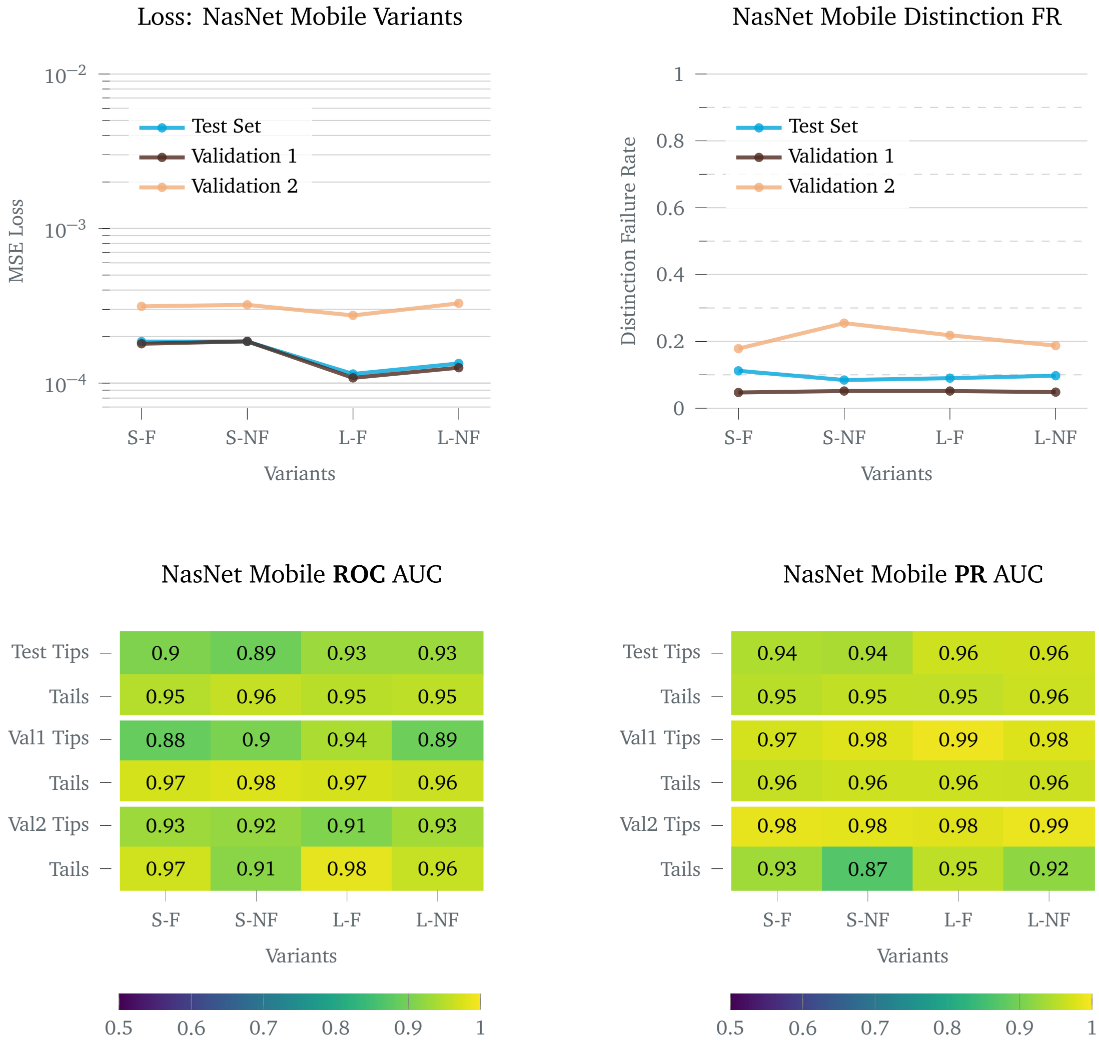


Figure 8.5. – MSE and detection results of *NasNet Mobile* variants – Top left: MSE loss. Top right: distinction failure rate. Bottom left: areas under ROC curve. Bottom right: areas under PR curve. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

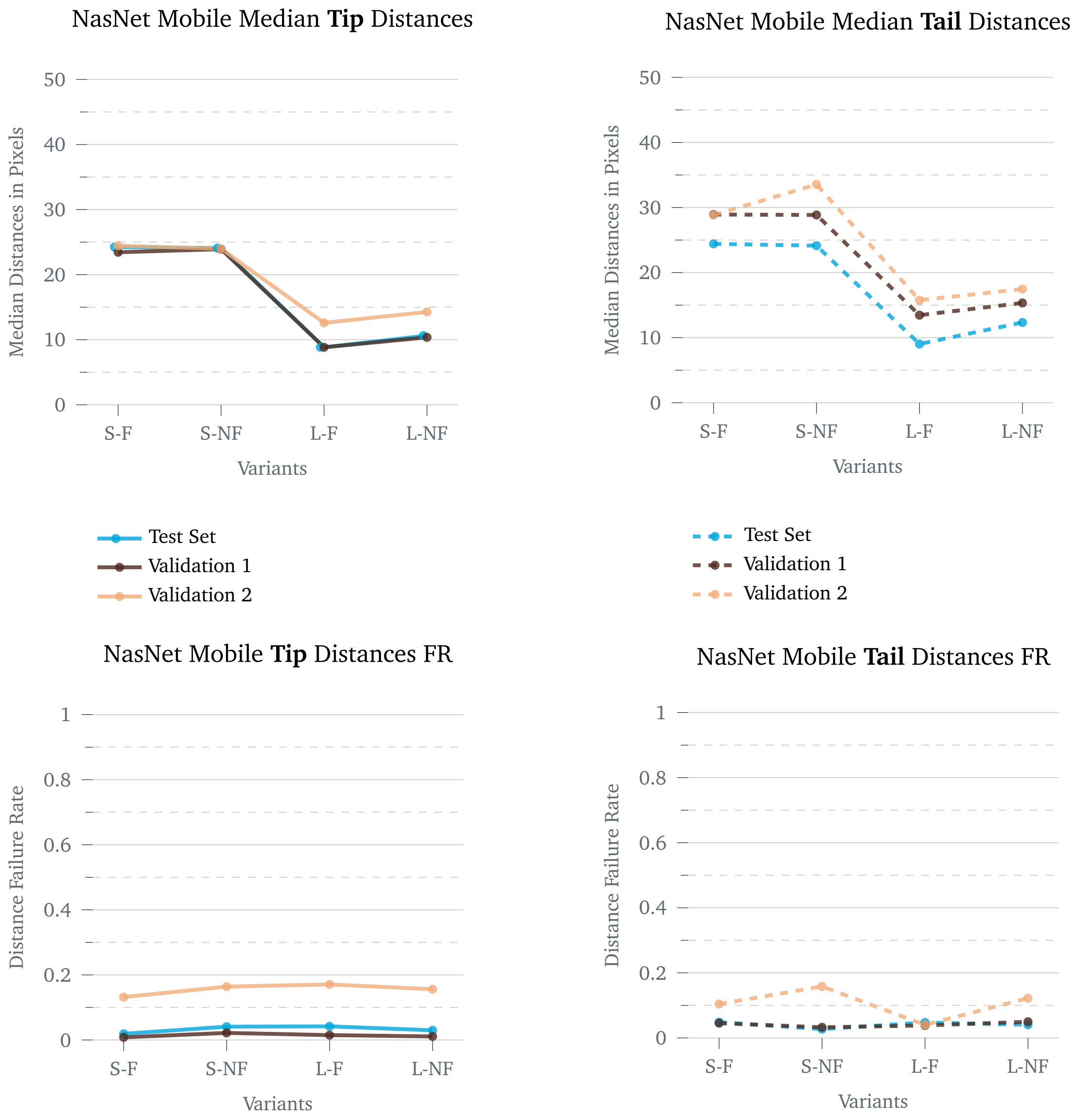


Figure 8.6. – Distance results of *NasNet Mobile* variants – Top left: median tip distances in pixels. Top right: median tail distances in pixels. Bottom left: distance failure rate among detected tips. Bottom right: distance failure rate among detected tails. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

Discussion The *NasNet Mobile* network is much smaller in terms of parameter count than *ResNet-50*. It reaches lower distance failure rates on validation 2 tips (13.2% for NasNet-S-F versus 15.7% for ResNet-50-S-F). It also consistently performs better on median distances and other metrics. The more modern architecture, optimized by automated architecture search, in combination with much lower parameter count, handles the limited dataset size better. There are no extreme outliers on any metric, indicating that irrespective of decoder size or trainable parameters, *NasNet Mobile* is well suited as a feature extraction backbone for the ROI keypoint detection task.

EfficientNet B0 Backbone

The *EfficientNet B0* feature extraction backbone is based on the smallest *EfficientNet* architecture published by Tan and Le (2019). The keypoint detection models that incorporated this architecture were evaluated with either `block1` and `block2` frozen, or with all parameters trainable. The lower batch normalization layers were kept frozen only in the first case. Otherwise, they were all kept trainable, to allow for an adaption to dataset shift, even though this meant increased convergence time.

With *EfficientNet B0*, the MSE lowered with increased model size for each set, with L-NF rating best (see figure 8.7, top left). The MSE for test and validation 1 followed the expected drop from small to large decoders, while the error on validation 2 decreased only slightly.

The distinction ability presented in fig. 8.7 (top right) is lowest for the models with frozen lower layers on validation 2. This might result from the increased focus on learning more abstract features, as the small initial filters are not malleable. Altogether, the distinction failure rate does not differ noticeably between variants.

The decision metric ROC AUC shows discernible differences for the smaller variants in the case of tip recognition. For each set, the smallest variant performs worst in this case, and the largest best, as can be seen in figure 8.7 (bottom left). For tails, there is no strongly noticeable downward variation. PR AUC shows even less deviation in figure 8.7 (bottom right). The only exception are the validation 2 tails, where S-NF outperforms the others in terms of average precision.

For distance evaluation, there is a clear improvement in measurements for the largest variant. Besides the expected drop in median tip and tail distance error for larger resolution decoders, distinction failure rates for validation 2 also follow this pattern.

Discussion In comparison to both *ResNet-50* and *NasNet Mobile*, the evaluation results demonstrate that *EfficientNet B0* has the best generalization performance. The median pixel distances and distance failure rates for validation 2 are closer to the results for test and validation 1, and overall are lower than the values of both of the previously analyzed architectures. AUC metrics show that the larger variants are close to perfect classifiers when deciding if a keypoint is present. The higher tip/tail distinction failure rate implies leftover potential for improvement in learning abstract feature representations of the keypoint classes. In summary, the *EfficientNet B0*-based L-NF keypoint detection architecture delivers ROI coordinates with a median deviation of $11.1px$ for validation 2 tips and $15.2px$ for tails, with success rates of 89.5% and 97.3%, respectively.

EfficientNet B5 Backbone

To contrast the *ResNet-50* performance with a modern architecture that has a similar amount of parameters, but larger input and output sizes, *EfficientNet B5* was evaluated as feature extraction backbone for the keypoint detection architecture. The process applied was identical to that of the *EfficientNet B0* evaluation, but the decoder architecture was changed to account for the larger input and final layers. Since the *B5* model works with larger 456x456 input images, this change was necessary to achieve similar input and output resolution on L-F and L-NF, but is a limitation on comparability.

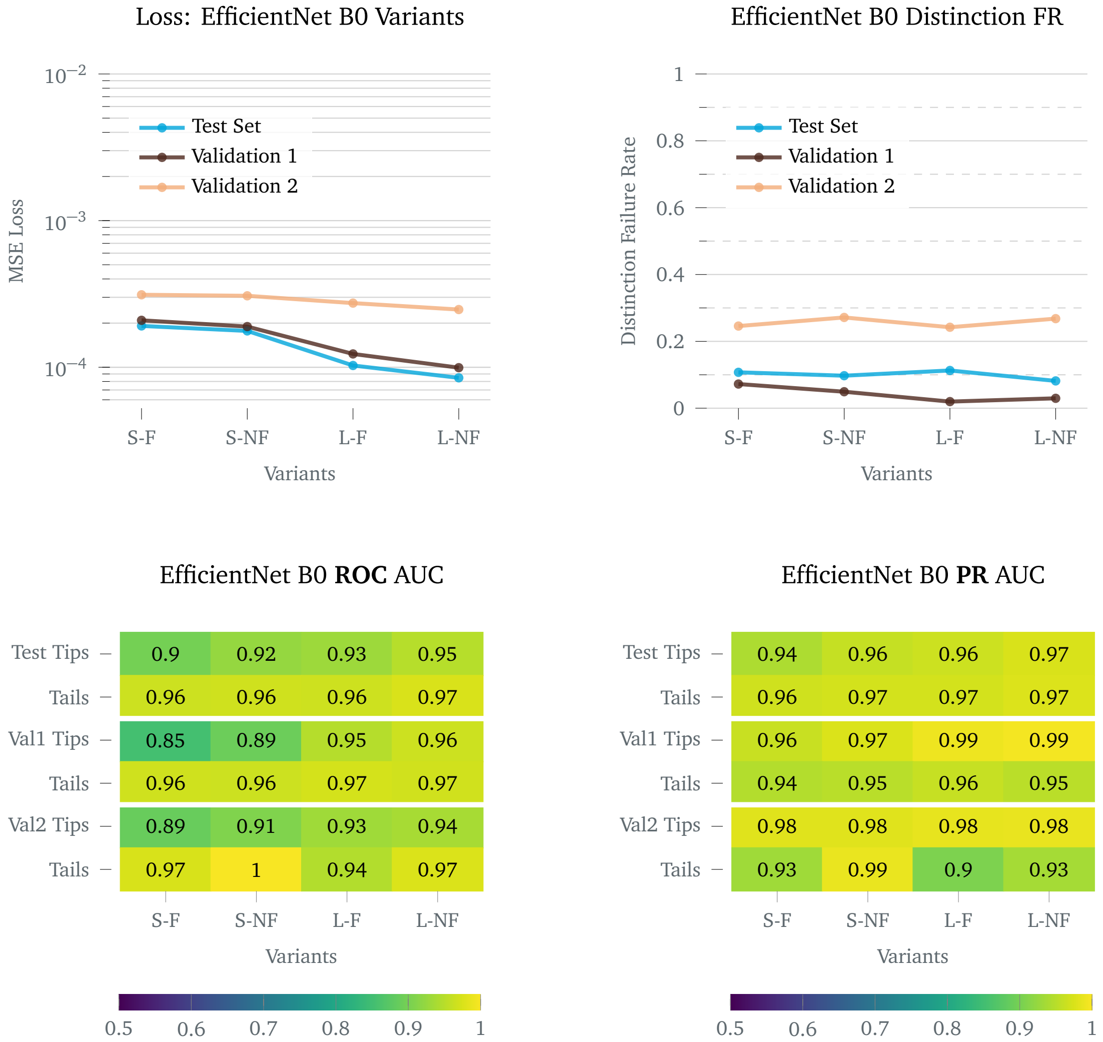
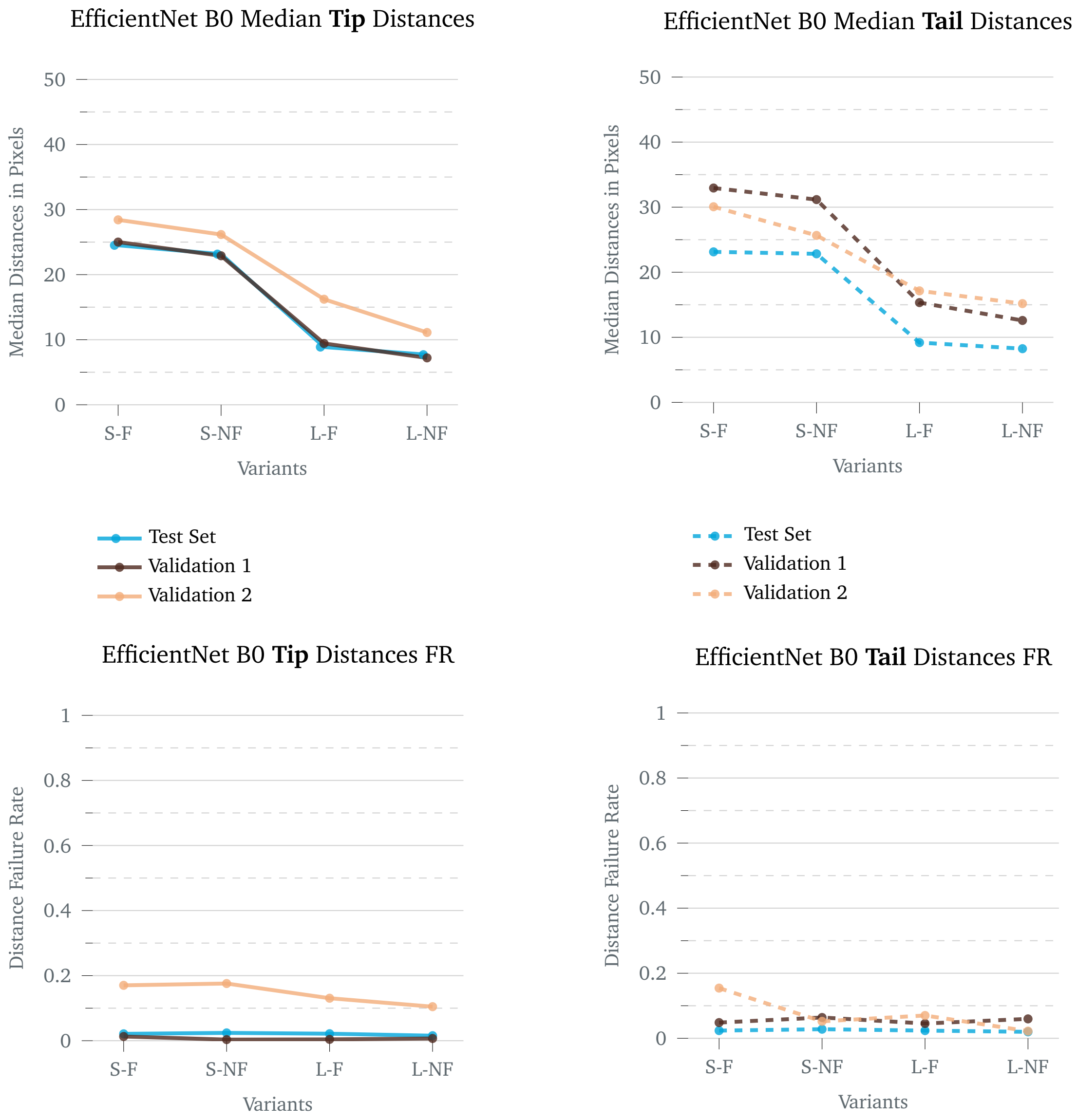


Figure 8.7. – MSE and detection results of *EfficientNet B0* variants – Top left: MSE loss. Top right: distinction failure rate. Bottom left: areas under ROC curve. Bottom right: areas under PR curve. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

Most noticeable, one of the trained variants – L-F – failed to reach a meaningful minimum at all. The network learned that the best way to conform to the optimization constraints was to always output zero, i.e. black heatmaps, which led to failure on all metrics. In the plots 8.9 and 8.10, this is either indicated by a grey hatched region where the plot line would be for distances, or by values of 1 for failure rates.



EfficientNet B0 Median **Tip** Distances

EfficientNet B0 Median **Tail** Distances

EfficientNet B0 **Tip** Distances FR

EfficientNet B0 **Tail** Distances FR

Figure 8.8. – Distance results of *EfficientNet B0* variants – Top left: median tip distances in pixels. Top right: median tail distances in pixels. Bottom left: distance failure rate among detected tips. Bottom right: distance failure rate among detected tails. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

The validation 2 MSE was worse with each trained variant that delivered outputs other than zero. The largest variant achieved very low MSE on test and validation 1 in comparison to the other network families evaluated previously, but its large error in relation to the other *B5* variants reveal that this is largely due to overfitting on the training set. This is evident by the large validation 2 distinction failure rates throughout, which are 4–5 times higher than those for test and validation 1 (see figure 8.9, top row).

While the median distances in figure 8.10 (top row) are not excessively high, here, too, overfitting produced disparity between test and validation 1 on one hand and validation 2 pixel distances on the other. The distance failure rates are prohibitively high, with the smallest variant S-F reaching the lowest validation 2 tip rate at 32.1%.

Discussion *EfficientNet B5*-based feature extractors were not able to extract powerful feature representations from the limited training set with the training regimen employed in this evaluation. In comparison to the other large parameter architecture evaluated in this study, it performed worse in terms of distance failure rate, the ROC and PR AUC metrics, and distinction failure rate. The better median pixel distances and lower MSE do not hide the fact that this backbone architecture is not well suited for the task and training set combination put to the test in this study.

8.4.2. Detail Configurations

The detail assessment followed the same general procedure as above, but focuses on pen tip keypoints. The patches extracted from the collected datasets are too small to include both pen tip and tail in the majority of cases. Additionally, for stroke reconstruction and accurate pen path tracking, the pen tip is most relevant. Four model configurations for each of four backbone architectures were trained on the full image set *POV-Keypoint*. The actual image material used as model input is different. Section 8.3.7 elaborates on the procedure that generates the patches

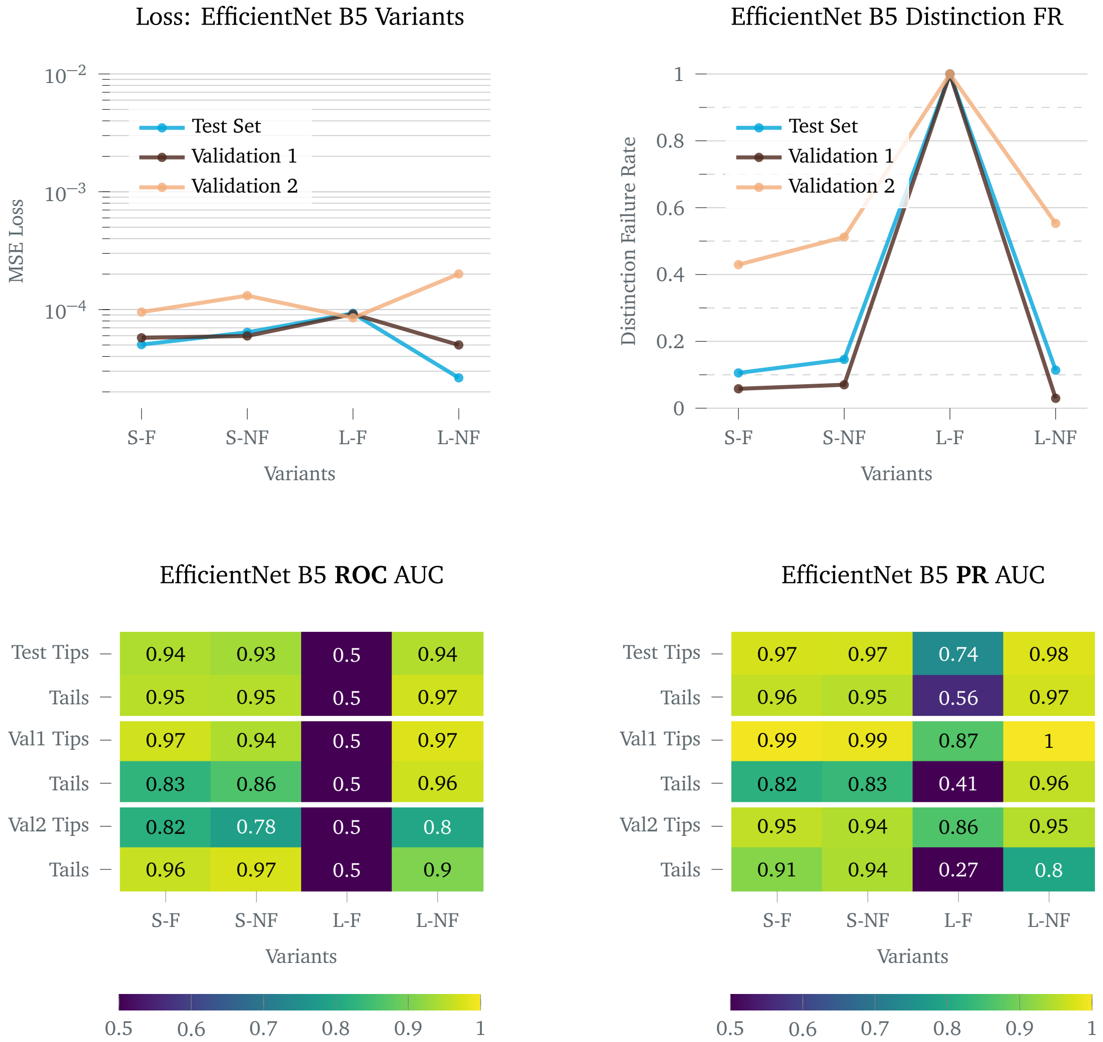


Figure 8.9. – MSE and detection results of *EfficientNet B5* variants – Top left: MSE loss. Top right: distinction failure rate. Bottom left: areas under ROC curve. Bottom right: areas under PR curve. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

for training. While a smaller decoder obviously results in lower pixel accuracy, its capacity and training influences model guesses and robustness. For this reason, the network families *ResNet-50, NasNet Mobile, EfficientNet B0, and B5* were also trained with a large decoder and with a small decoder for the detail recognition stage, all producing a single heatmap corresponding to the pen tip keypoint. The

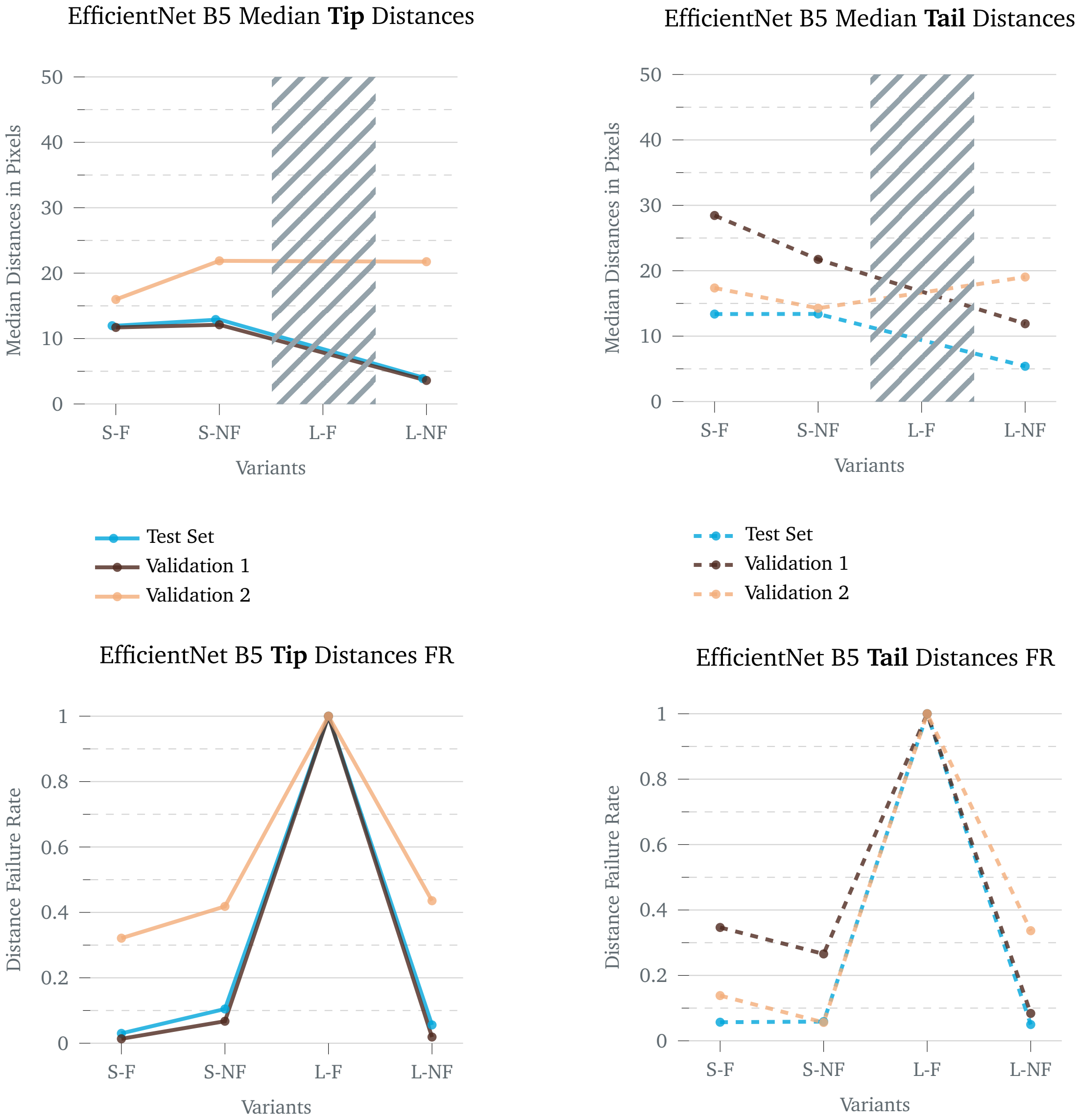


Figure 8.10. – Distance results of *EfficientNet B5* variants – Top left: median tip distances in pixels. Top right: median tail distances in pixels. Bottom left: distance failure rate among detected tips. Bottom right: distance failure rate among detected tails. All measurements are given for the *POV-Keypoint* test set, the *Simple-Keypoint*, and the *Wild-Keypoint* validation sets.

weight freezing configurations were kept the same as for the ROI stage, as the same reasoning applies for these detail detection encoder-decoder networks. This resulted in 16 trained configurations.

The visualizations of the detail stage assessment mostly follow the previous approach. The distance failure rate is not included, because in contrast to the ROI stage, there is no meaningful threshold. Since the assessment of the detail model performance concentrates on pen tip detection, the distinction failure rate is also excluded. The following paragraph explains the rationale behind the single new visualization introduced for the detail stage evaluation.

Distributions of Distances (decision metric *c*)) Since the pixel error is the most important criterion for model performance when it comes to accurately detecting pen tips, the distributions of the measured distances for all configurations and all datasets are provided as box plots. Again, only samples with a ground truth annotation representing a visible pen keypoint were considered.

ResNet-50 Backbone

The training of the *ResNet-50* feature extractor showed that in terms of loss, this architecture profits most from a combination of a backbone with frozen lower layers and a large decoder (see figure 8.11, left). The network variant L-NF (large/no frozen layers) with the largest amount of parameters performed worst in terms of loss, showing complete failure in its ability to rank a random positive sample higher than a negative one. This is indicated by the ROC AUC for L-NF being close to 0.5, i.e. a random classifier, for all sets, as visualized in figure 8.11 (right). The other networks achieve values close to one for the first two sets, while larger gaps remain for the *Wild-Keypoint* validation set.

This dataset serves as a great indicator of generalization ability: while the variants except L-NF have very good AUCs on the test and validation 1 sets, their performance clearly differs on validation 2, with S-NF exceeding other configurations. The average precision or PR AUC is close to one across the board, with again, the notable exception of L-NF.

The distance distributions visible in figure 8.12 correspond to these results. The model with best ROC AUC and average precision also achieves the lowest spread of distances in the upper quartile range, most notably on validation set 2. The median pixel error still lies above that of L-F, which is a symptom of the smaller output resolution.

Discussion The L-NF shows clear signs of overfitting. With the most trainable parameters, a much worse local minimum was found by the training algorithm, which is reflected in all evaluation metrics. The smaller decoder variant S-NF with more flexibility in the feature extractor than S-F showed the best relative performance, except for the higher MSE than L-F, which is explained by the smaller output resolution. The combined metrics of ROC AUC, average precision, and distance error distribution show that S-NF handles the *Wild-Keypoint* validation set best. The performance evaluation results demonstrate that the creation of a second, more difficult validation set was the correct choice. All variants except L-NF performed well on the test and the similar validation 1 set, while the challenging *Wild-Keypoint* set brought out evident differences.

NasNet Mobile Backbone

The second feature extractor architecture, *NasNet Mobile*, has a lot less parameters than *ResNet-50*. Here, the MSE and AUC plots show a different picture (see figure 8.13). In relation to each other, the mean squared error steadily decreases from smallest to largest model in terms of parameters. The AUC values are very close to one and are very similar between models. The only notable deviation is displayed by the ROC for the validation set 2, which is consistently lower, but shows the probability

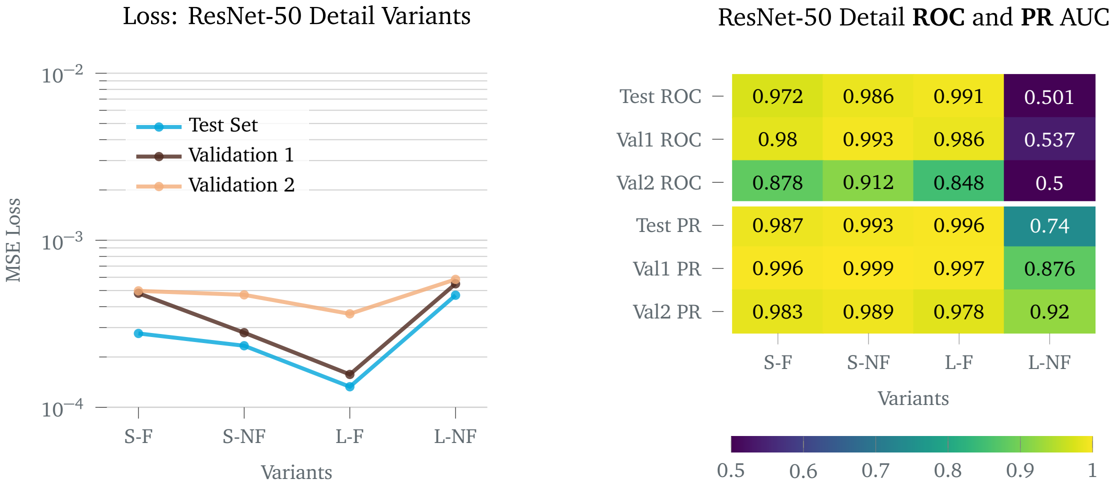


Figure 8.11. – Detail stage evaluation for *ResNet-50*. (L) MSE for all sets for each variant. (R) Top three rows are ROC AUC for all sets for each variant, bottom three rows show PR AUC. The color scale starts at 0.5, which represents a random classifier.

of the models ranking a random positive sample higher than a negative one is above 90% in all cases.

In fig. 8.14, the distance distributions exhibit the expected higher median for the smaller resolution output variants, as well as a larger upper quartile distribution for the second validation set throughout, with the smaller models packing distances more tightly. Test and first validation set evaluate to small spread and low median. The median distance for validation 2 – the most difficult case – is close to 5 pixels for S-F and S-NF, and between 3 and 4 pixels for the larger decoder variants L-F and L-NF.

Discussion No *NASNet Mobile* produces troublesome behavior, with L-NF achieving lowest MSE and lower validation 2 distance spread across lower and upper quartile. With a model that has no frozen lower layers and a large decoder in combination with a feature extractor that has a comparatively low parameter count, the training process

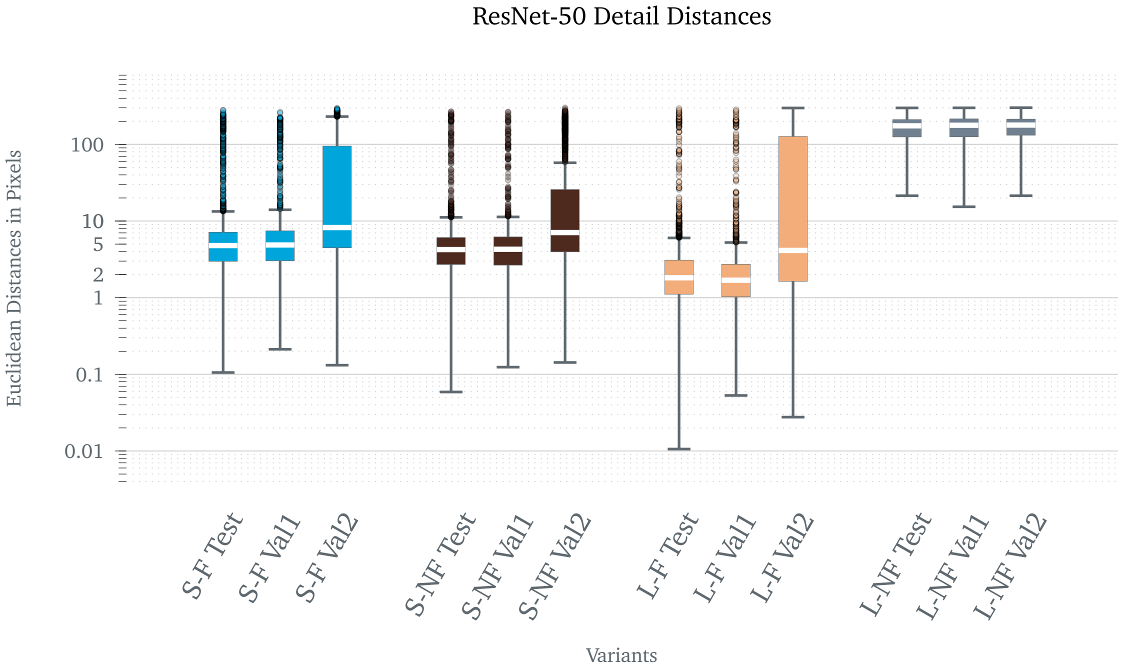


Figure 8.12. – Detail stage pixel error for *ResNet-50*. Box plots show distributions for euclidean distances in pixels between detected and annotated keypoints for each set for all variants. In the case of validation 2, when the input resolution was lower, the image was scaled proportionally so that all distances were measured in the same reference frame.

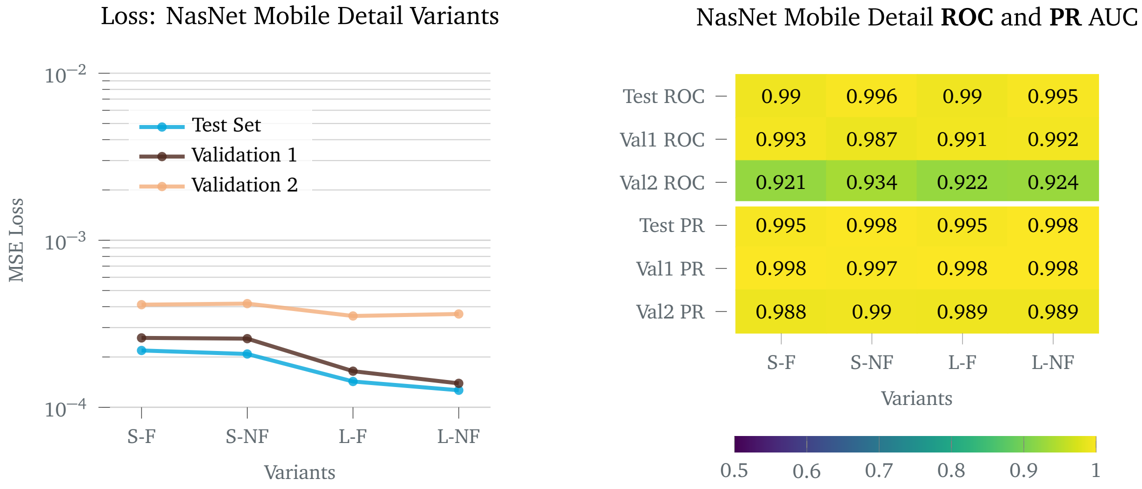


Figure 8.13. – Detail stage evaluation for *NasNet Mobile*. (L) MSE for all sets for each variant. (R) Top three rows are ROC AUC for all sets for each variant, bottom three rows show PR AUC. The color scale starts at 0.5, which represents a random classifier.

can find an overall beneficial minimum, indicating a good relationship between parameter count and training dataset size.

EfficientNet B0 Backbone

EfficientNet B0 is, too, an architecture with relatively low parameter count, when compared to the large networks in this assessment, *ResNet-50* and *EfficientNet B5*. It is more modern and overtook *NasNet Mobile* on the *ImageNet* benchmark. In terms of MSE, a correlation shows between a larger number of trainable parameters and lower minimal loss achieved (see figure 8.15, left). The L-NF variant also achieves the highest ROC AUC and average precision out of the four tested configurations, which is depicted in figure 8.15 (right). Overall decision strength is also close to one for all models, with the validation set 2 values lowest in line with expectations.

The distances box plot in figure 8.16 shows that medians for S-F and S-NF lie very close together, and the distributions for validation set 2 in particular are almost

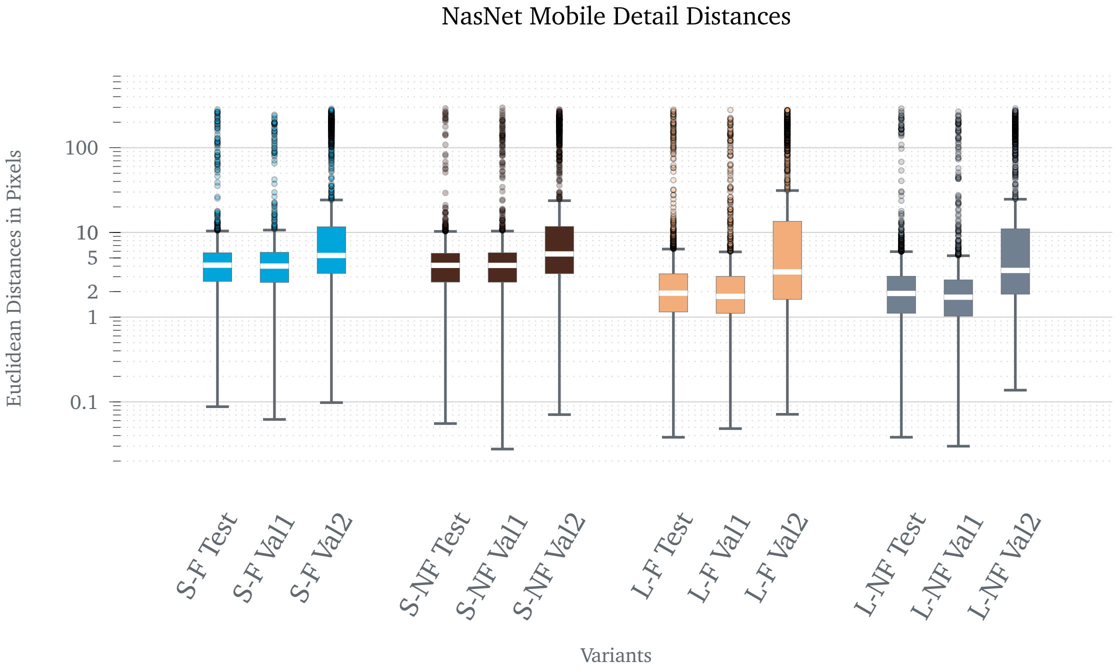


Figure 8.14. – Detail stage pixel error for *NasNet Mobile*. Box plots show distributions for euclidean distances in pixels between detected and annotated keypoints for each set for all variants. In the case of validation 2, when the input resolution was lower, the image was scaled proportionally so that all distances were measured in the same reference frame.

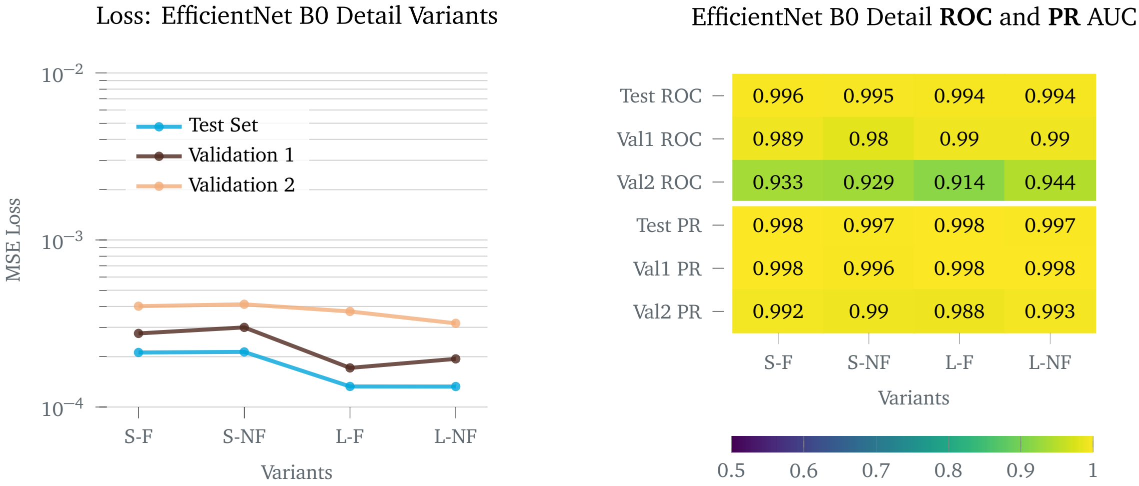


Figure 8.15. – Detail stage evaluation for *EfficientNet B0*. (L) MSE for all sets for each variant. (R) Top three rows are ROC AUC for all sets for each variant, bottom three rows show PR AUC. The color scale starts at 0.5, which represents a random classifier.

as tightly packed as for the other sets. For the larger L-F and L-NF configurations, median validation 2 pixel errors lie below 5 and 3 pixels respectively. Distribution for validation 2 is worst overall for L-F, and best for L-NF.

Discussion In the evaluation, the overall picture shows: a network with a smaller feature extractor can handle a larger decoder better, and benefits from more flexibility in training the feature extraction backbone. The generalization performance is best for the smaller variants, as is demonstrated by distance distribution, but absolute pixel error is lowest for L-NF, where the distribution is just slightly worse. The *EfficientNet B0*-based pen keypoint detection architecture demonstrates generalization ability as well as high detection accuracy.

EfficientNet B5 Backbone

To complete the model assessment, a more modern architecture with roughly the same parameter count as a *ResNet-50* in the feature extraction backbone was trained.

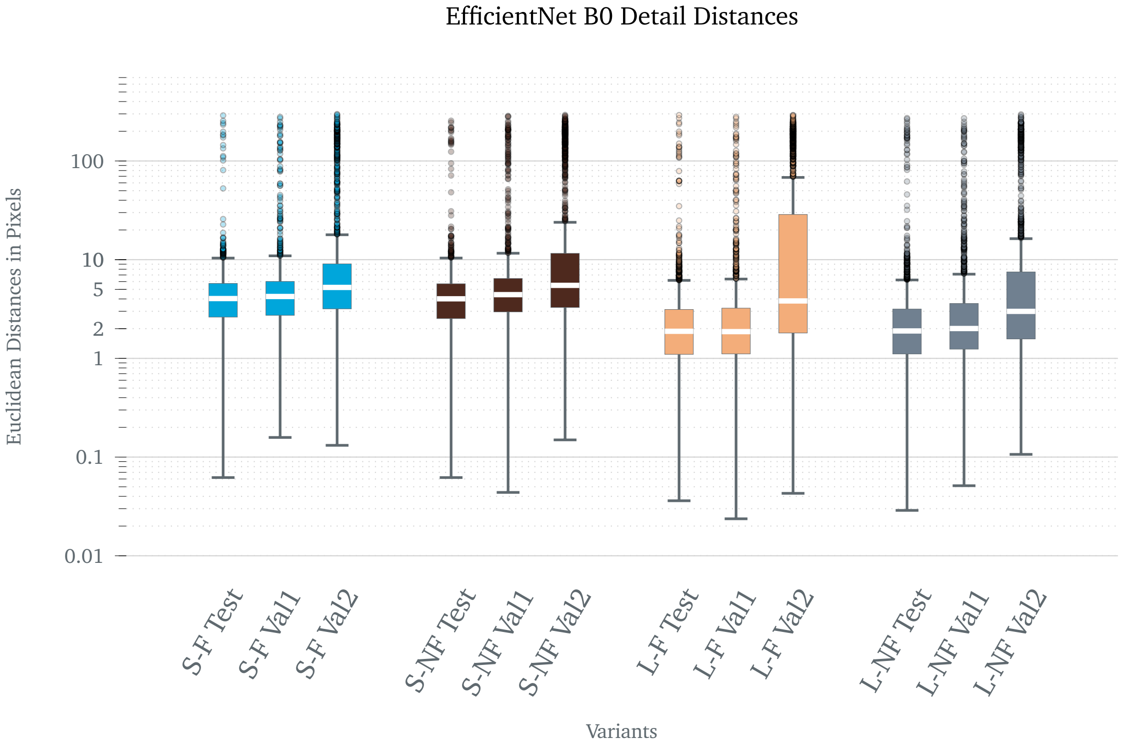


Figure 8.16. – Detail stage pixel error for *EfficientNet B0*. Box plots show distributions for euclidean distances in pixels between detected and annotated keypoints for each set for all variants. In the case of validation 2, when the input resolution was lower, the image was scaled proportionally so that all distances were measured in the same reference frame.

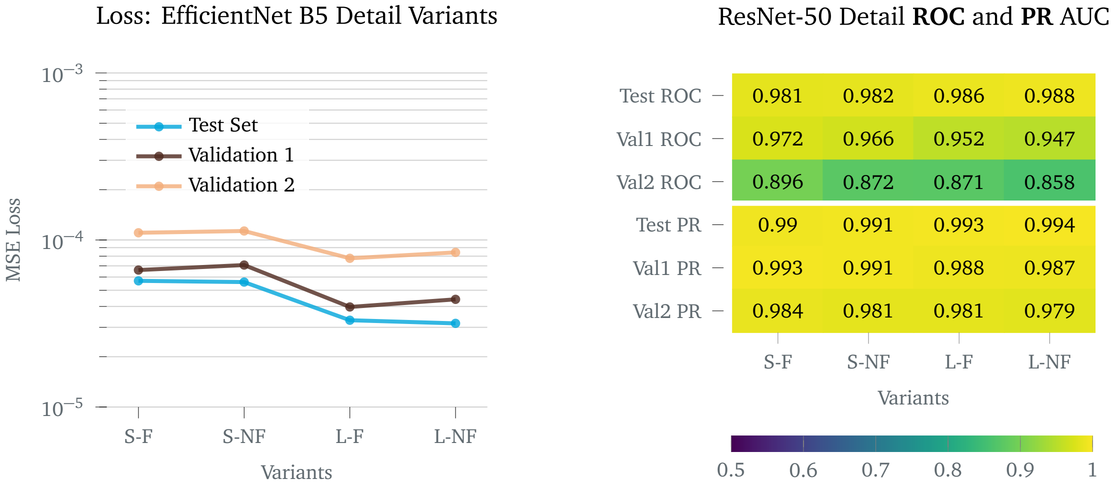
The MSE values show a dip for L-F with validation 1 and 2 (see figure 8.17, left). A clear drop in ROC AUC values from test to validation 2 and from smallest to largest model shows in figure 8.17 (right).

Distance distributions in fig. 8.18 show validation 2 spread increasing in tandem with trainable parameters, while test and validation 1 distances remain in expected dimensions.

Discussion The better MSE for L-F indicates that for *EfficientNet B5* the restriction to training only upper layers is beneficial with larger decoders. This can be explained with the large parameter count in general, for which the training procedure struggles to find a minimum with the limited training data. This is supported by the reduction in ROC AUC for the sets with data shift, i.e. validation 1 and 2, and demonstrates worse generalization ability. L-NF upper quartile limits lie above 200 pixels, which indicates unsuitability for accurate pen tip detection. The stark difference to test and validation 1 values implies the network training overfit on the training set. To sum up, the *EfficientNet B5*-based detail stage has too many parameters for the limited training data.

Summary

The detail model assessment illuminated the relationship between trainable parameters in the backbone, overall parameter count, and dataset size. For evaluating the networks tasked with detecting accurate pen tip locations, the two validation sets played an important role. The *POV-Keypoint/test* dataset deviates least from the training data, with validation 1 showing small and validation 2 exhibiting large dataset shift. All models except for the largest *ResNet-50* variant performed well on the test set and the simple validation set, sometimes producing slightly lower median on the simple set. The *Wild-Keypoint* validation set revealed stark differences between the trained nets.



	S-F	S-NF	L-F	L-NF
Test ROC	0.981	0.982	0.986	0.988
Val1 ROC	0.972	0.966	0.952	0.947
Val2 ROC	0.896	0.872	0.871	0.858
Test PR	0.99	0.991	0.993	0.994
Val1 PR	0.993	0.991	0.988	0.987
Val2 PR	0.984	0.981	0.981	0.979

Figure 8.17. – Detail stage evaluation for *EfficientNet B5*. (L) MSE for all sets for each variant. (R) Top three rows are ROC AUC for all sets for each variant, bottom three rows show PR AUC. The color scale starts at 0.5, which represents a random classifier.

In all models, the validation 2 distance median was higher than for the other sets. The difference lies in how much it improved between small and large decoders and how the validation 2 distances were distributed. The two model families with lightweight feature extraction backbones – *NasNet Mobile* and *EfficientNet B0* – performed best with large decoders and when no lower layers were frozen in the backbone. This shows that they can utilize the capacity available to generalize from the training set, and they do it best when they can adapt to domain-specific datasets by adjusting weights on all layers.

The training process struggled to find a configuration for validation 2 for variants where large backbones were connected to large decoders and with no frozen weights. This indicates overfitting problems, since good performance could be reached on datasets similar to the training data. The validation 2 set helped to expose this weakness. In contrast to the smaller networks, unfreezing lower layers did not help to adjust for dataset shift, as the learning process could not find a suitable minimum for the large amount of parameters. The magnitude of difference in distance error

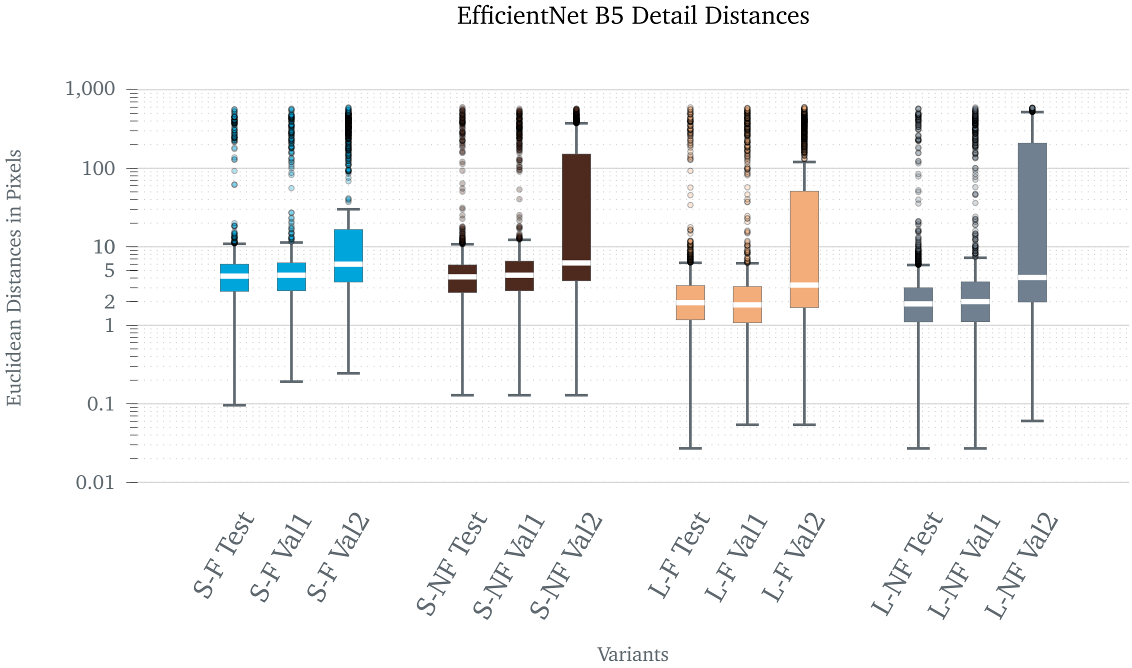


Figure 8.18. – Detail stage pixel error for *EfficientNet B5*. Box plots show distributions for euclidean distances in pixels between detected and annotated keypoints for each set for all variants. In the case of validation 2, when the input resolution was lower, the image was scaled proportionally so that all distances were measured in the same reference frame.

distribution for *EfficientNet B5* from validation 1 to validation 2 is much higher than for *EfficientNet B0*. Even though the validation 2 MSE is lower for *B5*-based nets, the better distance accuracy and the much lower parameter count indicates the latter is the best overall detail feature extraction backbone for the datasets at hand. Additionally, it takes much less computational effort to train *EfficientNet B0*.

8.5. Study Implications

Both in the ROI and the detail stage evaluation the proposed keypoint detection architecture's overall performance was analyzed with regards to prediction and decision metrics. Four different feature extraction backbone architecture families were trained for each stage, and their performance was assessed in combination with small and large decoder variants. To collect a balanced impression of the models, they were tested on the test set of *POV-Keypoint* as well as two different validation sets, which were gathered with participants not present in the training or test set. The validation 1 set exhibits slight dataset shift. The *Wild-Keypoint* set shows a large shift, since the source material was collected from a variety of devices, lighting situations, and camera perspectives.

The different levels of generalization ability of the proposed architecture variants were demonstrated in this study. An evaluation only looking at the performance on the test set, or a similar hold out set like validation 1, would not reveal the stark differences between feature backbone architectures – a model using *EfficientNet B5* with a large decoder would have left the impression of being well suited, as it reached distance failure rates below 5% and median distances also below $5px$ in test and validation 1. The additional effort of producing a shifted dataset for assessment paid off in that it exposed serious generalization issues in some models.

The study showed that for both ROI and detail processing, the *EfficientNet B0* feature extraction backbone with a large decoder produced the overall best results.

The *B0-L-NF* model architecture developed in this chapter achieved success rates of 89.5% (tips) and 97.3% (tails) for the *Wild-Keypoint* set for the task of finding the correct region of interest in an input image. The detail stage produced pen tip keypoint coordinates with a median distance error below $3px$ on a HD input, even when confronted with images using previously unknown pen types like fountain pens. These results show that this chapter contributes a novel, accurate, and reliable system with low parameter count and high flexibility for detecting pen keypoints in camera images. This subsystem represents a major building block towards a ubiquitous pen interface.

9. Handwritten Text Extraction

This chapter investigates a way to isolate the handwritten text from input images as a supporting feature for pen interface applications. By enabling the separation of handwriting from typography, the developed interface framework facilitates interaction concepts that integrate annotation workflows for printed documents into physical-digital applications. Tasks that require annotating physical elements range from reviewing scientific publications to annotating large, wall-mounted printouts of construction drawings. As Marshall (2010, p. 38) puts it, ”annotation on paper is a seamless, flexible, and well-developed practice“. She contrasts this with electronic annotations, which can break attention through shifting input modalities and force specific types of marks, but can open up a host of possibilities like ”inter- and intra-corpus linking, search capabilities, and analytic tools“ (Marshall, 2010, pp. 38, 72).

Besides supporting annotation interaction concepts, the handwritten text extraction can help with the processing of tracked pen strokes because ink records can be identified in images. The novel method of extracting handwritten text proposed and examined in the following pages formulates the isolation task as an optimization problem that is solved using supervised training of a neural network.

9.1. Related Work

The text extraction problem touches research in several image recognition tasks, but is most closely aligned with segmentation efforts. Today's neural networks are capable of distinguishing many classes and producing region masks for them. Region proposal networks for object instance segmentation (Ren, He, Girshick, & Sun, 2016; He et al., 2017) and feature pyramid based models (Kirillov, Girshick, He, & Dollár, 2019; Qiao, Chen, & Yuille, 2021) for panoptic segmentation incorporating both instance and semantic segmentation are among the most popular methods.

Text Spotting

There are also approaches more specifically concerned with text detection and recognition. In recent years, locating machine printed text or *text spotting* in input images developed as a research direction in ANNs. Models for scene text spotting can be broadly categorized as either character- or segmentation-based (Y. Liu et al., 2020). Bissacco, Cummins, Netzer, and Neven (2013) leveraged the output of three different non-neural text detection systems to produce region candidates, which are filtered by maximizing text recognition scores. Jaderberg, Simonyan, Vedaldi, and Zisserman (2016) combined neural network region proposal methods with word classification to identify phrases and their bounding boxes in photos and videos. An alternative to the previous multi-stage approaches, the end-to-end system by Hui Li, Wang, and Shen (2017) employed recurrent neural networks to solve text detection and word recognition. It was subsequently improved upon by P. Wang, Li, and Shen (2022). To make real-time detection and recognition feasible, Y. Liu et al. (2020) introduced a light-weight architecture that makes use of bezier curve detection to allow for flexible text spotting.

These text spotting systems have in common that they are only concerned with non-handwritten text regions. Handwritten word or text spotting has too been explored, especially for the case of identifying specific keywords in documents. Research in the context of large and handwritten historical corpora has produced methods to index and decipher content semi-automatically. Manmatha, Han, and Riseman (1996) proposed an early method for indexing historical corpora by matching equivalence classes for word images to user-provided text equivalents. More recently, hidden markov models (Fischer, Keller, Frinken, & Bunke, 2012) and recurrent neural networks (Frinken, Fischer, Manmatha, & Bunke, 2011) have been used for keyword spotting. Toselli, Vidal, Puigcerver, and Noya-García (2019) proposed an extension to existing systems with boolean multi-word queries, making searching large image collections more efficient.

Document Binarization

In the field of document binarization, many researchers have dealt with the problem of separating handwritten or machine printed text from interference. The most basic segmentation of an image is a pixel-level two-class or foreground to background distinction, a core task of traditional image processing. Global thresholding methods like the algorithm by Otsu (1979) perform well on some kinds of clean input, but require preprocessing (Sulaiman, Omar, & Nasrudin, 2019). Common issues in historical document binarization identified by Sulaiman et al. (2019) also afflict video streams of writing processes – they stem from lighting artifacts, blur, smudges, bleed-through, and contrast variation. The ICDAR competition results on document image binarization (DIBCO) provide an overview of recent methods for overcoming said issues, with a non-neural winner using three clustering approaches and a voting mechanism in 2019 (Pratikakis et al., 2019). Localized patch-wise binarization using neural networks was performed by several runner-up methods.

Task at Hand

However, the application scenario of the above methods is in contrast to the task of text extraction from a video stream recording a desk or whiteboard environment in the pen interface context. In a ubiquitous pen interface, recorded scenes might be even less predictable. Therefore, existing systems do not solve the handwritten text extraction problem, but provide valuable conceptual and architectural motivation for investigation. Most importantly, the distinction between handwritten text and typography is not part of the design of aforementioned models. This, and the removal of other interference unique to the pen interface context, like the pen occluding part of the text, makes an investigation worthwhile.

When handwritten text extraction is employed in an end-to-end pipeline from recording to reconstructed strokes and text, a binary or bounding box result may either not include or discard helpful information for later steps. As an added benefit for researching a specialized building block, feature maps created by networks that received domain-specific training can be leveraged for support of other modules like pen up/down classification before any thresholding, i.e. before any decision regarding pixel class is made.

9.2. Approach

The network used for the handwritten text extraction is based on the encoder-decoder models of the previous chapter and borrows from approach 10a in Pratikakis et al. (2019) as well as Vo, Kim, Yang, and Lee (2018) in its local-to-global nature. In the spirit of finding the simplest, yet high-performance baseline, multi-scale and attention based aspects are not considered in this exploratory study.

The analysis focuses on the ability of an encoder-decoder network to produce probability heatmaps that highlight all handwritten text as a region of interest and discard all other information. In a supervised training approach, an adapted data set consisting of binarizations of *POV-Keypoint* images for training is used as training target. To provide the highest possible detail, just as the detail part of the keypoint detection network (see section 8.3.3), this model operates on original-resolution patches of the input images. Consequently, it relies on an additional system providing the cutouts. This can either be a grid-based subdivision of video stream input, or a more elaborate approach considering only the area around extracted pen tip patches. For the latter, the ROI pen keypoint detection networks studied in section 8.4.1 can provide reliable input.

The text extraction capabilities are firstly evaluated on the *UbiPen-Binarize* dataset as is. A second assessment is done using the dataset augmented with *lorem ipsum* text, to highlight the capabilities of the model to discern between handwritten and typed text. For this study, first the network trained on the *vanilla* binarization dataset is tested on images with *lorem ipsum* text on them. Then, another model is trained using the original and augmented data, to establish if such augmentation is helpful for training text extraction networks.

9.3. Model

The architecture is based on the *EfficientNet B0* feature extractor that proved reliable and computationally efficient in keypoint studies (see section 8.4.1). The latent space features are used as input for a decoder that mirrors the layout used for keypoint detection. Four transposed convolution layers combined with feature map pooling produce a heatmap that reflects the network's guess to what class a pixel belongs to. This results in a two-dimensional floating point image that can be used for further processing.

9.4. Dataset

Patches from the binarized ground truth full resolution images were sampled randomly for training. Because of the high cost of annotation, the resulting training dataset for text extraction is smaller than that for ROI detection and pen tip extraction. The random sampling produced 5341 patches and their binarized ground truth. In contrast to the per-subject train/test split in the other *UbiPen* datasets, the input images for training and testing were split by taking 20% of the samples as test set. This was done to put emphasis on providing the network with a better mix of ink colors for training, as participants used one color per task.

The dataset was augmented using *RandAugment* operations, as described in section 8.3.7. For the *lorem ipsum* augmentation, the whole dataset was additionally imprinted with machine written text at varying sizes and colors. This does not entirely reflect the actual scenario, where text would not obscure the writing hand, but approximates it in an economical manner.

9.5. Training

The model was trained with the *UbiPen-Binarize* dataset using the Adam optimizer (Kingma & Ba, 2014) with a step-wise learning rate reduction. Optimization was guided using the mean squared error loss, following the keypoint heatmap training configuration introduced in section 8.1. The weights producing the lowest validation error during 200 epochs were used for evaluation.

9.6. Evaluation Approach

While MSE reduction on the test set during training can be an indication of convergence and of better quality predictions, minima in the error surface can still represent undesirable weight configurations, as additional requirements not factored into the MSE calculation still exist outside the optimization process. In the previous assessment of keypoint detection architectures, this was mitigated by including a variety of metrics that shed light on different aspects of model performance. For binarization, the ground truth creation is more difficult, still. In contrast to marking keypoints manually, resulting in some relatively small location uncertainty, marking pixels as either belonging to the text class or not discards the nuances and imaging difficulties involved.

Ground Truth

Smith (2010) investigated semi-automated and manual methods of binarized ground truth creation and concluded that above a certain level of fit, the ”best“ results are preference-based depending on the way the ground truth was created. The ground truth for the *UbiPen-Binarize* set was created using *Adobe Photoshop CS 6*, applying several combinations of filters, adapted to the individual image characteristics. In this way, the training input was designed to provide an opportunity to the network to learn to filter out shadows, adapt to varying lighting conditions, and ignore unwanted structures.

Metrics

As described in section 9.1, the text extraction task resembles a segmentation process. For semantic segmentation, one of the most commonly used metrics is IOU, which was defined for pixel-wise evaluation by e.g. Everingham, Van Gool, Williams, Winn, and Zisserman (2010) and which T.-Y. Lin et al. (2014) used for assessing segmentation quality for their large *COCO* dataset, highlighting the proportion of correctly identified pixels tied to their location. They "decouple segmentation evaluation from detection correctness" (p. 752) by discarding incorrect detections for calculating the IOU, ignoring samples with an IOU below a certain threshold. The evaluation in this chapter follows this approach, reporting both detection quality and correctness. The latter will be determined using the balanced accuracy metric, as it takes into account true and false positives as well as true and false negatives. Together, IOU and BA provide a succinct impression of algorithm performance.

Another metric for object detection and segmentation, average precision, requires a ranked output of class probabilities for a given input image (Everingham et al., 2010). According to Arnab and Torr (2017, p. 447), "[i]t does not require, nor evaluate, the ability of an algorithm to produce a globally coherent segmentation map of the image". The IOU is a pessimistic metric that can penalize small pixel errors when foreground objects are small and can not be easily exploited by predicting all positives or negatives, so it is assumed sufficient for this evaluation. Since the background class detection is trivial and would skew the assessment, the IOU reported in the following is only taken on the foreground or text class.

The key challenge to evaluating the text extraction approach is that for the use of segmentation metrics, a threshold needs to applied to the output heatmaps of the neural network. This threshold brings with it all the problems decision making based on computed probabilities carries. Nonetheless, in the face of automating a future decision process, an algorithmic approach is necessary. For this evaluation, the heatmaps were binarized using several established thresholding algorithms and the results compared to using the binarization algorithms directly without the feature extraction duties of the neural network.

Implementations of current document binarization approaches like the top result in the benchmark by Pratikakis et al. (2019) are not publicly available. Semantic segmentation networks do not posess a handwriting class. Both these factors and the novelty of the proposed interface make comparison with existing approaches difficult, so a basic comparison with traditional image processing approaches was chosen. In order to motivate research in this area and to give other scientists the opportunity to improve upon the results presented here, the dataset and trained model weights will be made available publicly.

9.6.1. Results

To produce the binarized network output suitable for calculating both IOU and BA, the heatmaps were postprocessed with standard thresholding algorithms like Otsu's (Otsu, 1979). For brevity, only the results of the best variant – using Otsu – will be reported. Noise reduction postprocessing was applied in the form of a minimum foreground pixel count of 1% for all algorithms to avoid overly pessimistic results. Otherwise, when the thresholding approach produced a negligible amount of foreground pixels, the IOU would be reported as 0 and thus heavily skewed.

The evaluation scores are listed in table 9.1. Together, balanced accuracy and IOU help to paint a clear picture of algorithm performance. Most noticeable, the reported IOU for thresholding the input image with Otsu without using any neural network is above 80%, which alone would give the impression of a highly capable algorithm. Considering the fact that match quality is assessed with a threshold on IOU of 0.5, true and false negatives need to be taken into account, too. The balanced accuracy shows for all basic image processing approaches that they do not reliably extract handwritten text from *POV-Keypoint* images. While in some cases, the produced binarization matches the ground truth well, it fails to capture true negatives and can not safely discern between pen and text. This can be gathered from the BA scores hovering around 0.5.

For the test set of *UbiPen-Binarize*, the network trained without typography augmentation (*TE-B0*) matches two thirds of the ground truth text area correctly on average, while reaching a balanced accuracy of around 79%. This by itself could be considered an adequate result in relation to the limited dataset size, as it extracts a large portion of relevant features. Faced with typography laid over the input images however, the network fails to perform in most cases. While the quality of matches stays at the same level, a lot less samples could be identified correctly. In contrast, the network trained with typography augmentation (*TE-B0-Lorem*) only displays a 3% drop in BA between both test sets and improves upon the IOU in both cases, performing better on the basic test set than *TE-B0*. This documents that it is not merely overfitting the training data.

Supporting this argument is the qualitative assessment of *TE-B0-Lorem* heatmaps on the validation 2 dataset, visualized in figure 9.1. Even in challenging conditions with stark contrasts between light and shadow as well as conflicting edge information, the heatmaps produced fit the handwritten text. However, the network is not entirely successful in segmenting out the pen tip when it is the same color as the text. One aspect of handwritten text extraction as performed here that is representative of a systemic problem is shown in figure 9.1 (top right). The participant's pen has a design that resembles decorative, handdrawn patterns. The neural network recognizes these somewhat correctly as handwritten lines, but it of course is a false positive detection in the context of a pen interface application. A possible way to solve this would be a ROI detection that detects and masks out objects on a global scale, providing the detail detection with ”clean“ patches to interpret.

In summary, the proposed handwritten text extraction approach was shown to heavily outperform standard imaging algorithms and produce promising results that are both reliable and have adequate quality. Future work includes enlarging the training corpus – Devising more challenging augmentation schemes beyond affine transformations or histogram operations, like the overlaid *lorem ipsum* text, was fruitful and motivates further research. Additionally, ROI mechanisms could fix systemic problems regarding relevance of handdrawn lines to the application context.

9.6.2. Limitations

The evaluation is limited to a dataset without significant shift. Qualitative assessments with the validation 2 set were done to mitigate this, but cannot replace a thorough study with additional ground truth material. Binarizing the validation 2 set manually is one measure to alleviate these limitations. Another step to support the generalizability of these findings would be to compare performance more extensively with other architectures, which would involve adapting and training other models to the dataset and supervision concept of this method. Nonetheless, this chapter has shown that this contribution represents a reliable approach to extract handwritten text from input images with a variety of ink marks, pen types, and image quality issues like blur. It points the way towards a class of pen interface applications that consider the written ink, like archival tools and annotation suites.

Approach	Test Set IOU	Test Set BA	Lorem Test Set IOU	Lorem Test Set BA
Otsu	**80.26**	48.72%	64.85	50.00%
Local Otsu	76.95	49.83%	65.36	50.00%
Sauvola	62.53	50.00%	59.52	50.00%
Niblack	57.36	50.00%	51.97	50.00%
TE-B0	66.51	78.99%	66.31	24.75%
TE-B0-Lorem	67.19	**81.16%**	**67.38**	**78.16%**

Table 9.1. – Results of the text extraction evaluation in percent. IOU signifies the intersection over union between ground truth and binarized inference, or, in the case of the standard algorithms, binarized input image. The threshold for including the IOU in the quality assessment was 0.5. This can skew results as it can lead to high values of average IOU even when most samples are discarded. BA is balanced accuracy, averaging true positive rate and true negative rate at the pixel level, and helps interpreting IOU results. The *Lorem* test set includes random structured text superimposed on the input data for the algorithms and the networks.

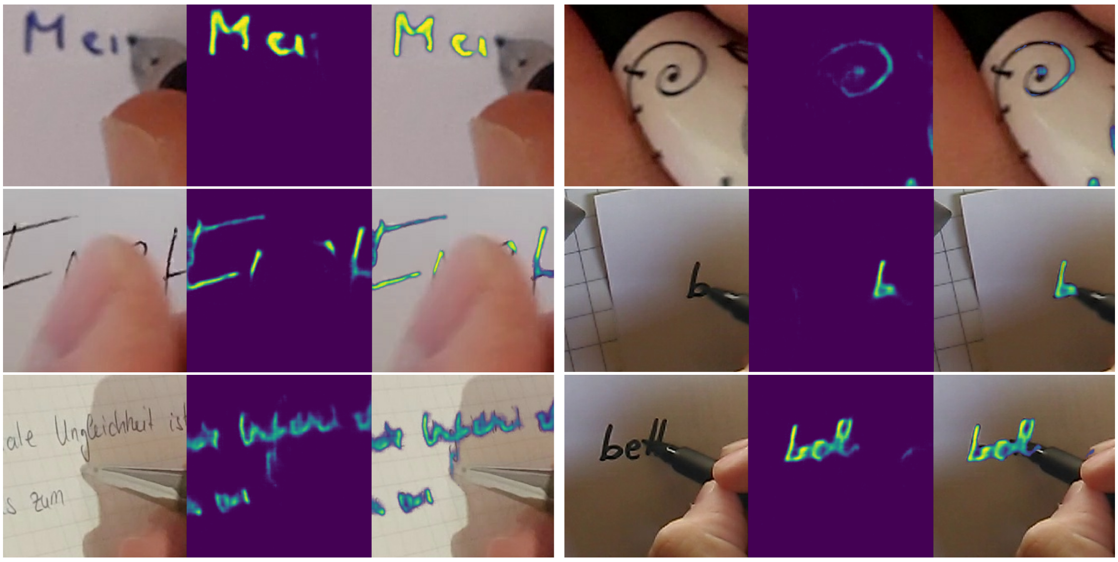


Figure 9.1. – Text extraction example results from the validation 2 set. The top left image shows solid extraction in the basic case. Text partially occluded by fingers is successfully marked in the left middle image. More challenging scenarios follow: The bottom left image shows good handling of large brightness variations through shadows. The sketch-like design of a participant's pen are wrongly identified as handwriting in the top right image – this illustrates the limits of the method, as in some cases, print mimicking hand drawn elements is not discarded. In the last two images on the right, while overall extraction is successful and ignores squared paper and sharp shadows, the network does not manage to exclude all of the black pen tip from the black writing.

10. Sequence Analysis: Pen State Prediction

The keypoint regression models developed in previous chapters form the basis for an accurate detection and localization of the pen tip. The text written can be segmented using the handwritten text extraction approach in chapter 9. In this chapter, the time-based aspects of writing sequences are investigated regarding their use for pen up/down prediction. Video analysis is an active area of ANN research, spanning from sequence based object tracking (Ciaparrone et al., 2020) to action recognition (Feichtenhofer et al., 2019).

Whether a pen tip touches the writing surface is not always apparent from a single picture, even for human observers. The first stroke could be occluded by the writing hand or pen, or the image could be of the very first moment the pen touches the paper. It is intuitively clear that viewing the frames before and after help understand the current up/down state.

In section 5.3.2, the general concept of RNNs was introduced. RNN types permit training with time-series, which can be one-dimensional, like audio or text data, have two dimensions, like images, or are volume-based, such as computed tomography (CT) or functional magnetic resonance imaging (fMRI) scans. They are based on specific artificial neural cell models with feedback loops that are able to ”remember“ some feature occurrences over time and ”forget“ others, guided by the training process. It is worthwhile to investigate their use in classifying pen states in the recording of a writing process because of its heavily time-dependent nature.

This section approaches the problem of pen up/down prediction by first creating a reference baseline using a time-agnostic single frame classifier that is trained on the shuffled *UbiPen-Sequence* set. This is documented in section 10.3. In subsequent sections, memory-based neural network approaches for the many-to-many sequence classification task (Sutskever, Vinyals, & Le, 2014) are explored.

10.1. Related Work

The pen state detection in itself is a classification or state prediction problem with a temporal component. It can be interpreted as an action recognition task, i.e. a task where semantic entities interact in a localized way that leaves traces in images captured over a period of time. Recent advances in the area of human action recognition have redefined the state-of-the-art using deep neural networks and broadly fall into the three categories of two-stream convolutional networks, 3D convolutional networks, and RNN approaches based on LSTM cells (H.-B. Zhang et al., 2019). They usually analyze video sequences as a whole and attach one or several labels to them indicating the recognized action or actions.

Two-Stream

Two-stream convolutional networks were introduced by Simonyan and Zisserman (2014) and use separate pathways to extract spatial and temporal features for classification. The *SlowFast* networks by Feichtenhofer et al. (2019) use a two-pathway architecture that improves on previous approaches by analyzing the input video at different frame rates – spatial structure is extracted using a few frames taken from the video sequence (slow) while temporal features are extracted with a light-weight high-frame-rate pathway (fast). Both parts of the network are joined through lateral connections from the fast to the slow section at several stages.

Tran, Bourdev, Fergus, Torresani, and Paluri (2015) proposed 3D convolutional networks that work with three-dimensional filters processing several stacked frames at once. With this approach, spatial and temporal features are analyzed in tandem. Building on this foundation, Tran et al. (2018), through analysis of various spatiotemporal convolution methods, arrived at a new convolutional block – *R(2+1)D* – that separates the combined 3D filtering into sequential extraction of features in space and time. 3D Convolution

In contrast to the convolutional approaches, techniques using LSTM units consider input videos as a sequence of separate frames where actions can be modeled by features changing from timestep to timestep (H.-B. Zhang et al., 2019). Yue-Hei Ng et al. (2015) introduced a model that takes frame and optical flow information output from a CNN feature extractor as an input for a layer of LSTM units and fuses the class score from both modalities. Donahue et al. (2017) established a simple architectural baseline for combining single frame CNN feature extractors with LSTM layers for classification. LSTM-Based

Convolutional approaches incorporating spatiotemporal analysis have immense memory requirements. A small *SlowFast* network with minimal *slow* pathway sample count of 2 is noted with ∼12.5 GFLOPs for a sequence of 64 frames. This is almost 200 MFLOPs per frame. For comparison, a single *EfficientNet B0* inference takes less than 4 MFLOPs. For the real-time inference requirements of the camera-based pen interface explored in this dissertation, especially considering partial execution on end-user hardware, two-stream and 3D convolution action recognition is not feasible as proposed in the respective publications. Real-Time Feasibility

Lastly, the use case for the network architecture investigated in this chapter is different than that of typical action recognition. Approaches like the *SlowFast* architecture use several *views* into the videos to be classified. This means they crop several uniformly sampled frame sequences from a video, and then average the class prediction. The pen interface use case requires a prediction for every single frame, while the sequence length is capped by latency considerations. Due to these differing requirements, in this chapter, several alternative light-weight architectures are explored, to gather a broad overview over real-time feasible frame classification algorithms. Nonetheless, the architectures developed and investigated here build upon several aspects of the literature discussed.

10.2. Performance Metrics

Evaluating binary classifiers such as the models explored in this chapter is a common task that has been treated extensively in literature. For a discussion of proper scoring rules and discontinuous confusion-matrix based metrics see chapter 7. For the sake of the *UbiPen-Sequence* dataset as well as the validation datasets introduced in chapter 6, it is helpful to use metrics that are robust to the class imbalances present. Although not extreme, the positive and negative cases are not distributed evenly due to the nature of the annotated source material, which was randomly sampled.

Ranking the models is done using the binary cross-entropy or negative log loss on the respective datasets. For a practical impression of their performance, the PR and ROC curve areas are reported. Additionally, the balanced accuracy is given for the classification task, accounting for class imbalances. As a sanity check, the loss and metrics of a naive predictor are given. Such a predictor would always predict the majority class for an imbalanced set or one of two classes with the chance of 0.5 for a balanced set.

10.3. Time-Agnostic Baseline

Architecture

Training a single-frame comparison network follows the spirit of the method by Donahue et al. (2017). The time-agnostic baseline uses an *EfficientNet B0* feature extractor pre-trained on the *ImageNet* dataset. The *EfficientNet* architecture was selected due to its combination of low computational effort paired with state-of-the-art accuracy regarding image classification, the task at hand. The relatively small capacity *B0* variant was chosen because of the limited dataset size and because it bested others in the analysis in chapter 8.

10.3.1. Variants

Finding a reasonable time-agnostic baseline (AB) was approached systematically by comparing the performance of several configurations of output layers. The original dense layers following the convolutional part of the network were removed, since they were only useful for classifying *ImageNet* images.

The first variant **AB-0** simply connects a single output neuron with a sigmoid activation function to the last activation layer of the *EfficientNet*. Because of the dense connection scheme in the fully connected layers following the spatial feature extraction, this variant has the least amount of parameters. AB-0

The second configuration **AB-1** includes an additional dense layer containing 1024 nodes before the sigmoid node producing the class probabilities. With this added capacity it is possible to check if modeling capability is limited by the single-neuron output in **AB-0**. AB-1

The third variant **AB-2** has two dense layers, containing 1024 and 512 nodes respectively. Two layers were chosen to explore the performance of deeper dense networks for decision making. Specifically, the goal is to see if it is beneficial to add ability for modeling multi-stage nonlinearity in the relation between the features in latent space at the end of the convolutional stage and the image class. This configuration has ~1.44 times as much parameters as **AB-0**. AB-2

10.3.2. Batch Normalization and Transfer Learning

The process used for creating the time-agnostic baseline is commonly regarded as transfer-learning, because a pre-trained *EfficientNet B0* is the starting point for further optimization using a smaller dataset. The creator of one of the neural network libraries used throughout this dissertation (*Keras*) recommends not re-training the batch normalization layers present in *EfficientNet B0* (Chollet, 2020). Batch normalization is a method employed in neural network layers to improve performance by adapting the mean and variance of incoming data (Ioffe & Szegedy, 2015).

Chollet (2020)'s argument is that ”the updates applied to the non-trainable weights [tracking mean and variance of inputs] will suddenly destroy what the model has learned“ when connecting new, randomly initialized layers. He argues this would lead to inefficient training, as important aspects would have to be relearned. The experiments for the baseline models have shown the *UbiPen-Sequence* dataset to be too different to *ImageNet* for training to converge without adapting batch normalization weights. Leaving batch normalization layers frozen lead to constant test error rates around the loss value consistent with random guessing. Reported results for the AB thus only include networks with adaptable batch normalization weights.

10.3.3. Training

With a binary cross entropy loss for training, the Adam optimizer was used to train the models on *UbiPen-Sequence* with light *RandAugment* image augmentation ($m = 6, 9$) until the test set loss did not improve further. The learning rate was scaled up in a warm-up phase for the first 5 epochs, and was then dropped step-wise during the training progress. A standard transfer-learning and fine-tuning procedure was used – First, the randomly initialized new layers were trained exclusively. Afterwards, the whole network was made trainable. Early stopping was employed with a patience period of 50 epochs, i.e. the training process allowed for 50 epochs of non-improvement before deciding the last best weights as optimum. Alternatively, when validation loss did not change within 10^{-3} for 5 epochs, the training was cut short. The relatively large patience period was chosen because of the trainable batch normalization layers that required a long period to re-stabilize.

Input Data

For the classification task, patches taken from the annotated frames of *UbiPen-Sequence* at HD resolution were used. The detail patches were sized 224x224 according to the *EfficientNet B0* input shape and centered around the pen tip locations with a uniformly random offset. The decision to use the detail patches focusing on the area around the pen tip was made because the small movements and ink changes that indicate a pen state change are lost when downsampling the full frame to the neural network input size.

Label smoothing was used to factor in the sometimes noisy labels, since the dataset only contains human-annotated pen states (Müller, Kornblith, & Hinton, 2019). To ensure unbiased metric results, the training set classes were weighted during optimization to balance the slight imbalance in positives and negatives (see table 6.3 in chapter 6 for exact positive/negative distribution).

Augmentation Magnitude

For further analysis, augmentation hyperparameters of the trained candidates were explored. The ideal *RandAugment* augmentation magnitude is smaller for networks of smaller capacity (Cubuk et al., 2020). For this aspect, $m = 9$, which is the value given by Cubuk et al. (2020) for *ResNet-50*,and $m = 6$ were tested.

10.3.4. Discussion

For a task that is intuitively difficult for human observers to perform on single images, the time-agnostic classification network *AB-2* was able to achieve loss below that of a naive predictor and reached an average precision of 0.8195 on the *UbiPen-Sequence* test set. The ROC curve area indicates that the probability of the predictor ranking a random positive sample higher than a random negative sample is 0.8425. This represents a time-agnostic classification baseline that is solidly above random

Model	**RandAugment Magnitude**	**Loss**	**PR AUC**	**ROC AUC**	**BA**	**Parameters**
Naive	N/A	0.693	0.5	0.5	0.5	N/A
AB-0	9	0.6393	0.8043	0.8465	75.24%	4.1M
AB-0	6	0.6266	0.8275	*0.8538*	*76.09%*	4.1M
AB-1	9	0.6622	*0.8378*	0.8532	74.32%	5.4M
AB-1	6	0.6266	0.8218	0.8368	74.51%	5.4M
AB-2	9	0.6225	0.8260	0.8349	74.21%	5.9M
AB-2	6	**0.6135**	0.8195	0.8425	75.35%	5.9M

Table 10.1. – The *UbiPen-Sequence* test set loss and other metrics for the agnostic baseline variants. The balanced accuracy (BA) is given for a decision threshold of 0.5.

guessing. For the decision threshold of 0.5, the **AB-2** candidate performed with a balanced accuracy of 75.35%. For the full results, see table 10.1.

The results confirmed the aforementioned findings of Cubuk et al. (2020) regarding data augmentation for the case of video frame classification. The training process with $m = 6$ lead to finding a minimum associated with a lower loss for all three candidates.

Baseline

The objective of creating a time-agnostic baseline is two-fold. One part is establishing a reference for measuring the effect of memory-based approaches. The other part is more practical – further optimization calls for model weights that represent a good starting point in the loss surface. Based on those goals, the model for further comparison was selected based on minimum loss achieved, i.e. the model that gives predictions that agree the most with the ground truth probabilities. Even though the balanced accuracy or AUC of other models might be higher, a model selected through this metric has the best predictive performance. This means the weights of the *EfficientNet B0* that were optimized as part of **AB-2** with augmentation magnitude $m = 6$ were chosen to be used for the sequence prediction models in the next section.

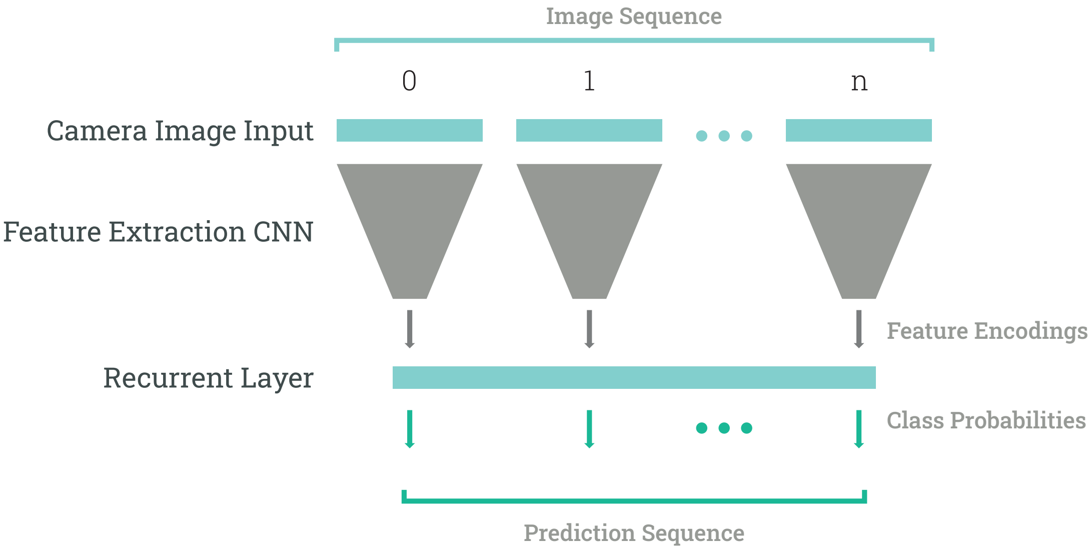


Figure 10.1. – General operating scheme of the time-based approaches F-LSTM and F-GRU in this chapter. An image sequence of length n is input into the model, which extracts features from each input image using a non-recurrent, pre-trained CNN. These compressed representations are the input sequence for one or more recurrent layers, which output a class probability sequence.

10.4. Time-Based Approaches

There are several architectural approaches currently used in video processing networks, as discussed in section 10.1 above. This section focuses on recurrent networks using LSTM and GRU cells and the spatial variant *ConvLSTM*. Although this limits the scope of this study, and some architectures outperform memory cell options in action recognition, there is a need to establish a reference for the real-time pen interface application. Literature on other approaches does not account for the concrete use cases investigated in this dissertation, necessitating either larger computing power, more memory, or several views of videos. The author assumes the recurrent approach to time-based sequences with state-of-the-art memory cells as a reasonable point of departure for prototype research regarding the proposed interface concept.

In the following, it is explored how sequence-based approaches help to improve upon a single frame predictor by incorporating temporal relationships into prediction. The problem is formulated as a task of sequence-to-sequence learning (Sutskever et al., 2014). In a production system, predictions for every frame taken of the writing process are required to potentially extract ink strokes.

10.4.1. CNN Feature Extractors Feeding Recurrent Layers

Architecture

The first variant of memory-based networks for classifying the pen state uses a flat layer of memory cells to learn time-based relationships between encodings of sequential images and the annotated pen state. The encodings are produced by flattening the output of the last layer before classification from a CNN image classifier. Spatial relationships are lost during this process. This approach is inspired by the baseline of Donahue et al. (2017) and was chosen for its simplicity and low memory footprint, keeping in line with the *simple baseline* approach taken throughout this dissertation.

Approach

For the investigation of this encodings-to-RNN approach, the best-performing **AB-2** *EfficientNet B0* model developed in section 10.3 was used as feature extractor. The effect of cell types in recurrent layers that were fed by the CNN was investigated by comparing layers made of GRU cells to layers using LSTM cells, which are more powerful than GRU cells (Weiss, Goldberg, & Yahav, 2018), but have more trainable parameters. Following Donahue et al. (2017), the recurrent layers were attached to the first dense layer trained for the agnostic-baseline. A single sigmoid neuron connected after the recurrent layer was trained to deliver the class probabilities. The general operating principle of the models in this section is shown in figure 10.1.

To determine the necessary cell count of the recurrent layer, several networks of different size were trained on the *UbiPen-Sequence* dataset on the same amount of time steps. With the knowledge gathered about capacity, the impact of window size on model performance was explored by training with 3, 10, 30, and 60 frame

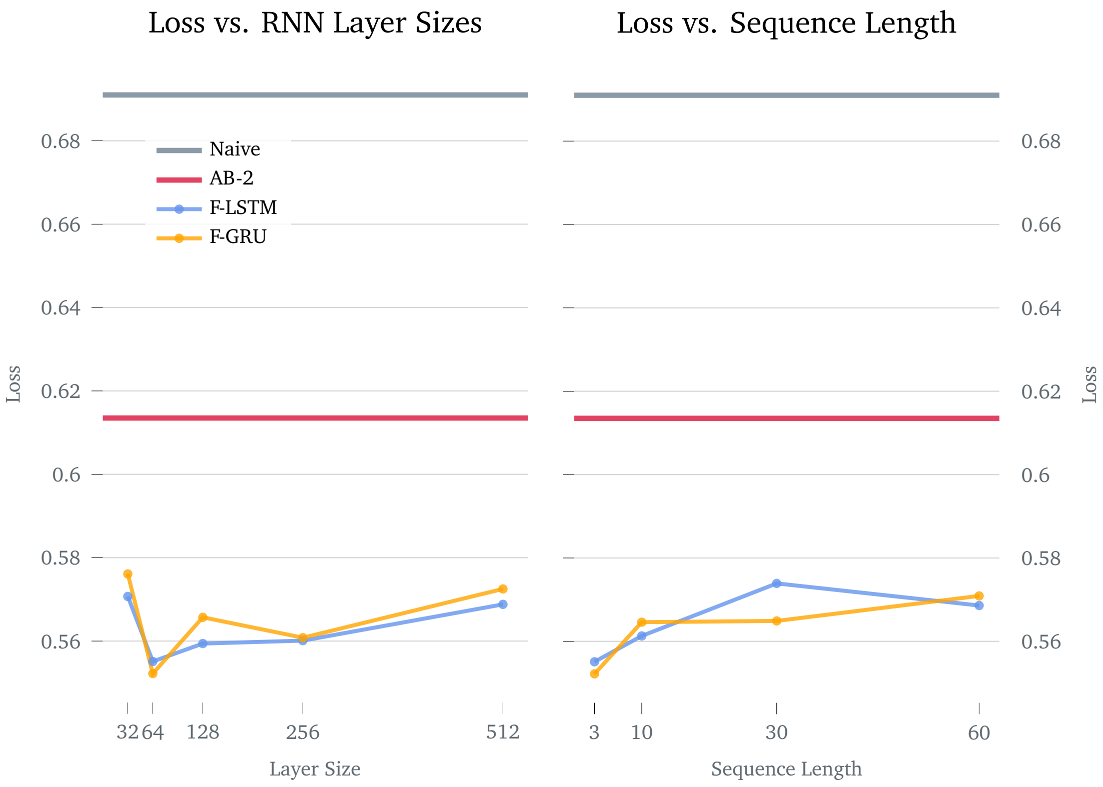


Figure 10.2. – *Left*: Binary cross entropy loss on the test set plotted against the trained recurrent layer sizes at 3 timesteps. **F-LSTM** identifies baseline **AB-2** coupled with a flat LSTM layer. **F-GRU** stands for **AB-2** coupled with a flat GRU layer. *Right*: Binary cross entropy loss of **F-LSTM** and **F-GRU** versions plotted against the trained recurrent layer sequence length, using 64 as layer size, following analysis of performance on the left. The loss axis does not start at origin in order to visualize the differences between LSTM and GRU variants as well as depict the relative improvement between naive (grey), time-agnostic (red), and recurrent candidates (blue and yellow).

sequences at once. This translates into 0.1s, 0.33s, 1.0s, and 2.0s of video time, with the latter being the maximum sequence length present in *UbiPen-Sequence*.

Training

For this fine-tuning approach, the batch normalization layers were frozen completely, in contrast to the **AB** training, as they were adapted to the new dataset already. The SGD optimizer with a binary cross entropy loss was used over Adam in the training regimen of the time-based models. SGD outperformed Adam in preliminary experiments performed on the *UbiPen-Sequence* dataset with LSTM layers attached to **AB-2**.

The last block of *EfficientNet B0* and the dense layer the recurrent network was attached to were unfrozen, allowing some adaption in the non-temporal part of the network to happen based on sequential data. The learning rate was scheduled in a step-wise fashion, following the training regimen of the time-agnostic model. Here, too, label smoothing was used to reflect the inherent uncertainty in the manual labeling.

Input Data

To create sequences for classification, again, detail patches were taken from *UbiPen-Sequence* images, which are stored in HD resolution. The dataset consists of several 60 frame sequences, which were either used as a whole, or split into smaller parts for training in cases where the impact of using shorter sequences was the point of focus.

Discussion

The models using recurrent layers were in most cases able to improve upon the time-agnostic candidate in terms of loss, meaning the the predictive performance increased. In figure 10.2, *UbiPen-Sequence* test set results are visualized for different layer sizes (left) and sequence lengths (right). This visualization was designed to detail the relative improvement, and thus does not start the loss-axis at zero, as this would obscure differences between models. The gray upper line marks the loss for the naive case, while the red middle line depicts the time-agnostic baseline loss. All variants using either GRU or LSTM cells trained for layer size and timestep count comparison reduced cross entropy loss further, although not as much as the time-agnostic baseline improved upon the naive case.

F-LSTM

The training results from the time-agnostic case proved a useful starting point optimizing the sequence classification. The loss improved most for a network with a flat LSTM layer size of 64 out of a variety tested between 32 and 512. The strongest performance was achieved by using a sequence length of 3 frames. This result is contrary to the assumption that more temporal information would always yield better insights.

F-GRU

The GRU networks performed in a similar fashion to the LSTM variants. In the best case, they trumped LSTM loss slightly. This could be explained by their reduced parameter count, which helps when data is scarce.

A case not depicted in fig. 10.2 was the *EfficientNet B0* solely pre-trained on *ImageNet* without the agnostic baseline weights. The optimization algorithm failed to find any minima and produced results comparable to random noise with several window sizes. This shows that using weights not adapted for this case does not provide a sufficient starting point for the neural network training process with this dataset structure and size. Both the frozen and unfrozen variants did not deliver any meaningful optimization.

Overfitting

In the experiments documented in this section, overfitting took place quickly, i.e. training and validation error diverged early. This is not unexpected for a fine-tuning approach, and meant that only a few training epochs were used to reach a local minimum in the loss surface. The reasons that the loss could not be reduced further are surmised to be the relatively small size of *UbiPen-Sequence* dataset, containing interdependent samples. This, combined with the noisy labels, lead to an optimization process quickly stagnating in local minima. This could potentially be solved by introducing additional information into the system, which could happen either through larger and better annotated training datasets, or through the use of model parts pre-trained on similar, but different data. Another approach is taken in the next section. Spatial information is discarded when flattening the feature extractor output for the recurrent layers tested above. This information also could support finding better local minima, and keeping it during training is investigated in the following.

The model with the lowest loss – F-GRU-3-64 – will be used for comparison with the other methods developed in the next sections. Loss and balanced accuracy were captured to provide a basis for suitability decisions when comparing all models trained in this chapter.

10.4.2. Retaining Spatial Information with ConvLSTM Blocks

Convolutional LSTM blocks were introduced by Xingjian et al. (2015). In contrast to the one-dimensional layers of memory cells used in the previous section, they are able to learn temporal relationships between inputs while retaining the spatial nature of extracted features, potentially giving them an advantage in terms of convergence and classification performance when working with spatial data, such as images (Xingjian et al., 2015). Based on this knowledge, the use of ConvLSTM blocks as combined temporal and spatial feature extractors for pen state detection was evaluated.

Architecture

The architectural concept is similar to that of the F-LSTM and F-GRU models. A ConvLSTM block was connected to the last block of an AB-2 *EfficientNet B0* pre-trained on *UbiPen-Sequence*. The notable difference was that the dense layer following the last convolutional block in AB-2 was discarded, because flattening spatial information to reconstruct it is an unnecessary complication of the training process. The kernel size was set at 3x3, which is as small as possible while still symmetrically considering neighborhood information. Following the ConvLSTM block was a single prediction neuron with a sigmoid activation function. The *sparse connectivity* of CNNs means that for a *ConvLSTM* layer's filter count of 512, the amount of trainable parameters is roughly equal to that of a 128 unit flat LSTM layer.

Approach

The approach taken for evaluating variants of the ConvLSTM-networks was guided by the lessons learned from F-LSTM and F-GRU. The sequence lengths identified as optimal were used as the basis for the evaluation. While several shorter sequence lengths were trained, the largest sequences were skipped. Loss and balanced accuracy were again collected for comparing the models trained in this chapter.

Training

The model architecture was trained 15 times, for the sequence lengths of 3, 10, and 30, and for each sequence length filter counts of 32–512 were trained. The binary cross entropy was the loss function used for optimization with the Adam algorithm, which resulted in better loss reduction than SGD, in contrast to the flat recurrent layers. Otherwise, input data, learning rate, label smoothing, unfrozen layers, and frozen batch normalization were kept the same as with F-variants.

Discussion

Overall, the performance of the ConvLSTM-based recurrent neural networks for pen state sequence classification did not differ substantially from the results gathered with the flattened layers. In fig. 10.3, the binary cross entropy loss for three timestep variants for each layer size is plotted in relation to the agnostic baseline, with the best performing F-GRU-3-64 from the section above marked separately. While there was no dramatic improvement, three models performed better than F-GRU-3-64. Two of those used the same very short sequence length of 3 time steps. However, the best network in terms of loss and balanced accuracy was the variant using 30 timesteps and a filter size of 64. This implies that the additional structural information helped find a better local minimum, and that a network retaining this information can potentially extract and use feature relationships further apart in time. The best model scores a balanced accuracy of 79.0% at a decision threshold of 0.5, which is 4.65% above the agnostic baseline. In summary, the added spatial information proved beneficial to solving the task of pen state prediction, and helped to further improve upon the agnostic baseline.

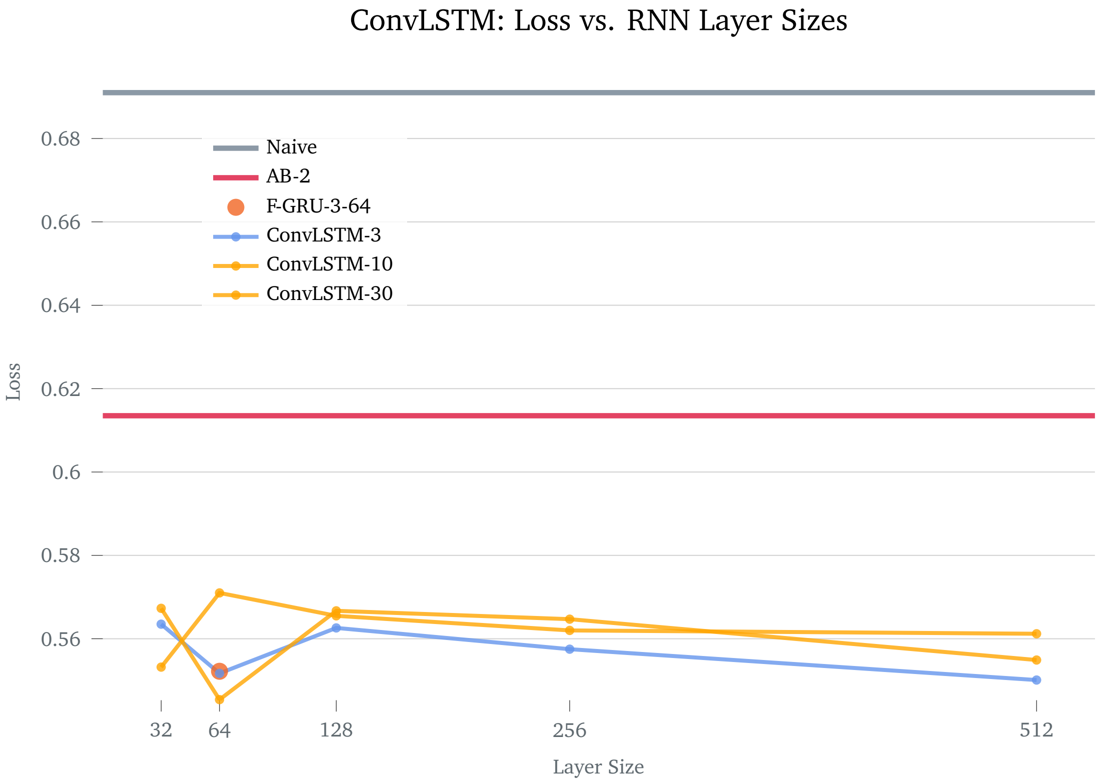


Figure 10.3. – Binary cross entropy plotted against the trained ConvLSTM layer sizes. For each time sequence length, one line shows the loss development. The single marked point represents the best result from the flat recurrent layer evaluation. As before, naive (grey) and time-agnostic (red) loss references were included. The loss axis does not start at the origin to visualize the relative differences between the candidate results.

10.5. Multi-Mode Feature Extractors

The previous sections assessed the performance of recurrent network architectures based on combining a single feature extraction backbone trained on *UbiPen-Sequence* with LSTM, GRU, and ConvLSTM layers. Results showed that using time-based information improved upon the simple classification network AB-2, and that retaining spatial relationships helped the prediction task further. In this section, adding feature encodings produced by networks trained on different datasets to the previously analysed recurrent architectures is investigated. In this way, two different interpretations of the input data are presented to the recurrent stage of the combined model.

Combined Classifier

In the first of two experiments, the combination of a recurrent ConvLSTM classifier with a keypoint detector backbone was examined. Training time for the combined network was a lot higher, since the feature extraction backbone is double in size and the recurrent part also has a higher node count. This is owed to the output of both *B0* extractors being concatenated before being input into a ConvLSTM stage, leading to a much higher number of connections. Memory constraints prohibited training on time sequences larger than 30 frames. Training was performed with the keypoint detector frozen, as the goal was to assess the impact of added information on the later, recurrent stages. The B0-backbone of the ConvLSTM classifier was kept adaptable in the last block to allow for some flexibility. The network did not manage to improve upon the other recurrent approaches with a test set loss of 0.5769 and a balanced accuracy of 76.58%.

Combined Keypoint Detector

The second experiment aimed at exploring a different perspective on the data also failed to deliver better results than previous best approaches. While using the same combination of classifier and keypoint detector backbones, an upsampling decoder using ConvLSTM blocks was added afterwards to produce a combined output of keypoint detection heatmaps and pen state prediction. Calculating the loss on the test set regarding classification, the model managed to reach a loss of 0.5983, which is better than the agnostic baseline, but worse than other methods.

Model	Time Steps	Loss	BA
Naive Predictor	N/A	0.693	N/A
Time-Agnostic	1	0.6135	75.35%
Combo-CLS-KP	30	0.5983	71.61%
Combo-CLS	10	0.5750	77.10%
GRU 64	3	0.5522	76.76%
ConvLSTM 64	30	**0.5454**	**79.02%**

Table 10.2. – The *UbiPen-Sequence* test set results for the approaches evaluated in this chapter. The naive predictor loss represents the random baseline. Time-agnostic designates an *EfficientNet B0* network trained on the sequence dataset frame-by-frame. Combo-CLS-KP is the recurrent network with combined sequence classification and keypoint detection backbones producing sequence heatmaps. Combo-CLS uses the same backbones but has a classification output rather than keypoint mappings. GRU is the best F-GRU variant and ConvLSTM the best convolutional recurrent model.

Interpretation

The added information by additional feature extractors did not manage to offset the impact of adding capacity to models that were trained on a relatively small dataset. The spirit of baseline development used throughout the technical part of this dissertation informs the decision to relegate investigation of further combinations of backbone networks to future work.

10.6. Summary

In this chapter, utilizing recurrent neural networks to analyze writing process video streams regarding pen states – i.e. does the pen touch the paper? – was investigated. For this task, a separate dataset annotation tool was developed (see chapter 6) and 14040 frames of 234 sequences from 39 subjects were annotated manually.

Five architectural templates were assessed by training 33 variants and comparing their binary cross entropy loss and balanced accuracy. In table 10.2, the best scoring variants are listed. As a reference, an architecture using a feature extraction backbone based on the *EfficientNet B0* architecture was trained on the annotated frames one-by-one, to be able to gauge the impact of employing recurrent network layers classifying whole sequences. The best loss and balanced accuracy was delivered by an approach using convolutional LSTM layers after the time-agnostic feature encoding stage. In contrast to flat recurrent layers, it retained spatial information by the convolutional neural network encoder. This, coupled with the sparse connectivity of convolutional layers, proved beneficial to the task of predicting the pen state, reaching a balanced accuracy of 79.02%. This measure takes into account the true positive rate and the true negative rate, so moderate class imbalances in the test dataset do not produce an optimistic score.

The scope of this study is limited by the relatively small dataset for training and testing – the samples are sequence-based, and thus are not independent. A more thorough investigation of the performance under data shift is advised for future work.

This chapter describes the artefactual contribution of a neural network based pen state classifier towards building a ubiquitous pen interface. With the ability to accurately find pen keypoints, adequately extract handwritten text, and to classify close to four out of five frames correctly regarding the pen state, most major building blocks for a prototype interface have been established. The next chapter deals with a concept to support execution of user commands as the last component explored in this dissertation.

11. Pen Gestures

Interfaces must allow user input. Between non-command interfaces (Nielsen, 1993) and traditional windows, icons, menus, pointer (WIMP) style concepts lies a wide array of possible interaction modes. They have in common that they are governed by a set of underlying constraints with respect to the user that informs their design (ISO, 2018). Human computer interaction research has long been concerned with pen interfaces (I. E. Sutherland, 1963), and has succeeded in designing usable tablet PCs using digital pens as well as real paper tools that record written words digitally.

The products that use analog paper take the place of a digital recorder of analog strokes. Interaction only happens afterwards when working with the digital traces. As discussed previously, they are limited to certain writing utensils and thus limit user expression. For their analog-digital pen interface *PapierCraft*, Liao and Guimbretièere (2012, p. 4) concluded that "a command system was necessary to address the variety of commands required by paper interactions". Much previous work on physical-digital pen interfaces relied on pen gestures to e.g. tag content in the printed document (Steimle, Mühlhäuser, & Hollan, 2012, p. 53). Besides gesture systems, Heinrichs (2015) identified two other main classes of interaction techniques for pen and paper interfaces: *Pidgets and Proxies*, which imbue e.g. paper buttons with functionality and *Cross-media links*, which connect physical and digital documents through marks (Heinrichs, 2015). The former are contrary to the *no modification* guideline the proposed pen interface follows. While the implementation of the latter is out of scope for this dissertation project, gestures investigated in this chapter facilitate such functionality. A camera-based interface enables using commands spatially related to writing processes, and can adapt commonly known gesture based interaction paradigms to handwritten text production.

As an example for an application, pen gesture commands could initiate automated analysis of the current document. In a situation where a laptop or other device is nearby, a command could execute a literature or Wikipedia search for a phrase just written. A similar concept could allow for searching a word marked by gesture in other documents, for spell checking, or simplifying equations.

In this chapter, an in-air pen gesture concept and its technological foundation based on the keypoint detection building block from chapter 8 is proposed.

11.1. Related Work

11.1.1. Collecting and Evaluating Pen Gestures

Pen gestures, as in drawing shapes with a pen-pointer on a screen, have been shown to be appreciated by users as being ”powerful, easy to learn, efficient, easy to use, convenient“ (Long, Landay, & Rowe, 1998, p. 6). Long et al. also reported criticism regarding recognition accuracy and memorizability. Pen interfaces had a phase of popularity at the beginning of the 2000s and inspired researchers to find efficient and easy ways to enter commands using a stylus (Tu, Ren, & Zhai, 2015). Long, Landay, Rowe, and Michiels (2000) analyzed visual similarity of pen gestures and modeled it to help interaction designers predict how users will perceive them, and to make an informed choice on gesture similarity.

Gesture Complexity

A complexity model for single stroke pen gestures was developed by X. Cao and Zhai (2007). They used the stroke elements consisting of curves, lines, and corners (CLC) to ”characterize the efficiency of a given gesture or a gesture set“ (X. Cao & Zhai, 2007, p. 1495) by predicting the time needed to perform the gesture within some error bounds.

Vatavu, Vogel, Casiez, and Grisoni (2011) addressed how users perceived the difficulty of pen gestures and developed a model by examining previous descriptors and their own empirical findings. Their model enables designers to rank several gestures by difficulty and assign a difficulty class for single gestures.

Pen vs. Finger

More recently, Tu et al. (2015) studied how pen and finger gestures differ when used on stationary or mobile devices employing the CLC model. Depending on the setting and the device used, finger gestures were either comparable in time and accuracy to pen input or performed worse, with the exception of thumb gestures while walking. Their studies showed that there are differences in gesture size ratio, and that generally, pens allow for more detail.

Visual Clutter

To combat visual clutter when using ink for commands, Tsandilas and Mackay (2010) developed an alternative kind of pen gestures that users integrate into the handwritten elements by adding circular dots. They argue that ”can be used as delimiters and command selectors“ (Tsandilas & Mackay, 2010, p. 4) at any point during strokes, in contrast to other delimiters such as *pigtails* (Hinckley, Baudisch, Ramos, & Guimbretière, 2005). The concept for the proposed pen interface does not use real marks to avoid this kind of clutter. This also satisfies the constraints of modifying notes set out in section 2.2.2.

Pen Tails

Pen tail gestures were explored as an approach to allow users to potentially eliminate mode switching and to prevent interruptions of working with the tip by Tian et al. (2013). They projected 3D pen tail movement into a 2D shape that was classified using template matching. The gestures used were designed according to constraints regarding natural pen movement gathered from users of pen interfaces.

Elicitation

Talkad Sukumar et al. (2018) repeated elicitation studies reaching back 30 years to gather knowledge about the impact of advances in interface design and technological possibilities on gestures elicited from users of handwriting systems. The findings conclude the gestures elicited for text editing or annotation still are based on the concept of ”annotating on paper“, and that there is considerable overlap with gestures elicited decades ago.

Hand Gestures

Hand gestures while using a digital pen with the other hand where elicited by Aslan, Schmidt, Woehrle, Vogel, and André (2018), implicating an entirely different approach to pen computing with gesture-based commands. Because of the additional processing necessary for capturing both hands in a camera stream as well as the impractical camera positioning for the pen interface case as discussed here, this is not considered as an alternative input method.

11.1.2. Systems using Pen Gestures

Several digital-analog systems incorporated pen gestures as their command system.

Digital Ink

Hinckley et al. (2005) used their pen interaction testbed *Scriboli* to analyze digital ink delimiters for ”scope selection, command activation and direct manipulation“ (p. 1) and found the *pigtails* gesture a superior user experience to e.g. button delimiters.

In their literature study, C. J. Sutherland, Luxton-Reilly, and Plimmer (2016) identified 12 systems using digital ink as commands for a pen interface. The mode switching approaches found were through *pen buttons*, *separate display space*, *special gestures*, and *pen and touch*, the latter describing the case where text and touch input are combined.

Physical Ink

PapierCraft selected their operations based on the fact that no real-time feedback was available to the user (Liao et al., 2005). The system allowed to hyperlink, copy/paste, and creating collages from physical documents. In an offline process, strokes collected with an *Anoto* pen on the physical document were later executed when synchronizing with a daemon server to edit the digital document. They explicitly switched interaction modes using a gesture foot pedal. Evaluating an improved version where feedback was available through a LED on the pen top, they showed that their gesture system required some training time, but that users ultimately performed comparably to tablet pc users.

The physical editing of *PowerPoint* presentation slides using an *Anoto* pen was studied by Signer and Norrie (2007). The *Paperpoint* system integrated paper buttons that allowed users to change or show slides. Annotations on printouts could be transferred to the digital slide during the presentation. Gesture support however was constricted to tapping marked areas on the printout.

Paperproof used common pen gestures such as strike-through or arrow-heads for editing *OpenOffice* documents physically by mapping coordinates to a document structure model through a server. Similar to *PapierCraft*, the strokes of an *Anoto* pen were collected and synchronized at once with a digital document. The ecosystem of *OpenOffice* plug-ins developed by the authors allowed for continuous editing and tracking of changes alternating between digital and physical documents.

Karatzas, d'Andecy, Rusinol, Chica, and Vazquez (2016) used hand or infrared pen-based gestures for their augmented paper system, but only considered the pen as a replacement pointer instead of fingers. They did not take into account the peculiarities of pen interaction for gesture design. No Ink

11.2. Pen-Gesture Concept

A key difference to the literature reviewed, with the exception of pen tail gestures, is that the gestures always incorporate drawing them with a stylus on a digital surface. As such, there is a limit to their transferability. However, the underlying principles of interacting with a pen did not change and the mental model of users still largely is that of ”annotating physical paper“ (Talkad Sukumar et al., 2018, p. 104). By limiting the interaction concept evaluation to easy and familiar gestures that are readily distinguishable, the proposed basic building block of a gesture command system is applicable to a wide range of scenarios by following a user-centered design approach motivated by previous work. This also takes into account that ”gesture based interaction also imposes problems regarding *learnability* and *recallability*“ (Heinrichs, 2015; Norman & Nielsen, 2010, p. 66).

Gesture Selection

Distinctive gestures of low complexity were selected from strokes that Talkad Sukumar et al. (2018) confirmed were as relevant as 30 years ago and those that were identified as easy and familiar by Vatavu et al. (2011). Borrowing from text-related applications, *Select Word/Phrase* with *Circle/Oval* and *Delete Word/Phrase* with *Line* were selected. For other, general commands, *Rectangle* and *Triangle* were included.

Mode Switching

Mode switching is performed explicitly, following systems by e.g. Hinckley et al. (2005) and Liao et al. (2005). There is no command button, as there is in *PapierCraft*. However, users could perform mid-air gestures with either a flipped pen, i.e. the pen tail pointing downwards, or with a voice command that includes as a clear indication that command input is happening.

Feedback Options

Since the user scenario for the proposed pen interface includes a digital device nearby, auditory feedback is the most natural choice. New digital assistants like *Alexa* or *Siri* are now commonplace and could take on the role of spelling marked words and delivering search results. Other common problems in interface design laid out by Norman and Nielsen (2010) are adressed as follows: Visibility only applies in limited ways as the interface commands leave no marks, like a speech interface. Consistency is reached by using few well-known gestures that can be combined and have been elicited time and time again. The latter point also ties in with discoverability – once users know how to activate command mode, these very basic shapes will be among the first they try, as literature shows. Non-destructive operations are at the heart of the interface concept, since the physical document is not altered by the commands. The shapes are scalable and should be recognized regardless of their size, as long as they are in the FOV of the camera. This last point represents an important limitation to the concept, but is inherent to the tradeoff between unmodified materials and user constraints to certain pens and surfaces. Reliability was shown for keypoint detection, and is investigated in this chapter regarding gesture recognition.

11.3. Gesture Recognition Study

For the gesture recognition building block to work properly, the pen paths extracted with the keypoint detection component need to be classified correctly as belonging to a certain gesture class. Reliable systems exist for the recognition of the paths themselves, so this part is delegated to the *$1 Recognizer* (Wobbrock, Wilson, & Li, 2007). The *$1 Recognizer* is a geometric template matcher that is fast and lightweight, making it suitable for real-time processing even in resource-constricted environments. Neural networks for action recognition offer another possibility to classify gestures. However, it is prudent to investigate more lightweight alternatives first, given the computational expense and the latency RNNs introduce – as well as the cost of creating training data. The pen keypoint paths are available to the system in any case, making their use for gesture recognition an efficient choice. The proposed system is concerned with producing keypoint paths suitable as recognizer input. To evaluate this, a study collecting recordings from participants' pen gestures was conducted.

11.3.1. Data Collection

To assess the quality of extracted keypoint paths, four users (2 male, 2 female) were recruited. Based on the positive experience with the *Wild-Keypoint* study, this study was, too, conducted remotely and users received illustrated instructions, which can be seen in figure 11.2. Parameters such as pens used and lighting conditions were purposely kept undefined, but some questions arose regarding lighting and it was recommended to those participants to perform the experiment in a well lit area. Participants were asked to make sure the pen keypoints were visible in all videos. Subjects were invited to perform 20 repetitions of each of the selected gestures for

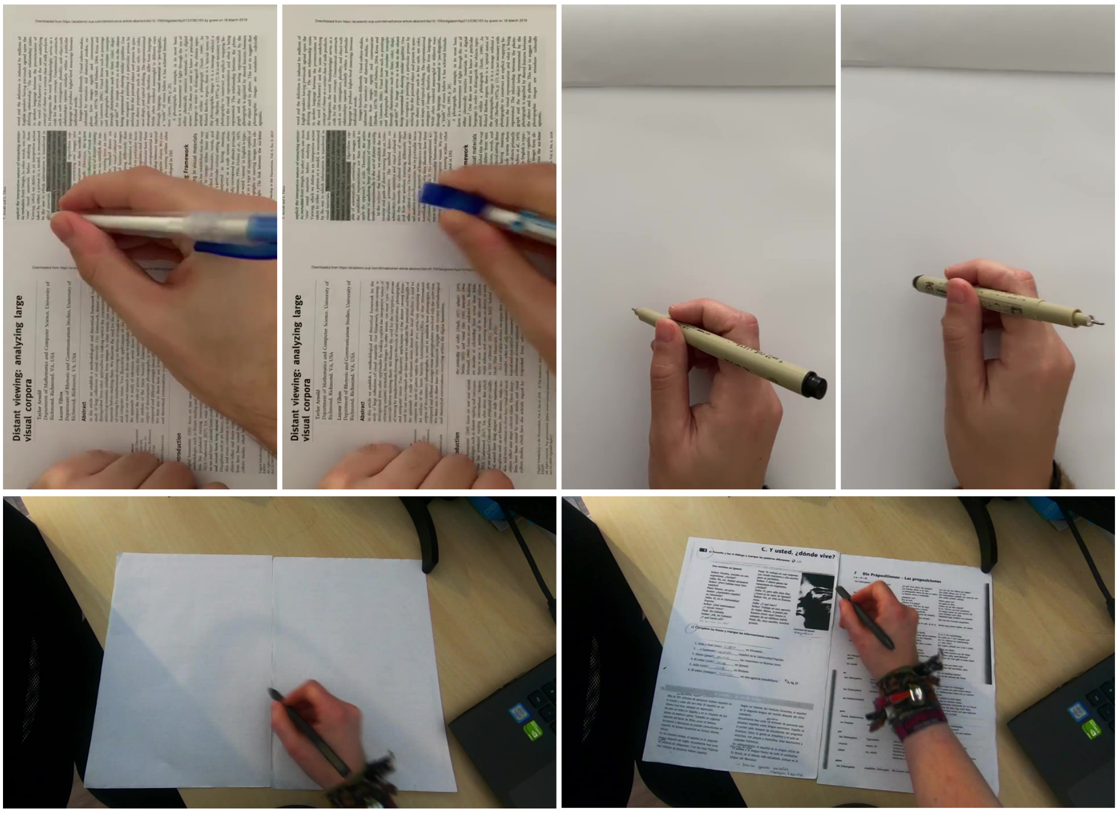

Distant viewing: analyzing large visual corpora
Distant viewing: analyzing large visual corpora
C. Y usted, ¿dónde vive?

Figure 11.1. – Sample frames of the data collection, highlighting the different setups, resolution, and aspect ratios of the evaluation data. Images with printed pages belong to the *complex background* subset. Images with blank pages belong to the *proof-of-concept* set.

each of *normal pen pose* and *flipped pen pose*. For 10, they were asked to use a blank sheet of paper as background. For the other 10, a high interference background of printed text was to be used. Through this, a challenging, but realistic dataset was collected. In figure 11.1, examples of recorded frames are shown. One subject subset contained large amounts of frames were the pen tip was occluded and was not used for evaluation.

The video sequences were annotated with keypoint and gesture type as well as background used. All in all, 240 usable video sequences were produced during this study. Two subjects provided data with a strongly divergent aspect ratio and of much lower resolution (480x848) than the network was trained with, while one subject delivered HD videos of gestures.

To produce pen keypoint paths, the videos were processed by the *B0-L-NF* ROI keypoint detection network, which performed best in the network assessment study. On the output, a threshold of 10^{-2} was applied to discard close-to-zero activations from the keypoint path. After rescaling, the resulting detected points were given to the *$1* recognizer, which compared them to predefined templates of the selected gestures.

11.3.2. Results

Blank Sheet

In the proof-of-concept case using a blank sheet of paper as background, the network was able to produce paths that the classifier recognized correctly in the majority of cases. When using the pen tip as gesture instrument, 95.0% of sequences with a blank background were recognized correctly. The flipped pen tail gestures were recognized in 71.67% of cases. Circles drawn with the flipped pen pose were the most difficult to detect, with only 40.0% being correctly classified, while all squares and triangles input with the pen tip were successfully placed. For the comparison, see figure 11.3 (a).

Printed Sheet

Participants provided their own sheets of printed paper backgrounds, which led to varied data sets. This part of the dataset proved particularly challenging for the detector/recognizer combination, as can be seen in figure 11.3 (b). 46.67% of gestures performed with the standard pen pose were recognized, while only 31.67% of gestures with a flipped pen could be categorized correctly. The analysis of provided video sequences indicates that the aspect ratio matters when the network is confronted with unusual input – the gestures of the subject who provided landscape oriented data were recognized correctly in 80% of sequences with a printed background. The other videos, where the results were worse, were filmed in portrait mode.

Discussion

The lack of printed documents in the *UbiPen-Keypoint* training set proved problematic for gesture recognition. In combination with untrained image aspect ratios, this led to a detection rate of less than half for tip gestures and less than one third for flipped pen gestures. As was demonstrated with the printed text augmentation for the handwritten text extraction network in chapter 9, this effect could possibly be mitigated without collecting additional data by using beneficial augmentation techniques. The author argues that annotating the videos already collected for this study and training the keypoint detector with them would lead to a generalized keypoint detector capable of providing well-defined keypoint paths even for the aforementioned cases.

For the basic proof-of-concept sequences, the gesture recognition performed well – averaging over all cases, 83.34% of gestures were categorized correctly using the ROI keypoint paths. Reusing network inferences that need to be performed regardless is beneficial to the overall resource requirements.

11.4. Summary and Future Work

This exploration of in-air pen gesture feasibility represents a realization of the pen-and-paper interface design guideline derived by Steimle et al. (2012, p. 88) to ”[d]esign a modular interface using simple and flexible building blocks“. It enables combining multiple, simple, and easy-to-learn gestures for a rich feature set, albeit constrained to expected aspect ratios and low background interference. Good results when using a flipped pen pose on a blank background indicate that the detection is not tied to the keypoint location in relation the the hand holding it. Although high quality recognition could be achieved for some cases, the study highlighted the limits of the generalization ability of the keypoint detector network. Dataset shift could be handled by the network with regards to lighting conditions, pen types, and resolution, as was shown by the *UbiPen-Wild* validation results. Document content that was entirely different in character to what was present in the dataset collection study for *UbiPen-Keypoint* lead to average recognition rates between 31 and 46 percent, leaving lots of room for improvement. Future work includes training the keypoint detector with the collected dataset and additional print-oriented augmentation. With this network, it makes sense to repeat the gesture recognition study, comparing effects of aspect ratios and noisy backgrounds.

In future expansions on the pen interface components developed here, more interactions, like incorporating multiple sheets and tangible tools, should be investigated based on the guideline to ”provide for rich interactions“ (Steimle et al., 2012, p. 88).

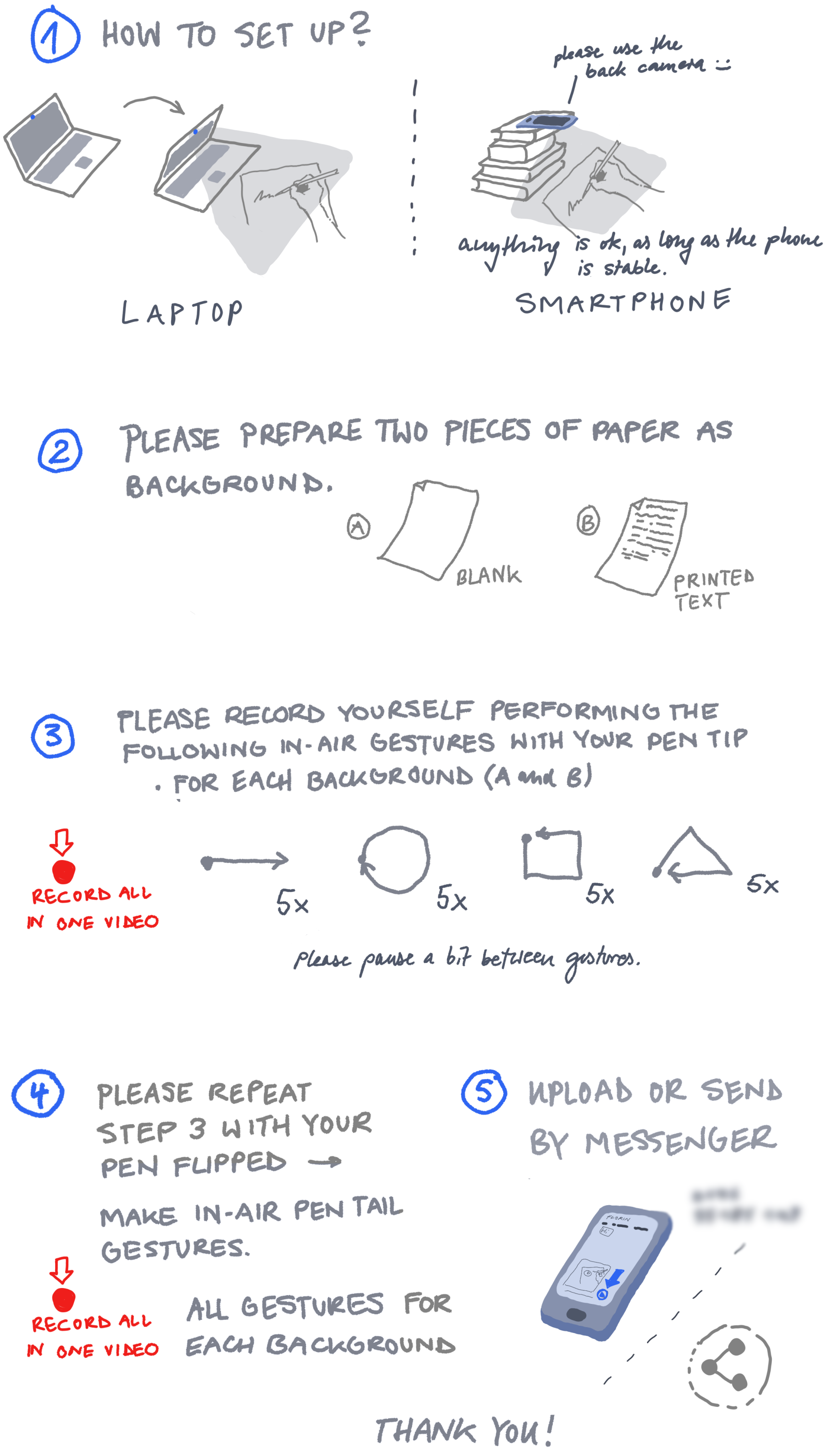
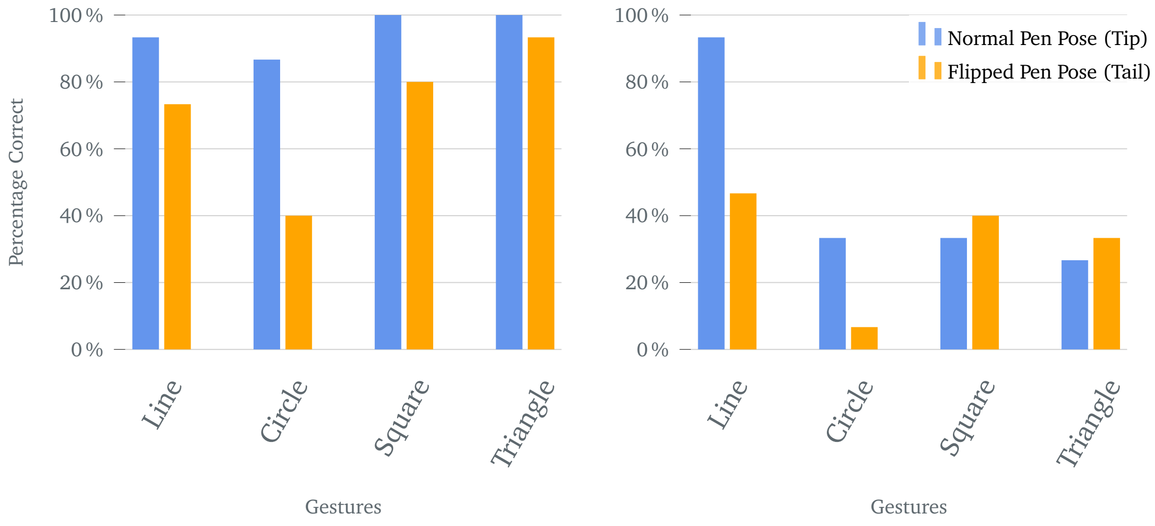

1 HOW TO SET UP?
please use the back camera :)
LAPTOP
anything is ok, as long as the phone is stable.
SMARTPHONE
2 PLEASE PREPARE TWO PIECES OF PAPER AS BACKGROUND.
A BLANK
B PRINTED TEXT
3 PLEASE RECORD YOURSELF PERFORMING THE FOLLOWING IN-AIR GESTURES WITH YOUR PEN TIP
· FOR EACH BACKGROUND (A and B)
RECORD ALL IN ONE VIDEO
5x
5x
5x
5x
Please pause a bit between gestures.
4 PLEASE REPEAT STEP 3 WITH YOUR PEN FLIPPED →
MAKE IN-AIR PEN TAIL GESTURES.
RECORD ALL IN ONE VIDEO
ALL GESTURES FOR EACH BACKGROUND
5 UPLOAD OR SEND BY MESSENGER
THANK YOU!

Figure 11.2. – Instructions for the pen gesture data collection study.



(a) Proof-Of-Concept **(b)** Complex Background

Figure 11.3. – Results of the gesture recognition study using the keypoint paths extracted with *B0-L-NF-ROI* and a *$1-Recognizer*. (a) shows the performance of the proof-of-concept design with a blank paper background. (b) shows the percentage of correctly recognized gestures when using a complex, printed background.

12. Prototype

The previous chapters focused on individual building blocks that could make up the missing link between physical ink strokes and online handwriting recognition. Evaluation studies highlighted capabilities and limitations of each one and established a baseline in terms of architectures both on the neural network and the system level. In this chapter, a prototypical framework for connecting the building blocks is presented.

Computational performance was always considered while exploring possible pen interface aspects in this dissertation. Still, end-user devices like laptops or mobile phones generally do not possess the computing power to handle all neural network and other image processing by themselves. Based on the requirements of real-time stream handling of camera data, a system distributing the workload across devices was developed.

This system is not intended to be a production-ready implementation that performs all potential tasks a ubiquitous pen interface would. It is a proof-of-concept that the building blocks developed can be connected in a useful manner and engineered to work in real-time. To design such a system, first, the constraints on video input and operating latency are established. Then, a prototype architecture is developed by identifying necessary processing steps and determining how they are realized with the devices available to the author. A technical analysis of frame processing times shows abilities and limitations of the implementation as presented. Following that, the system processing time and latency are discussed in the context of the developed parallelization approach.

12.1. Design Constraints

Input Stream

Ideally, input video streams run at 60 frames per second or more. Like this, hand movements do not introduce excess blur, which could cause uncertainty when tracking objects in the frame. The resolution of video streams, as discussed in section 8.3.2 in the context of neural network input, should allow detection at the scale of handwritten letter sizes in a typical setup. For a 60 FPS video at HD resolution, this theoretically adds up to 355.96 MB/s of uncompressed data that needs to be transferred, preprocessed, and resized, followed by neural network inferences and postprocessing steps.

Input Lag

The conceptual pen interface does not require *displaying* output images at a real-time framerate. Nonetheless, it should process commands and extract pen movements at a rate that makes interactivity possible. MacKenzie and Ware (1993) studied the effect of lag on motor-sensory task performance in interactive systems and showed that it degrades considerably above 225 ms. In their experiment, MacKenzie and Ware determined that below a system latency of 75 ms, the user error rate was under 5%. These values can act as a guideline for further development.

Frame Time

As the main focus of the investigated pen interface concept, pen and writing surface allow immediate interaction unaffected by any digital processing. They are not susceptible to motor-sensory task performance degradation through interface lag. However, the building blocks of the writing process support system ideally need to operate within 1/60th of a second, i.e. ∼16.67 ms to keep lag from adding up for pen path extraction. In the following section, the measures taken to achieve this are laid out.

Sequence Analysis

There are also considerations beyond single frame computation times: The gesture recognition for command entry (see chapter 11) and the pen state prediction (see chapter 10) both rely on sequences of frames. The former component was designed in a way that solely relies on the continuously extracted keypoint paths, and does not incur additional lag besides running the template matching of some coordinates.

This matching means performing some additions and multiplications on a small amount of points (here: 64) for each template (here: 4), which is negligible in terms of computation time (Wobbrock et al., 2007). The pen state sequence analysis, depending on the time sequence length chosen, takes between 3 and 30 frames until a long enough sequence is collected for processing. That means that the first frame of written text could be available digitally a half second afterwards in the case of a 60 FPS recording. When the user executes a command with their pen, the previously written text would have been processed already at the time the user finishes their gesture, given parallel execution of the sequence inference step. In the case of simply storing the digitized text, such a time period would not impact the operation. This delay thus does not represent a major obstacle for the realization of a ubiquitous pen interface.

12.2. Prototype Architecture

In this section, the necessary processing steps are determined and their distribution across the available devices is documented.

12.2.1. Processing Steps

The processing taking place to make strokes and recognized gestures available to a potential interface application is shown schematically in figure 12.1. This view illustrates which components are necessary on a macro level to deliver the extracted information: A camera streams video frames into a system, which performs text and keypoint detection on each image. Successive sequences of images are analyzed to perform pen state prediction. Both the reconstructed keypoint paths and the predicted pen states are then used to reconstruct ink strokes digitally. The gesture recognition component makes use of the reconstructed paths, too. After explicit

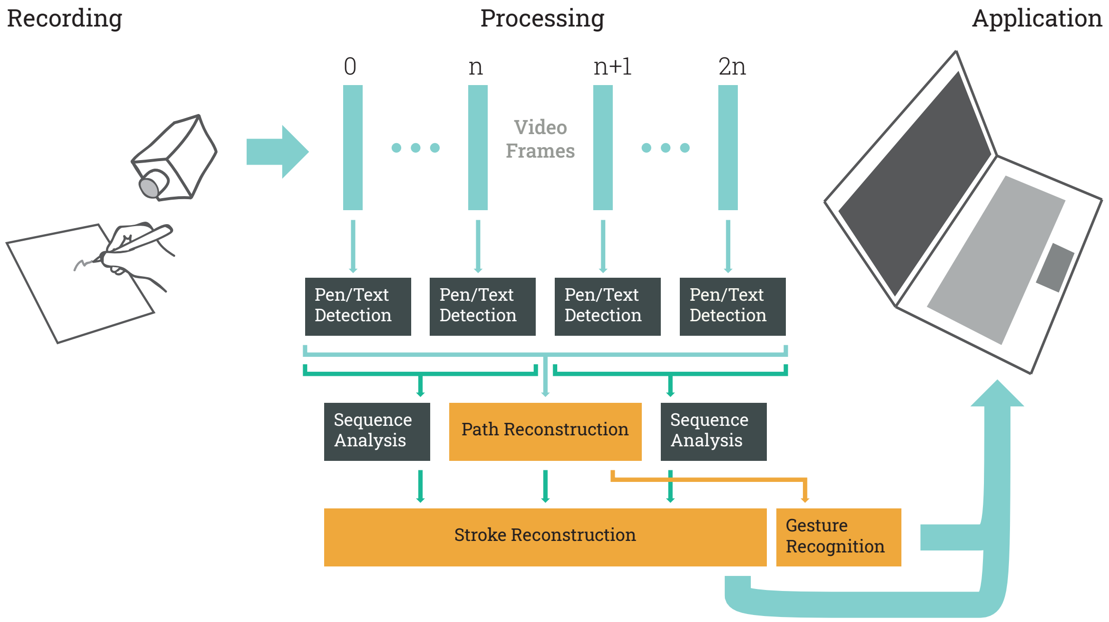


Figure 12.1. – A schematic view of the components necessary to process video streams in the prototype system to make strokes and recognized gestures available to a potential application using the interface. Dark grey components involve neural network inferences. Pre- and post-processing was omitted for clarity.

activation and deactivation, it performs geometric template matching on the detected keypoint path and provides the gesture description to the application.

12.2.2. Device Distribution and Thin Clients

The different components are distributed among available devices. The prototype system, schematically drawn in figure 12.2, is centered around a laptop for displaying pen paths. A pen path display was integrated to quickly inspect and analyze detected keypoints. The laptop represents the end-user device running a pen-computing application in the pen interface concept and is connected to the other components through an Ethernet connection.

Detection Split

The pen keypoint detection is split up between two systems. Since it would introduce significant lag into the system, the proposed architecture avoids sending full resolution images across network connections. The computer connected to the recording device – in a production-level system, this would be the camera of the end-user laptop or smartphone – detects only the region of interest in the video frame and sends a small patch to a backend system for further processing. A system set up in such a way can be integrated into a cloud computing setting with minimal effort.

Thin Clients

This modular organization also allows even more reduced scenarios where an end-user device simply receives the extracted information over a network connection, i.e. implements a *thin client*, while the camera stream comes from a fixed camera that e.g. is looking at a whiteboard for team meetings or a desk. Notes can be streamed to a smartphone, and commands could be given through the phone interface in addition to pen gestures.

System Config

The proof-of-concept system in this chapter is a realization of such a fixed camera scenario. A *Logitech BRIO* webcam was mounted looking towards the writing surface and was directly connected to the system performing the ROI detection, enabling efficient camera-to-system transfer of the high resolution images. It is the same camera that was used for the large data acquisition study. The ROI extraction system is designated as *System-A* in the following and runs Debian Linux with a Intel i7-8700 CPU, 16 GB of RAM and a GeForce GTX 1080 GPU.

The backend server that is only reachable over the local area network of the University of Regensburg runs Ubuntu 18.04, has four RTX 2080 GPUs and an Intel Xeon CPU. Two of the GPUs were used in the tested iteration of the proof-of-concept prototype. It is designated as *System-B*.

Communication

The laptop, System-A and System-B are connected via asynchronous publisher and subscriber patterns. When System-A has detected a region of interest in a camera image, it publishes the ROI patch. System-B, as a subscriber to System-A, receives the image patch and can then perform detail pen detection and text extraction on the incoming patch. Additionally, it can accumulate recent patches and execute sequence analysis in a parallel process, using another GPU. Parallelization of processes on System-B is implemented by adding clients that run in their own process and communicate by asynchronous request/reply patterns. To organize their workload, a broker also running on System-B sends requests to them, distributing patches to do e.g. detail keypoint inference and text extraction in parallel. The broker collects the replies and publishes the results to its subscribers, i.e. the laptop or other devices that want to know about the extracted elements. A parallelization setup like this carries the modular, building block-oriented architecture to the processing step level.

The user's laptop is subscribed to updates from both systems and connects the detected keypoints to a pen path, which can then be processed further. Basic outlier culling is performed here by cutting out keypoints which are placed at extreme distances.

The tested iteration of this system was capable of processing input at 23.8 FPS on consumer level hardware when pen tip and mark extraction are performed remotely and the patches and inferences are transmitted over a network connection. This iteration was built with earlier variants of the *ResNet-50-L-NF* detection networks,

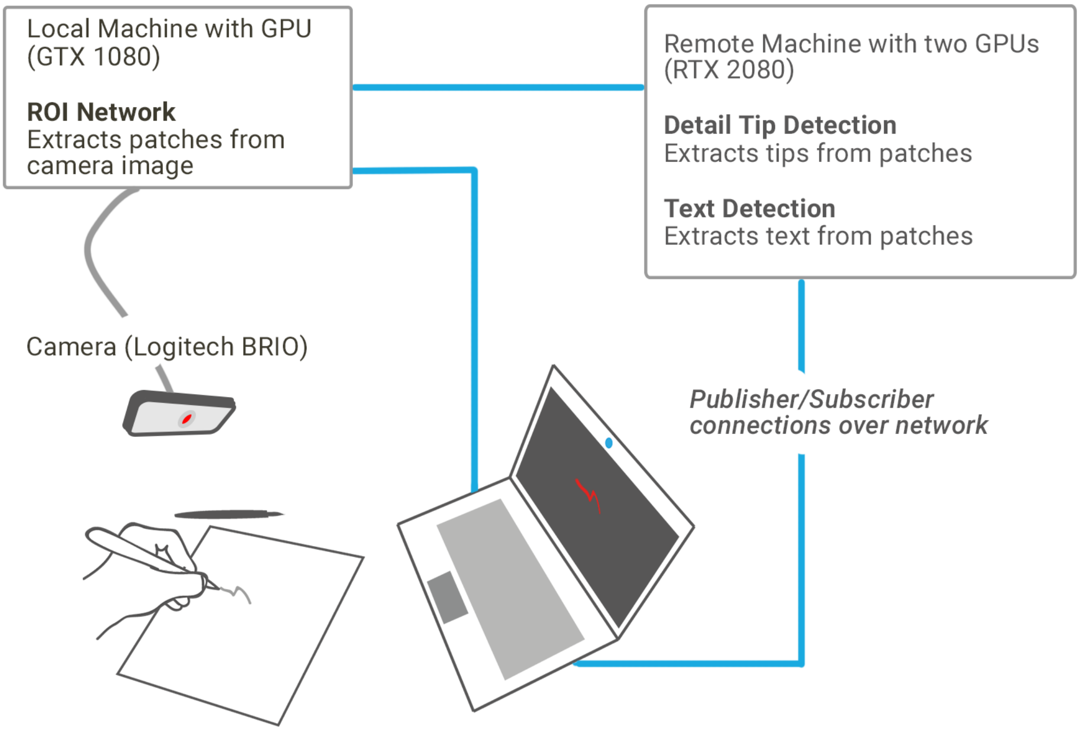


Figure 12.2. – The prototype system: A laptop receives updates from a local machine extracting patches from a camera stream and a remote server performing tip and text extraction. Blue lines indicate network connections.

which were quicker, but less accurate than the *B0-L-NF* models that were used later. Due to technical constraints, the B0-L-NF models could not be optimized to the same level of inference speed.

12.3. Single Frame Processing Times

To achieve a high throughput, each step of computation needed to be assessed in terms of time needed. Components were implemented in Python, with a focus on using efficient libraries implementing most of their core processing in compiled code, like *Pillow-SIMD* for image resizing (Karpinsky, 2021) and *Tensorflow* for data pipelines and neural network development. In that way, Python code acts as an easy-to-prototype glue connecting high-performance code together. The communication between components was implemented using asynchronous messaging principles with the *Zero Messaging Queue* (ZMQ) library. This allowed for clear synchronization points between processes both over network connections and on the same system, while maintaining autonomous execution and memory management, reducing development complexity.

12.3.1. Image Preprocessing

Image Acquisition

Measurements showed that under normal working conditions on the recording system, the Logitech BRIO camera is able to deliver close to 60 FPS. The theoretical frame time of 16.67 ms thus was used as a reference throughout the performance optimization process of the prototype.

Image Resizing

After acquiring the video frame from the imaging device, it needed to be prepared for neural network input and ROI detection. Chapter 8 showed that the architecture B0-L-NF with an *EfficientNet B0* backbone performed best in terms of generalization ability and accuracy. Its input resolution is 224x224, so all frames need to be scaled

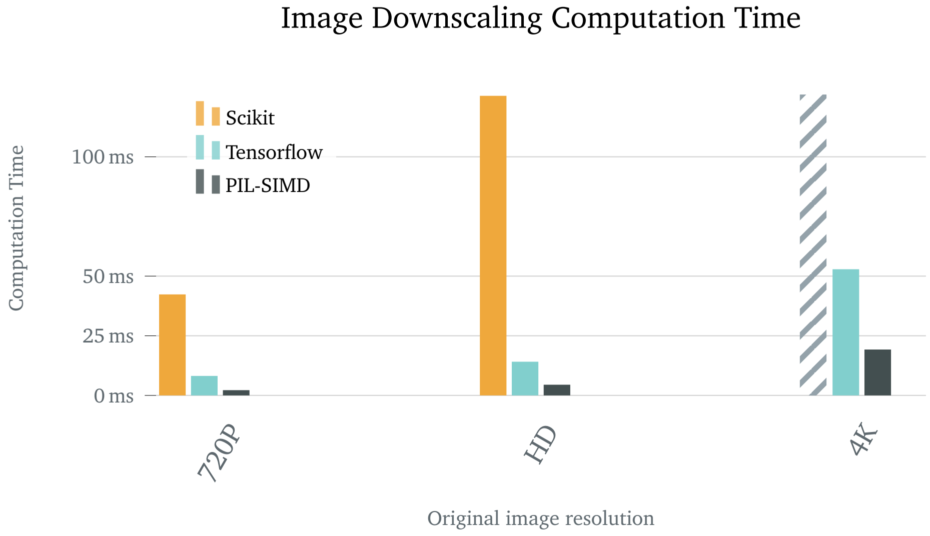


Figure 12.3. – The mean time in milliseconds the tested libraries took to downscale an input image of 720p, HD, or 4K resolution to the input resolution of *EfficientNet B0* (224x224). The benchmark measurements were produced by *pytest-benchmark* (Mărieş, 2020), averaging several runs. The grayed out bar signifies an impractical Scikit processing time above 1 s, which was left out for readability.

down accordingly. This operation can take considerable time in relation to the available 16.67 ms.

A comparison of available, popular Python libraries that provide image resizing functionality showed the clear advantage PIL-SIMD provides. In figure 12.3, the mean times in milliseconds the methods from Scikit, Tensorflow, and PIL-SIMD took to downscale an image to *EfficientNet B0* input size are compared. Resizing was executed with antialiasing enabled, to prevent artifacts from distorting the resulting image. The benchmarks were performed on System-A using *pytest-benchmark* (Mărieş, 2020). PIL-SIMD resizes a HD image to the target format in 4.48 ms on this system.

Model	Average Inference Time (ms)	Average Inference Time (TensorRT) (ms)
ROI	35.44	7.88
Detail	35.57	7.86
Text	35.39	7.82
Sequence	128.99 (4.30 per Frame)	N/A

Table 12.1. – Average frame-by-frame times, i.e. without batching, for the 4 neural network model components. The TensorRT times are the result of graph optimizations. The optimization could not be performed for the sequence model due to technical problems. The benchmark measurements were produced by *pytest-benchmark* (Mărieş, 2020).

12.3.2. Inference Times

The computational makeup of backpropagation and inference algorithms used for neural networks allows the execution on massively parallelized computing architectures. Drivers and frameworks available for end-user and general processing GPUs make real-time performance of several image processing tasks feasible (Sierra-Canto, Madera-Ramirez, & Uc-Cetina, 2010; Krpan & Jakobovic, 2012; S. Zhang, Gunupudi, & Zhang, 2015).

On System-A, the average frame-by-frame inference time for the ROI, detail, text, and sequence networks was measured using *pytest-benchmark* again. For the assessment of inference times, no batching was applied to mimic realistic runtime conditions. Experiments on RTX 2080 hardware supporting 16-bit arithmetic did not result in faster inference. The measurements can be seen in table 12.1, together with results which were produced by applying TensorRT (NVIDIA, 2020) optimization to the trained models to improve speed. The B0-L-NF based ROI model achieved an improved inference time of 7.88 ms. The other models performed correspondingly. Theoretically, sequential processing of all three models on a single GPU leads to a feature extraction time of 23.56 ms, 6.86 ms above the target limit of 16.7 ms. For

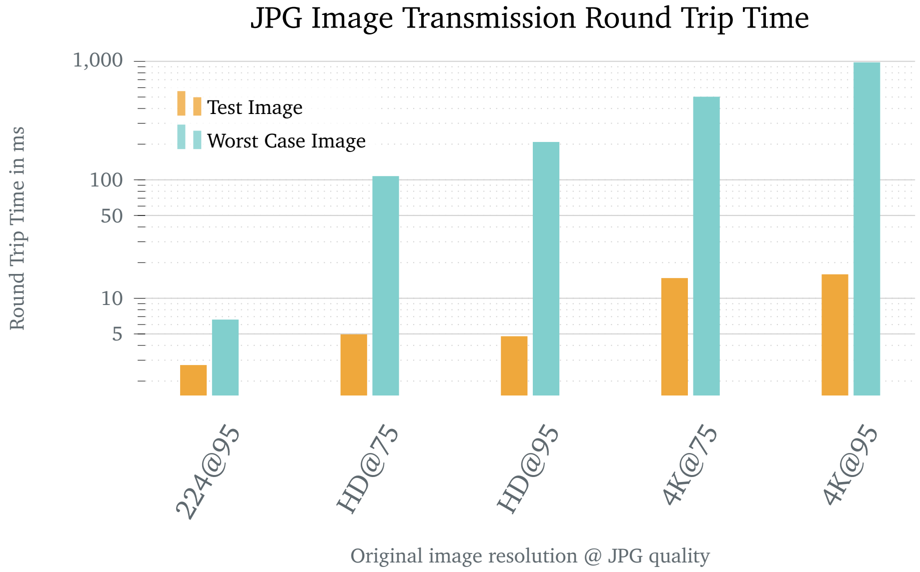


Figure 12.4. – Average image transmission round trip time for various resolutions. For the higher resolutions, two JPG compression levels were tested. The *worst case image* measurements designate transmission tests where uniformly random RGB noise was used as image content, making it hard to compress and providing an upper limit to estimate the impact of variations in image content.

this reason, parallelizing the inference process was a priority for the development of this prototype.

The sequence analysis of a 30 frame snippet was performed in 128.99 ms by the ConvLSTM-30-64 network, which performed best among the architectures evaluated in chapter 10. This means per single frame, the model took 4.30 ms.

12.3.3. Image Transmission

By distributing ROI detection and detail keypoint extraction, the amount of data that needs to be transmitted across devices and, in the case of this prototype, across network connections, is reduced. In figure 12.4, the transmission times of a JPG-compressed images at HD resolution at various quality settings is compared to those of an image patch at B0-L-NF input resolution, i.e. 224x224. Images at 4K resolution were included for reference. The transmission times include waiting for a confirmation from the recipient and were measured inside the university of Regensburg's network between two devices connected through Gigabit Ethernet adapters. Any network measurement taken in real-life conditions needs to be interpreted with caution, as conditions and network load can change drastically. This analysis illustrates the point that transmission times are, even during a pandemic in a local area network of a university, considerable and impact the frame processing times. The isolated experiment showed that image patches at model resolution (224x224) can potentially be transferred at 1.25 ms on average and at 6.61 ms at worst. Regular HD images can be transferred at 4.78 ms, and *worst case images* can take up to 208.7 ms on average.

The measurements for the worst case image were made with a programmatically generated image with uniformly distributed RGB noise. This kind of image is hard to compress, since it potentially contains every color at every luminosity and has no larger areas of the same color. The worst case image served as an upper limit for the variance one can expect when working with realistic image data. Notably, figure 12.4 shows that while the test image transmission for the smallest image and the HD image were not far apart, the larger format is much more susceptible to content-dependent fluctuations. If the compression can't work well with some image content, the impact on network transmission time is worse when that image is large. This supports the argument for keeping network transfers small.

12.3.4. System Latency and Summarized Processing Time

Figure 12.5 illustrates the makeup of processing times and transmission latency for the prototype. The distributed system operates in three distinct main loops. On the local machine (as seen in 12.2) connected to the camera via USB 3.0 port, frame acquisition and ROI network inference takes place. The processing time here is the sum of time between frames, ROI network inference time, and the time it takes to locate and cut a patch from the camera image. Transmission time after

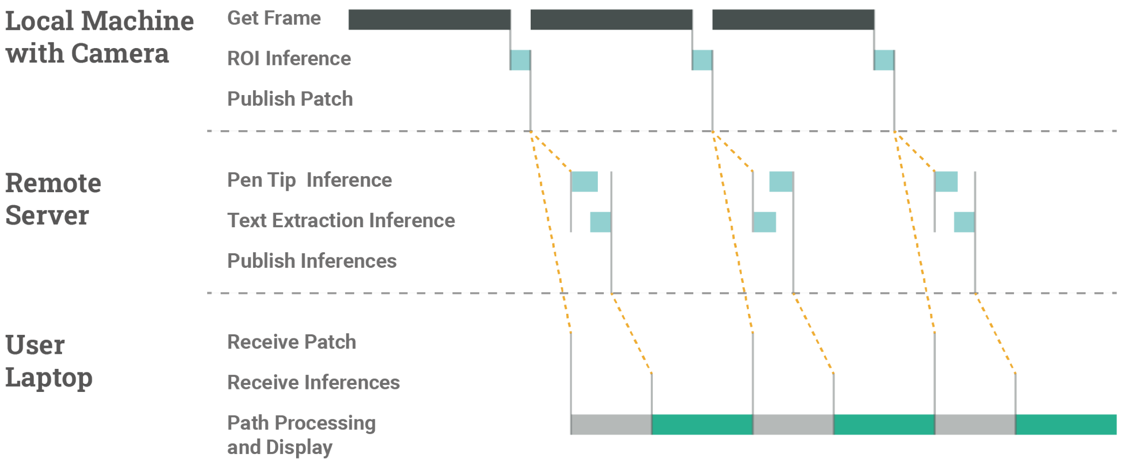


Figure 12.5. – Schematic latency and processing time overview of the three main system loops. Some processing steps were omitted for clarity.

publishing the patch to the remote server as well as the user laptop adds to the system latency. On the remote server, both the pen tip network and the text extraction network run in parallel, distributed among two RTX 2080 GPUs. The broker program synchronizes receiving the patch, distributing it to the two machine learning models, and collecting and publishing their results. The user laptop waits for matching pairs of patch and inference transmissions, before it processes the input to extract a new tip keypoint location and the handwritten text. Depending on the complexity of path processing done clientside, the path reconstruction can take up most of the processing time needed for a single frame. Transmission times vary wildly between network configurations. The prototype system processing time and latency is most dependent on the network performance and the clientside processing. The tested iteration of this system reached 23.8 FPS, sending all data between devices over network connections.

Note that in figure 12.5, sequence analysis is not included. The tested iteration of this prototype did not incorporate the pen state prediction building block, which at the time was not available. The pen state network can be included as an additional

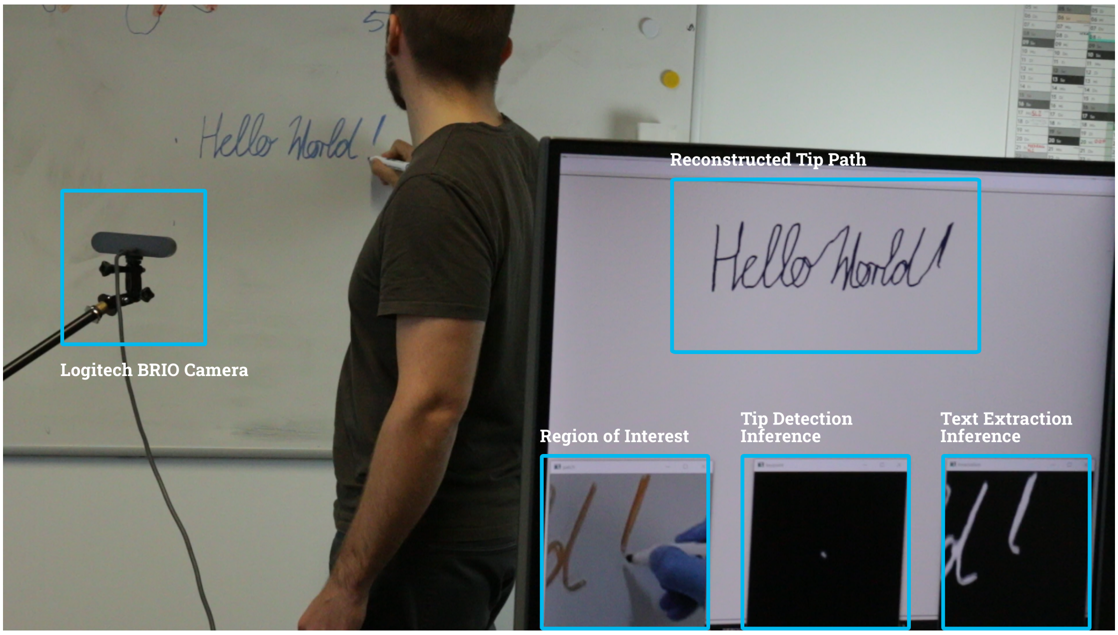


Figure 12.6. – The prototype system running and digitizing a whiteboard writing process.

inference server connected via ZMQ sockets and added to the broker for distribution of patches. System-B would collect the patches and use a third GPU to perform inference, once the sequence count is complete. The pen state information, i.e. probabilities that the pen is touching the writing surface, can then be sent via network.

12.3.5. Example Scenarios

In the following paragraphs, example scenarios for the prototype system are shown. A video of the running system can be reviewed in the file referenced in appendix B.6.

Whiteboard In figure 12.6, a typical scenario for a whiteboard application of the pen interface is shown, with additional displays of the intermediate products of the system components. The additional displays contain the region of interest as identified by the ROI network, the inference of the detail tip detection network as well as the inference of the text extraction network. In the center of the screen,

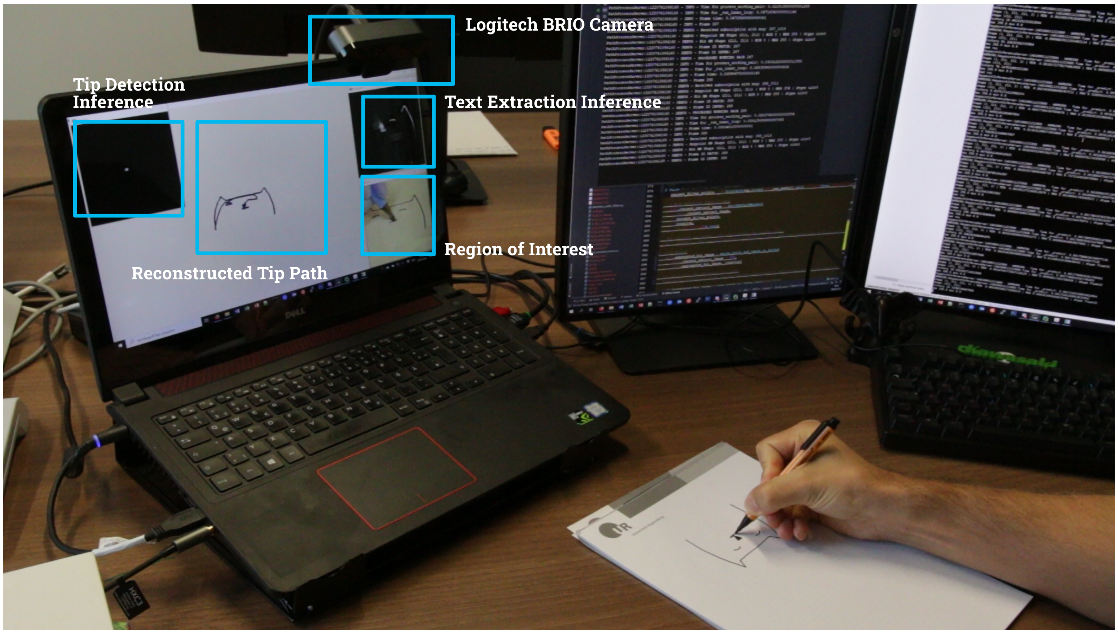


Figure 12.7. – The prototype system running and digitizing a paper pad writing process.

the reconstructed tip path is shown. The path, with outliers removed, does not contain information about pen up/pen down events. This can be best seen at the points between the two words ”Hello“ and ”World“, and between the latter and the exclamation mark.

Paper Pad and Laptop A second scenario, writing on a paper pad with a computer and a webcam nearby, is shown in figure 12.7. It demonstrates the versatility of the system by tracking the pen and ink upside down. In both example scenarios, the same backend models were operating, and no change in training was made, nor were the networks primed for a particular setting.

12.4. Summary

This proof-of-concept system represents an artifact in the design science research sense that reveals capabilities and limitations as well as new directions for future research.

A running prototype that distributed computation across devices connected through Ethernet adapters for keypoint detection, text extraction, and stroke reconstruction delivered close to 24 FPS and demonstrated the feasibility of a parallelization approach that lends itself to integration into cloud computing applications. The single steps were optimized for low lag and high throughput to provide future researchers with more knowledge to navigate the pitfalls of constructing a pen interface prototype. In two example scenarios with different writing surfaces and utensils, pen paths were reconstructed in real-time.

13. Conclusion

This dissertation presents a novel approach for combining the advantages of physical and digital systems in the context of scientific note taking. Built upon a foundation of literature work, a theoretical model of note taking was developed, looking at the process from the perspective of several disciplines. This approach helped identify aspects that lead away from a trivialization as mere storage activity, reflecting on the highly idiosyncratic properties as signs of an extension of mind into the tools employed.

To empirically support the insights gathered in this course of action, interviews with scholars at the Master's and PhD level regarding their research illuminated the place note taking has in contemporary scholarly workflows. The findings confirmed the motivation of supporting scientific note taking in a careful way by using pen-and-paper interaction together with digital systems as relevant and helped consolidate requirements for a ubiquitous pen interface.

From these requirements, a course of action was charted for investigating building blocks that implement a missing link between analog ink strokes and digitized ink, which can readily be input to text recognition. The choice fell on camera-based tracking of both pen and writing surface, as cameras are available in every laptop and smartphone and lend themselves to mobile and flexible interfacing while taking notes. In this way, the interface is not dependent on a digital stylus or integrated sensors.

Today's deep learning methods promise robust feature extraction. Since they need extensive training to realize their potential, randomized data collection studies with 64 participants were performed and the data subsequently annotated. The produced datasets range from those collected in controlled lab environments to remotely collected sets rich in variety pertaining to devices, resolution, and environmental influences. They allow gathering a complete picture of the performance of potential machine learning models.

For the missing link building blocks, an architecture search that produced accurate and reliable keypoint detection for pen tips and tails was performed. Segmenting out the handwritten text for applications that e.g. only care for archiving handwritten notes, the text extraction model developed in chapter 9 reached a balanced accuracy of close to 80% for a challenging set that meant it had to distinguish between handwritten text and typography. Pen state classification was explored by building recurrent neural networks working with snippets of camera streams to integrate knowledge about previous and later positions, resulting in a system that could correctly classify 4 out of 5 images in a test dataset. A study that showed pen gestures as a viable mode of interaction based on extracted pen keypoint paths represented the last building block. While currently limited to clear paper backgrounds, results from the text extraction approach promise improvement on print background with a straight-forward augmentation scheme.

In a prototype implementation, the components were interconnected with a parallelization scheme that allowed distributing computational effort between devices and included platform-agnostic network interfaces for the component parts. This proof-of-concept was able to run at real-time speeds, with part of the traffic sent over network connections.

While the technological investigation and data collection resulted in a useful baseline for further exploration, the balanced accuracy of pen state prediction and the outliers in pen keypoint tracking do not yet represent a system that is ”good enough“ for daily tasks. In an early assessment of user acceptance of recognition accuracy in pen interfaces, Frankish, Hull, and Morgan (1995) found that, depending on the payoff, users were willing to accept 5–20% error rates. Standard *Windows Vista* handwriting recognition was just as fast as using a software keyboard and had a 1.0% mean error rate in hour-long text entry tasks (Kristensson & Denby, 2009). As an important future step, it is necessary to explore how the performance of the created system influences the error rate of established handwriting recognition systems.

14. Future Work

This dissertation offers a foundational body of work motivated by an understanding of note taking that surpasses mere storage of knowledge. The established requirements, data collected, and technological investigation can act as stepping stones towards fully integrated ubiquitous pen interfaces.

Besides concrete measures for improvement like implementing additional data augmentation as recommended in chapter 11 and adding more datasets for training, another viable path of inquiry is the integration of visual feedback through AR systems. In figure 14.1, a possible application scenario is outlined using a portable AR device, like the *Microsoft HoloLens* (Kipman, 2016). Figure 14.2 shows how possible AR applications could look for indexing and working with tags for selected content, and for linking external documents and earlier uses of a selected phrase.

This scenario profits from the modular character of the building blocks proposed in this dissertation. Running a variant of ROI detection on device and distributing other computation into cloud services using the developed parallelization scheme is worth exploring. The benefits are mobility of the user while working, instant visual feedback, and the hardware-based integrated tracking of the surroundings of the *HoloLens*.

However, the integrated camera would likely be insufficient for the high-speed high-resolution tracking and would need to be accompanied by a separate camera. The unique user-focused concerns of AR applications need to be taken into account when developing for this platform.

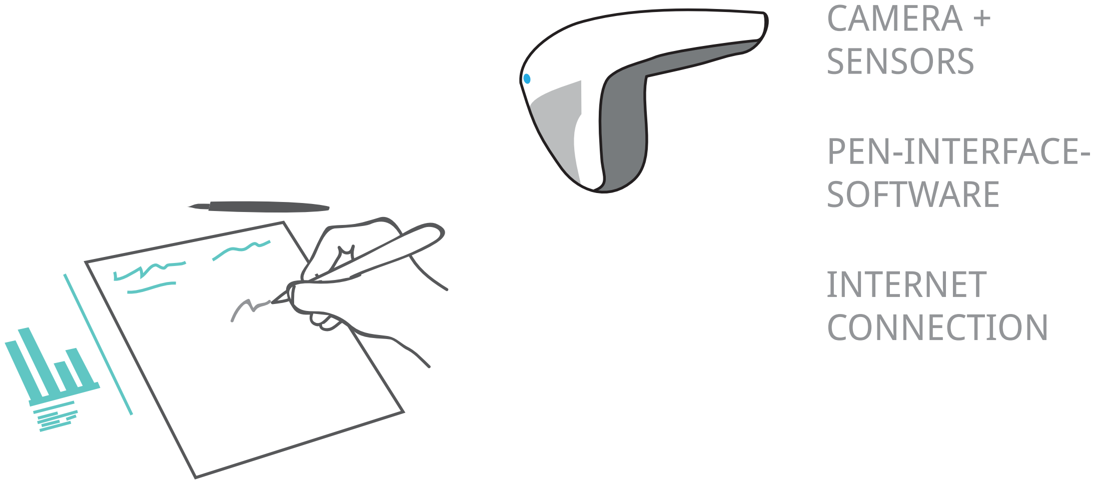


Figure 14.1. – Possible application scenario of a pen interface integrated into a portable AR device like the *Microsoft HoloLens*. Augmented content is colored blue.

A possible research agenda for such a project based on the results of this dissertation is listed in the following:

(1) **Establish feasibility by assessing integration into the *HoloLens* platform** *Unity* is a straight forward authoring tool for *HoloLens* applications. Encapsulate inference with pre-trained weights from *B0-L-NF* and evaluate performance of *Unity*-integrated operation.

(2) **Determine minimum viable application** Create mock-up *Unity* applications motivated by requirements and improvement proposals from chapter 3 and perform expert walkthroughs to gather knowledge for iteration.

(3) **Collect domain-specific data** After choosing an application concept, collect additional training data to adapt CNN performance.

(4) **Create application prototype** Using the domain-adapted building blocks, connect them using the parallelization scheme presented and implemented for chapter 12 – the interface between components is platform-agnostic, and can

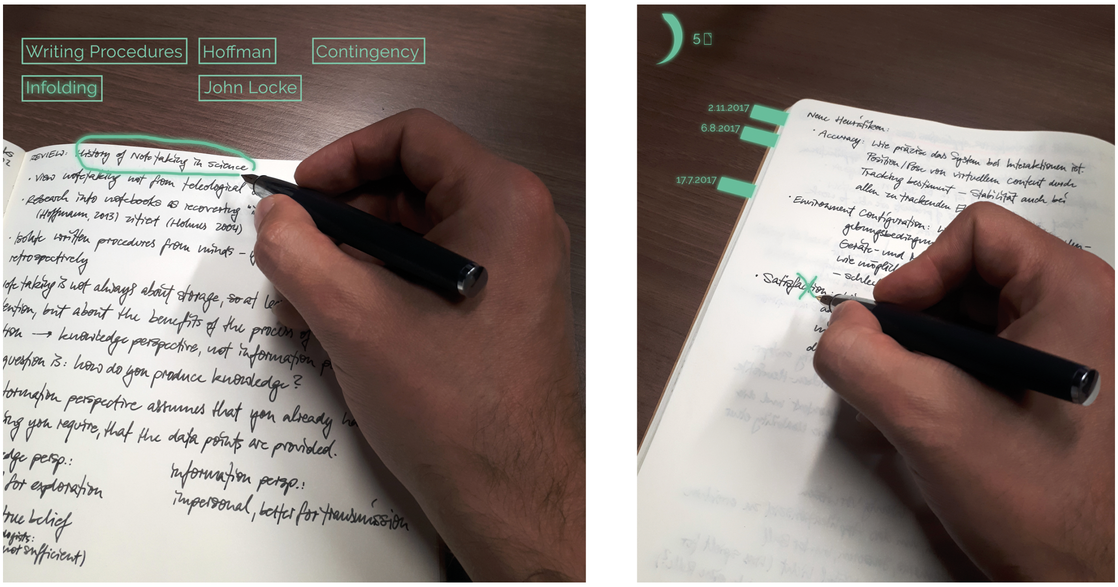


Figure 14.2. – Mock ups for possible AR applications. Left: tagging and indexing selected content. Right: displaying earlier occurrences of a marked phrase and links to external documents.

easily connect to cloud-based services that often run *Linux*, as opposed to the *Windows*-based platform of the *HoloLens*.

(5) **Evaluate prototype with users** Depending on the maturity of the prototype, evaluate it with either traditional usability testing methods like task performance assessment or expert walkthroughs. Let users speak about their experiences and juxtapose the results with results from the scholarly workflow study in chapter 3.

14.1. Ethics of Neural Network Training

As a closing note, this section briefly goes into what implications the recent increase in power of machine learning systems has in a wider sense. Many companies, especially those that collect our data on a massive scale, use it to train neural networks that predict our behaviour to sell products through advertising (Cyphers, 2020). Other systems assess possibility of recidivism of prisoners (Ozkan, 2017; Shih, Chiu, & Chou, 2019). Vigilance regarding what happens in these systems is necessary to avoid integrating biases, because therein lies the potential for automated suppression of demographic groups.

Arguably one of the most popular and widespread datasets in machine learning – *ImageNet* – has been around since 2009 and has been used for the training of numerous state-of-the-art architectures. For those architectures, weights pretrained on *ImageNet* are available in neural network frameworks like *Keras* or *Pytorch* and carry a distillation of the dataset to machine learning students and practitioners around the world.

A subset of *ImageNet* with 1000 object classes has been used for the performance challenge ILSVRC, which was held from 2010–2017 (Russakovsky et al., 2015). Motivated by this competition, researchers published major advances in convolutional neural networks and image classification. While the challenge classes refer only to dog breeds or airplanes and categories that are similarly non-contentious, the complete dataset is much larger and possessed a *person* category with over 2500 subcategories, which were generated from *WordNet*-synsets (Miller, 1998). These subcategories often contain moral judgments and images were labeled with concepts for which no straightforward way exists to attribute them. Examples include ”Slav“, ”kleptomaniac“, ”coward“, ”good person“, ”nonsmoker“, ”Togolese“, ”weakling“, and so on. It is hard to say how many deployed systems using neural networks have been trained based on *ImageNet* weights and carry features based on this highly problematic labeling in their core. After nine years of making this questionable part of the dataset available, the original authors started to take action by removing offensive categories and have since (a) published a measure for imageability of a concept and (b) analyzed balance and ethics in datasets (Yang et al., 2020). In a treatment of the inherent political nature of datasets, Crawford and Paglen (2019) maintain that this is not enough:

> Datasets aren't simply raw materials to feed algorithms, but are political interventions. As such, much of the discussion around ”bias“ in AI systems misses the mark: there is no ”neutral,“ ”natural,“ or ”apolitical“ vantage point that training data can be built upon. There is no easy technical ”fix“ by shifting demographics, deleting offensive terms, or seeking equal representation by skin tone. The whole endeavor of collecting images, categorizing them, and labeling them is itself a form of politics, filled with questions about who gets to decide what images mean and what kinds of social and political work those representations perform.

While a training set for a ubiquitous pen interface does not require labeling people, inherent bias regarding e.g. skin tone or handedness is troublesome for a production-ready system. The scope of prototype systems in this dissertation is limited by available resources, and not all eventualities are considered during creation and annotation. Still, this section serves as a reminder that care needs to be taken when transforming artifacts from this design science process into production-ready systems. Weights and training sets need to be adapted for real-world use, even when tracking and classification performance would suffice with given validation sets. And even then, just as Crawford and Paglen argue, it will still be an imperfect system that carries tacit assumptions about the nature of individuals in it and will at times be unjust or inept, and has the potential for misuse. In 1984 by George Orwell, handwritten notes were the last frontier in the combat against big brother, as surveillance systems were not able to read the sometimes crooked handwriting.

Bibliography

Achmann, M. (2021). Qualitative Analyse und Modellierung des wissenschaftlichen Arbeitens. Retrieved from https://epub.uni-regensburg.de/45682/

Ackoff, R. L. (1989). From data to wisdom. *Journal of Applied Systems Analysis, 16*(1), 3–9.

Agrawal, P., Girshick, R., & Malik, J. (2014). Analyzing the performance of multilayer neural networks for object recognition. In *European Conference on Computer Vision: Zurich, CH, 2014* (pp. 329–344). ECCV '14. Cham, DE: Springer. https://doi.org/10.1007/978-3-319-10584-0_22

Aiken, E. G., Thomas, G. S., & Shennum, W. A. (1975). Memory for a lecture: Effects of notes, lecture rate, and informational density. *Journal of Educational Psychology, 67*(3), 439–444. https://doi.org/10.1037/h0076613

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., . . . Asari, V. K. (2018). The history began from AlexNet: A comprehensive survey on deep learning approaches. arXiv preprint: 1803.01164

Altenberger, F. & Lenz, C. (2018). A non-technical survey on deep convolutional neural network architectures. arXiv preprint: 1803.02129

Amazon Web Services. (2020). Last accessed on May 20th, 2020. Retrieved from https://aws.amazon.com

Arnab, A. & Torr, P. H. (2017). Pixelwise instance segmentation with a dynamically instantiated network. In *Proceedings of the IEEE Conference on Computer Vision* *and Pattern Recognition: Honolulu, HI, 2017* (pp. 441–450). CVPR '17. Piscataway, NJ: IEEE. https://doi.org/10.1109/CVPR.2017.100

Aslan, I., Schmidt, T., Woehrle, J., Vogel, L., & André, E. (2018). Pen + mid-air gestures: Eliciting contextual gestures. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction: Boulder, CO, 2018* (pp. 135–144). ICMI '18. New York, NY: ACM. https://doi.org/10.1145/3242969.3242979

Audi, R. (2010). *Epistemology: A contemporary introduction to the theory of knowledge*. Abingdon-on-Thames, England, UK: Routledge.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(12), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, *39*(3), 930–945.

Bates, M. J. (2006). Fundamental forms of information. *Journal of the American Society for Information Science and Technology*, *57*(8), 1033–1045. https://doi.org/10.1002/asi.20369

Bates, M. J. (2015). The information professions: Knowledge, memory, heritage. *Information Research: An International Electronic Journal*, *20*(1). Retrieved from http://www.informationr.net/ir/20-1/paper655.html

Bates, M. J. (2017). Information. In *Encyclopedia of Library and Information Sciences, Fourth Edition* (pp. 2048–2063). Boca Raton, FL: CRC Press.

Bateson, G. (1972). *Steps to an ecology of mind: Collected essays in anthropology, psychiatry, evolution, and epistemology*. Chicago, IL: University of Chicago Press.

Belagiannis, V. & Zisserman, A. (2017). Recurrent human pose estimation. In *Proceedings of the 12th IEEE International Conference on Automatic Face & Gesture*

Recognition: Washington, DC, 2017 (pp. 468–475). FG ’17. Piscataway, NJ: IEEE. `https://doi.org/10.1109/FG.2017.64`

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, *2*(1), 1–127. `https://doi.org/10.1561/2200000006`

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166. `https://doi.org/10.1109/72.279181`

Berthelot, D., Schumm, T., & Metz, L. (2017). BEGAN: Boundary equilibrium generative adversarial networks. arXiv preprint: `1703.10717`

Bissacco, A., Cummins, M., Netzer, Y., & Neven, H. (2013). PhotoOCR: Reading text in uncontrolled conditions. In *2013 IEEE International Conference on Computer Vision: Sydney, AUS* (pp. 785–792). ICCV ’13. Los Alamitos, CA: IEEE Computer Society. `https://doi.org/10.1109/iccv.2013.102`

Block, H. (1970). A review of ”perceptrons: An introduction to computational geometry“. *Information and Control*, *17*(5), 501–522.

Blum, A. & Rivest, R. L. (1988). Training a 3-node neural network is NP-complete. In *Proceedings of the First Annual Workshop on Computational Learning Theory: Cambridge, MA, 1988* (pp. 9–18). COLT ’88. San Francisco, CA: Morgan Kaufmann Publishers Inc.

Boyd, K., Eng, K. H., & Page, C. D. (2013). Area under the precision-recall curve: Point estimates and confidence intervals. In H. Blockeel, K. Kersting, S. Nijssen, & F. Železný (Eds.), *Machine Learning and Knowledge Discovery in Databases* (pp. 451–466). Berlin, DE: Springer.

Brandl, P., Richter, C., & Haller, M. (2010). NiCEBook: Supporting natural note taking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems:*

Atlanta, GA, 2010 (pp. 599–608). CHI '10. New York, NY: ACM. `https://doi.org/10.1145/1753326.1753417`

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, *78*(1), 1–3. `https://doi.org/10.1175/1520-0493(1950)078<0001:vofeit>2.0.co;2`

Brodersen, K. H., Ong, C. S., Stephan, K. E., & Buhmann, J. M. (2010). The balanced accuracy and its posterior distribution. In *Proceedings of the 20th International Conference on Pattern Recognition: Istanbul, TR, 2010* (pp. 3121–3124). ICPR '10. Washington, DC: IEEE Computer Society. `https://doi.org/10.1109/icpr.2010.764`

Brooks, F. P., Jr. (1996). The computer scientist as toolsmith II. *Communications of the ACM*, *39*(3), 61–68. `https://doi.org/10.1145/227234.227243`

Burke, L. I. & Rangwala, S. (1991). Tool condition monitoring in metal cutting: A neural network approach. *Journal of Intelligent Manufacturing*, *2*(5), 269–280. `https://doi.org/10.1007/BF01471175`

Cao, X. & Zhai, S. (2007). Modeling human performance of pen stroke gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: San Jose, CA, 2007* (pp. 1495–1504). CHI '07. New York, NY: ACM. `https://doi.org/10.1145/1240624.1240850`

Cao, Z., Simon, T., Wei, S.-E., & Sheikh, Y. (2017). Realtime multi-person 2D pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: Honolulu, HI, 2017* (pp. 7291–7299). CVPR '17. Piscataway, NJ: IEEE. `https://doi.org/10.1109/cvpr.2017.143`

Charmaz, K. (2006). *Constructing grounded theory: A practical guide through qualitative analysis*. Introducing Qualitative Methods series. Thousand Oaks, CA: SAGE Publications.

Chaudhari, P. & Soatto, S. (2018). Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *Proceedings of the 2018 Information Theory and Applications Workshop: San Diego, CA, 2018* (pp. 1–10). ITA '18. Piscataway, NJ: IEEE.
`https://doi.org/10.1109/ita.2018.8503224`

Chen, Y. [Yu], Shen, C., Chen, H., Wei, X., Liu, L., & Yang, J. (2019). Adversarial learning of structure-aware fully convolutional networks for landmark localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(7), 1654–1669.
`https://doi.org/10.1109/tpami.2019.2901875`

Chen, Y. [Yunpeng], Li, J., Xiao, H., Jin, X., Yan, S., & Feng, J. (2017). Dual path networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems: Long Beach, CA, 2017* (pp. 4470–4478). NIPS'17. Red Hook, NY: Curran Associates Inc.
`https://doi.org/10.5555/3294996.3295200`

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing: Doha, QA* (pp. 1724–1734). Stroudsburg, PA: Association for Computational Linguistics.
`https://doi.org/10.3115/v1/D14-1179`

Chollet, F. (2020). Transfer learning & fine-tuning. Last accessed on November 25th, 2020. Retrieved from `https://keras.io/guides/transfer%5C_learning/`

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint: `1412.3555`

Ciaparrone, G., Sánchez, F. L., Tabik, S., Troiano, L., Tagliaferri, R., & Herrera, F. (2020). Deep learning in video multi-object tracking: A survey. *Neurocomputing*,

381, 61–88.
https://doi.org/10.1016/j.neucom.2019.11.023

Clark, A. & Chalmers, D. (1998). The extended mind. *Analysis*, *58*(1), 7–19.

COCO Challenge Keypoint Leaderboard. (2020). Last accessed on April 21st, 2020. Retrieved from http://cocodataset.org/#keypoints-leaderboard

Corbin, J. & Strauss, A. (1990). Grounded theory research: Procedures, canons and evaluative criteria. *Zeitschrift für Soziologie*, *19*(6), 418–427.
https://doi.org/10.1515/zfsoz-1990-0602

Corbin, J. & Strauss, A. (2008). *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Thousand Oaks, CA: Sage Publications, Inc.

Crawford, K. & Paglen, T. (2019). Excavating AI: The politics of training sets for machine learning. Last accessed on June 2nd, 2020. Retrieved from https://excavating.ai

Crevier, D. (1993). *AI: The tumultuous history of the search for artificial intelligence*. New York, NY: Basic Books.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019). AutoAugment: Learning augmentation strategies from data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: Long Beach, CA, 2019* (pp. 113–123). CVPR '19. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvpr.2019.00020

Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). RandAugment: Practical automated data augmentation with a reduced search space. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW): Online* (pp. 702–703). CVPRW '20. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvprw50498.2020.00359

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, *2*(4), 303–314.
https://doi.org/10.1007/bf02551274

Cyphers, B. (2020). Google says it doesn't 'sell' your data. here's how the company shares, monetizes, and exploits it. Last accessed on February 12th, 2021. Retrieved from https://www.eff.org/de/deeplinks/2020/03/google-says-it-doesnt-sell-your-data-heres-how-company-shares-monetizes-and

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops): Miami, FL* (pp. 248–255). CVPR '09. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/cvprw.2009.5206848

DIN. (2005). *Lines for handwriting - part 1: General lines (DIN 16552-1:2005-05)*.

Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., & Ferrer, C. C. (2020). The deepfake detection challenge dataset. arXiv preprint: 2006.07397

Donahue, J., Hendricks, L., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., & Darrell, T. (2017). Long-term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, *39*(04), 677–691. https://doi.org/10.1109/TPAMI.2016.2599174

Drucker, P. F. (1959). *The landmarks of tomorrow*. New York, NY: Harper.

Drucker, P. F. (1999). Knowledge-worker productivity: The biggest challenge. *California Management Review*, *41*(2), 79–94. https://doi.org/10.2307/41165987

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). CenterNet: Keypoint triplets for object detection. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV): Seoul, KOR* (pp. 6568–6577). ICCV '19. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/ICCV.2019.00667

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.
`https://doi.org/10.1007/s11263-009-0275-4`

Fang, H., Xie, S., Tai, Y., & Lu, C. (2017). RMPE: Regional multi-person pose estimation. In *2017 IEEE International Conference on Computer Vision (ICCV): Venice, IT* (pp. 2353–2362). ICCV '17. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/iccv.2017.256`

Fawcett, T. (2004). ROC graphs: Notes and practical considerations for researchers. Retrieved from `https://www.hpl.hp.com/techreports/2003/HPL-2003-4.pdf`

Feichtenhofer, C., Fan, H., Malik, J., & He, K. (2019). Slowfast networks for video recognition. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV): Seoul, KOR* (pp. 6202–6211). ICCV '19. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/iccv.2019.00630`

Fischer, A., Keller, A., Frinken, V., & Bunke, H. (2012). Lexicon-free handwritten word spotting using character HMMs. *Pattern Recognition Letters*, *33*(7), 934–942.
`https://doi.org/10.1016/j.patrec.2011.09.009`

Flor, N. V. & Hutchins, E. L. (1991). Analyzing distributed cognition in software teams: a case study of team programming during perfective software maintenance. In *Empirical Studies of Programmers: Fourth Workshop* (pp. 36–64).

Frankish, C., Hull, R., & Morgan, P. (1995). Recognition accuracy and user acceptance of pen interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Denver, CO, 1995* (pp. 503–510). CHI '95. New York, NY: ACM Press.
`https://doi.org/10.1145/223904.223972`

Frické, M. (2009). The knowledge pyramid: A critique of the DIKW hierarchy. *Journal of Information Science*, *35*(2), 131–142. `https://doi.org/10.1177/0165551508094050`

Friedman, M. C. (2014). Notes on note-taking: Review of research and insights for students and instructors. *Harvard Initiative for Learning and Teaching*, 1–34.

Frinken, V., Fischer, A., Manmatha, R., & Bunke, H. (2011). A novel word spotting method based on recurrent neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(2), 211–224. `https://doi.org/10.1109/TPAMI.2011.113`

Fukushima, K. (1975). Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics*, *20*(3-4), 121–136. `https://doi.org/10.1007/bf00342633`

Fukushima, K. & Miyake, S. (1982). Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In S. Amari & M. Arbib (Eds.), *Competition and Cooperation in Neural Nets* (pp. 267–285). Berlin, Heidelberg, DE: Springer. `https://doi.org/10.1007/978-3-642-46466-9_18`

Galison, P. (2000). The suppressed drawing: Paul dirac's hidden geometry. *Representations*, (72), 145–166. `https://doi.org/10.2307/2902912`

Gallager, R. G. (2001). Claude E. Shannon: A retrospective on his life, work, and impact. *IEEE Transactions on Information Theory*, *47*(7), 2681–2695. `https://doi.org/10.1109/18.959253`

Gao, P., Yuan, R., Wang, F., Xiao, L., Fujita, H., & Zhang, Y. (2020). Siamese attentional keypoint network for high performance visual tracking. *Knowledge-Based Systems*, *193*, 105448. `https://doi.org/10.1016/j.knosys.2019.105448`

Garcia, J., Tsandilas, T., Agon, C., & Mackay, W. E. (2014). PaperComposer: Creating interactive paper interfaces for music composition. In *26e Conférence Francophone sur l'Interaction Homme-Machine: Lille, FR* (pp. 1–8). IHM '14. New York, NY: ACM.
`https://doi.org/10.1145/2670444.2670450`

Gibson, J. (1977). The theory of affordances. In R. Shaw & J. Bransford (Eds.), *Perceiving, Acting, and Knowing: Toward and Ecological Psychology* (pp. 62–82). Hillsdale, NJ: Erlbaum.

Giere, R. N. (2006). The role of agency in distributed cognitive systems. *Philosophy of Science*, *73*(5), 710–719.
`https://doi.org/10.1086/518772`

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Columbus, OH* (pp. 580–587). CVPR '14. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/cvpr.2014.81`

Gkioxari, G., Toshev, A., & Jaitly, N. (2016). Chained predictions using convolutional neural networks. In *Proceedings of the IEEE European Conference on Computer Vision: Amsterdam, NL* (pp. 728–743). ECCV '16. Cham, DE: Springer.
`https://doi.org/10.1007/978-3-319-46493-0_44`

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. `http://www.deeplearningbook.org`. Cambridge, MA: MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems* (Vol. 27, pp. 2672–2680). Red Hook, NY: Curran Associates, Inc. Retrieved from `https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf`

Goodfellow, I., Vinyals, O., & Saxe, A. M. (2015). Qualitatively characterizing neural network optimization problems. In *International Conference on Learning Representations: San Diego, CA*. ICLR '15. arXiv preprint: `1412.6544`

Goonatilake, S. (1991). *The evolution of information : Lineages in gene, culture, and artefact*. London, UK: Pinter Publishers.

Graves, A. (2013). Generating sequences with recurrent neural networks. arXiv preprint: `1308.0850`

Guimbretière, F. (2003). Paper augmented digital documents. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology: Vancouver, CAN* (pp. 51–60). UIST '03. New York, NY: ACM. `https://doi.org/10.1145/964696.964702`

Harrell, F. (2017a). Classification vs. prediction. Last accessed on November 11th, 2020. Retrieved from `https://www.fharrell.com/post/classification/`

Harrell, F. (2017b). Damage caused by classification accuracy and other discontinuous improper accuracy scoring rules. Last accessed on November 11th, 2020. Retrieved from `https://www.fharrell.com/post/class-damage/`

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. New York, NY: Springer Science & Business Media.

Hayes, G. R., Pierce, J. S., & Abowd, G. D. (2003). Practices for capturing short important thoughts. In *Extended Abstracts on Human Factors in Computing Systems: Ft. Lauderdale, FL* (pp. 904–905). CHI EA '03. New York, NY: ACM. `https://doi.org/10.1145/765891.766062`

Haykin, S. (2009). *Neural networks: A comprehensive foundation (3rd edition)*. Hoboken, NJ: Prentice-Hall, Inc.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV): Venice, IT* (pp. 2961–2969).

ICCV '17. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/iccv.2017.322

He, K., Zhang, X. [X.], Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Las Vegas, NV* (pp. 770–778). CVPR '16. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvpr.2016.90

He, K., Zhang, X. [Xiangyu], Ren, S., & Sun, J. (2016). Identity mappings in deep residual networks. In *Proceedings of the IEEE European Conference on Computer Vision: Amsterdam, NL* (pp. 630–645). ECCV '16. Cham, DE: Springer.
https://doi.org/10.1007/978-3-319-46493-0_38

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. New York, NY, London, UK: J. Wiley; Chapman & Hall.

Heiner, J. M., Hudson, S. E., & Tanaka, K. (1999). Linking and messaging from real paper in the paper PDA. In *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology: Asheville, NC* (pp. 179–186). UIST '99. New York, NY: ACM.
https://doi.org/10.1145/320719.322600

Heinrichs, F. H. F. (2015). *Mobile pen and paper interaction* (Doctoral dissertation, Technische Universität Darmstadt).

Herczeg, M. (2009). Medieninformatik in Forschung, Lehre und Praxis. In *Workshop-Proceedings der Tagung Mensch & Computer 2009: Berlin, DE*. Berlin, DE: Logos Verlag.

Hevner, A. & Chatterjee, S. (2010). Design science research in information systems. In *Design research in information systems: Theory and Practice* (pp. 9–22). New York, NY: Springer.
https://doi.org/10.1007/978-1-4419-5653-8_2

Hinckley, K., Baudisch, P., Ramos, G., & Guimbretière, F. (2005). Design and analysis of delimiters for selection-action pen gesture phrases in Scriboli. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Portland, OR* (pp. 451–460). CHI '05. New York, NY: ACM. `https://doi.org/10.1145/1054972.1055035`

Hinton, G. E. (1992). How neural networks learn from experience. *Scientific American, 267*(3), 144–151. `https://doi.org/10.1038/scientificamerican0992-144`

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation, 18*(7), 1527–1554. `https://doi.org/10.1162/neco.2006.18.7.1527`

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint: `1207.0580`

Hjørland, B. (2011). Theoretical clarity is not 'manicheanism': A reply to Marcia Bates. *Journal of Information Science, 37*(5), 546–550. `https://doi.org/10.1177/0165551511423169`

Ho, D., Liang, E., Chen, X., Stoica, I., & Abbeel, P. (2019). Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning: Long Beach, CA, 2019* (pp. 2731–2741). ICML '19. arXiv preprint: `1905.05393`

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München.

Hochreiter, S. & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780. `https://doi.org/10.1162/neco.1997.9.8.1735`

Hoffer, E., Hubara, I., & Soudry, D. (2017). Train longer, generalize better: Closing the generalization gap in large batch training of neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems: Long Beach, CA* (pp. 1729–1739). NIPS '17. Red Hook, NY: Curran Associates Inc.

Hoffmann, C. (2013). Processes on paper: Writing procedures as non-material research devices. *Science in Context*, *26*(2), 279–303. https://doi.org/10.1017/s0269889713000069

Hollan, J., Hutchins, E., & Kirsh, D. (2000). Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction*, *7*(2), 174–196. https://doi.org/10.1145/353485.353487

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*(8), 2554–2558. https://doi.org/10.1073/pnas.79.8.2554

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint: 1704.04861

Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2020). Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, *42*(08), 2011–2023. https://doi.org/10.1109/TPAMI.2019.2913372

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Honolulu, HI* (pp. 2261–2269). CVPR '17. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/cvpr.2017.243

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., … Guadarrama, S., et al. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Honolulu, HI* (pp. 7310–7311). CVPR '17. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/cvpr.2017.351

Hunter, D. (1978). *Papermaking: The history and technique of an ancient craft*. Dover Books Explaining Science. Mineola, NY: Dover Publications.

Hutchins, E. (1995a). *Cognition in the wild*. Cambridge, MA: MIT Press.

Hutchins, E. (1995b). How a cockpit remembers its speeds. *Cognitive Science*, *19*(3), 265–288. https://doi.org/10.1207/s15516709cog1903_1

Ichikawa, J. J. & Steup, M. (2018). The analysis of knowledge. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Summer 2018). Stanford, CA: Metaphysics Research Lab, Stanford University. Retrieved from https://plato.stanford.edu/archives/sum2018/entries/knowledge-analysis/

Image Classification on ImageNet. (2020). Last accessed on May 20th, 2020. Retrieved from https://paperswithcode.com/sota/image-classification-on-imagenet

Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach & D. Blei (Eds.), *Proceedings of the 32nd International Conference on Machine Learning: Lille, FR, 2015* (Vol. 37, pp. 448–456). ICML '15. Cambridge, MA: PMLR. Retrieved from http://proceedings.mlr.press/v37/ioffe15.html

ISO. (2018). *Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts (Standard No. 9241-11:2018)*. International Organization for Standardization. Geneva, CH.

Ivakhnenko, A. G. (1968). The group method of data handling-a rival of the method of stochastic approximation. *Soviet Automatic Control*, *1*(3), 43–55.

Ivakhnenko, A. G. & Lapa, V. G. (1965). *Cybernetic predicting devices*. New York, NY: CCM Information Corporation.

Ivakhnenko, A. G., Lapa, V. G., & McDonough, R. N. (1967). *Cybernetics and forecasting techniques*. New York, NY: American Elsevier Publishing Company.

Jaderberg, M., Simonyan, K., Vedaldi, A., & Zisserman, A. (2016). Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision*, *116*(1), 1–20. https://doi.org/10.1007/s11263-015-0823-z

Jakubovitz, D., Giryes, R., & Rodrigues, M. R. D. (2019). Generalization error in deep learning. In H. Boche, G. Caire, R. Calderbank, G. Kutyniok, R. Mathar, & P. Petersen (Eds.), *Compressed Sensing and Its Applications* (pp. 153–193). Cham, DE: Springer. https://doi.org/10.1007/978-3-319-73074-5_5

Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., & Qu, R. (2019). A survey of deep learning-based object detection. *IEEE Access*, *7*, 128837–128868. https://doi.org/10.1109/access.2019.2939201

Johnson, W., Jellinek, H., Klotz, L., Rao, R., & Card, S. K. (1993). Bridging the paper and electronic worlds: The paper user interface. In *Proceedings of the ACM Conference on Human Factors in Computing Systems: Amsterdam, NL, 1993* (pp. 507–512). CHI ’93. New York, NY: ACM. https://doi.org/10.1145/169059.169445

Jones, L. K. (1997). The computational intractability of training sigmoidal neural networks. *IEEE Transactions on Information Theory*, *43*, 167–173. https://doi.org/10.1109/18.567673

Jordan, M. I. (1986). Serial order: A parallel distributed processing approach. Technical report. San Diego, CA: California University.

Kallenberg, M., Petersen, K., Nielsen, M., Ng, A. Y., Diao, P., Igel, C., . . . Karssemeijer, N., et al. (2016). Unsupervised deep learning applied to breast density segmentation and mammographic risk scoring. *IEEE Transactions on Medical Imaging*, *35*(5), 1322–1331.
`https://doi.org/10.1109/tmi.2016.2532122`

Karatzas, D., d'Andecy, V. P., Rusinol, M., Chica, A., & Vazquez, P.-P. (2016). Human-document interaction systems–a new frontier for document image analysis. In *Proceedings of the 12th IAPR Workshop on Document Analysis Systems (DAS): Santorini, GR* (pp. 369–374). IAPR '16. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/das.2016.65`

Karpinsky, A. (2021). Pillow performance. Last accessed on January 25th, 2021. Retrieved from `https://python-pillow.org/pillow-perf/`

Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: Long Beach, CA, 2019* (pp. 4401–4410). CVPR '19. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/cvpr.2019.00453`

Keysers, D., Deselaers, T., Rowley, H., Wang, L.-L., & Carbune, V. (2017). Multi-language online handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. TPAMI '17, *39*, 1180–1194.
`https://doi.org/10.1109/tpami.2016.2572693`

Khan, A., Sohail, A., Zahoora, U., & Saeed Qureshi, A. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*.
`https://doi.org/10.1007/s10462-020-09825-6`

Kiewra, K. A. (1989). A review of note-taking: The encoding-storage paradigm and beyond. *Educational Psychology Review*, *1*(2), 147–172. https://doi.org/10.1007/bf01326640

Kiewra, K. A., DuBois, N. F., Christian, D., McShane, A., Meyerhoffer, M., & Roskelley, D. (1991). Note-taking functions and techniques. *Journal of Educational Psychology*, *83*(2), 240–245. https://doi.org/10.1037/0022-0663.83.2.240

Kim, C., Chiu, P., & Oda, H. (2017). Capturing handwritten ink strokes with a fast video camera. In *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition: Kyoto, JP* (pp. 1269–1274). ICDAR '17. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/icdar.2017.209

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint: 1412.6980

Kingma, D. P. & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint: 1312.6114

Kipman, A. (2016). Announcing microsoft hololens development edition. Last accessed on March 3rd, 2021. Microsoft, Redmond, United States. Retrieved from https://blogs.windows.com/devices/2016/02/29/announcing-microsoft-hololens-development-edition-open-for-pre-order-shipping-march-30/

Kirillov, A., Girshick, R., He, K., & Dollár, P. (2019). Panoptic feature pyramid networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: Long Beach, CA, 2019* (pp. 6399–6408). CVPR '19. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/cvpr.2019.00656

Kittler, F. A. (1991). Synergie von Mensch und Maschine. In F. Rötzer & S. Rogenhofer (Eds.), *Kunst machen ? Gespräche und Essays*. München: Klaus Boer.

Kittler, F. A. (1993). Geschichte der Kommunikationsmedien. In J. Huber & A. M. Müller (Eds.), *Raum und Verfahren* (pp. 169–188). Basel, CH: Stroemfeld/Roter Stern.

Klamka, K. & Dachselt, R. (2017). IllumiPaper: Illuminated interactive paper. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems: Denver, CO* (pp. 5605–5618). CHI '17. New York, NY: ACM. `https://doi.org/10.1145/3025453.3025525`

Kolassa, S. (2019). Example when using accuracy as an outcome measure will lead to a wrong conclusion. Cross Validated. Version: 2019-06-29. eprint: `https://stats.stackexchange.com/q/368979`

Krauthausen, K. & Nasim, O. W. (2010). Notieren, Skizzieren: Schreiben und Zeichnen als Verfahren des Entwurfs. Zurich, CH/Berlin, DE: Diaphanes. Retrieved from `http://kar.kent.ac.uk/48627/`

Kristensson, P. O. & Denby, L. C. (2009). Text entry performance of state of the art unconstrained handwriting recognition: A longitudinal user study. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Boston, MA, 2009* (pp. 567–570). CHI '09. New York, NY: ACM. `https://doi.org/10.1145/1518701.1518788`

Kristensson, P. O. & Vertanen, K. (2012). Performance comparisons of phrase sets and presentation styles for text entry evaluations. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces: Lisbon, PT* (pp. 29–32). IUI '12. New York, NY: ACM. `https://doi.org/10.1145/2166966.2166972`

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Neural Information Processing Systems, 25*. `https://doi.org/10.1145/3065386`

Krpan, N. & Jakobovic, D. (2012). Parallel neural network training with OpenCL. In *Proceedings of the 35th International Convention MIPRO: Opatija, HR, 2012* (pp. 1053–1057). Piscataway, NJ: IEEE.

Kuhlen, R. (2013). Information - Informationswissenschaft. In R. Kuhlen (Ed.), *Grundlagen der praktischen Information und Dokumentation : Handbuch zur Einführung in die Informationswissenschaft und -praxis* (6th ed., pp. 1–24). München, DE: De Gruyter Saur.

Law, H. & Deng, J. (2020). CornerNet: Detecting Objects as Paired Keypoints. *International Journal of Computer Vision*, *128*(3), 642–656.

LeCun, Y., Bengio, Y., & Hinton, G. E. (2015). Deep learning. *Nature*, *521*(7553), 436–444. `https://doi.org/10.1038/nature14539`

LeCun, Y., Boser, B., Denker, J. S., Howard, R. E., Habbard, W., Jackel, L. D., & Henderson, D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2* (pp. 396–404). San Francisco, CA: Morgan Kaufmann Publishers Inc.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551. `https://doi.org/10.1162/neco.1989.1.4.541`

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. `https://doi.org/10.1109/5.726791`

Li, F.-F., Krishna, R., & Xu, D. (2020). CS231n: Convolutional neural networks for visual recognition. Last accessed on February 18th, 2021. Stanford University, Stanford, United States. Retrieved from `https://cs231n.github.io/neural-networks-3/`

Li, H. [Hao], Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2018). Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems* (pp. 6389–6399).

Li, H. [Hui], Wang, P., & Shen, C. (2017). Towards end-to-end text spotting with convolutional recurrent neural networks. In *2017 IEEE International Conference on Computer Vision (ICCV): Venice, IT* (pp. 5238–5246). ICCV '17. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/iccv.2017.560`

Li, Z., Annett, M., Hinckley, K., Singh, K., & Wigdor, D. (2019). HoloDoc: Enabling mixed reality workspaces that harness physical and digital content. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems: Glasgow, UK* (687:1–687:14). CHI '19. New York, NY: ACM.
`https://doi.org/10.1145/3290605.3300917`

Liao, C. & Guimbretièere, F. (2012). Evaluating and understanding the usability of a pen-based command system for interactive paper. *ACM Transactions on Computer-Human Interaction*, *19*(1).
`https://doi.org/10.1145/2147783.2147786`

Liao, C., Guimbretière, F., & Hinckley, K. (2005). PapierCraft: A command system for interactive paper. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology: Seattle, WA* (pp. 241–244). UIST '05. New York, NY: ACM.
`https://doi.org/10.1145/1095034.1095074`

Liao, C., Guimbretière, F., Hinckley, K., & Hollan, J. (2008). Papiercraft: A gesture-based command system for interactive paper. *ACM Transactions on Computer-Human Interaction*, *14*(4).
`https://doi.org/10.1145/1314683.1314686`

Liao, C., Guimbretière, F., & Loeckenhoff, C. E. (2006). Pen-top feedback for paper-based interfaces. In *Proceedings of the 19th Annual ACM Symposium on User*

Interface Software and Technology: Montreux, CH (pp. 201–210). UIST ’06. New York, NY: ACM.
https://doi.org/10.1145/1166253.1166285

Lifshitz, I., Fetaya, E., & Ullman, S. (2016). Human pose estimation using deep consensus voting. In *Proceedings of the IEEE European Conference on Computer Vision: Amsterdam, NL* (pp. 246–260). ECCV ’16. Cham, DE: Springer.
https://doi.org/10.1007/978-3-319-46475-6_16

Lim, S., Kim, I., Kim, T., Kim, C., & Kim, S. (2019). Fast AutoAugment. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 32, pp. 6665–6675). Red Hook, NY: Curran Associates, Inc.

Lin, M., Chen, Q., & Yan, S. (2013). Network in network. arXiv preprint: 1312.4400

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., … Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *European Conference on Computer Vision: Zurich, CH, 2014* (pp. 740–755). ECCV ’14. Cham, DE: Springer.
https://doi.org/10.1007/978-3-319-10602-1_48

Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.-J., … Murphy, K. (2018). Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision: Munich, DE, 2018* (pp. 19–34). ECCV ’18. Cham, DE: Springer.

Liu, Y., Chen, H., Shen, C., He, T., Jin, L., & Wang, L. (2020). ABCNet: Real-time scene text spotting with adaptive bezier-curve network. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): Online* (pp. 9809–9818). Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvpr42600.2020.00983

Livescribe Dot Paper. (2020). Last accessed on February 11th, 2021. Retrieved from https://www.livescribe.com/en-us/support/ls3/dot%5C_paper.html

Livescribe Smartpens. (2020). Last accessed on April 20th, 2020. Retrieved from https://www.livescribe.com

Livni, R., Shalev-Shwartz, S., & Shamir, O. (2014). On the computational efficiency of training neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1: Montreal, CAN* (pp. 855–863). NIPS '14. Cambridge, MA: MIT Press.

Long, A. C., Landay, J. A., & Rowe, L. A. (1998). *PDA and gesture uses in practice: Insights for designers of pen-based user interfaces. Technical report*. Berkeley, CA: University of California.

Long, A. C., Landay, J. A., Rowe, L. A., & Michiels, J. (2000). Visual similarity of pen gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: The Hague, NL, 2000* (pp. 360–367). CHI '00. New York, NY: ACM. https://doi.org/10.1145/332040.332458

Mackay, W. E. (2003). The missing link: Integrating paper and electronic documents. In *Proceedings of the 15th Conference on l'Interaction Homme-Machine: Caen, FR, 2003* (pp. 1–8). IHM '03. New York, NY: ACM. https://doi.org/10.1145/1063669.1063671

Mackay, W. E., Pagani, D. S., Faber, L., Inwood, B., Launiainen, P., Brenta, L., & Pouzol, V. (1995). Ariel: Augmenting paper engineering drawings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Denver, CO, 1995* (pp. 421–422). CHI '95. New York, NY: ACM. https://doi.org/10.1145/223355.223763

Mackay, W. E., Pothier, G., Letondal, C., Bøegh, K., & Sørensen, H. E. (2002). The missing link: Augmenting biology laboratory notebooks. In *Proceedings of the*

15th Annual ACM Symposium on User Interface Software and Technology: Paris, FR, 2002 (pp. 41–50). UIST '02. New York, NY: ACM Press.
`https://doi.org/10.1145/571985.571992`

MacKenzie, I. S. & Soukoreff, R. W. (2003). Phrase sets for evaluating text entry techniques. In *Extended Abstracts on Human Factors in Computing Systems: Ft. Lauderdale, FL, 2003* (pp. 754–755). CHI EA '03. New York, NY: ACM.
`https://doi.org/10.1145/765891.765971`

MacKenzie, I. S. & Ware, C. (1993). Lag as a determinant of human performance in interactive systems. In *Proceedings of the 1993 Conference on Human Factors in Computing Systems: Amsterdam, NL* (pp. 488–493). CHI '93. New York, NY: ACM.
`https://doi.org/10.1145/169059.169431`

Macukow, B. (2016). Neural networks – state of art, brief history, basic models and architecture. In *International Conference on Computer Information Systems and Industrial Management: Vilnius, LT* (pp. 3–14). IFIP '16. Cham, DE: Springer.
`https://doi.org/10.1007/978-3-319-45378-1_1`

Mangen, A., Anda, L., Oxborough, G., & Brønnick, K. (2015). Handwriting versus keyboard writing: Effect on word recall. *Journal of Writing Research*, *7*(2), 227–247.
`https://doi.org/10.17239/jowr-2015.07.02.1`

Manmatha, R., Han, C., & Riseman, E. M. (1996). Word spotting: A new approach to indexing handwriting. In *1996 IEEE Conference on Computer Vision and Pattern Recognition: San Francisco, CA* (pp. 631–637). CVPR '96. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/cvpr.1996.517139`

Mardani, M., Gong, E., Cheng, J. Y., Vasanawala, S. S., Zaharchuk, G., Xing, L., & Pauly, J. M. (2018). Deep generative adversarial neural networks for compressive sensing mri. *IEEE Transactions on Medical Imaging*, *38*(1), 167–179. https://doi.org/10.1109/TMI.2018.2858752

Mărieş, I. C. (2020). Pytest-Benchmark. Last accessed on January 26th, 2021. Retrieved from https://pytest-benchmark.readthedocs.io/en/latest/index.html/

Marshall, C. (2010). *Reading and writing the electronic book*. Synthesis lectures on information concepts, retrieval, and services. San Rafael, CA: Morgan & Claypool.

Masci, J., Meier, U., Ciresan, D. C., & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In T. Honkela, W. Duch, M. A. Girolami, & S. Kaski (Eds.), *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, FI, June 14-17, 2011, Proceedings, Part I* (Vol. 6791, pp. 52–59). Lecture Notes in Computer Science. Berlin, Heidelberg, DE: Springer. https://doi.org/10.1007/978-3-642-21735-7_7

McCulloch, W. S. & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115–133. https://doi.org/10.1007/bf02478259

McGuire, J. & Tamny, M. (1983). *Certain philosophical questions: Newton's trinity notebook*. Newton's Trinity Notebook. Cambridge, UK: Cambridge University Press.

Microsoft. (2020). Take handwritten notes in OneNote. Last accessed on February 26th, 2020. Retrieved from https://support.microsoft.com/en-us/office/take-handwritten-notes-in-onenote-0ec88c54-05f3-4cac-b452-9ee62cebbd4c

Microsoft Azure. (2020). Last accessed on May 20th, 2020. Retrieved from https://azure.microsoft.com

Miller, G. A. (1998). *WordNet: An electronic lexical database*. Cambridge, MA: MIT Press.

Minsky, M. (1961). Steps toward artificial intelligence. *Proceedings of the IRE*, *49*(1), 8–30.
`https://doi.org/10.1109/jrproc.1961.287775`

Minsky, M. & Papert, S. A. (1969). *Perceptrons: An introduction to computational geometry*. Cambridge, MA: MIT Press.

Minsky, M. & Papert, S. A. (1988). *Perceptrons: An introduction to computational geometry*. Cambridge, MA: MIT Press.

Mueller, P. A. & Oppenheimer, D. M. (2014). The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. *Psychological Science*, *25*(6), 1159–1168.
`https://doi.org/10.1177/0956797614524581`

Müller, R., Kornblith, S., & Hinton, G. E. (2019). When does label smoothing help? *32*, 4694–4703.

Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., & Snoek, J. (2020). Evaluating prediction-time batch normalization for robustness under covariate shift. arXiv preprint: `2006.10963`

Nair, V. & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning: Haifa, ISR* (pp. 807–814). ICML '10. Madison, WI: Omnipress.

Nardi, B. A. (1995). Context and consciousness. In B. A. Nardi (Ed.), (Chap. Studying Context: A Comparison of Activity Theory, Situated Action Models, and Distributed Cognition, pp. 69–102). Cambridge, MA: Massachusetts Institute of Technology.

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. In *Proceedings of the IEEE European Conference on Computer Vision:*

Amsterdam, NL (pp. 483–499). ECCV '16. Cham, DE: Springer. https://doi.org/10.1007/978-3-319-46484-8_29

Nibali, A., He, Z., Morgan, S., & Prendergast, L. (2018). Numerical coordinate regression with convolutional neural networks. arXiv preprint: 1801.07372

Nielsen, J. (1993). Noncommand user interfaces. *Communications of the ACM, 36*(4), 83–99. https://doi.org/10.1145/255950.153582

Norman, D. A. (1993). *Things that make us smart: Defending human attributes in the age of the machine*. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

Norman, D. A. & Nielsen, J. (2010). Gestural interfaces: A step backward in usability. *Interactions, 17*(5), 46–49. https://doi.org/10.1145/1836216.1836228

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., & Sohl-Dickstein, J. (2018). Sensitivity and generalization in neural networks: An empirical study. In *6th International Conference on Learning Representations, ICLR 2018: Vancouver, BC, Canada, Conference Track Proceedings*. ICLR '18. OpenReview.net.

Nunamaker Jr, J. F., Chen, M., & Purdin, T. D. (1990). Systems development in information systems research. *Journal of Management Information Systems, 7*(3), 89–106. https://doi.org//10.1080/07421222.1990.11517898

NVIDIA. (2020). NVIDIA TensorRT. Last accessed on May 5th, 2020. Retrieved from https://developer.nvidia.com/tensorrt

Ogunyemi, A. A., Lamas, D., Lárusdóttir, M. K., & Loizides, F. (2019). A systematic mapping study of HCI practice research. *International Journal of Human-Computer Interaction, 35*(16), 1461–1486. https://doi.org/10.1080/10447318.2018.1541544

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, *9*(1), 62–66. https://doi.org/10.1109/tsmc.1979.4310076

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., … Snoek, J. (2019). Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems* (pp. 13991–14002).

Ozkan, T. (2017). *Predicting recidivism through machine learning* (Doctoral dissertation, University of Texas, Dallas, TX).

Pan, S. J. & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359. https://doi.org/10.1109/TKDE.2009.191

Park, S. & Shin, D. (2015). Effects of text input system on learner's memory: Handwriting versus typing on tablet pc. In *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication: Bali, IDN* (30:1–30:4). IMCOM '15. New York, NY: ACM. https://doi.org/10.1145/2701126.2701198

Parker, E. B. (1974). Information and Society. In C. A. Caudra & M. J. Bates (Eds.), *Library and Information Service Needs of the Nation. Proceedings of a Conference on the Needs of Occupational, Ethnic, and other Groups in the United States.* Washington, DC: Eric.

Pascual, S., Bonafonte, A., & Serrà, J. (2017). SEGAN: speech enhancement generative adversarial network. In F. Lacerda (Ed.), *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association: Stockholm, Sweden, 2017* (pp. 3642–3646). ISCA. https://doi.org/10.21437/interspeech.2017-1428

Patterson, J. & Gibson, A. (2017). *Deep learning: A practitioner's approach*. Sebastopol, CA: O'Reilly.

Pavlakos, G., Zhou, X., Chan, A., Derpanis, K. G., & Daniilidis, K. (2017). 6-DoF object pose from semantic keypoints. In *2017 IEEE International Conference on Robotics and Automation (ICRA): Singapore, SGP* (pp. 2011–2018). Piscataway, NJ: IEEE Press.
`https://doi.org/10.1109/ICRA.2017.7989233`

Peffers, K., Tuunanen, T., Rothenberger, M., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, *24*(3), 45–77.
`https://doi.org/10.2753/mis0742-1222240302`

Peng, C. & Wen, X. (1999). Recent applications of artificial neural networks in forest resource management: An overview. *Transfer*, *1*(X2), W1.

Piolat, A., Olive, T., & Kellogg, R. T. (2005). Cognitive effort during note taking. *Applied Cognitive Psychology*, *19*(3), 291–312.
`https://doi.org/10.1002/acp.1086`

Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M. P., ... Iyengar, S. S. (2018). A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys*, *51*(5), 1–36.
`https://doi.org/10.1145/3234150`

Pratikakis, I., Zagoris, K., Karagiannis, X., Tsochatzidis, L., Mondal, T., & Marthot-Santaniello, I. (2019). ICDAR 2019 competition on document image binarization (DIBCO 2019). In *2019 International Conference on Document Analysis and Recognition: Sydney, AUS* (pp. 1547–1556). ICDAR '19. Los Alamitos, CA: IEEE Computing Society.
`https://doi.org/10.1109/icdar.2019.00249`

Priya, A., Mishra, S., Raj, S., Mandal, S., & Datta, S. (2016). Online and offline character recognition: A survey. In *2016 International Conference on Communication and Signal Processing: Melmaruvathur, IND* (pp. 967–970). ICCSP '16.

Piscataway, NJ: IEEE Press.
https://doi.org/10.1109/iccsp.2016.7754291

Qiao, S., Chen, L., & Yuille, A. (2021). DetectoRS: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): Nashville, TN* (pp. 10208–10219). CVPR '21. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/CVPR46437.2021.01008

Ramsundar, B. & Zadeh, R. (2018). *Tensorflow for deep learning: From linear regression to reinforcement learning*. Sebastopol, CA: O'Reilly.

Ranzato, M., Poultney, C., Chopra, S., & LeCun, Y. (2006). Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. C. Platt, & T. Hofmann (Eds.), *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems: Vancouver, BC, CAN, 2006* (pp. 1137–1144). NIPS '06. Cambridge, MA: MIT Press.

Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y. L., Tan, J., … Kurakin, A. (2017). Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017: Sydney, NSW, AUS* (pp. 2902–2911). ICML '17. Cambridge, MA: PMLR.

Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149.
https://doi.org/10.1109/tpami.2016.2577031

Riche, Y., Henry Riche, N., Hinckley, K., Panabaker, S., Fuelling, S., & Williams, S. (2017). As we may ink?: Learning from everyday analog pen use to improve digital ink experiences. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems: Denver, CO* (pp. 3241–3253). CHI '17. New York,

NY: ACM.
`https://doi.org/10.1145/3025453.3025716`

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386.

Rowley, J. (2007). The Wisdom Hierarchy: Representations of the DIKW Hierarchy. *Journal of Information Science*, *33*(2), 163–180.
`https://doi.org/10.1177/0165551506070706`

Rowley, J. & Hartley, R. (2008). *Organizing knowledge: An introduction to managing access to information*. Farnham, UK: Ashgate.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.
`https://doi.org/10.1038/323533a0`

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*, 211–252.
`https://doi.org/10.1007/s11263-015-0816-y`. eprint: `1409.0575`

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117.
`https://doi.org/10.1016/j.neunet.2014.09.003`

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., ... Dennison, D. (2015). Hidden technical debt in machine learning systems. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015: Montreal, CAN* (Vol. 28, pp. 2503–2511). NIPS '15. Red Hook, NY: Curran Associates, Inc.

Sejnowski, T. J. [Terrence J] & Rosenberg, C. R. (1987). Parallel networks that learn to pronounce English text. *Complex Systems*, *1*(1), 145–168.

Sellen, A. J. & Harper, R. H. (2003). *The myth of the paperless office*. Cambridge, MA: MIT Press.

Seok, J.-H., Levasseur, S., Kim, K.-E., & Kim, J. (2008). Tracing handwriting on paper document under video camera. In *Proceedings of the 11th International Conference on Frontiers in Handwriting Recognition: Montreal, CAN, 2008*. ICFHR '08. Montreal, CAN: CENPARMI, Concordia University.

Shalev-Shwartz, S. & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge, UK: Cambridge University Press. https://doi.org/10.1017/cbo9781107298019

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, *27*(3), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Shih, P.-C., Chiu, C.-Y., & Chou, C.-H. (2019). Using dynamic adjusting NGHS-ANN for predicting the recidivism rate of commuted prisoners. *Mathematics*, *7*(12), 1187. https://doi.org/10.3390/math7121187

Shorten, C. & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*, 1–48. https://doi.org/10.1186/s40537-019-0197-0

Sierra-Canto, X., Madera-Ramirez, F., & Uc-Cetina, V. (2010). Parallel training of a back-propagation neural network using CUDA. In *The Ninth International Conference on Machine Learning and Applications, ICMLA 2010: Washington, DC* (pp. 307–312). ICMLA '10. Los Alamitos, CA: IEEE Computer Society. https://doi.org/10.1109/icmla.2010.52

Sifre, L. & Mallat, S. (2014). *Rigid-motion scattering for image classification* (Doctoral dissertation).

Signer, B. (2005). *Fundamental concepts for interactive paper and cross-media information spaces* (Doctoral dissertation). `https://doi.org/10.3929/ethz-a-005174378`

Signer, B. & Norrie, M. C. (2007). PaperPoint: A paper-based presentation and interactive paper prototyping tool. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction 2007: Baton Rouge, LA* (pp. 57–64). TEI '07. New York, NY: ACM. `https://doi.org/10.1145/1226969.1226981`

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... Lanctot, M., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489. `https://doi.org/10.1038/nature16961`

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., ... Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, *362*(6419), 1140–1144. `https://doi.org/10.1126/science.aar6404`

Simonyan, K. & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems* (pp. 568–576).

Simonyan, K. & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Y. Bengio & Y. LeCun (Eds.), *3rd international conference on learning representations, ICLR 2015: San diego, ca, conference track proceedings*. Retrieved from `http://arxiv.org/abs/1409.1556`

Sloman, A. (2011). What's information, for an organism or intelligent machine? how can a machine or organism mean? In *Information and Computation: Essays on Scientific and Philosophical Understanding of Foundations of Information and Computation* (pp. 393–438). Singapore, SGP: World Scientific. `https://doi.org/10.1142/9789814295482_0015`

Smith, E. H. B. (2010). An analysis of binarization ground truthing. In *The Ninth IAPR International Workshop on Document Analysis Systems, DAS 2010: Boston, MA* (pp. 27–34). DAS '10. New York, NY: ACM.
`https://doi.org/10.1145/1815330.1815334`

Socha, D., Frever, T., & Zhang, C. (2015). Using a large whiteboard wall to support software development teams. In *48th Hawaii International Conference on System Sciences, HICSS 2015: Kauai, HI* (pp. 5065–5072). HICSS '15. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/hicss.2015.600`

Söderström, U., Hellgren, M., & Mejtoft, T. (2019). Evaluating electronic ink display technology for use in drawing and note taking. In *Proceedings of the 31st European Conference on Cognitive Ergonomics, ECCE 2019: Belfast, UK* (pp. 108–113). New York, NY: ACM.
`https://doi.org/10.1145/3335082.3335105`

Srivastava, R. K., Greff, K., & Schmidhuber, J. (2015). Training very deep networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2: Montreal, CAN* (pp. 2377–2385). NIPS '15. Cambridge, MA: MIT Press.

Steimle, J., Brdiczka, O., & Muhlhauser, M. (2009). CoScribe: Integrating paper and digital documents for collaborative knowledge work. *IEEE Transactions on Learning Technologies, 2*(3), 174–188.
`https://doi.org/10.1109/tlt.2009.27`

Steimle, J., Mühlhäuser, M., & Hollan, J. (2012). *Pen-and-paper user interfaces: Integrating printed and digital documents*. Human-Computer Interaction Series. Berlin: Springer.

Su, D., Zhang, H., Chen, H., Yi, J., Chen, P.-Y., & Gao, Y. (2018). Is robustness the cost of accuracy? – a comprehensive study on the robustness of 18 deep image classification models. In *Proceedings of the European Conference on Computer*

Vision: Munich, DE, 2018 (pp. 631–648). ECCV '18. Cham, DE: Springer. `https://doi.org/10.1007/978-3-030-01258-8_39`

Sulaiman, A., Omar, K., & Nasrudin, M. F. (2019). Degraded historical document binarization: A review on issues, challenges, techniques, and future directions. *Journal of Imaging*, *5*(4), 48. `https://doi.org/10.3390/jimaging5040048`

Sun, Y., Wang, X., & Tang, X. (2013). Deep convolutional network cascade for facial point detection. In *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Portland, OR* (pp. 3476–3483). CVPR '13. Los Alamitos, CA: IEEE Computer Society. `https://doi.org/10.1109/cvpr.2013.446`

Sutherland, C. J., Luxton-Reilly, A., & Plimmer, B. (2016). Freeform digital ink annotations in electronic documents: A systematic mapping study. *Computers & Graphics*, *55*, 1–20. `https://doi.org/10.1016/j.cag.2015.10.014`

Sutherland, I. E. (1963). *Sketchpad, a man-machine graphical communication system* (Doctoral dissertation, Massachusetts Institute of Technology, Cambridge, MA).

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems* (pp. 3104–3112).

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Boston, MA* (pp. 1–9). CVPR '15. Los Alamitos, CA: IEEE Computer Society. `https://doi.org/10.1109/cvpr.2015.7298594`

Talkad Sukumar, P., Liu, A., & Metoyer, R. (2018). Replicating user-defined gestures for text editing. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces: Tokyo, JPN* (pp. 97–106). ISS '18. New York,

NY: ACM.
https://doi.org/10.1145/3279778.3279793

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., & Le, Q. V. (2019). MnasNet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: Long Beach, CA, 2019* (pp. 2820–2828). CVPR '19. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvpr.2019.00293

Tan, M. & Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning, ICML 2019: Long beach, ca* (Vol. 97, pp. 6105–6114). Proceedings of Machine Learning Research. Cambridge, MA: PMLR.

Tensmeyer, C. & Martinez, T. (2019). Robust keypoint detection. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW): Sydney, AUS* (Vol. 5, pp. 1–7). ICDARW '19. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/icdarw.2019.40072

Tesauro, G. & Sejnowski, T. J. [Terrence J.]. (1989). A parallel network that learns to play backgammon. *Artificial Intelligence*, *39*(3), 357–390.
https://doi.org/10.1016/0004-3702(89)90017-9

Tharwat, A. (2020). Classification assessment methods. *Applied Computing and Informatics*, *17*(1), (168–192).
https://doi.org/10.1016/j.aci.2018.08.003

Thornton, C., Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2013). Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: Chicago, IL* (pp. 847–855). KDD '13. New York, NY: ACM.
https://doi.org/10.1145/2487575.2487629

Tian, F., Lu, F., Jiang, Y., Zhang, X. (, Cao, X., Dai, G., & Wang, H. (2013). An exploration of pen tail gestures for interactions. *International Journal of Human-Computer Studies*, *71*(5), 551–569.
`https://doi.org/10.1016/j.ijhcs.2012.12.004`

Tompson, J. J., Jain, A., LeCun, Y., & Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems* (pp. 1799–1807).

Toselli, A. H., Vidal, E., Puigcerver, J., & Noya-García, E. (2019). Probabilistic multi-word spotting in handwritten text images. *Pattern Analysis and Applications*, *22*(1), 23–32.
`https://doi.org/10.1007/s10044-018-0742-z`

Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV): Santiago, CHL* (pp. 4489–4497). ICCV '15. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/iccv.2015.510`

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., & Paluri, M. (2018). A closer look at spatiotemporal convolutions for action recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): Salt Lake City, UT* (pp. 6450–6459). CVPR '18. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/cvpr.2018.00675`

Trippi, R. R. & Turban, E. (1992). *Neural networks in finance and investing: Using artificial intelligence to improve real world performance*. New York, NY: McGraw-Hill, Inc.

Trullemans, S. & Signer, B. (2014). From user needs to opportunities in personal information management: A case study on organisational strategies in cross-media information spaces. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries: London, UK, 2014* (pp. 87–96). JCDL '14. Piscataway, NJ:

IEEE Press.
https://doi.org/10.1109/jcdl.2014.6970154

Tsandilas, T. & Mackay, W. E. (2010). Knotty gestures: Subtle traces to support interactive use of paper. In *Proceedings of the International Conference on Advanced Visual Interfaces: Rome, IT, 2010* (pp. 147–154). AVI '10. New York, NY: ACM.
https://doi.org/10.1145/1842993.1843020

Tu, H., Ren, X., & Zhai, S. (2015). Differences and similarities between finger and pen stroke gestures on stationary and mobile devices. *ACM Transactions on Computer-Human Interaction*, *22*(5), 22:1–22:39.
https://doi.org/10.1145/2797138

Vaishnavi, V. K. & Kuechler, W. (2015). *Design science research methods and patterns: Innovating information and communication technology, 2nd edition* (2nd). Boca Raton, FL: CRC Press, Inc.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop: Sunnyvale, CA, 2016*. ISCA. Retrieved from http://www.isca-speech.org/archive/SSW%5C_2016/abstracts/ssw9%5C_DS-4%5C_van%5C_den%5C_Oord.html

Vapnik, V. & Chervonenkis, A. J. (1991). The necessary and sufficient conditions for consistency in the empirical risk minimization method. *Pattern Recognition and Image Analysis*, *1*, 285–305.

Vatavu, R.-D., Vogel, D., Casiez, G., & Grisoni, L. (2011). Estimating the perceived difficulty of pen gestures. In *Human-Computer Interaction - INTERACT 2011 - 13th IFIP TC 13 International Conference, Lisbon, Portugal, Proceedings, Part II* (pp. 89–106). Berlin, Heidelberg, DE: Springer.
https://doi.org/10.1007/978-3-642-23771-3_9

Venable, J., Pries-Heje, J., & Baskerville, R. (2017). Choosing a design science research methodology. In *The 28th Australasian Conference on Information Systems: Hobart, AUS, 2017*. Hobart, AUS: University of Tasmania.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., & Bottou, L. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, *11*(12).

Vo, Q. N., Kim, S., Yang, H. J., & Lee, G. (2018). Binarization of degraded document images based on hierarchical deep supervised network. *Pattern Recognition*, *74*, 568–586.
`https://doi.org/10.1016/j.patcog.2017.08.025`

Wacom Smart Pads. (2020). Last accessed on April 20th, 2020. Retrieved from `https://www.wacom.com/en-en/products/smartpads`

Wang, P., Li, H., & Shen, C. (2022). Towards end-to-end text spotting in natural scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(10), 7266–7281.
`https://doi.org/10.1109/TPAMI.2021.3095916`

Wang, X., Bo, L., & Fuxin, L. (2019). Adaptive wing loss for robust face alignment via heatmap regression. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV): Seoul, KOR* (pp. 6971–6981). ICCV '19. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/iccv.2019.00707`

Ware, C. (2012). *Information visualization: Perception for design* (3rd ed.). San Francisco, CA: Morgan Kaufmann Publishers Inc.

Wasserman, P. D. & Schwartz, T. (1987). Neural networks, part 1: What are they and why is everybody so interested in them now? *IEEE Expert*, *2*(4), 10–11.

Wei, L., Xiao, A., Xie, L., Chen, X., Zhang, X., & Tian, Q. (2020). Circumventing outliers of AutoAugment with knowledge distillation. arXiv preprint: `2003.11342`

Weiss, G., Goldberg, Y., & Yahav, E. (2018). On the practical computational power of finite precision RNNs for language recognition. In I. Gurevych & Y. Miyao (Eds.), *Proceedings of the 56th annual meeting of the association for computational linguistics, ACL 2018: Melbourne, australia, volume 2: Short papers* (pp. 740–745). ACL '18. Stroudsburg, PA: Association for Computational Linguistics. `https://doi.org/10.18653/v1/P18-2117`

Wellner, P. (1993). Interacting with paper on the digitaldesk. *Communications of the ACM*, *36*(7), 87–96. `https://doi.org/10.1145/159544.159630`

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550–1560. `https://doi.org/10.1109/5.58337`

Wersig, G. (1971). *Information, Kommunikation, Dokumentation : ein Beitrag zur Orientierung der Informations- und Dokumentationswissenschaft*. Beiträge zur Informations- und Dokumentationswissenschaft 5. München, DE: Dokumentation Saur KG.

Widrow, B. (1960). *An adaptive "adaline" neuron using chemical "memistors"* (tech. rep. No. 1553-2). Stanford, CA: Stanford Electronics Laboratories, Stanford University.

Widrow, B. (1962). Generalization and information storage in networks of adaline "neurons". In M. Yovits, G. Jacobi, & G. Goldstein (Eds.), *Self-Organizing Systems*. Washington DC: Spartan Books.

Widrow, B. & Hoff, M. E. (1960). *Adaptive switching circuits* (tech. rep. No. 1553-1). Stanford, CA: Stanford Electronics Laboratories, Stanford University.

Wobbrock, J. O., Wilson, A. D., & Li, Y. (2007). Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology:*

Newport, RI, 2007 (pp. 159–168). UIST ’07. New York, NY: ACM. `https://doi.org/10.1145/1294211.1294238`

Wolff, C. (2009). ”embedded media computing“ – die Regensburger Ausrichtung der Medieninformatik.

Wu, P.-C., Wang, R., Kin, K., Twigg, C., Han, S., Yang, M.-H., & Chien, S.-Y. (2017). DodecaPen: Accurate 6DoF tracking of a passive stylus. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology: Québec City, CAN, 2017* (pp. 365–374). UIST ’17. New York, NY: ACM. `https://doi.org/10.1145/3126594.3126664`

Wu, Y. & Ji, Q. (2019). Facial landmark detection: A literature survey. *International Journal of Computer Vision*, *127*(2), 115–142. `https://doi.org/10.1007/s11263-018-1097-z`

Xiao, B., Wu, H., & Wei, Y. (2018). Simple baselines for human pose estimation and tracking. In *Proceedings of the European Conference on Computer Vision: Munich, DE, 2018* (pp. 466–481). ECCV ’18. Cham, DE: Springer. `https://doi.org/10.1007/978-3-030-01231-1_29`

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., & Woo, W.-c. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems* (pp. 802–810).

Yang, K., Qinami, K., Fei-Fei, L., Deng, J., & Russakovsky, O. (2020). Towards fairer datasets: Filtering and balancing the distribution of the people subtree in the ImageNet hierarchy. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency* (pp. 547–558). `https://doi.org/10.1145/3351095.3375709`

Ye, J. C. & Sung, W. K. (2019). Understanding geometry of encoder-decoder CNNs. In *International Conference on Machine Learning: Long Beach, CA, 2019* (pp. 7064–7073). ICML ’19. Cambridge, MA: PMLR.

Yeo, R. (2014). Introduction. In *Notebooks, English Virtuosi, and Early Modern Science* (pp. 1–35). Chicago/London: University of Chicago Press.

Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., & Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR): Boston, MA* (pp. 4694–4702). CVPR '15. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/cvpr.2015.7299101`

Zeiler, M. D. & Fergus, R. (2013). Visualizing and understanding convolutional networks. In *European Conference on Computer Vision: Zurich, CH, 2014*. ECCV '14. Cham, DE: Springer.
`https://doi.org/10.1007/978-3-319-10590-1\_53`. arXiv preprint: `1311.2901`

Zeiler, M. D., Taylor, G. W., & Fergus, R. (2011). Adaptive deconvolutional networks for mid and high level feature learning. In *IEEE International Conference on Computer Vision, ICCV 2011: Barcelona, ES* (pp. 2018–2025). ICCV '11. Los Alamitos, CA: IEEE Computer Society.
`https://doi.org/10.1109/iccv.2011.6126474`

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2020). *Dive into deep learning 0.15.1*. `https://d2l.ai` PDF version.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *5th international conference on learning representations, ICLR 2017: Toulon, fr, conference track proceedings*. ICLR '17. OpenReview.net. arXiv preprint: `1611.03530`

Zhang, F., Zhu, X., Dai, H., Ye, M., & Zhu, C. (2020). Distribution-aware coordinate representation for human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR): Online* (pp. 7093–

7102). CVPR ’20. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvpr42600.2020.00712

Zhang, H.-B., Zhang, Y.-X., Zhong, B., Lei, Q., Yang, L., Du, J.-X., & Chen, D.-S. (2019). A comprehensive survey of vision-based human action recognition methods. *Sensors*, *19*(5), 1005.
https://doi.org/10.3390/s19051005

Zhang, S., Gunupudi, P., & Zhang, Q.-J. (2015). Parallel back-propagation neural network training technique using CUDA on multiple GPUs. In *2015 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization: Ottawa, CAN* (pp. 1–3). NEMO ’15. Piscataway, NJ: IEEE Press.
https://doi.org/10.1109/nemo.2015.7415056

Zhang, Y. [Yu], Chan, W., & Jaitly, N. (2017). Very deep convolutional networks for end-to-end speech recognition. In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing: New Orleans, LA* (pp. 4845–4849). ICASSP ’17. Piscataway, NJ: IEEE Press.
https://doi.org/10.1109/icassp.2017.7953077

Zhang, Y. [Yuhang], Sun, H., Zuo, J., Wang, H., Xu, G., & Sun, X. (2018). Aircraft type recognition in remote sensing images based on feature learning with conditional generative adversarial networks. *Remote Sensing*, *10*(7), 1123.
https://doi.org/10.3390/rs10071123

Zhou, X., Zhuo, J., & Krahenbuhl, P. (2019). Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition: Long Beach, CA, 2019* (pp. 850–859). CVPR ’19. Los Alamitos, CA: IEEE Computer Society.
https://doi.org/10.1109/cvpr.2019.00094

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. In *2018 IEEE/CVF Conference on Computer*

Vision and Pattern Recognition (CVPR): Salt Lake City, UT (pp. 8697–8710). CVPR ’18. Los Alamitos, CA: IEEE Computer Society. `https://doi.org/10.1109/cvpr.2018.00907`

A. Appendix A: Qualitative Study

A.1. Interview Codings

The interview codings for the qualitative study of contemporary scholarly workflows are provided in digital form on the accompanying media. The file is stored under `/interview-study/transcripts.pdf` and is best navigated by using the PDF bookmarks for the coded interview transcripts of the participants S1–S11. This content is a result of the Master's thesis of Michael Achmann, which was supervised by the author. If the accompanying digital storage is not available, please refer to Achmann (2021) for the codings.

A.2. Interview Framework

The German interview framework developed in collaboration with Michael Achmann is stored under `/interview-study/framework.pdf`. The questions relevant for note taking are translated here:

Do you take notes during your research?

1. How do you take notes (on what medium)?
   a) Do you take notes in books you have read or in printed copies?
   b) Do you use a pad, scratch paper or notebook for your notes?

Note Type	**Category**	Reported by Subject
Conversation notes	Communication	S9 (P23, 29, 71)
Lecture notes	Communication	S1(P64), S11(P54)
Translation notes	Research Task	S1 (P 64), S11 (P34, P76)
Literature Notes (Excerpts)	Research Task	S1 (P2, 60), S2 (P24, P44), S4 (P26), S5, S6(P30, P46), S7(P36, P52), S8 (P12, 14, 42), S9(P6), S10(P22, 24), S11 (P50, P58)
To-Do lists	Structure	S2 (P54) S3 (P80), S6 (P26, 86), S9 (P79)
Mind Maps	Structure	S2 (P28), S5 (P98), S9 (P81)
Timelines	Structure	S1 (P2, P68)
Thinking notes	Insight	S4 (P60), S5 (P66, P72), S7 (P58), S10 (P6)
Notes to Self	Insight	S1 (P48), S2 (P34, P50) S3 (P26, P80), S4 (P62), S6(P28, P86)

Table A.1. – Types of notes the participants reported making during their scholarly workflow.

c) Do you use a computer to take notes?

d) Do you take notes electronically directly in an e-book, PDF, or on scanned pages?

e) Do you use a tablet to take notes?

2. When do you take notes? (e.g. before/after reading literature)

3. What do you note down?

   a) Do you use abbreviations or codes?

   b) Would the notes be readable by others?

   c) Are the notes understandable without associated text/source?

   d) Would they be understandable to others?

   e) Are the notes made to be accessed later?

4. What things do you take notes on in your research?

5. Would you like to improve anything about this process?

B. Appendix B: Neural Networks and Digital Support Material

This part of the dissertation documents the digital support material that is relevant to the technological investigations. Every paragraph is named according to the directory on the accompanying media.

B.1. Models and Evaluation

handwritten text extraction

- `evaluation.xlsx` Evaluation results for IOU and BA.
- `dataset` The training dataset (UbiPen-Binarize) with binarized ground truth patches and the dataset augmented with typography.
- `te-b0` Weights in h5-format and training history
- `te-b0-lorem` Weights in h5-format and training history (typography augmentation)
- `text-ex-parameters.xlsx` Parameter documentation.

keypoint-baseline/roi

The weight naming scheme for the keypoint detection variants is: *backbone_stage_decoder-size_freeze_augmentation-magnitude*. Decoder size *u2* corresponds to small decoders, *u4* is large. Frozen lower layers are designated by *ft (freeze true)*, and unfrozen lower layers by *ff (freeze false)*.

- `effnet-b0` Weights for EfficientNet-B0 backbone variants for ROI detection and training history. Detailed evaluation CSV files.
- `effnet-b5` Weights for EfficientNet-B5 backbone variants for ROI detection and training history. Detailed evaluation CSV files.
- `nasnetmobile` Weights for NasNet Mobile backbone variants for ROI detection and training history. Detailed evaluation CSV files.
- `resnetv2` Weights for EfficientNet-B0 backbone variants for ROI detection and training history. Detailed evaluation CSV files.
- `roi-parameters.xlsx` Parameter documentation.

keypoint-baseline/detail

The naming scheme follows the same as for the ROI variants.

- `effnet-b0` Weights for EfficientNet-B0 backbone variants for detail detection and training history. Detailed evaluation CSV files.
- `effnet-b5` Weights for EfficientNet-B5 backbone variants for detail detection and training history. Detailed evaluation CSV files.
- `nasnetmobile` Weights for NasNet Mobile backbone variants for detail detection and training history. Detailed evaluation CSV files.

- `resnetv2` Weights for EfficientNet-B0 backbone variants for detail detection and training history. Detailed evaluation CSV files.
- `detail-parameters.xlsx` Parameter documentation.

prototype-analysis

- `transmission_test` The test image for non-random transmission tests and the transmission timings
- `image_resize_results.xlsx` Image resize test run results for PIL, TF, and Scikit-Image
- `inference_time.xlsx` Inference timings for EfficientNet-B0 variants for keypoint detection and handwritten text extraction as well as sequence inferences. TRT-optimized timings for EfficientNet-B0 variants.

sequence-analysis

- `convlstm layers` Weights for ConvLSTM-variants trained with 3, 10, and 30 timesteps, as well as training parameters and detailed evaluation results.
- `flat recurrent layers` Weights for LSTM- and GRU-variants trained with 3, 10, 30 and 60 timesteps, as well as training parameters and detailed evaluation results.
- `time-agnostic baseline` Weights for the AB-0, 1, and 2 time-agnostic baselines, as well as training parameters and detailed evaluation results.
- `concat-models` Weights for the Combo-CLS-KP and Combo-CLS models, as well as training parameters and detailed evaluation results.

B.2. Datasets

pov-prestudy

This directory contains all annotated samples from the POV-Prestudy.

pov-keypoint

This directory contains all annotated samples from the POV-Keypoint dataset in TFRecord format.

ubipen-sequence

This directory contains all annotated samples from the UbiPen-Sequence dataset in TFRecord format.

simple-keypoint

This directory contains all annotated samples from the Simple-Keypoint dataset in TFRecord format.

wild-keypoint

This directory contains all annotated samples from the Wild-Keypoint dataset in TFRecord format.

gesture-study

This directory contains all gesture videos for the pen gesture study. Annotations are provided via file names with the naming scheme *subject-id_keypoint-name_background_gesture-name_gesture-id*.

TFRecord Descriptors

These file descriptors in *Python* code describe the layout of the TFRecord examples inside the dataset collections. `image_feature_description` is for keypoint samples, `sequence_image_feature_description` is for sequential data with keypoint and pen state annotation, and `bin_set_pair_feature_description` is for the samples used to train the handwritten text extraction.

```
image_feature_description = {
    'image/encoded': tf.io.FixedLenFeature([], tf.string),
    'image/format': tf.io.FixedLenFeature([], tf.string),
    'image/subject_id': tf.io.FixedLenFeature([], tf.int64),
    'image/task_id': tf.io.FixedLenFeature([], tf.int64),
    'image/frame_id': tf.io.FixedLenFeature([], tf.int64),
    'image/ptip_x': tf.io.FixedLenFeature([], tf.float32),
    'image/ptip_y': tf.io.FixedLenFeature([], tf.float32),
    'image/ptail_x': tf.io.FixedLenFeature([], tf.float32),
    'image/ptail_y': tf.io.FixedLenFeature([], tf.float32),
    'image/height': tf.io.FixedLenFeature([], tf.int64),
    'image/width': tf.io.FixedLenFeature([], tf.int64),
}
```

```
sequence_image_feature_description = {
    'image/encoded': tf.io.FixedLenFeature([], tf.string),
    'image/format': tf.io.FixedLenFeature([], tf.string),
    'image/subject_id': tf.io.FixedLenFeature([], tf.int64),
    'image/task_id': tf.io.FixedLenFeature([], tf.int64),
    'image/frame_id': tf.io.FixedLenFeature([], tf.int64),
    'image/sequence_id': tf.io.FixedLenFeature([], tf.int64),
    'image/ptip_x': tf.io.FixedLenFeature([], tf.float32),
    'image/ptip_y': tf.io.FixedLenFeature([], tf.float32),
    'image/ptail_x': tf.io.FixedLenFeature([], tf.float32),
    'image/ptail_y': tf.io.FixedLenFeature([], tf.float32),
    'image/pen_down': tf.io.FixedLenFeature([], tf.int64),
    'image/height': tf.io.FixedLenFeature([], tf.int64),
    'image/width': tf.io.FixedLenFeature([], tf.int64),
}

bin_set_pair_feature_description = {
    'image/frame_id': tf.io.FixedLenFeature([], tf.int64),
    'image/subject_id': tf.io.FixedLenFeature([], tf.int64),
    'image/encoded': tf.io.FixedLenFeature([], tf.string),
    'bin/encoded': tf.io.FixedLenFeature([], tf.string),
    'image/height': tf.io.FixedLenFeature([], tf.int64),
    'image/width': tf.io.FixedLenFeature([], tf.int64)
}
```

B.3. Code

src

This directory contains the python source code for the parts of the technological investigation. Some parts were implemented using the older Tensorflow version 1.14 and there are some parts that were implemented using Tensorflow 2.1 and 2.3. Further documentation about the general structure of the code is available in the `readme` file.

This directory contains all necessary code to create the neural network models evaluated in this dissertation. It also contains the prototype implementation and parallelization classes as well as benchmarks, supporting classes, and data preparation.

B.4. File Formats

The two relevant file formats are `TFRecord` and `H5`. Information about the former can be found under `https://www.tensorflow.org/tutorials/load_data/tfrecord`. `H5` stands for *hierarchical data format* in the version 5.
See `https://de.wikipedia.org/wiki/Hierarchical_Data_Format` for more information.

B.5. Dissertation PDF

The PDF version of this dissertation lies in the root directory of the accompanying media under `dissertation-florin-schwappach.pdf`.

B.6. Prototype Video

A video showcasing the prototype in action is available in the root directory under `prototype-showcase.mp4`.