UPPSALA
UNIVERSITET

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 1475

Interpreting the Script

Image Analysis and Machine Learning for Quantitative Studies of Pre-modern Manuscripts

FREDRIK WAHLBERG

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2017

ISSN 1651-6214
ISBN 978-91-554-9814-6
urn:nbn:se:uu:diva-314211

Dissertation presented at Uppsala University to be publicly examined in Tidskriftläsesalen, Carolina rediviva, Dag Hammarskjölds väg 1, Uppsala, Friday, 24 March 2017 at 10:15 for the degree of Doctor of Philosophy. The examination will be conducted in English. Faculty examiner: Professor Apostolos Antonacopoulos (University of Salford, Manchester, UK).

Abstract
Wahlberg, F. 2017. Interpreting the Script. Image Analysis and Machine Learning for Quantitative Studies of Pre-modern Manuscripts. *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 1475. 95 pp. Uppsala: Acta Universitatis Upsaliensis. ISBN 978-91-554-9814-6.

The humanities have for a long time been a collection of fields that have not gained from the advancements in computational power, as predicted by Moore´s law. Fields like medicine, biology, physics, chemistry, geology and economics have all developed quantitative tools that take advantage of the exponential increase of processing power over time. Recent advances in computerized pattern recognition, in combination with a rapid digitization of historical document collections around the world, is about to change this.

The first part of this dissertation focuses on constructing a full system for finding handwritten words in historical manuscripts. A novel segmentation algorithm is presented, capable of finding and separating text lines in pre-modern manuscripts. Text recognition is performed by translating the image data of the text lines into sequences of numbers, called features. Commonly used features are analysed and evaluated on manuscript sources from the Uppsala University library Carolina Rediviva and the US Library of Congress. Decoding the text in the vast number of photographed manuscripts from our libraries makes computational linguistics and social network analysis directly applicable to historical sources. Hence, text recognition is considered a key technology for the future of computerized research methods in the humanities.

The second part of this thesis addresses digital palaeography, using a computers superior capacity for endlessly performing measurements on ink stroke shapes. Objective criteria of character shapes only partly catches what a palaeographer use for assessing similarity. The palaeographer often gets a feel for the scribe's style. This is, however, hard to quantify. A method for identifying the scribal hands of a pre-modern copy of the revelations of saint Bridget of Sweden, using semi-supervised learning, is presented. Methods for production year estimation are presented and evaluated on a collection with close to 11000 medieval charters. The production dates are estimated using a Gaussian process, where the uncertainty is inferred together with the most likely production year.

In summary, this dissertation presents several novel methods related to image analysis and machine learning. In combination with recent advances of the field, they enable efficient computational analysis of very large collections of historical documents.

Keywords: document analysis, machine learning, image analysis, digital humanities, document dating, writer identification, text recognition

Fredrik Wahlberg, Department of Information Technology, Division of Visual Information and Interaction, Box 337, Uppsala University, SE-751 05 Uppsala, Sweden. Department of Information Technology, Computerized Image Analysis and Human-Computer Interaction, Box 337, Uppsala University, SE-75105 Uppsala, Sweden.

ISSN 1651-6214
ISBN 978-91-554-9814-6
urn:nbn:se:uu:diva-314211 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-314211)

To Emily, Jorik and Iris

List of papers

This dissertation is based on the following papers, which are referred to in the text by their Roman numerals.

I **Fredrik Wahlberg**, Mats Dahllöf, Lasse Mårtensson and Anders Brun. Data mining medieval documents by word spotting. In *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing (HIP '11)*, pages 75–82, New York, NY, USA, 2011. Association for Computing Machinery (ACM). DOI: 10.1145/2037342.2037355

II **Fredrik Wahlberg** and Anders Brun. Graph based line segmentation on cluttered handwritten manuscripts. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1570–1573, Nov 2012. ISBN: 978-4-9906441-0-9, IEEE Xplore: 6460444.

III **Fredrik Wahlberg** and Anders Brun. Feature weight optimization and pruning in historical text recognition. In *International Symposium on Visual Computing*, pages 98–107. Springer, 2013. DOI: 10.1007/978-3-642-41939-3_10

IV **Fredrik Wahlberg** and Anders Brun. Feature space denoising improves word spotting. In *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing (HIP '13)*, pages 59–66, New York, NY, USA, 2013. Association for Computing Machinery (ACM). DOI: 10.1145/2501115.2501118

V **Fredrik Wahlberg**, Mats Dahllöf, Lasse Mårtensson and Anders Brun. Spotting words in medieval manuscripts. *Studia Neophilologica*, 86(sup1):171–186, 2014. DOI: 10.1080/00393274.2013.871975

VI **Fredrik Wahlberg**, Lasse Mårtensson and Anders Brun. Scribal attribution using a novel 3-d quill-curvature feature histogram. In *14th International Conference on Frontiers in Handwriting Recognition (ICFHR 2014)*, Crete, Greece, September 1-4, 2014, pages 732–737. Institute of Electrical & Electronics Engineers (IEEE), Sep 2014. DOI: 10.1109/ICFHR.2014.128

VII **Fredrik Wahlberg**, Lasse Mårtensson and Anders Brun. Large scale style based dating of medieval manuscripts. In *The 3rd International Workshop on Historical Document Imaging and Processing (HIP '15)*, 2015. Pages 107-114. DOI: 0.1145/2809544.2809560

VIII **Fredrik Wahlberg**, Lasse Mårtensson and Anders Brun. Large scale continuous dating of medieval scribes using a combined image and language model. In *12th IAPR Workshop on Document Analysis Systems (DAS 2016)*. Institute of Electrical & Electronics Engineers (IEEE), Apr 2016. DOI: 10.1109/DAS.2016.71

IX **Fredrik Wahlberg**, Tomas Wilkinson and Anders Brun. Historical Manuscript Production Date Estimation using Deep Convolutional Neural Networks. In *15th International Conference on Frontiers in Handwriting Recognition (ICFHR 2016)*. DOI: 10.1109/ICFHR.2016.0048

Associated (peer-reviewed) acts

Fredrik Wahlberg, Alexander Medvedev and Olov Rosén. A LEGO-based mobile robotic platform for evaluation of parallel control and estimation algorithms, *50th IEEE Conference on Decision and Control and European Control Conference*, Orlando, FL, 2011, pp. 4548-4553. DOI: 10.1109/CDC.2011.6161252

Lasse Mårtensson, **Fredrik Wahlberg** and Anders Brun. Digital palaeography and the old swedish script: The quill feature method as a tool for scribal attribution. *Arkiv för nordisk filologi*, 2015. ISSN 0066-7668, Volume 130, pages 79-100. URN: urn:nbn:se:hig:diva-19435

Technical reports

Fredrik Wahlberg, Mats Dahllöf, Lasse Mårtensson and Anders Brun. Word spotting in pre-modern manuscripts using dynamic time warping. *Swedish Symposium on Image Analysis*, 2012.

Fredrik Wahlberg and Anders Brun. Feature space de-noising for text recognition. *Swedish Symposium on Image Analysis*, 2014.

Fredrik Wahlberg, Anders Brun and Lasse Mårtensson. Writer identification using the quill-curvature feature in old manuscripts. *Swedish Symposium on Image Analysis*, 2015.

Fredrik Wahlberg, Lasse Mårtensson and Anders Brun. Estimating manuscript production dates using both image and language data. *Swedish Symposium on Image Analysis*, march 2016.

Summary of contributions

The roman numerals correspond to the numbers in the list of papers.

I Sole contributor to the experiment implementation. Significant contributions to the theoretical design, experiment planning, writing and conclusions. Minor contributions to paper idea and disposition.

II Sole contributor to the paper idea and experiment implementation. Significant contributions to the theoretical design, experiment planning, disposition, writing and conclusions.

III Sole contributor to the experiment implementation. Significant contributions to the paper idea, theoretical design, experiment planning, disposition, writing and conclusions.

IV Sole contributor to the experiment planning and implementation. Significant contributions to the paper idea, theoretical design, disposition, writing and conclusions.

V Sole contributor to the experiment planning and implementation. Significant contributions to the paper idea, theoretical design, disposition and conclusions. Minor contributions to the writing.

VI Sole contributor to the theoretical design, experiment planning and implementation. Significant contributions to the paper idea, disposition, writing and conclusions.

VII Sole contributor to the theoretical design, disposition, experiment planning and implementation. Significant contributions to the paper idea, writing and conclusions.

VIII Sole contributor to the paper idea, theoretical design, disposition, experiment planning and implementation. Significant contributions to the writing and conclusions.

IX Significant contributions to the paper idea, theoretical design, experiment planning and implementation, disposition, writing and conclusions.

Sammanfattning på svenska (Summary in Swedish)

Humaniora är en samling fält som inte har gagnats av de senaste decenniernas framsteg inom datorkraft, så som områdena medicin, biologi, fysik, kemi, geologi eller ekonomi har. Datoriserad analys av text har haft visst genomslag, vilket lett till fältet datorlingvistiks uppkomst. Däremot har andra typer av data (t.ex. bilder, CT-volymer, markpenetrerande radar) sällan analyserats automatiskt. På senare tid har framsteg inom datoriserad mönsterigenkänning öppnat upp för nya möjligheter att automatisera analysen av olika typer av data och i allt större mängder. Denna utveckling mot datorbaserade metoder för forskning inom humaniora har fått namnet *digital humaniora*. Varför ska då allt det material som idag finns på museum och i bibliotek digitaliseras? Eftersom det ger våra bevarande institutioner möjligheten att spara en digital kopia de objekt de har i sina samlingar, sker det idag på bred front. Denna digitalisering för att bevara vårt kulturarv sker idag i hela världen.

Humaniora är ett mycket brett begrepp som ofta beskrivs som en samling fält där det fundamentalt mänskliga studeras. Hur ska då en dator kunna bidra i den forskningen. Grunden till detta är att människor och datorer är bra på olika typer av analys. En dator kan inte tolka vad en människa menat eller känt, däremot är den mycket bra på att outtröttligt göra mätningar och processa stora samlingar av statistik. Hela projektet bygger därför på att göra kvantitativa studier möjliga där ett kvalitativt tillvägagångssätt varit det rådande paradigmet. Dessutom kan kvantitativa studier, som av materiella skäl inte kunnat skalas upp till större samlingar, med datorns hjälp kunna greppa mer än vad enskilda människor kan. Visionen är att det nu starkt växande området digital humaniora kommer att utvecklas analogt med hur biologi och datavetenskap bildade vetenskapsområdet bioinformatik. Idag är det otänkbart att inte använda datorbaserade metoder inom stora delar av biologi. Det är dock en empirisk fråga om detta är möjligt på bred front även inom humaniora.

I den här avhandlingen ligger fokus på två områden: *textigenkänning* och *identifiering av skrivarhänder*. Textigenkänning är processen där den handskrivna text som finns i digitala bilder avkodas till digital text. Detta gör texten sökbar och öppnar i förlängningen också för mer avancerad analys likt den för sociala nätverk. Identifiering av skrivarhänder är en kartläggning av stil i syfte att binda händer till dokument eller att uppskatta det år som ett dokument skrevs. Idag är datering av manuskript ett mycket svårt problem där forskare i slutändan ofta är utelämnade till textens innehåll. Datoriserad mätning av likheter och trender i stil öppnar för att se vem som skrev och när detta skedde. Detta möjliggör en kartläggning av de som stod för det skrivna ordet i historien (här ligger fokus dock på medeltida skrifter).

Den första delen av denna avhandling fokuserar på textigenkänning av handskriven text (i bemärkelsen penna i handen, ej skrivmaskin). Den grupp av metoder som fokuseras på avvänder en metodologi som börjar med hela bilden för att sedan i flera olika steg bryta ned den till an hierarki av mindre bildobjekt.

Textrader isoleras från varandra (s.k. segmentering) för att sedan klippas ihop på rad efter varandra. På så vis representeras texten som en bild av en enda lång textrad. Denna långa rad mäts enligt ett förutbestämt schema och byter skepnad till en lång sekvens av mätningar av textens form. Mindre delar av denna sekvens kan nu, inte olikt DNA-analys, jämföras med hela sekvensen för att hitta områden som liknar varandra. Dessa områden där sekvensen av mätningar liknar varandra korresponderar mot områden som liknar varandra i originalbildens textrader.

Olika sätt att förbättra denna avkodningsprocess har undersökts och presenteras nedan, och i de fem första artiklarna som inkluderas i avhandlingen. Här presenteras också en ny metod för att hitta och separera textrader i en bild. De metoder som utvecklats har utvärderats på material från Uppsala Universitets bibliotek och ifrån Library of Congress. Detta material är "Summula de ministris et sacramentis ecclesiasticis" (C64) som avhandlar kyrkans göromål under 1400-talet, heliga Birgittas uppenbarelser (C61) i en kopia från tidigt 1500-tal och en del av "George Washington Letters" som är en brevsamling från 1754–1755.

Den andra delen av denna avhandling fokuserar på att identifiera stilen hos skrivarhänder. Till viss del handlar det om att identifiera enskilda skrivarhänder i materialet C61, men till större del att identifiera tidsvarierande stildrag för att datera manuskript. Samlingen "Svenskt Diplomatariums Huvudkartotek" (SDHK) innehåller närmare 11000 digitaliserade brev från svensk medeltid. Dessa brev är officiell kommunikation som är daterad på dagen innan brevet först skickades. Begreppet avsändare behöver här delas upp i kategorierna författare och skrivare. Under medeltiden (framför allt tidig sådan) var det vanligt att använda sig av en skrivare även om avsändaren kunde skriva själv. Författaren behöver alltså inte ha varit inblandad i själva skrivandet, utan enbart dikterat.

Den paleografiskt relevanta bakgrunden till denna studie är att kriterier för formen av skrivtecken endast delvis fångar vad en expert inom paleografi skulle använda för att bedöma likheten mellan två manuskript. När man tittar på två manuskript kan man ofta få en känsla för stilen hos en skrivarhand, genom att kort inspektera några enskilda sidor. Denna känsla är dock mycket svår att kvantifiera. Hur sätts en siffra på exempelvis rundheten hos en stil? Datorns kapacitet till att outtröttligt göra mätningar kan här utnyttjas för en sådan kvantifiering.

Delvis identifieras skrivarhänder i boken C61, vilket oberoende visade samma uppsättning skrivare som den mest auktoritativa identifiering gjord av en människa. Detta gjordes med hjälp av semi-övervakat lärande, en maskininlärningsterm som betyder att hänsyn tas till struktur i datamaterialet tillsammans med den identifiering av skrivarhänder som en människa kan ge algoritmen. Till större del handlar den senare delen av arbetet om att identifiera produktionsår för manuskript. Detta görs både med djupa neurala nätverk och med en för hand designad egenskapsutvinning. Osäkerheten i dateringen kvantifie-

ras av en Gaussisk process i syfte att uppskatta var datorn behöver hjälp av en människa. Den slutgiltiga dateringen rapporteras för varje undersökt brev som en normalfördelning över tidslinjen.

I denna avhandling presenteras metodutveckling för datorstödd forskning inom humaniora. De fält som arbetet direkt bygger på faller dock inom kategorierna dokumentanalys, maskininlärning, datorseende och datoriserad bildanalys.

Acknowledgements

Hard work is a necessary condition for finishing a dissertation, but not a sufficient one. Many people have been involved in making this dissertation possible, as I (like those before me) truly stand on the shoulders of giants. With every day (and some nights) being focused on this endeavour, for the past five years, I would, firstly, like to thank my wife and family for the joy and inspiration they bring to my life every day. **Emily Lekström** you are my rock and without your support, love and guidance I would never have been able to finish this dissertation. Thank you **Jorik** and **Iris**, for how you have never allowed me to forget that playing is much more important than work.

To my parents, **Anita Wahlberg** and **Janne Wahlberg**, who are literally the pillars to the world to which I stand. Thank you for each and every important life lesson you have imparted upon me. I am forever grateful for you helping and believing in me.

Lars Djerf, **Johan Fång**, **Kajsa Lamm** and **Timo Järpeskog**, thanks to each and every one of you and your individual roles that helped to develop my computing and critical analysis skills.

Every life has some critical moments. Those where the figurative Markov process of life makes the more unlikely transitions. A special thanks to my 7^{th} to 9^{th} grade science teacher **Stig Lindgren**, who saw that I was bored and encouraged me to study trigonometry and special relativity outside of the curriculum. After being classified as a hopeless case in mathematics in earlier grades, you gave me the confidence to once again be fascinated. In my university life, a critical person was **Marcus Berg**, who helped me get back into the undergraduate program after I failed miserably back in 2006. Also, thanks to **Tom Smedsaas**, who saw my nerd skills and took me in as a TA.

I would like to thank my main supervisor **Anders Brun** for his support in both my research and navigating the academic world (I can't tell which one has been more important). Also, my co-supervisors **Ewert Bengtsson** and **Lasse Mårtensson**, for sharing your experience and scientific expertise. Thank you fellow PhD-students and office mates **Tomas Wilkinson** and **Kalyan Ram Ayyalasomayajula**. What would I have done without you guys? You have really lifted this research group, though getting work done can sometimes be hard, since you have so many interesting things to say. Thanks also to current and former members of our research group: **Mats Dahllöf**, **Bojana Simsic**, **Alicia Fornés**, **Jonas Lindström**, **Carl Nettelblad** and **Anders Hast**.

Colleagues, friends and guides on the dark paths of academia **Pelle Mattsson**, **Dave Zachariah** and **Ted White**. Time flies in your company and I've learned much about everything from materialist analysis to the intricacies of the academic world, or lack thereof.

Thanks to my colleagues **Omer Ishaq** and **Sadjit Sadanandan Kecheril** for fika, maybe the greatest of Swedish traditions, especially combined with machine learning. Our discussions has given me new perspectives and inspi-

ration. Thanks also to my former office mate **Vlada Curic**, how motivated me to delve deeper into mathematics.

Thanks to our great collaborators and source material experts. **Zeth Alvered**, for providing me with data and expertise in my time of need. **Per Cullhed** and **Krister Östlund**, for discussions, ideas and being our link into the amazing collections at Carolina Rediviva. **Per-Axel Wiktorsson**, for providing help with the "Svenskt Diplomatariums Huvudkartotek" (SDHK) collection, a great set of books on medieval writers and helping me with my own interest in old Nordic writings. **Sara Risberg**, for your assistance with the digitized SDHK and being our link into the Swedish national archive.

Thank you to my parents-in-law **Bengt Lekström** and **Helena Lekström** for helping me with proofreading.

At every workplace, there are people in critical positions that make the day to day actives run smoothly. These pillars of the organization are rarely the management. For me, these people have been **Lena Nordström**, **Henrik Hedlund**, **Joel Fredrikson** and the **cleaning staff**.

No research is performed in a vacuum. The welcoming atmosphere of our research community in document analysis has made conferences fun and a great inspiration. Special thanks to **Jean-Paul van Oosten**, **Andreas Fischer**, **Leonard Rothacker**, **Volkmar Frinken**, **Patrick Doetch**, **Angelica Garz**, **Sheng He**, **Sebastian Sudholt**, **Gernot Fink** and **Seiichi Uchida**. I will finish another conference paper as soon as I can.

Contents

1. Research context

In the following chapter, the context and research questions of the work presented in this dissertation will be described. The authors personal views and experiences of digital humanities will also be developed on. The perspective is from a computer scientist's (or even engineer's), though there is no unspoken claim of talking for all computer scientists. There are several other rather long texts on the subject that will not be commented on, lest this turn into a book on only the definition of digital humanities[1]. Lastly, at the bottom of the section, a brief description of the capabilities of computerized image analysis and data driven research will be presented.

1.1 Aims of this dissertation and a short project history

The research question for this dissertation is in short: Where is the state-of-the-art in applying computerized image analysis to data relevant for any discipline in the humanities, and how can this be used and developed further to leverage research on Swedish pre-modern text sources?

If one were to form a dendrogram according to the themes of the presented herein papers, the first branching would be two well separated groups. The first, text recognition on binarized image, and the second, writer/date attribution. This split is represented by the chaptering (i.e. chapters 3 and 4). The split also mirrors the project history and, to some extent, the development in the field.

I came in to this project during the spring of 2011, doing small implementations of ideas as a pilot project. The aim was to strengthen a large application for funding in handwritten text recognition. I was successful in this endeavour, leading to that we in the summer of 2011 wrote paper I. Due to the momentum shown in our group, a decision was made for a strategic investment (UFV 2011/1913) from three faculties and the vice chancellor for funding one Ph.D. student (with supervision). I applied for this position and got it in early 2012. My first papers in this project concern breaking down a binarized page and performing recognition of handwritten text. This work is presented in chapter 3 and papers I-V.

In the autumn of 2012, we got a significant grant from the Swedish Science Council (Dnr 2012-5743) with my supervisor as the principal investigator (PI).

[1] This, I have no doubt, will be viewed as provocative to some, and I assure the reader I am under no illusion that this is the final word on the matter.

This grant was mainly for handwritten text recognition and data mining on historical sources. Due to this, the group could grow and we could welcome two more Ph.D. students in to the project. Suddenly we went from only me on full time to being the Handwritten Text Recognition (HTR) project, a real research group.

In late 2013, Lasse Mårtensson, our in-house philologist, was curious about a method for identifying scribal hands, based on extracting statistics from a full manuscript page. This was a methodological break from what had been doing so far. I promised to help him implement this and run it on the C61 manuscript, assuming at the time that it would take less than three weeks of work. It not only took longer, but redefined my research since then. This work led to paper VI, later used in another grant application to "Riksbankens Jubileumsfond". In 2015, this grant was given (NHS14-2068:1) with Lasse as the PI, enabling a continuation of the ongoing work on scribal hands. This work is presented in chapter 4 and papers VI-IX.

At the time of writing, the international research community working on image analysis on historical sources is small. However, this small community is connected to broader communities of researchers, from philology to machine learning. When I started working with the group, there were no big grants and no one on full time. This has truly been a fascinating journey. In the following chapters, I will describe my contributions, as a part of this community, to making the humanities more digital.

1.2 Thoughts on digital humanities and imaging

The field of digital humanities (DH) has a new name but is not a new direction of research. Since the invention of computers, text processing has been an important application. However, with the increasing computational power and storage capacity gained over time, it is now possible to handle large collections of images. This was impossible only 20 years ago. An important invention for this dissertation was the charge-coupled device (CCD) in the 1960's[2]. The CCD chip (and its CMOS successor) enables the digital imaging we use today for preserving our heritage. Libraries like that of the Vatican, and several other large libraries, are continuing to digitizing documents in the range of petabytes (a petabyte is one million gigabytes). All this data can then potentially be processed as images, and not only as transcribed text. This development opens up for new and exciting research possibilities, assuming we can find the computerized methods to deal with this large amount of data. This is where DH comes in, as the field of finding the relevant research questions and methods to deal with such data. Going back to the example of text processing, it has

[2]Nobel prize in physics in 2009, http://www.nobelprize.org/nobel_prizes/physics/laureates/2009/

been around for some time but only recently has it been used for large scale quantitative analysis of books, as in the paper by Michel et al. (2011)[3].

DH is sometimes described as the intersection between the humanities and computer science, as illustrated in figure 1.1 as a Venn diagram. Defining what digital humanities is (or is not) in detail is an ongoing process. Today, there is no consensus. Some are even worried that it might just be another buzzword. Defining a multi-disciplinary effort in an excluding way is likely divisive and counterproductive. We do not know where the introduction of computational capabilities in many of the disciplines concerning the fundamentally human will lead. However, if the term is defined in an including way, several examples of research can form a foundation upon which to build.

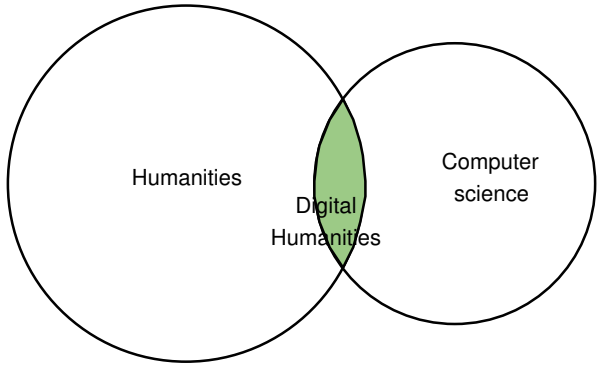

Figure 1.1. A Venn diagram of the common conception that digital humanities is the intersection of the Humanities and Computer Science.

Forty years ago, it was not evident that bioinformatics (i.e. computational biology) would become a part of standard biological research. Today, it would be unthinkable not to include computational research methods in the field, in parallel to the laboratory and field work. There were two main reasons for this inclusion of computer science, the possibility to sequence the genome and suitable computing capacity. By a stroke of luck, the computational technology of the day was good at matching, often very long, sequences of text data. This was easily extended to sequences of base pairs[4].

Some find it tempting to leave the collaboration part of interdisciplinary work to experts with knowledge in computer science and some field of the humanities, e.g. someone who is both a computer scientist and a philologist. Though finding such people is possible (but often hard) I think the wish in it self is a product of an anxiety about interdisciplinary work. It is hard to communicate for people trained in very different disciplines. Misunderstandings

[3]Studying 4% of all books ever printed for various purposes e.g. evolution of grammar and epidemiology.

[4]The statistical models, however, is a continuing research field e.g. Bayesian hidden Markov models etc.

occur often and there might be large differences in methodological traditions. However, all scientific disciplines are in motion. It can be hard to keep up with one, let alone two, fields. Why not skip the middle man and work on trying to understand each other? Medical doctors and physicists do it when working on medical equipment. Bioinformaticians and computer scientists do it for analysis of proteins or RNA. If a digital humanities researcher does not work on interdisciplinary communication, it will be very hard to produce research that is relevant and novel in several fields at the same time. A better way of doing it is, from my experience of our interdisciplinary research group, to just make an effort to get to know the basics of each other's fields.

Most of the work in this dissertation can be seen as method development. The intention is not to solve a research task for a palaeographer. It is to formulate a research question in collaboration, and add expertise from computer science and mathematics to explore that direction.

The definition of DH as the intersection of computer science and the humanities, as it is illustrated by the Venn diagram in figure 1.1, is intuitive but not without its problems. It makes poetic sense, but a lot of the research that would be DH from the perspective of, lets say, a historian, is not computer science. Research enabled by creating a large database of genealogical data with socio-economic information a significant cross referencing capabilities is a clear case of DH, though creating a database (on this scale) is not a part of computer science research, any more than using a word processor is. Also, it is easy for the machine learning researcher to imagine a case where specific neural network architectures need to be developed for a class of data from palaeography, though this would hardly fit under the humanities umbrella. For these reasons, I think it is better to picture DH as is shown in the expanded Venn diagram of figure 1.2.

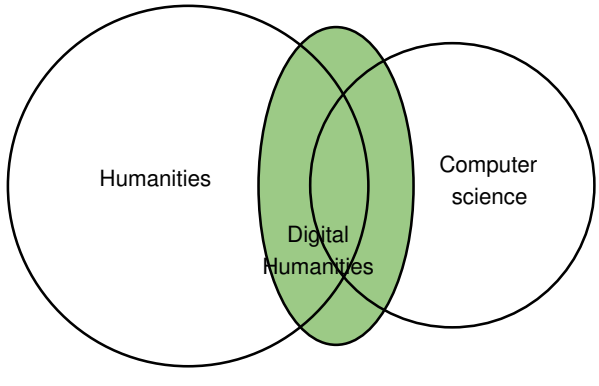

Figure 1.2. A Venn diagram of an expanded definition of Digital Humanities, encompassing the definition from figure 1.1. The digital humanities bubble has been expanded to include research that fit in one of the categories computer science or the humanities, but not necessarily both. Also, a new part outside the two main areas are added as a reminder that new research is, likely, yet to be found.

In figure 1.2, the DH area is expanded to include research that is not necessarily what we today would call either humanities or computer science, in contrast to the diagram in figure 1.1. The examples of projects given above fit this definition, and there is room for the old definition in the new. The new area, which is neither humanities or computer science, can be projects like large scale quantitative analysis of historical documents using yet to be developed methods. In this hypothetical future, the computer science and, for example, history fields would likely be expanded to include this research topic. However, I think having some part of the expanded DH definition outside the current definition of the fields can serve as a reminder that there are new kinds of research out there. We should not be tempted to only see the introduction of computational methods as a way of doing more of the same. With new methods, it is likely that there are opportunities for completely new research.

There has also been some talk of DH being a buzz word and hype. Assuming this is the case, how to exploit the advantages while still mitigating the detrimental effects? A related example is the multiple hype cycles that neural networks has gone though. The hypes was so strong, at some point, that when the bubble did burst, the following years have been described as "neural network winter". Funder confidence was completely depleted. The recipe for success is simple: "Do not promise anything you can not reasonably deliver!". This "no fun" solution might not give you funding either, in which case, the "boom and bust" is inevitable.

A real threat to DH projects is the difference in research cultures between disciplines. The luxury of having a clear definition of success, as in computer science and most often in physics, is not one that is present in, for instance, philosophy. The double meaning of hard sciences, in the all too popular dichotomy of hard and soft sciences, is in some ways misleading. The hard sciences might as well be the humanities or social sciences, since it does not lend itself well to quantification. Galileo's cannonball fell in the same way every time he threw it, in his early work on mechanics. He was able to manipulate something external, repeat his results many times and express them mathematically. However, the repeated experience of making sense of the world though this kind of experimentation can easily lead to a desire to apply the same thinking where problems look very different. This *engineering thinking* is necessary for including computer science in DH, but scepticism of the same is needed for the inclusion of general humanities in DH.

Another aspect of the divide, described in the preceding paragraph, is the need for (and problem with) defining a criterion for success. A discussion on how to define the items and categories that a research project are working with, in some data collection, is necessary. However, to a computer scientist, a long discussion in what way some concept is problematic, might just sound like bickering. There is a real problem of finding a *common definition* of the issues at hand. If a problem can not be stated clearly, translating it into a statistical

model (or any algorithm) is close to impossible. If a computer is going to be involved, there needs to be a clear quantifiable goal.

A final pitfall is the problem of *field specific jargon*. Not terminology, that can easily be corrected, but the use of words that convey values or belonging, sometimes together with scientific concepts. A part of hacker culture even had a manual for its own jargon[5]. The danger here is that in the process of assimilating concepts and researchers from several fields, jargon will be misinterpreted and misused. We risk reinforcing already existing prejudices we have of each other. Luckily, we only need to be generous to our co-workers for this not to be a growing, but a diminishing, problem. In the end, we all want to progress in expanding human knowledge.

Despite the pitfalls described above, there is now great potential to do research, previously restricted to a handful of textual sources, on a large scale. A computer can not interpret, while a human can only interpret a small part of a large collection, due to material circumstances. This is where cross disciplinary collaboration can create something greater than the sum of its parts by amplifying the application expert's capabilities though the use of computational methods. Digital Humanities is this collaboration space.

1.3 Proxy research tasks

In a time of increasing automation and the threat of machines making a large part of the labour market obsolete, it is tempting to assume that this will also be the case in humanistic research. That digital humanities (DH) would be the precursor to this "revolution" in science, forcing strict "physics-style" quantitative method on all scientific disciplines. These are nonsensical fantasies from people who have been listening to too much marketing talk (if you ask those who want you to invest, a technological revolution is always around the corner). These ideas shows a fundamental lack of understanding of the scientific endeavour and research including the human experience. The computer is a fantastic achievement of human intellect and carefully crafted. However, I think it is well worth remembering that a computer not only shares its basic components with rocks (e.g. silicon), but it is also as dumb as one. We can indeed craft machines capable of great feats in pattern recognition, though we are far from the thinking machines of science fiction (yet?).

Some scepticism against DH is based on the notion that the research subject is something "fundamentally human". How do we get computers to help us when they, by definition, can not? From my perspective, the answer to this question is the concept of *proxy tasks*. A task that a computer can perform, where the result is close to indistinguishable from what a human would produce. A task for the computer that can act as a proxy for the task a human

[5] http://jargon-file.org/

would preform. An analogy can be made to the dishwasher. This machine and a human achieve the same goal, yet the processes of achieving clean plates look very different. In chapter 4, the methods from the included papers on computerized dating of manuscripts are presented. A human can estimate the production dates for manuscripts (if properly trained), and so can a computer. The results are very similar, but the procedures are different. A human uses textual clues or palaeography, a computer uses a regression in a high dimensional space spanned by the output from layers of feature extractors. Both "output" years, while using completely different means of getting there. While it is tempting to talk about a computer as a human, we must not forget that it is only a metaphor. The challenge for the computer scientist, is to translate a task some human needs performed into something a machine can help with.

In this perspective, a programmer is an interpreter. The original is a clear description of the research task, and the translation is the code. Note that the machine never searches for a word in a text, it matches a short sequence of binary numbers to some other longer sequence of binary numbers, given some comparison algorithm. The challenge then for a DH researcher is to find a similar task to the original one, that is suitable for a computer to solve while also giving a result as similar as possible to what was sought for.

1.3.1 Imaging from a computational perspective

From the perspective of a computerized image processing pipeline, the input data (i.e. the image) is only a grid of coloured points. The points have a spatial relation on this grid. There is no prior understanding of objects, what a colour might mean or that the image usually is a two-dimensional representation of something in three dimensions. Think of it as an image where you have zoomed in on a detail so much it has lost its meaning. You can still see what colours there are and how the image points are laid out beside each other, but it does not make sense any more. The shapes have no meaning. This is what a computer starts with.

To make sense of this data, a computer only has a limited number of operations that can be performed. There is no "separate object" or "find symmetry" operations available, as in a human brain. A hierarchy of increasingly complex operations is needed to make sense of the image data. The field of computerized image analysis is concerned with developing these operations and evaluating their use for research and industrial applications. A researcher in computer science needs to , in a sense, master the art of building levels of abstraction, from the data to the desired interpretation.

1.3.2 Data driven research with large data sets

Computational power is increasing exponentially, as described by law of Moore et al. (1998), and so is the possibility to process large sets of data. To be able to make sense of this data, indifferent of the type of digital data or task in mind, new computational methods have needed (and still need) to be developed. The traditional pipelines for data processing can not keep up. Another challenge is that a significant portion of the data that is being created have new modalities. Mass storage of images was impossible not that long ago, at the same time as the image types are being expanded into for example mass acquisition of volumetric (3D) data. Not only does this development require better processing, but also new ways of making the machine understand differently structured data than spreadsheets or text documents.

A future challenge for digital humanities is to go towards big data[6]. The knowledge from analysing the structure of large data sets is something different from the currently prevailing forms of qualitative analysis. This is a paradigm shift in many disciplines[7]. However, success stories ranging from machine translation[8] to image captioning[9] shows that great gains might be possible.

Computerized recognition of historical handwritten text is the key technology that enables analysis of the full body of pre-modern text in one go. Making it possible to study the spread of ideas across time and space in human history. This has already been done on more modern English literature, through the use of topic modelling for tracking themes in literature over time and space, e.g. by Blei (2012) and Blei et al. (2003). Identifying writer styles would make it possible to date previously undated manuscripts and track the movements of scribes. More importantly, the spread of scribal styles indicate communication between places at different times. Getting any certainty in these computerized methods, however, demands huge amounts of data.

In summary, big data really refers to quantitative methods, applied on a larger scale, where details that would be lost in smaller collections become visible. The data driven paradigm is about focusing more on what the data itself can show, rather than relying on theory. Thereby it reduces the risk of reaching conclusions built on cognitive bias.

---

[6]Big data is simply a name for using large collections of data. What is considered "big" in the concept of big data is a moving goal post relative to what has already been done in the specific field.

[7]A renowned researcher in history once told me that maybe the traditional (and very successful) methods of analysing historical data *risked* being fetishized and seen as the only way to do meaningful research in the field. That a silent norm have been established in some places, preventing computerized methods from being seen as "real" historical research.

[8]A machine inferring a translation and working out meanings of words from large collections of data, without a human feeding it any set of rules on grammar, e.g. Bahdanau et al. (2014).

[9]Neural networks that can describe your holiday photos with coherent sentences, e.g. Karpathy and Fei-Fei (2015).

2. Data sets

the work presented in the dissertation has been focussed in pre-modern handwritten sources, primarily the Swedish medieval period. This is roughly all manuscripts produced before the year 1523 (the year of Gustav the first's coronation), though in practice the period constitutes the years 1000–1523, since very little was written (and even less preserved) before the year 1000 in what currently is Sweden. This data is described below together with one modern source, the so called "George Washington" data set, which consists of 18^{th} century handwriting.

2.1 "Summula de ministris et sacramentis ecclesiasticis" (C64)

The book "Summula de ministris et sacramentis ecclesiasticis", from the collections of the Uppsala University library, is a manuscript containing a handbook for priests of the medieval Swedish church. It was written in the 14^{th} century at a time when the bureaucracy of the church was almost indistinguishable from that of the state. Its identifier in the archives is C64, catalogued by Andersson-Schmitt and Hedlund (1989).

The manuscript consists of 230 folios with writing on both sides. All was written by a single hand, attributed to "Laurentius of Vaksala" (Vaksala is today an area of Uppsala, Sweden), in the *Northern Textualis* script style. Like many manuscripts of this type, the margins are significant with prominent help lines for text, between markings made by puncturing the page. Two spreads can be seen in figure 2.1. Note the red initials, these mark the beginning of a new part of the text[1].

Writings in this style often have a very complex system for abbreviations. One of the standard dictionaries of such abbreviations is the dictionary by Cappelli (1928), listing thousands. This complicates the computerized decoding of the handwritten characters since the regularities present in modern writing are not there. The spelling can be confusing as it is and abbreviations depend on the context of where they are found. Hence, the computer must solve a problem of interpreting the image information and simultaneously parse the meaning of the words.

The images were taken by the reproduction unit at the Uppsala University library Carolina Rediviva. The photographs are large (almost 18 Mpixels) with

[1] Incidentally, the red colouring is where the name rubric, as in heading/title, comes from.

a very good colour quality. Only 16 spreads of the collection are available in digital form, not enough for a text recognition research but is used in paper II for segmentation experiments. In figure 2.1, two spreads are shown as an example of the reproduction quality.

2.2 "The revelations of Saint Bridget of Sweden" (C61)

The manuscript C61, catalogued by Andersson-Schmitt and Hedlund (1989), contains book 1–8 of the revelations of Saint Bridget of Sweden (1303–1373). It was written in old Swedish and is generally regarded a translation from Latin. This copy was written sometime in the beginning of the 16^{th} century (the Swedish medieval era ends 1523) and made into the current book format in the 17^{th} century by order of Johannes Loccenius. It was given to the Uppsala University Library in 1674 by Loccenius. The script differs considerably from C64 and is categorized as *Cursiva Recentior*, by the criteria set by Derolez (2003). In general, writing from this time in Swedish did not use many abbreviations, as opposed to texts written in Latin. At the time, however, there was no canonized spelling, making the choice of characters very much up to the scribes.

The images were taken by the Uppsala University library Carolina Rediviva with a 15 Mpixel camera. In total, there are 555 images, all are spreads with most having text on both pages. The colour quality is good. Some examples are shown in figure 2.2.

The variation in cursive writing (due to the quick execution) is a source of difficulty when doing scribal attributions (i.e. determining who wrote what). Even though more of the scribes habits show when there is little time to think about each character, formulating a clear morphology that can be compared is hard. The most authoritative attribution of C61 is given by Klemming (1883), shown in table 2.1.

2.3 "Svenskt Diplomatariums Huvudkartotek" (SDHK)

The charter collection "Svenskt Diplomatariums Huvudkartotek" (SDHK) is held by the Swedish national archive. It is a collection with almost 11000 digitized medieval charters. The full collection is even larger (all is not digitized). The charters are from the Swedish medieval period and is the largest collection of its kind in Sweden (there are some much smaller collections concerning specific families). They are all written on parchment. Some charters are large but most are small. A large number (slightly above 5300) are transcribed. An estimation has been done on the scribal hands for more than that (though these are not available in digital form, only in a printed book) by Wiktorsson (2015), who is the foremost expert on the collection. Many charters have intact sigils

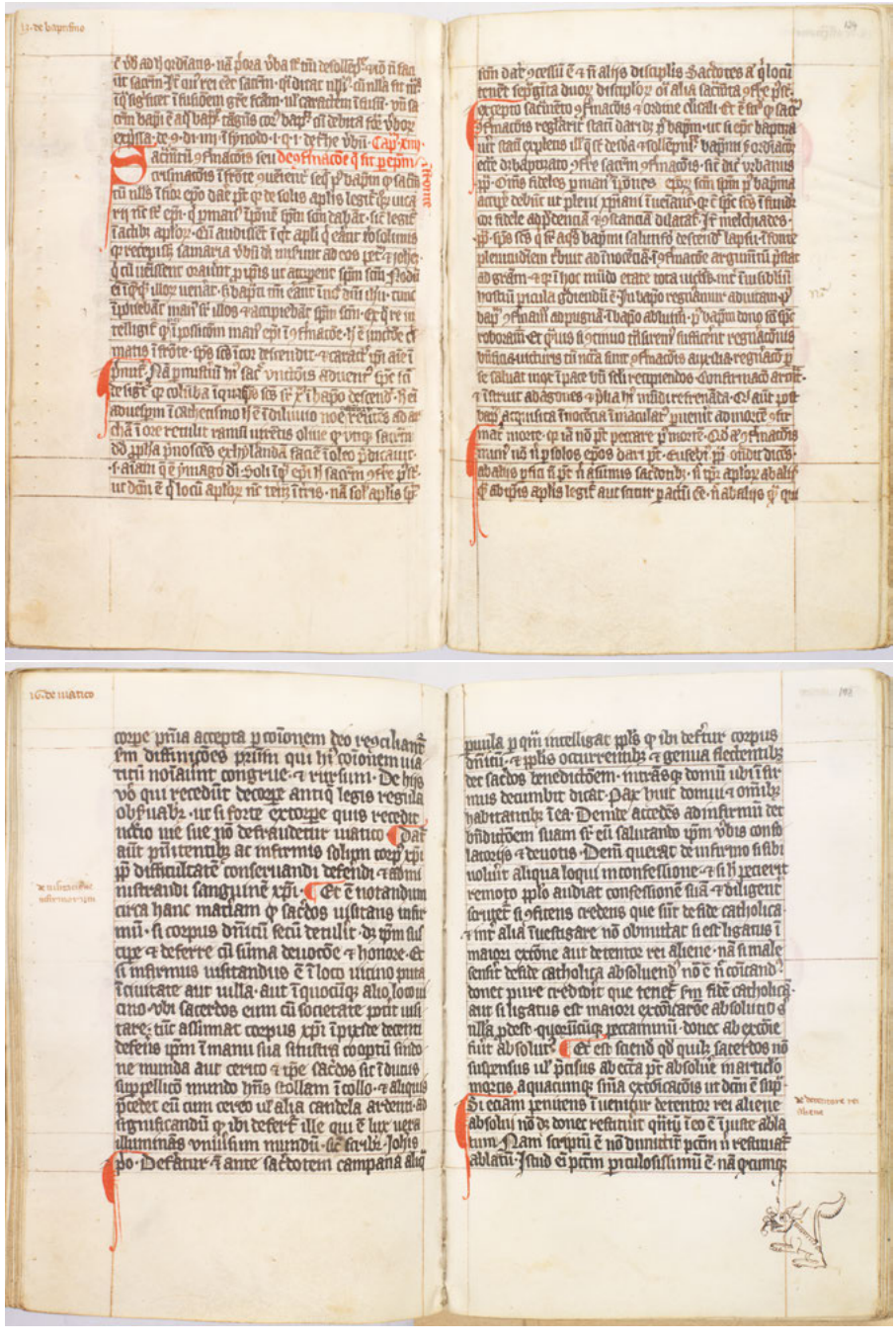

Figure 2.1. Examples of two spreads from C64 (section 2.1). Note the large margins, symbolizing the importance of the text and the cost of production. However, the wish to save space (parchment was expensive) sometimes led to very compressed spaces between words (sometimes spaces are completely removed).

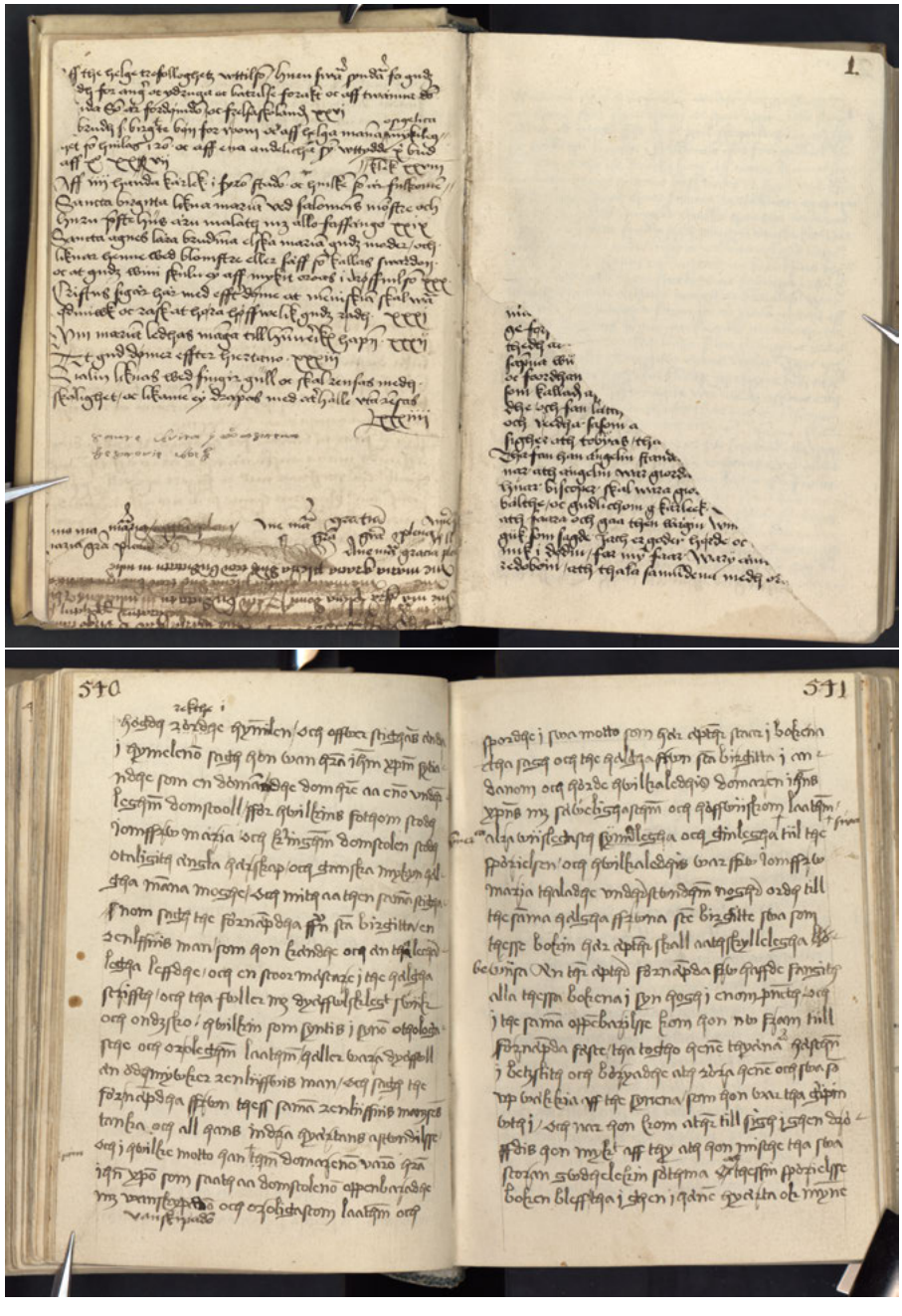

Figure 2.2. Two spreads from the data material C61, pages 0–1 and 540–541 (section 2.2). The obvious difference in writer style is because several scribes worked on finishing this copy. Note the distortions of the text lines, stemming from the binding of the book. For text recognition, these might have to be "straightened out".

Pages	Hand	Pages	Hand	Pages	Hand
1 – 129	1	425 – 428	1	453 – 472	1
130 – 134	4	429 – 430	2	473 – 476	2
135 – 298	1	431 – 438	1	477 – 516	1
299 – 304	2	439 – 440	2	517 – 530	2
305 – 354	1	441 – 444	1	531 – 534	1
355 – 414	2	445 – 446	2	535 – 536	2
415 – 422	1	447 – 450	1	537 – 538	blank
423 – 424	2	451 – 452	2	539 – 1104	3

Table 2.1. *An overview of the scribes who wrote the pages of C61 (section 2.2), as attributed by Klemming (1883). These were attributed to 4 different scribes, on palaeographic grounds. Drawing a metaphorical line where the style difference is enough to introduce a new scribe is very hard. Illustrating this, attributions of pages 130–133 are a matter of dispute and arguments have been put forth that scribe 1, 2, and 4 are all the same person. In paper VI, the classification given in the table is shown to be measurable, supporting the claims by Klemming.*

attached by strips of parchment, showing the authenticity. This collection is exemplary for a computerized study of manuscripts since all charters are dated on the day, except for a very small undated subset. The dating was written by the original authors and was of importance at the time due to the unreliable mail distribution system.

The collection has been published on the web in a low resolution (with jpeg artefacts) format, with the meta data in html[2]. The high resolution images can be bought from the Swedish national archive and are in 4 Mpixel jpeg photographs produced over the last 10 years. Most photos include a brightness reference for calibration. Examples of 8 charters can be seen in figure 2.3. The meta data have been harvested from the website. It includes production dates, places from where the respective charters were sent from, transcriptions (for some), earlier reproductions etc.

2.4 George Washington Letters

The George Washington data set, published as a data base for text recognition by Fischer et al. (2012), is a set consisting of 20 pages from the United States Library of Congress written by George Washington and one secretary (who has an almost indistinguishable style of hand writing from the main author).

The images come from series 2, letterbook 1, pages 270–279 and 300–309, written between August 11 of 1754 and December 25 of 1755. The reproduction is of low quality with grey scale images of less than 1 Mpixel. As a database, the text lines and words have been segmented manually. Images of the cropped words and lines are supplied (with normalization, a technique

[2]https://sok.riksarkivet.se/SDHK

Figure 2.3. A sample of charters found in the "Svenskt Diplomatariums Huvudkartotek" (SHDK) collection (section 2.3). These are all medieval charters showing varying degrees of degradation.

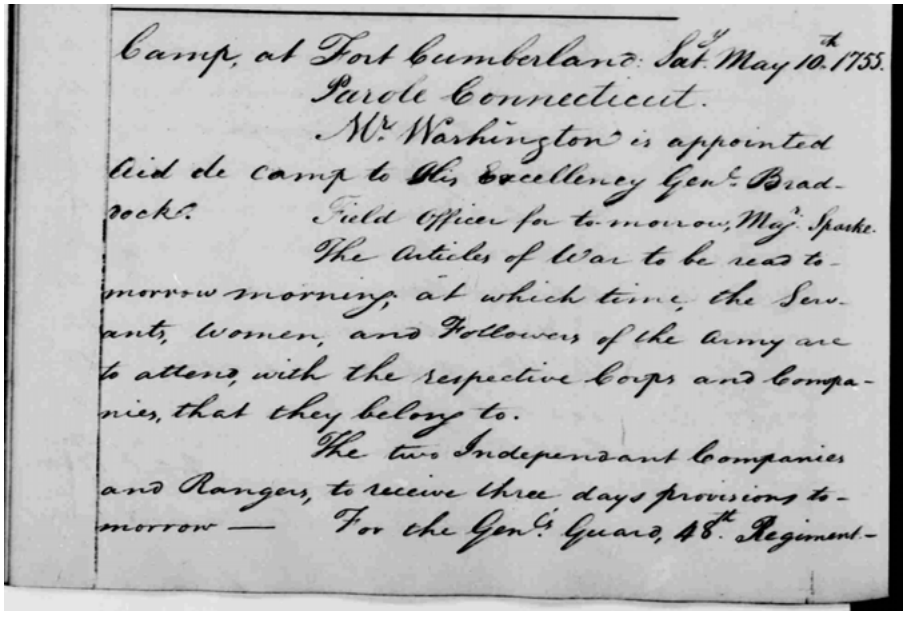
Camp, at Fort Cumberland: Sat.y May 10.th 1755.
Parole Connecticut.
Mr. Washington is appointed
Aid de camp to His Excellency Genl. Brad-
dock.
Field Officer for to-morrow, Majr. Sparke.
The Articles of War to be read to-
morrow morning, at which time, the Serv-
ants, Women, and Followers of the Army are
to attend, with the respective Corps and Compa-
nies, that they belong to.
The two Independant Companies
and Rangers, to receive three days provisions to-
morrow — For the Genl. Guard, 48th. Regiment-

Figure 2.4. An example image from the Washington data set (section 2.4).

described in section 3.2.1). This is a fairly small database, but often used due to its detailed annotation. In total, 4894 word images from 656 lines of text, containing 1471 unique words.

The original collection can be found on the website of the United States Library of Congress[3]. As an example, a cropped image from letterbook 1, page 132, is shown in figure 2.4.

[3]https://memory.loc.gov/ammem/gwhtml/gwhome.html

3. Text recognition

A common way of doing image analysis , and the methodology taught in many basic courses, is to break down your task into independent steps in a pipe-line. The first box, as illustrated in figure 3.1, would typically be image acquisition followed by some pre-processing (e.g. histogram equalization), binarization (if applicable), segmenting out the relevant parts (e.g. connected components, watersheds), measuring the relevant parts (e.g. area or texture features) and, finally, draw some conclusions from whatever was quantified. It can be seen as a divide and conquering approach to image processing, breaking down the image until the smallest interesting parts of information are identified.

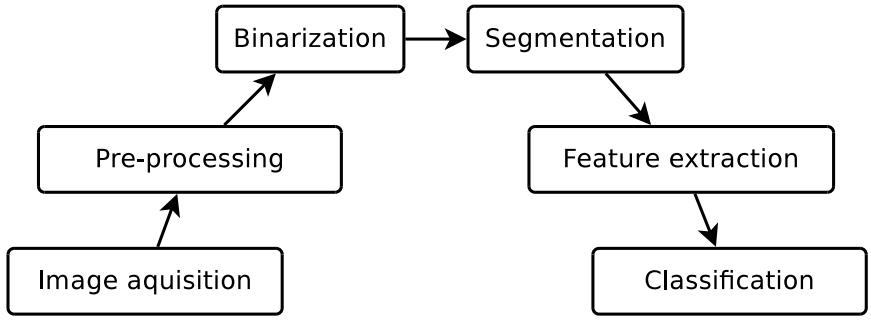

Figure 3.1. A classical image analysis pipe-line for some classification problem. This set up has, in part, been used for the work described in this section.

In the earlier part of the project, a clear pipe-line was the preferred way of viewing text recognition. Work was put in to designing a top-down approach where the manuscript page was broken down into smaller fragments (page and line segmentation) on which some handwritten text recognition technique could be applied. This can be viewed in contrast to the latter work presented in the dissertation, focusing more on machine learning techniques with a more integrated alternative to the pipe-line.

In the following chapter, the work on the top-down approach to text recognition and some context to it will be described. An exhaustive description of the context falls outside the scope of this dissertation, though some perspective on current trends will be included[1].

[1]During the time I have been working on this project, Recurrent Neural Networks (RNN) have proven themselves to be a much better approach to text recognition, than what is presented here. In 2011–2013, this was not the consensus yet. I have heard it described as ”this is not evolution,

3.1 Segmentation

Segmentation is, in short, the art of dividing and conquering the image. Regions of interest are identified, successively breaking down the data into separable (though often overlapping) objects.

3.1.1 Pre-processing

Both the line segmentation and feature evaluation (papers I-V) require binarization. This is a common, but often unnecessary, design choice. A binarization is a two class classification problem, where each pixel is set as foreground or background. If binarization can be performed, the following segmentation steps are considerably simplified. However, on larger materials, binarization can be a very hard problem due to the variability of the documents. This insight led to our latter work of trying to eliminate this step so that larger data sets would be easier to handle.

Many feature extraction methods are based on the assumption that the image can be binarized sufficiently well. This poses obvious problems when working with manuscripts that are almost impossible to binarize. The state-of-the-art requiring binarization, can not even be used as a meaningful comparison without a lot of work on the binarization itself, a task that is not necessarily at the core of the project at hand[2]. In paper VII for example, significant work was put into adapting the binarization proposed by Shafait et al. (2008) to SDHK, since we used quill features (section 4.1.1) as a comparison to the proposed method.

The result of pixel or patch based binarization methods are often broken connected components, due to the focus on only using local information. For the data sets C61 and C64, Otsu's method, by Otsu (1975), with some extra clean-up using mathematical morphology (with a structuring element looking like a quill pen tip) was sufficient. For better performance (according to the metrics of the DIBCO competitions for printed and handwritten text, presented by Pratikakis et al. (2013) and Ntirogiannis et al. (2014)), something with more of a model in the binarization can be used, such as the work by Howe (2012) on graph cuts for finding accurate binary objects or the method proposed by Shafait et al. (2008) found in the ocropus library[3].

---

but a revolution" and a revolution is indeed what has taken place during the time of my PhD-studies. RNN, well described for text recognition by Graves and Schmidhuber (2009) and for historical text by Frinken et al. (2012), is an approach where a network is given memory by recursively feeding the output back to the network. In parallel to its own output, a sequence of image data containing the text line is fed to the network sequentially as the primary input. At the time of writing, this approach wins all competitions.

[2] In my opinion, binarization has its place but as a research community, we must move away from it, except in very special cases. I am glad that this, from what I hear at conferences, is not a controversial opinion any more.

[3] https://github.com/tmbdev/ocropy

3.1.2 Line segmentation

In paper II, we proposed a line segmentation method based on multiple steps of refinement. Popular methods for text recognition often process the text on a page as a sequence of concatenated lines. This is due to the sequential nature of written text. In many ways the problem is processed very similar to that of a speech recognition problem, a different but overlapping scientific community, e.g. see Jurafsky and Martin (2000). For this sequential processing to be possible, the text lines need to be segmented so the page can be cut into strips. Not all text recognition or word spotting is performed this way (e.g. see work by Rusiñol et al. (2015) for descriptor based searching and by Wilkinson and Brun (2015) for deep learning based object detection). Great overviews of the many different techniques for segmenting handwriting can be found in the contest reports by Gatos et al. (2011) and by Gatos et al. (2010).

The method presented in paper II was actually implemented and tested for the data in paper I but it was only mentioned briefly. Because of the feedback given at the Historical Image Processing workshop in Beijing in 2011, the line segmentation method was evaluated more thoroughly and published separately the following year. For evaluating the method, 10 pages from C61 (section 2.2) and 10 pages from C64 (section 2.1) were segmented by hand[4].

In short the algorithm works as follows:

1. **Binarize the image** using the algorithm of your choice. In papers I–II, Otsu's methods was used followed by a morphology step for cleaning up the results.
2. **Estimate the text height** using connected components. This was defined as the median height of all bounding boxes of the connected components. In this way, under-segmented and over-segmented connected components were not allowed to dominate the estimate.
3. **Calculate the horizontal projection profile** on the middle part of a threefold vertical split of the page.
4. **Smooth the profile** by convolving it using a Gaussian filter with the standard deviation set to the text height (from step 2).
5. **Identify maxima** in the projection profile, this step is more robust after preceding smoothing. The number of maxima is used as an estimate of the number of lines on the page.
6. **Calculate more projection profiles** with 15 vertical splits, to get more detail. The previous projection profile was performed on a larger area for a more robust estimate.
7. **Smooth the profiles** from the 15 splits by convolving the profiles using a Gaussian filter as with the wider profile from step 4.

[4] Sadly, due to unclear image rights, we have not had the opportunity to release this these images in full.

8. **Grow text line estimates** based in the positions of the maxima in the 15 projection profiles, with their starting points at the maxima of the wider profile. The estimates are grown from the middle towards the edges by associating the maxima from adjacent splits with the lowest vertical position difference. The lines grown using the maxima mark the positions of text lines. Correspondingly, the minima between the line estimates make up a piece-wise linear cut for creating cropped text line images.
9. **Create a cost map** as described by equation 3.1, below.
10. **Create final line cuts** by finding lowest cost paths between the left and right page edges in the areas between the estimated text lines. The shortest path was found using Dijkstra's algorithm, allowing for very flexible estimates.

The initial binarization of the image was performed using Otsu's method. This method has the useful feature of being without parameters, but is far from the only algorithm for binarizing old handwritten data. Other techniques like Howe (2012), with clustering improvements by Ayyalasomayajula and Brun (2014), use much more domain specific knowledge, hence the algorithms are better for handling domain specific noise (e.g. page bleed-through). The images of both C61 and C64 (our evaluation data sources) have good contrast and high resolution, making Otsu's method good enough for paper II.

Projection based methods are fairly common for finding objects in many domains, likely because of the very low demands on computational power making the methods usable very early in the life of the field. Some papers, like those by Pal and Datta (2003) and by Arivazhagan et al. (2007), propose a methodology where semi-connected binary objects are searched for using summing over scan lines, both similar to the method presented here.

The graph is created with each pixel as an 8-connected node. The final weight map W is given by equation 3.1, where pixel positions are indexed by the coordinate pair (i, j). In equation 3.1, each pixel has been given an exclusive membership in the foreground or background set (i.e. binarization), called F and B respectively. A distance transform is run on both foreground and background, resulting in the distance maps D^{fg} and D^{bf}.

$$W_{ij} = \begin{cases} 2D^{\text{fg}}_{ij}, & (i,j) \in F \\ 1 - \frac{D^{\text{bg}}_{ij}}{10}, & (i,j) \in B \wedge D^{\text{bg}}_{ij} < 10 \\ 0.01, & (i,j) \in B \wedge D^{\text{bg}}_{ij} \geq 10 \end{cases} \tag{3.1}$$

The initial estimation, the piece-wise linear estimates using projections of the 15 splits, can give a very good estimate of the positions of the text lines. However, for the types of feature extraction used in papers I and III-V, clear cut lines where each pixel can be said to belong to only one line are needed.

This is where graph optimization comes in. Such an optimization would not be needed if connected components did not reach over several lines. In C64, connected components, even with the best of binarization algorithms, can overlap more than half of the page (vertically). Equation 3.1 is constructed as to give the cut between lines some specific characteristics. It is assumed that the connected components spanning between two adjacent text lines are of either, touching or overlapping ink strokes. Overlapping strokes are harder to separate due to the large overlaps, these would need a character model for better separation. Touching ink strokes have smaller regions of overlap, making a separation possible by finding the touching overlaps and cutting the shortest path though the connected component. The lowest cost path is steered away from characters to create a "buffer zone" for errors in the binarization (indifferent of if the errors are from algorithm weaknesses or degradation). Where there is overlap, the lowest cost path cuts through as few pixels as possible while still trying to cut close to the projecting minima (i.e. between lines). An example of the processing steps can be found in figure 3.2.

The work proposed by Avidan and Shamir (2007) on seam carving for content aware image resizing was adapted for cutting apart text lines by Asi et al. (2011) and by Saabni and El-Sana (2011). This methods was used as a comparison to our own work in paper II. Since seam carving only defines one pixel per pixel column for a separation set of pixels, the method in paper II worked better on our sources, since cuts using a shortest path can be made around interlocking shapes. Another interesting way of doing graph based separation is the approach proposed in Fernández-Mota et al. (2014), where the background is skeletonized and the shortest path though this skeleton is used as the cut between lines. For a more complete survey, in the papers by Likforman-Sulem et al. (2007) and by Razaka et al. (2008) two groups have compiled good overviews (up to the years of publishing).

3.2 Sequential features

Since text is a sequence of characters, text recognition is often set up as a sequence learning problem. The text lines are segmented and reassembled as a long line of concatenated text images, along which, some statistics are extracted that form a feature sequence. This feature sequence correspond to the appearance of the original text image[5]. Sequences of data are something that has been studied for a long time in fields spanning from speech recognition to

[5]This is not always the case and there are for example some approaches that directly do word spotting for whole words, as in the work by Kovalchuk et al. (2014) on creating an embedding space for comparison and going for a neural network approach as by Wilkinson and Brun (2016). At the time of writing, these kinds of approaches are the most interesting. This opinion is supported by that more than half of the papers at the international conference for frontiers in handwriting recognition (ICFHR) in 2016 had the word "deep" in the title.

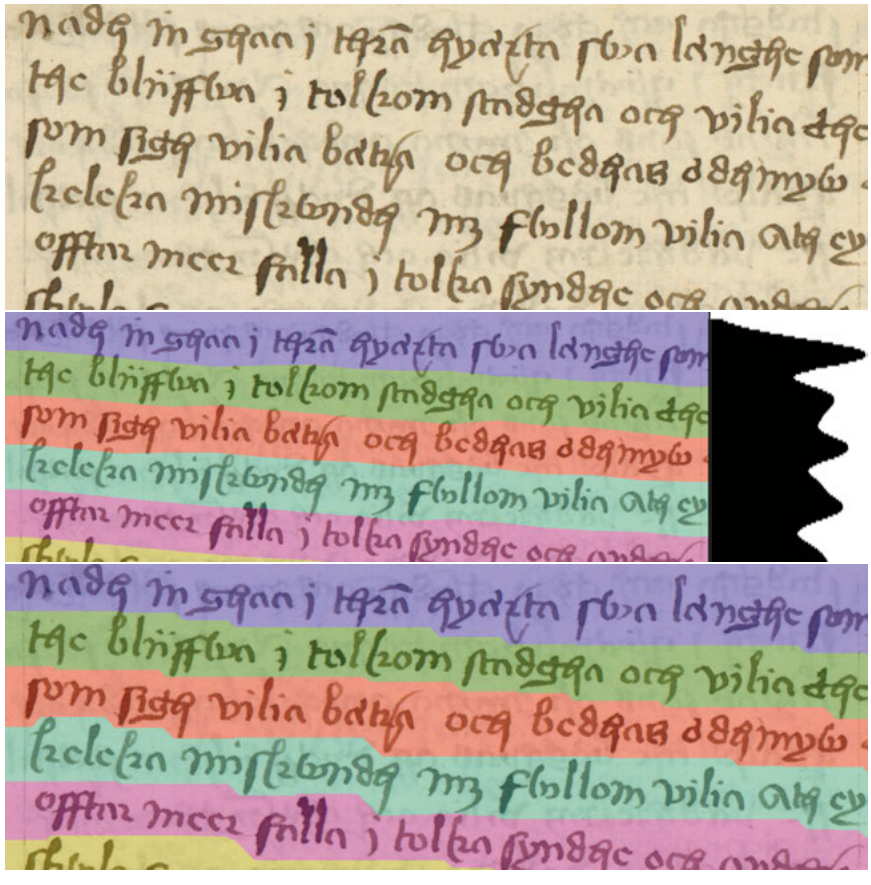

Figure 3.2. An example of the progression through the segmentation refinements from section 3.1.2. From the top, the first image is the original, followed by the first projection step and the refinement from growing the lines using a more detailed projection and finally the result of the graph based refinement step.

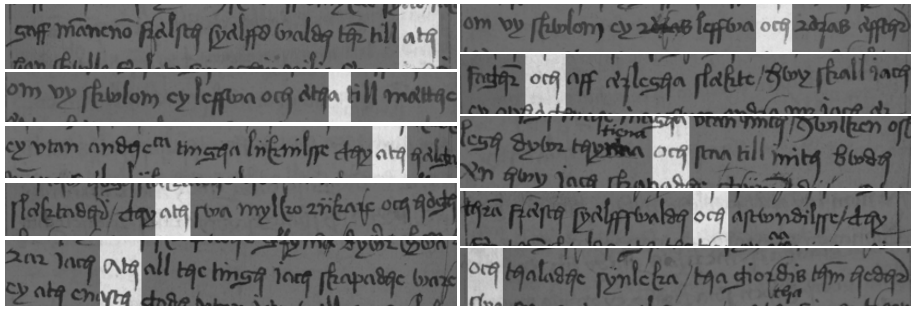

Figure 3.3. Some results from paper V on query-by-example word spotting. In word spotting, single (or short sequences of) words are sought for. The data material here is the manuscript C61 (section 2.2). The left column shows hits for the word “att” (Swedish for “to”, as in to do something) and the right column shows the word “och” (Swedish for “and”).

bioinformatics, in addition to the recognition of writing. There exists a multitude of methods for comparing such data, e.g. see the survey by Trentin and Gori (2001). For less learning based approaches (as those described here), a distance metric is defined between individual feature vectors, as opposed to neural network based methods where the metric is learned in full. This distance metric allows for a new metric on a meta level, comparing sequences of feature vectors. When a sequence is cut out , corrsponding to some part of the text line, a template matching scheme can be used for comparing it to every other position along the text lines. Likely matches are at positions where the distance is low.

The way of finding words that is presented here is called word spotting. Since, full recognition of historical text has proven itself to be a very hard problem, the research challenges have been split up into full text recognition and searching large document collections with high accuracy, i.e. spotting interesting words. An example of the results of word spotting can be seen in figure 3.3, showing results from paper V. Word spotting is commonly divided into query-by-example, where an example of the search query must be selected from the manuscript, and query-by-string, where models for the characters are learned and used for searching. For a query-by-string approach, a model that connects the individual characters and a text image must be learned. This is often a more complex procedure than concatenating some example images of characters and then running the word spotting in a query-by-example manner. It is rarely enough to learn from single instances of characters. For good results, the expected variation of each character needs to be present in the training data.

3.2.1 Features for text recognition

The most common set of text recognition features are the so called Marti-Bunke features, proposed by Marti and Bunke (2001). These are a set of statistics on the shape of the foreground pixels in a sequence of pixel columns. This is the set used in papers I-V. A so-called sliding window is run along the text line and a feature vector is created from each small window (i.e. neighbourhood) at each pixel column. Some of the features are defined in relation to a lower and upper baseline. These baselines are the common text help lines, the standard baseline and an imaginary line limiting the height of the lower case characters. The features are the following:

Projection profile The vertical projection of the pixel column (i.e. the sum of foreground pixels in each column).

Upper contour The position of the foreground pixel with the highest vertical position.

Lower contour The position of the foreground pixel with the lowest vertical position.

Upper projection The projection profile above the upper baseline (i.e. projections of ascenders).

Lower projection The projection profile below the lower baseline (i.e. projections of descenders).

Centre of Weight The mean vertical position of all foreground pixels in a column.

Foreground/background transitions The number of transitions between the foreground and the background in the pixel column.

Second moment The variance of the vertical positions of the foreground pixels in a column.

Gradient of the upper contour First derivative of the upper contour curve. This calculation needs a window size of more than one column.

Gradient of the lower contour First derivative of the lower contour curve. This calculation needs a window size of more than one column.

Foreground fraction The ratio of pixels belonging to the foreground, between the upper and lower contours.

In figure 3.4, an example of feature extraction is shown (using the upper and lower contour features). As stated in their original paper, these features are not very robust against varying slant, skew, character size or non-smooth baselines. This is mitigated by a pre-processing scheme, resulting in normalized text line images. This normalization is in short, estimation of the slant, estimation of the lower and upper lower baselines and, finally, warping the whole word image to compensate for these estimated variations. The result is a binarized image of a text line (or single word) where the base line is at a set height in the image. More importantly, the characters have the same

Figure 3.4. An example of feature extraction for text recognition on two instances of the words “corpus christi” from C64 (section 2.1). The upper most images are cropped from the original source, the middle images are the binarization and the bottom images are the contour features. In pixel columns where there are no contours, the contour line was interpolated linearly, as to not break the sequence.

size in all images. The slant correction can sometimes make ascenders slant ”backwards”, though the end result is still a better recognition accuracy. This normalization is already applied to the George Washington data set (section 2.4), hence, it is used in papers III and IV.

The Marti-Bunke feature set is used extensively in the literature. A lot of research has gone into finding other types of features. Some of the more interesting are using histograms over a codebook of descriptors, proposed by Rothacker et al. (2013), a more learning based approach like those by Doetsch et al. (2014) and by Voigtlaender et al. (2016) or going for a HMM and neural network hybrid as in Espana-Boquera et al. (2011). An evaluation for features can be found in the work by Ayyalasomayajula et al. (2016). The biologically inspired features by van der Zant et al. (2008) was another very interesting research direction before feature learning using deep learning started to completely dominate handwritten text recognition.

3.2.2 Relative importance of feature types

The Marti-Bunke features have been a very common choice of text recognition features. In paper III, the relative importance of the individual Marti-Bunke features were examined. By setting up a word spotting evaluation framework, using the George Washington database (section 2.4). By implementing the word-spotting approach by Manmatha and Croft (1997) and developed further by Rath and Manmatha (2003b) and by Rath and Manmatha (2003a), the features could be re-weighted and a word spotting performance number associated with every point in the weight space. This was done in order to better understand which features contributed more to the outcome.

In the George Washington database, 4894 cropped words belonging to 1471 word classes are given as normalized binary images. For every word, the 11 features were extracted. These feature sequences were weighted, given a weight vector supplied for suppressing or reweighing single feature dimensions. Each word pair was compared using dynamic time warping (described below in section 3.3.1), giving a single number for the pair-wise difference. A quadratic and symmetric matrix, with a side of 4894, was set up to contain all the pair-wise differences between word images. Dynamic time warping (DTW) is a way of identifying the differences between sequences, given some feature vector distance metric, without any learning. Hence, we thought the effects of this evaluation would be as "raw" as possible, without any learning hiding the effects of the feature weighting.

The true positive rate (TPR), true positives from the classification as proportion of ground truth positives, and false positive rate (FPR), false positives from the classification as proportion of ground truth negatives, were calculated by classifying the cells in the word difference matrix using thresholding. By successively relaxing the threshold, for each level (all possible threshold levels were used) a TPR-FPR value pair was calculated. These value pairs were ordered according to the threshold level and looked like the curves in figure 3.5. For each weight vector, the area under the curve (AUC), based on receiver operating characteristic, was calculated as a single number performance metric. For a more thorough overview of AUC and its uses, see Bradley (1997). In the figure, the red area under the curve is the AUC, i.e. AUC is the red area as proportion of the unit square. The word spotting, given the database and other design choices, can now be seen as a function $f_{\text{wordspotting}} : [0,1]^{11} \subset \mathbb{R}^{11} \rightarrow [0,1] \subset \mathbb{R}$, mapping every possible weight vector to an AUC.

By choosing a weight vector with only 0 and 1 as values, the performance from fully removing features can be evaluated. This is shown in table 3.1, where results using only single features and leave-one-out validation are presented. When the weights are seen as a parameter space, with restrictions on the weight vector $\mathbf{w}$ as $\mathbf{w} \in (0,1)^{11} \subset \mathbb{R}^{11}$, optimization can be used to to maximize performance. This space was piecewise flat, due to the evaluation depending on thresholding. Small areas within the weight space gave constant performance, making gradient based optimization non-applicable. Hence, we could use a sampling coordinated descent approach to maximize the word spotting performance. Some results of the optimization are shown in table 3.2. Several runs were performed but they all looked approximately like those presented in the table. Hence, it looks like the projection based features were the most useful for finding word matches.

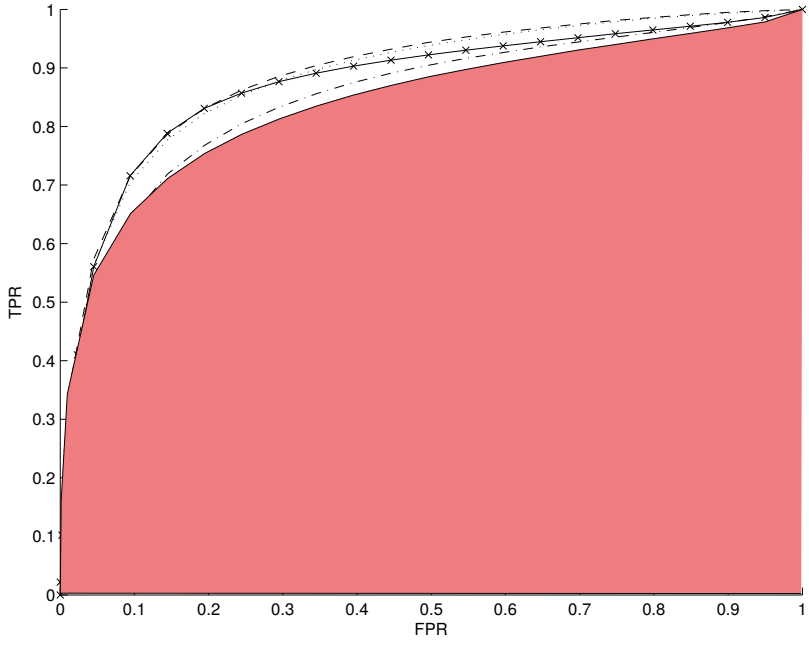

Figure 3.5. In sections 3.2.2 and 3.2.3, the word spotting evaluation is quantified using the area under the curve (AUC). By thresholding the pair-wise word differences, a classification was made giving a true positive ratio (TPR) false positive ratio (FPR). Successively relaxing the threshold creates curves with TPR on the y-axis and FPR on the x-axis. AUC is shown as a red area under the lowest of the performance curves in the plot. The other curves are examples of improved performance when using optimization for finding a reweighing of the features.

Feature	Single	Leave one out	Leave one out (without "lower gradient")
Projection	**0.877**	0.839	**0.858**
Upper contour	**0.853**	0.840	**0.859**
Lower contour	0.788	0.842	**0.861**
Upper projection	**0.871**	0.840	**0.856**
Lower projection	0.648	0.836	**0.856**
Centre of weight	0.784	0.842	**0.861**
Background/Foreground transi.	0.802	**0.843**	**0.863**
Second moment	0.792	**0.843**	**0.862**
Upper gradient	0.794	0.841	**0.867**
Lower gradient	0.796	**0.861**	-
Foreground fraction	0.827	0.826	**0.843**

Table 3.1. *Area under the curve (AUC) performance for word spotting on the George Washington Letters dataset (section 2.4), for evaluating feature importance. By only trying out one feature at a time, their respective importance for discriminating word images can be evaluated. The effect of leaving one feature out from the full set was also investigated by leave-one-out (LOO) validation. The right-most column shows LOO after "lower gradient" feature was removed, the lowest scoring feature from the middle column. Results that were better than the base line of **0.8418** are in bold font. The full set of words for the evaluation contained 4894 elements.*

	Marti-Bunke features (ordered as the list in section 3.2.1)											AUC
Weights	0.6	0	0	1	1	0.4	0	0	0.2	0	0.3	0.899
	0.9	0	0	0.9	1	0	0	0	0.2	0	0.3	0.900
	0.8	0	0.4	0.9	1	0	0	0	0.2	0	0.3	0.899

Table 3.2. *Weight vectors found by coordinate descent optimization. From these weight vectors, the features that were most important were the three projection profiles.*

3.2.3 Feature filtering

When working with noisy images, a very common first step is to apply some filtering to remove noise. In paper IV, we expanded on the Marti-Bunke feature extraction approach by adding filtering to the sequence of feature vectors, instead of applying them on the images. By adapting four types of filtering (Gaussian, bi-lateral, median and non-local means) to the feature vector sequences, the performance in terms of word spotting could be increased. The evaluation framework presented in the last section was used for evaluating the effects of the different filters. The full set of 11 features were reduced to a subset of 4 for this study. The features for paper IV were "Projection profile", "Upper contour", "Lower contour" and "Foreground/background transitions", as recommended by Rath and Manmatha (2003a).

The Gaussian filter

A Gaussian filter is applied as a convolution on the data domain. It is a weighted average of neighbouring data points, where the weights are taken from a Gaussian kernel. Assuming neighbouring data points are similar and that the noise is independent for each data point, the noise cancels out in the resulting weighted average. However, this leads to smoothing of sharp differences (e.g. at edges in an image). Because of this, the Gaussian filter is often called Gaussian smoothing. When the Gaussian filter is applied to gray scale images, all pixel values in some neighbourhood are considered in the convolution. For colour images, the colour channels are treated as independent. For the feature sequences in paper IV, the filtering was performed per dimension. The Gaussian filter have many applications for sequences in signal processing, e.g. see Blinchikoff and Zverev (1976), very similar to what is considered here. For the feature vector sequence $\mathbf{D}_i$ indexed by position i, the filtered sequence $\hat{\mathbf{D}}_i$ is given by:

$$\hat{\mathbf{D}}_i = \frac{1}{Z} \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \mathbf{D}_{(i+j)} \exp\left(-\frac{j^2}{2\sigma^2}\right) \tag{3.2}$$

where the normalization constant Z is given by:

$$Z = \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \exp\left(-\frac{j^2}{2\sigma^2}\right) \tag{3.3}$$

The width of the filter, N, gives the size of the local neighbourhood (in the unit adjacent feature vectors) for each weighted average. The σ parameter is the width of the Gaussian kernel, preferably much smaller than N (e.g. $4\sigma \leq N$) for the convolution kernel to have Gaussian characteristics (otherwise it will be cut off before reaching the tails of the distribution). Selected evaluation results are shown in table 3.3.

Filter type	σ	AUC
Gaussian	0.34	**0.853**
Gaussian	2	**0.876**
Gaussian	8	0.849

Table 3.3. *Some word spotting results for the Gaussian filter (section 3.2.3). Numbers in bold font indicates performance above the baseline of 0.852.*

The bilateral filter

The Gaussian filter assumes that adjacent data points are similar, creating problems along image object edges and other sharp steps in the data. The bilateral filter proposed by Tomasi and Manduchi (1998), mitigates this by adding a data similarity component to the weighted average. Since this new component depends on the data, the operation can no longer be implemented simply as a convolution, increasing the computational complexity in comparison to the Gaussian filter. However, the performance along edges is considerably improved by only applying the smoothing on selected parts of a local neighbourhood. The similarity is not defined per dimension in this implementation. For the feature vector sequence $\mathbf{D}_i$ indexed by position i, the filtered sequence $\hat{\mathbf{D}}_i$ is given by:

$$\hat{\mathbf{D}}_i = \frac{1}{Z_i} \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \mathbf{D}_{(i+j)} \exp\left(-\frac{j^2}{2\sigma_s^2} - \frac{\left\| \mathbf{D}_i - \mathbf{D}_{(i+j)} \right\|^2}{2\sigma_d^2} \right) \tag{3.4}$$

where the normalization constant Z_i depends on the data and is given by:

$$Z_i = \sum_{j=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} \exp\left(-\frac{j^2}{2\sigma_s^2} - \frac{\left\| \mathbf{D}_i - \mathbf{D}_{(i+j)} \right\|^2}{2\sigma_d^2} \right) \tag{3.5}$$

The exponential function now includes two terms. In the first, the width parameter σ_s is analogue to the σ in the above definition of a Gaussian filter. The second term is the sensitivity to similarity between the data point being processed and its neighbours. The parameter σ_d is the sensitivity to this data similarity. Selected results are shown in table 3.4.

Filter type	σ_s	σ_d	AUC
Bilateral	2	4	**0.876**
Bilateral	8	0.4	**0.861**
Bilateral	8	4	0.852

Table 3.4. *Selected evaluation results for the bilateral filter (section 3.2.3). The parameter σ_s is the spatial standard deviation (in number of sequence positions) and σ_d the standard deviation for the data similarity. Numbers in bold font indicates performance above the baseline of 0.852.*

The non-local means filter

The non-local means filter was proposed by Buades et al. (05) and takes the bilateral filter one step further. By removing the part of the weight function that defines locality, the filtering at every point now depends on every other point in the data. Hence, the computational complexity becomes quadratic. In some implementations, a window is defined for the similarity comparison to limit the data sequence, e.g. to 10% of the full data. An interpretation of this type of filtering, is that the variation in the data is reduced by making the structure of the variations more homogeneous. In this way, similar areas can now be brought closer to each other in terms of feature distance. For the feature vector sequence $\mathbf{D}_i$ indexed by position i, the filtered sequence $\hat{\mathbf{D}}_i$ is given by:

$$\hat{\mathbf{D}}_i = \frac{1}{Z_i} \sum_{\mathbf{d} \in \mathbf{D}} \mathbf{d} \exp\left(-\frac{\|\mathbf{D}_i - \mathbf{d}\|^2}{2\sigma_d^2} \right) \tag{3.6}$$

where the normalization constant Z_i depends on the data and is given by:

$$Z_i = \sum_{\mathbf{d} \in \mathbf{D}} \exp\left(-\frac{\|\mathbf{D}_i - \mathbf{d}\|^2}{2\sigma_d^2} \right) \tag{3.7}$$

Note that $\mathbf{d} \in \mathbf{D}$ indicates summation over all vectors in the sequence. Also, while the sequences for the other filters are defined per word, the non-local means filter uses all feature vectors from all words for the similarity comparison. The σ_d parameter is the standard deviation in the feature space and represents the sensitivity to similarity.

The similarity function can be defined between single vectors, but also between several concatenated vectors. Inspired by the work of Tibell et al. (2009), neighbouring feature vectors in a sequence can be used together in the similarity calculation. Considering a small neighbourhood of feature vectors proved to be a good strategy, but the effect of tuning its size was dwarfed by tuning the σ_d parameter. Selected results are shown in table 3.5.

Filter type	Neighbourhood	σ_d	AUC
Non-local means	3	4	**0.913**
Non-local means	1	4	**0.903**
Non-local means	3	0.05	0.847

Table 3.5. *Selected evaluation results for the non-local means filter (section 3.2.3). Numbers in bold font indicates performance above the baseline of 0.852.*

The median filter

In image processing, the median filter can be used for removing random spikes in the data (also called "salt and pepper" noise). The advantages to this type of filtering is that edges are preserved at the expense of also removing sudden spikes in the signal. A median operation on scalar values is a very simple

operation, while the same can not be said for the median of vectors, that can be defined in several ways. We followed the approach by Astola et al. (1990) and Astola et al. (1988) for median filters for multi channel signals.

Depending on application, the l_1 or l_2 norm can be used for similarity between vectors. We implemented both, along with a dimension wise median filter (i.e. the standard scalar median filter run on a sliding window for each data dimension of the sequence). For the feature vector sequence $\mathbf{D}$, a replacement vector index j for each processed index i is sought for. The replacement, creating $\hat{\mathbf{D}}_i$, is defined as:

$$\hat{\mathbf{D}}_i = \mathbf{D}_j \tag{3.8}$$

where the replacement index j, in some index set X_i depending on i and the neighbourhood size N such that $X_i = \{x | x \in \mathbb{N} : i - \lfloor N/2 \rfloor \leq x \leq i + \lfloor N/2 \rfloor\}$, is given by:

$$\underset{j \in X_i}{\operatorname{argmin}} \sum_{k \in X_i} \|\mathbf{D}_j - \mathbf{D}_k\| \tag{3.9}$$

where $i, j, k \in \mathbb{N}$ and are all valid sequence indices. A more intuitive explanation to this procedure is that the middle-most vector, defined by some metric, from a local neighbourhood replaces the vector at the index being processed. Some selected results are shown in table 3.6.

Filter type	Width	AUC
Dimension wise mean	7	**0.875**
Dimension wise median	5	**0.859**
l_1 vector median	3	0.8497
l_2 vector median	3	0.8497

Table 3.6. *Selected evaluation results for the median filter (section 3.2.3). Numbers in bold font indicates performance above the baseline of 0.852.*

In figure 3.6, the effects of filtering is shown on a single dimension of a feature signal, as an illustration. The difference in performance between the Gaussian and the bi-lateral filter, tables 3.3 and 3.4, are not clear. There might be some improvement but it is hard to say from this evaluation. For the median filter, the dimension-wise filtering performed much better than the vector based one. This might be because spiking noise in the feature signal occurs in one dimension at a time and are obscured when the full feature set (i.e. the whole vector) is used for calculating differences. On the other hand, is the filter can not pick up the difference, the word spotting comparison algorithm might not either. The non-local means filter looks like a whole new signal in the illustration in figure 3.6, however, the increased word spotting performance, shown in table 3.5, is clear compared to every other filter configuration.

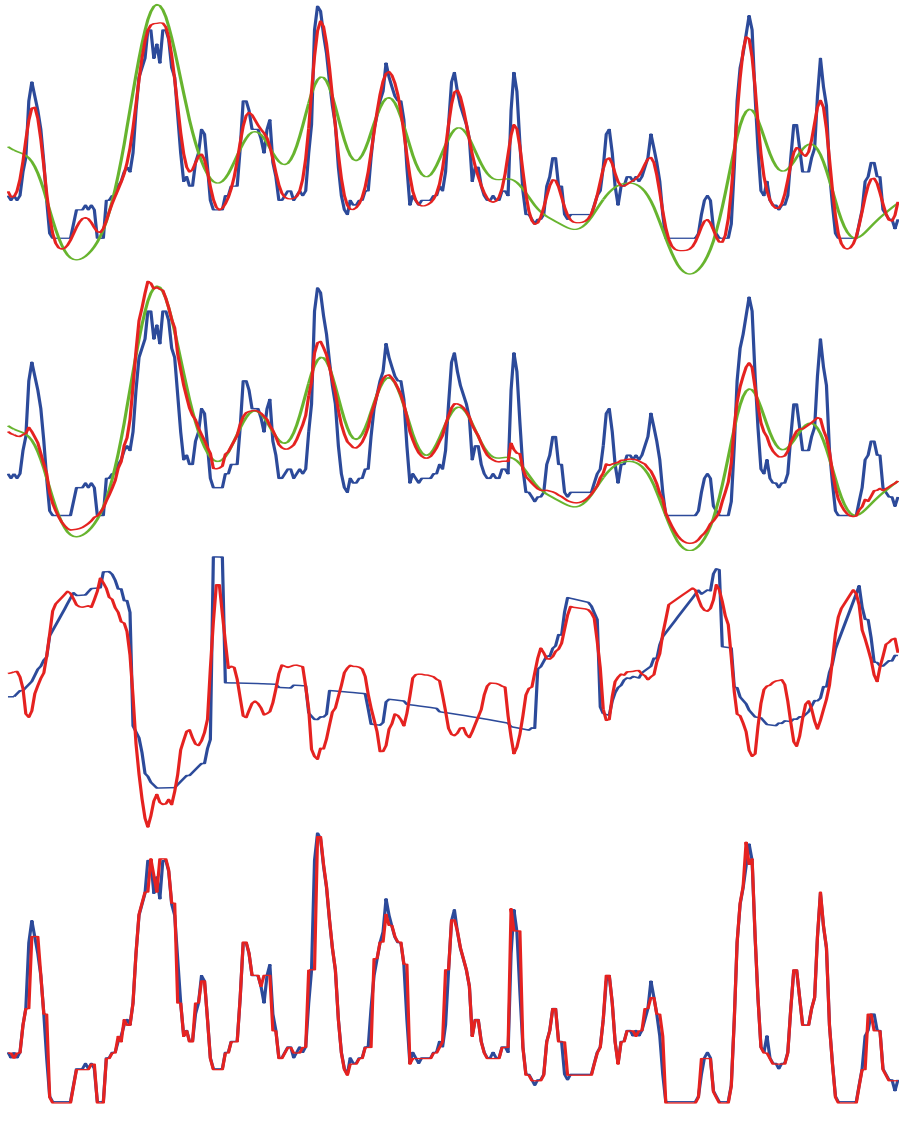

Figure 3.6. Examples of the original signal and filtered results, with varying parameters, on small cut outs of one dimension of the feature sequences (section 3.2.3). **Top:** the Gaussian filter, **Upper middle:** the bi-lateral filter, **Lower middle:** the non-local means filter, **Bottom:** the vector median filter. The original signal is in blue, while the other colours are filtering results with varying parameters. Note how the Gaussian and bi-lateral filter smooth the signal, while the median filter only cuts some extreme values. The red curve in the results from the non-local means filtering is very different from the blue curve (the original signal). This is due to that the filter uses the full feature sequence from all words to reshape the output. It looks a bit erratic but is a homogenization of the feature sequence.

3.3 Sequence analysis as text recognition

3.3.1 Dynamic time warping

When images of text is represented by sequences of feature vectors, a way to compare these sequences is needed to be able to compare the text content. The technique used in papers I–V for quantifying the similarities (or dissimilarities) was dynamic time warping (DTW). DTW is a way of finding the minimum number of changes that need to be performed in order to "warp" two sequences (possibly of different lengths) to be as similar as possible. Below, the needed distance metrics and allowed changes to sequences will be explained, but first, an introductory example.

To explain DTW, it is easier to start the description from its predecessor, the concept of Levenshtein distance, proposed by Levenshtein (1966). The Levenshtein distance is a distance metric between strings of text, more specifically, the minimum number of edit operations that are needed to transform some string A into some string B. As such, one of its uses is as a similarity metric for spell checkers. This quantified difference between strings is symmetric, i.e. every minimal cost change set $S_{A \to B}$ has a corresponding change set $S_{A \to B}$ with equal cost. The change set is found using dynamic programming, guaranteeing the minimal cost. The edit operations are often defined as: adding a character, removing a character, changing a character and swapping places of two subsequent characters. The necessary set of operations are only adding and removing characters. However, since swapping characters is such a common spelling mistake, it might be better to allow this as a separate low cost edit operation in a spell checker. More operations can be added if the application demands it. Also, each edit operation can be weighted, e.g. for allowing low cost changes for characters that are phonetically similar or where the keyboard keys are close together.

In figure 3.7, an example of the Levenshtein calculation is shown. The strings to be compared are first set up along the vertical and horizontal dimensions of a table, so that every character pair have a cell in the table. The default is that cells are filled with the number 1, in this simplified example. For all cells where rows and columns belong to matching characters, the cell is filled with a 0. This corresponds to the change cost, mismatch has a cost but matches do not. The next step is to calculate the accumulated cost at each cell. Each cell C of position i, j in the table is now used to calculated the accumulated cost in a new table A as $A_{i,j} = C_{i,j} + min(A_{i,j-1}, A_{i-i,j}, A_{i-1,j-1})$. The value of each cell in A, contains the minimum cost for the change set up to that point. This accumulated cost table is shown in the middle part of figure 3.7. The last part of the distance calculation is to find the minimum cost path through table A. The lowest cost will always be the number in the lowest and rightmost cell. Backtracking though A is performed by stepping from the last cell backwards. Each step is always to a cell will equal or lower cost than the current. Tracking

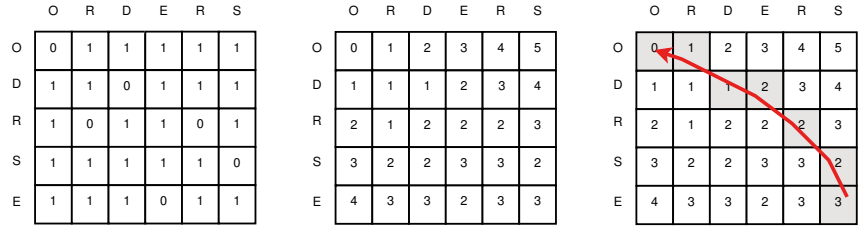

Figure 3.7. An example of Levenshtein distance (section 3.3.1), calculated between the words “orders“ and the misspelling ”odrse“. Through dynamic programming, the minimum cost of the change set needed to transform one string onto the other, can be guaranteed. The left-most table shows the element difference cost, the middle-most table show the accumulated cost and the right-most table shows the warping path. Note that there are several paths with equal cost. How this should be handled is a design choice that falls outside the method itself. However, a common way of reducing the number of alternative paths is to add more change operations and find better difference cost than only using 1 and 0, as in the left most table.

the lowest cost path back to the upper left corner gives the full path, with two corresponding change sets $S_{A\to B}$ and $S_{B\to A}$.

The Levenshtein distance metric between strings is extended in dynamic time warping (DTW) to discretely sampled continuous signals. The signals can have more than one dimension, as the sequences of feature vectors described above. Instead of working with discrete symbols, as in the case with text characters, feature vectors take their place in the tables from the Levenshtein example. The difference between feature vectors can be defined as binary within some ϵ, though it makes a bit more sense to use proper distance metrics (e.g. euclidean distance). In paper I, some variations to the standard euclidean distance were evaluated for a word spotting task. The evaluation there led to the squared euclidean distance being used in the subsequent papers.

In papers I-V, when sequences are of equal length, the DTW difference is defined as the accumulated squared euclidean distance at each sequence sample point. The allowed changes was skipping one sample from either sequence. This corresponds to allowing the first sequence to be ”stretched out” to match the second, and the second to be compressed into the first (assuming the first sequence is shorter than the second). In algorithm 1, the warping procedure is shown. Note that the algorithm does not return the warping change set (though this would be an easy extension), since it is not needed. Only the minimal cost

after warping is used to compare feature vector sequences corresponding to word images.

Algorithm 1: An algorithm for calculating the dynamic time warping difference between two sequences u and v, of lengths n and m, in some domain S given some distance function $d : S \times S \rightarrow \mathbb{R}$ (section 3.3.1). In papers I and III-V, squared euclidean distance was used. The set of changes needed for the warping could be obtained by backtracking in D, which is not needed for finding the DTW cost/distance.

Data: $u \in S^n, v \in S^m$
Result: $C \in \mathbb{R}$

$D \longleftarrow AllocateMatrix(n, m)$

$D[0,0] \longleftarrow 0$ `/* Set starting point to 0 */`

for $i \in \{x \in \mathbb{Z} : 0 \leq x < n\}$ **do**
  for $j \in \{x \in \mathbb{Z} : 0 \leq x < m\}$ **do**
    $D[i,j] := d(u[i], v[j]) + min(D[i-1,j],$ /* insertion/deletion */
    $D[i,j-1],$ /* deletion/insertion */
    $D[i-1,j-1])$ /* match */

$C \longleftarrow D[n-1, m-1]$

Dynamic Time Warping have been used extensively for handwriting recognition in the literature, see for example Kane et al. (2001), Rath and Manmatha (2003b) and Manmatha and Croft (1997)[6]. In these papers, segmented words are compared. In paper I, a way of running a fast word spotting on full lines were presented. To do this, a word that is selected as the search template is extended with dummy signal elements as the first and last sequence element. The cost for ”stretching” these are set to zero. Since the warping path though the cost matrix (the cost matrix is the name for the cost table of Levenshtein distance in DTW) is forced from the upper left corner to the lower right corner, the optimal warping path is allowed to ignore large parts of the sequence of a full text line. The weight calculations can also be reused to decrease the necessary calculations. In figure 3.8, an example of this is shown as the cost matrix in grey scale and the minimum cost path in colour. The properties of the DTW distance still hold and the matches, between a template and a line can be ranked according to the distance value.

The cost matrix is expensive to calculate in full. However, restricting the warping as to not allow matches between signal samples with a large time difference can be beneficial to the matching and reduces the number of calculations. This is achieved by limiting the distance of the warping paths from the diagonal of the cost matrix. The matrix is not calculated in full, only the

[6]There are many other uses for sequence alignment that I do not present here. A more unexpected application is aligning 3D images of colons for surgery preparation ”fly though”, proposed by Nain et al. (2002).

Figure 3.8. An example of a warping path "cutting" though a full text line, using dynamic time warping (section 3.3.1). The cost matrix, i.e. element wise differences between feature vectors, is shown in gray scale in the background with the warping path in colour.

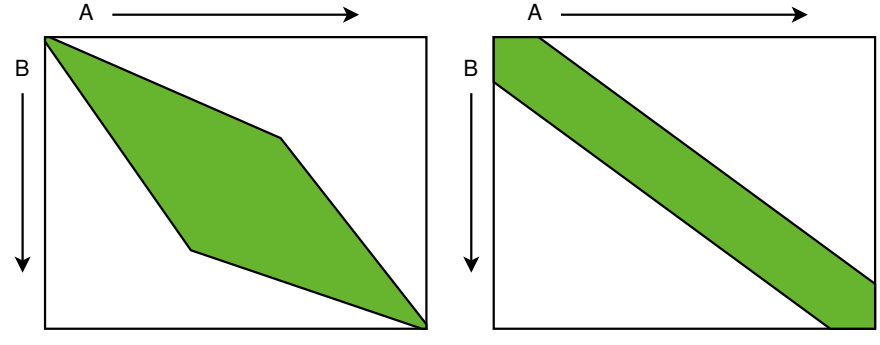

Figure 3.9. An illustration of the Sakoe-Chiba restrictions to the cost matrix using in dynamic time warping. To the right is the diagonal shape, allowing for a constant limit on the warping. To the left is the diamond shape, allowing for a higher warping in the middle of the sequences than in the beginning and end parts (section 3.3.1).

extended diagonal, as proposed by Sakoe and Chiba (1978). A variant of this restriction is the diamond shape, where the allowed warping is higher half way along the signals than in the beginning and end parts. Another is the diagonal shape, where warping is restricted by using a constant breadth of the extended diagonal. An illustration for which parts of the cost matrix that are calculated using the diagonal and diamond shapes are shown in figure 3.9. A lower cost match might be possible if a full warping would be allowed, the authors argue that this is a pathological warping and should not be allowed.

In the full line DTW word spotting of paper I, this restriction to the cost matrix can only be used in the beginning and end of the line. Hence, the savings from any restriction shape was very limited. In papers III and IV, however, the word wise matching approach proposed by Manmatha and Croft (1997), was used and the savings to computational power were considerable.

As is shown in algorithm 1, the implementation of DTW is straight forward and there is no need for a deeper understanding of the mathematics. As this is a great strength of this method, considering its usefulness, it is also its weak-

ness. No need for learning also means that there is no adaptation to the data, more than that design choice in the allowed set of changes and distance metric. Considerable work has been put into finding better distance metrics for word spotting application, for an overview see (Haji, 2012), though it falls short compared to more learning based approaches, as hidden Markov models (described below). However, for the applications presented in papers I, III and V, adaptation to the data was undesirable. Both the evaluation of the Marti-Bunke feature and filtering techniques demanded a more raw use of matching. Using a more learning based approach in these scenarios would make controlling which factors affected the end results much harder.

3.3.2 Hidden Markov models

A very popular technique for modelling sequences, in fields as diverse as speech recognition to genetics, is the Hidden Markov Model (HMM)[7]. The observed sequential data is modelled as if it was generated by some unobservable state machine with a data emitter attached to each state. The model parameters are probabilities for transitioning from each state to any other as a transition matrix, state initial probabilities and the data emitter distributions connected to each state (emitting observable data). The emission distribution can be of any form that fits the domain of the observed data (e.g. multinomial or Gaussian). Since the states can not be observed directly, there is no way of knowing, at any given point, from which state an observation is emitted. In fact, there is no limit to how similar emission distributions from different states can be. Sometimes it is even beneficial to tie them together, making two emission distributions have the same model parameters. Hence, two models that emit the same data sequences, can look very different in the hidden state space. For the many variations and implementation details, see the excellent tutorial by Rabiner (1989).

In the case of character recognition, it is common to choose a fixed topology before training the model, e.g. see the survey by Plötz and Fink (2009). In this way, some domain specific knowledge is encoded through model restrictions. In figure 3.10, an example of an HMM for a single character is shown. This is a so called character HMM, as in the work by Fischer et al. (2011) and Fischer et al. (2012). A character HMM is a way of modelling every single character using a common state space template. This template is then copied and each one trained to fit a specific character that is present in the training data set. The model shown in the figure is one instance of this template. The circles marked $S1..S7$ are the hidden states. From these, arrows go to other states,

[7] At the time papers I-V were written, HMMs for handwriting recognition were still considered a state-of-the-art method. Though I started working on writer identification techniques (next chapter) before getting around to publish on HMMs, I would still like to publish the 6 months worth of HMM code that lives in a dark corner of my hard-drive (assuming that the gods of funding are willing).

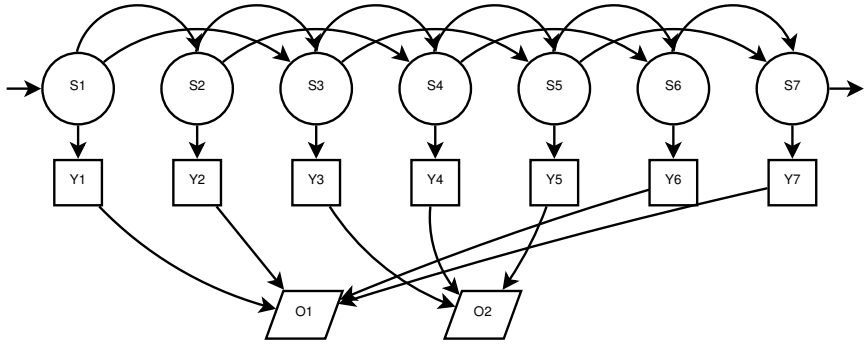

Figure 3.10. An illustration of a so called character Hidden Markov Model. The circles marked $S1..S7$ are the hidden states, the rectangles marked $Y1..Y7$ are emission distributions and the rhombi marked $O1, O2$ are the observations. In this example, the modelled character is an upper case "H". A sliding window over "H" only gives two types of pixel columns and is symmetric, leading to only two types of observations even though there are seven emission distributions. A more detailed description is found in section 3.3.2.

specifically the two following. Also, a self transition is allowed, i.e. staying in a state. This type of constraints, not allowing "backward" transitions, is called a Bakis topology. When the transition probabilities (for the allowed transitions) are learned from data, this topology allows for contracting the length of a likely path of state transitions between $S1$ and $S7$ from six to three (i.e. the state sequence $S1, S3, S5, S7$). In this way, longer and shorter characters can be modelled using the same template. The squares marked $Y1..Y7$ are the emission distributions. These might be Gaussian mixtures for some feature set (e.g. Marti-Bunke features as described above), multinomial distributions for some codebook or some other, more or less, exotic distribution. The observations, rhombi marked $O1, O2$, are in this example from a sliding window going over the character "H". Hence, the observed data is symmetric and with only two different outputs. This example shows how different the observations can be from the underlying state machine progression.

An HMM is usually trained using an algorithm called the Baum-Welch algorithm. This algorithm is an instance of expectation maximization (EM), but under another name since EM is a generalization of this algorithm and several others. As such, the emissions are locked while tuning the transitions, before locking the transitions and tuning the emissions, see Rabiner (1989). A great advantage over the DTW approach, described above, is that emissions, transitions and topology can be learned from data, giving flexibility within strong constraints.

When the HMM is trained, the most likely state sequence, given some observations, can be found using the Viterbi algorithm. This is how the most likely character sequence, given the data, is decoded. The Viterbi algorithm

is an instance of dynamic programming and can be used due to the Markov property of the state space, illustrated in equation 3.10.

$$p(s_n|s_{n-1}, s_{n-2}, \ldots, s_0) = p(s_n|s_{n-1}) \tag{3.10}$$

The Markov property, equation 3.10, states that the probability of being in any state, at position n in the data sequence, only depends on the preceding state. The HMM is described as "without memory", the future state progression only depends on the present. This makes decoding using dynamic programming possible, in this case the Viterbi algorithm, making decoding much easier and faster than would otherwise be possible. However, it also introduces a problem with an exponential decay in the probability of staying in any particular state. The probability of staying m time steps in one node with self transition probability $p(s_n|s_n) < 1$ is $p(s_n|s_n)^m$.

An alternative approach to fixing the topology is to put a prior on the transition matrix. This process, described by Huang et al. (2001), makes it possible to suppress some transitions instead making a binary choice on if a particular transition is allowed or not. However, there are some papers casting doubt on the importance of choosing a good topology. Work by Van Oosten and Schomaker (2014) show that in many instances, the topology does not matter much for the performance of the HMM model.

4. Identifying writer hands and production dates

The methodology of the work laid out in chapter 3 was based on a hierarchy of methods. For instance, page and line segmentation were prerequisites to the text recognition. In this chapter, methods will be described where information is extracted from a manuscript page as a whole. In particular, these methods will bypass line segmentation. These global methods are used for identifying writers and estimating production dates of pre-modern texts in large document collections. The following chapter will cover the work of papers VI-IX.

Segmentation methods (i.e. breaking a page down into a set of classes of objects) are often very hard to develop for large sets of data. Most of the variation in the images needs to be taken into account for a result that is successful on more than a small set used in the development. This demands a very flexible segmentation method or a very homogeneous data collection. An almost impossible requirement for real data. There are many potential ways of getting around this problem, e.g. learning a flexible model, several specialized systems for one data set or simply relaxing the demands on segmentation, and compensating elsewhere. In paper VI, binarization (i.e. the process of separating foreground from background) was used to find the ink strokes. Statistics on these ink strokes were used as features for separating scribal hands (i.e. writer identification). Evaluation was performed on a homogeneous data set, C61 (section 2.2). In papers VII-IX, the segmentation problem is largely avoided by introducing a sampling mechanism based on edge detection. This approach made it possible to perform feature extraction on a large dataset with varying contrast, light conditions and several types of degradations (the SHDK data set, section 2.3).

Binarization is an ongoing story in the document analysis community, many times seen as a solved issue but always re-opened as new data sets become available to the community. An important part of the novelty presented in papers VII-IX, is that the parts of the problem usually solved by segmentation, are pushed onto lower level stroke detection and feature selection. The number of features extracted was let to be very high, leaving sorting the good from the bad to a learning step (i.e. a Gaussian process).

The writer identification task is all about finding some distance metric between scribal hands, making clustering of similar hands possible. The estimation of production dates is the corresponding regression problem to this writer classification. Here, the aim is to quantify the *aspect* of a page for discriminating between writer styles. The aspect here is a term from palaeography for the

overall impression of a page, something there is currently no successful way of quantifying more objectively. Instead, the shapes of specific characters are used as criteria for discriminating in a quantitative way, e.g. the criteria summarized by Derolez (2003). Solving this problem using a computer requires the characters to be found and, in the worst case, the whole text transcription problem needs to be solved. Text recognition is sometimes said to be solved, a half truth. Assuming the text is scanned, black on white, 12 pt text in the Arial font without degradations, it is a solved problem. In all other cases your mileage may vary. Automatic transcription of pre-modern manuscripts can only be performed, with any real accuracy, on very few texts. Then, focussing on the page as a whole, effectively avoiding the transcription problem, makes analysis possible on more manuscripts. However, this is not a generally accepted way of doing studies in palaeography, opening up for a whole new realm of pedagogical challenges.

In philology, there is a long tradition of performing writer identification. Sometimes the, perhaps more correct term, scribal attribution is used. Writers have not identified themselves, but text is attributed to a scribal hand without saying anything about the person connected to said hand[1]. Common criteria are orthography, language forms and palaeography, as explained by Mårtensson (2011). Palaeography is described as the central tool for scribal attribution, the others as complementary. However, defining formal criteria for shape is hard, leading to disagreement on criteria, let alone attribution. Another problem is the variation in one hand. A formal style with little variation in shape between different instances of characters, is preferable to informal writing where the scribe might not have put as much work into uniformity and readability of the text. These conditions demand different methodologies when approaching different manuscript styles, even if the writer is the same. For Nordic medieval material, the criteria compiled by Åström (2003) are a source of common ground for researchers. However, Åström is negative to only using palaeographical criteria for attribution, emphasizing the value of also using language forms, orthography, manuscript or paper quality, watermarks etc. These criteria are often too vague for doing a production date estimation, with any certainty. This is where computerized methods come in, with their capability to measure small variances in the ink strokes exactly, without tiring.

Computational methods have been used for some time for identifying scibal hands, e.g. by Ciula (2005) for analysing formal script in the context of digital humanities. The base in their method was that of creating average letter forms, and similar work has been done before the dominance of computers, e.g. by Gilissen (1973). For forensic writer identification, systems have been assisting in finding matching hands in databases (thereby reducing the need for human

[1]It is important to note that for pre-modern text, the person signing the text might not be the scribe. It was not uncommon to dictate to a scribe, even for an educated person that could potentially write their own text.

labour), e.g. in a system proposed by Greening et al. (1995), built on criteria for cursive writing by Eldridge et al. (1984). They clustered writers by finding regularities within a scribal hand by looking at, among many other features, the relative difference between minims and ascenders in the script, and the angles with which diacritic signs attach to ascenders. For a good overview of the field, the survey by Franke and Srihari (2008). In the document analysis subfield of computer science, there is a tradition for feature extraction where an image is binarized and connected components are measured (with significant methodological variation). The quill features, described and used below, fits well with this tradition.

Significant work exist in the literature on how to hand-craft a good feature set for writer identification and production year estimation. A good feature should catch the relevant variance in a scribal hand while ignoring standard properties of writing. Classification and regression is often performed using support vector machines, or other non-linear kernel method,for finding a good separation[2]. The feature can be explicit in what is sought for, e.g. as by Bulacu and Schomaker (2007) with the Hinge feature for writer identification or the work by He et al. (2016c) on estimating production years. In contrast, the bag-of-features approach, feature extraction is made in a much less structured way. This approach, pioneered by Li and Perona (2005) for images, the relevant features are learned from a representation of the image data that is geometrically unstructured and does not take as much less time to engineer. Both approaches have been used in papers VI-IX.

4.1 Features and distance metrics

The purpose of feature extraction is to quantify something in an image that give similar values for similar images. Below, the similarities sought for are writer hand characteristics.

4.1.1 Quill based features

The quill feature, proposed and developed by Brink (2011) and Brink et al. (2012), is a way of quantifying the angle of pen strokes in a page. This type of feature comes from a long standing tradition in image analysis of measuring the shape of distinct objects in a binary image, see for instance Schomaker and Bulacu (2004). A standard way of constructing this type of pipeline is the following:

[2]Some also try to make the features linearly separable.

1. **Binarize** the image, e.g. according to Otsu (1975)
2. **Find each connected component**
3. **Extract statistics** on the object borders (e.g. chain codes) and/or shape (e.g. moments).
4. **Collect the measurements** from all sample points, from all connected components, into a histogram with some predefined number of bins (to get smooth transitions between bins, interpolation can be used).

The resulting histogram is used as a feature vector for describing a page. Comparing histograms is not an exact science. Euclidean distance can be used (as by He et al. (2016d)), though in a probability density the relevant information can often be found in the subtle differences. In section 4.1.4, some other distance metrics that are more suitable for a space of probability densities are described . The shape information deemed relevant in the quill feature is the joint distribution over width and angle of the pen stroke, working from the fact that the quill pen (used throughout the medieval era) has a distinctive angle of the tip. The process of creating a quill feature histogram is described in algorithm 2.

In algorithm 2, the boundaries of the connected components from a binarized image are traversed clockwise. At each point, the local boundary angle ϕ is estimated together with the maximal length w of a Bresenham line perpendicular to the ϕ angle though the character. In this way, each boundary point is associated with a feature point that is the pair of angle and width estimates (ϕ, w). All these pairs are collected as a histogram. The end results is a two dimensional histogram, with p number of bins for w and q number of bins for ϕ, that can be flattened to a feature vector of dimensionality $p \times q$. This feature vector, associated with a page, can now be compared using any divergence metric between discrete distributions, giving a metric of dissimilarity between

pages. Euclidean distance gives reasonable performance with this procedure, though in the original paper by Brink, they used χ^2-distance.

Algorithm 2: The feature extraction process for the quill feature (section 4.1.1).

Data: I: image, p: integer number of widths bins, q: integer number of angle bins, l: integer leg length
Result: H: Histogram of the joint distribution over pen angle and stroke width, where $H \in \mathbb{R}^{p \times q}$

$S \longleftarrow \emptyset$
$J \longleftarrow Preprocessing(Otsu(I))$
$C \longleftarrow SetOfConnectedComponents(J)$
for $c \in C$ **do**
  $b \longleftarrow OrderedBoundry(c)$
  for $i \in \{x : x \in \mathbb{Z} \wedge 0 \leq x < \text{len}(b)\}$ **do**
    $\phi_1 \longleftarrow angle(b[(i-l) \text{ \% len}(b)], b[i])$
    $\phi_2 \longleftarrow angle(b[i], b[(i+l) \text{ \% len}(b)])$
    if $|\phi_1 + \phi_2| < \pi$ **then**
      $\phi \longleftarrow (\phi_1 + \phi_2)/2$
    else
      $\phi \longleftarrow (\phi_1 + \phi_2)/2 + \pi$
    $(x, y) \longleftarrow b[i]$
    $\hat{x} \longleftarrow x + p \cdot cos(\phi + 1.5\pi)$
    $\hat{y} \longleftarrow y + p \cdot sin(\phi + 1.5\pi)$
    $w \longleftarrow BresenhamRaycast(x, y, \hat{x}, \hat{y})$
    $S \longleftarrow S \cup \{(\phi, w)\}$
  // If curvature is needed, calculate it here
$H \longleftarrow HistogramFromCoordinateSet(S, p, q)$

In paper VI, we extended the quill feature by adding local curvature to each sample point on the connected component. Now the histogram is in three dimensions and the number of bins $p \times q \times r$, where r is the number of bins in the curvature. The curvature is estimated by convolving the estimated angles at each boundary point with a differentiating mask, inspired by finite difference methods. We have presented this type of feature approach, together with a discussion on its potential impact on digital palaeography, for an audience of humanist researchers in Mårtensson et al. (2015).

4.1.2 Sets of image patches as features

In the following section, the process from papers VII and VIII of using sets of image patches as features, will be described. The process is as follows:

1. **Identify the boundaries** of ink strokes (i.e. the edges).
2. **Sample patches** at ink stroke boundaries for all images.
3. **Cluster patches** using a shape sensitive metric.
4. **Find the proportions** of patches most similar to each cluster centre, for each image.

The result is a bag-of-features type vector, representing each page in some collection. These vectors are used for classification or regression (as in papers VII and VIII for estimating the production date).

Working form the hypothesis that the ink strokes are the interesting parts of a manuscript, the edges of the strokes were extracted. Others have chosen a similar approach but extracted the skeleton of the ink stroke instead of the edges, e.g. Christlein et al. (2015) and He and Schomaker (2015). Since the method for manuscript dating presented here is evaluated on the database "Svenskt diplomatariums Huvudkartotek" (SDHK, section 2.3) some special challenges with degradations had to be tackled. Even identifying full ink strokes was a problem in many of the charters and hence, our approach could not rely on binarization (usually a prerequisite to finding the skeleton). Building the method on edge detection, lets us extract data from more damaged parts of the manuscripts. This approach was loosely inspired by the state-of-the-art binarization method proposed by Howe (2012). There, the author uses edge detection together with a graph cut to find the foreground.

Canny edge detection, proposed by Canny (1986), was used for edge detection of the ink strokes. This algorithm is well tested and understood, but requires threshold parameters. The parameters correspond to a level of gradient magnitude, extracted from the image using Sobel filters and l_2-norm, above which all pixels are edge candidates and a second threshold, below which, pixels are not edge pixels. In the magnitude range between the thresholds, ridges that connect pixels above the higher threshold are defined as edges. The thresholds can be set manually, assuming a smaller data set with little variance in contrast. This is, however, not the case with SDHK. Instead, we developed a statistical model for adapting the threshold to each charter. After the image gradient magnitudes was extracted (in the same way as Canny), a Gaussian mixture model (equation 4.1) was trained to fit the distribution over gradient magnitudes. The thresholds for the Canny edge detection algorithm were set relative to the structure identified by the mixture model. Effectively adapting the thresholds to the light level, contrast and level of degradations of each individual charter. The thresholds T_{low} and T_{high} were defined as is shown in equations 4.3 and 4.2.

$$p(x) = \omega_1 \mathcal{N}(x \mid \mu_1, \sigma_1^2) + \omega_2 \mathcal{N}(x \mid \mu_2, \sigma_2^2) \tag{4.1}$$

$$\omega_1 \mathcal{N}(T_{low} \mid \mu_1, \sigma_1^2) = \omega_2 \mathcal{N}(T_{low} \mid \mu_2, \sigma_2^2) \tag{4.2}$$

$$T_{high} = max\,(\mu_1, \mu_2) \tag{4.3}$$

Assuming the lower mixture component would converge to the spike in lower magnitudes, corresponding to colour-wise flat areas, the higher mixture component would fit the higher values, corresponding to areas with edges. The lower threshold was then set as the point at which the probability of a magnitude belonging to either mixture was equal (i.e. the point of transition between background and higher edge magnitudes). The higher threshold is set as the mean of the higher mixture component, hence, it is assumed to be part of some ink stroke edge. This approach did not only extract edges along ink strokes, but would also often find edges on sigils and manuscript edges.

To mitigate the problem of non-stroke edges, the Stroke Width Transform (SWT) was used for de-noising the edge traced images. The SWT, proposed by Epshtein et al. (2010), was originally developed for text detection in video. The transform finds, for every pixel in an image, how far away the corresponding other side of a potential character edge is, assuming that the current pixel is a character edge pixel and looking in the direction of the gradient. Adapted to the problem here, this means that for each potential edge pixel, given by the edge detection, the distance to the corresponding other side of the ink stroke was measured. The distance is only defined if two edges can be bound together by a line that is approximately perpendicular to both edges ($\pm\frac{\pi}{6}$). This procedure removed edge pixels on the charter's boundary, stemming from colour change found at the parchment's physical edge. After this de-noising, character ink edges and some noise coming from the sigils remain. The SWT procedure is illustrated in figure 4.1.

In papers VII and VIII, the stroke image was cut up into all possible patches with a centre pixel on a stroke edge. To cluster this cloud of patches, some distance metric was needed that took shape into account. Hence, a standard metric like Euclidean or divergence metrics were unsuitable. The shape context descriptor was proposed by Belongie et al. (2002), and describes pixels surrounding a point as probability distribution (i.e. histogram). This was meant as a key point descriptor but can just as well be used as a distance metric between patches of semi binary data (the patches only have none-zero values along edges but the values can be an intensity).

The shape context is a two dimensional histogram of the coordinates of none-zero pixels in a small area, transformed into a log-polar space. For each image patch $s \in \mathbb{R}^{N \times N}$, coordinate triplets are extracted for each none-zero pixel as $(x_i, y_i, I_i) \in S$, where x_i, y_i are positions relative to the center, I_i is an intensity and S is the set of triplets for the patch s. The positions x_i, y_i are then transformed into log-polar coordinates as $\theta_i = \text{atan2}(y_i, x_i)$ and $r_i = \log \sqrt{x_i^2 + y_i^2}$. The new triplet $(\theta_i, r_i, I_i) \in S^*$ is aggregated into a histogram

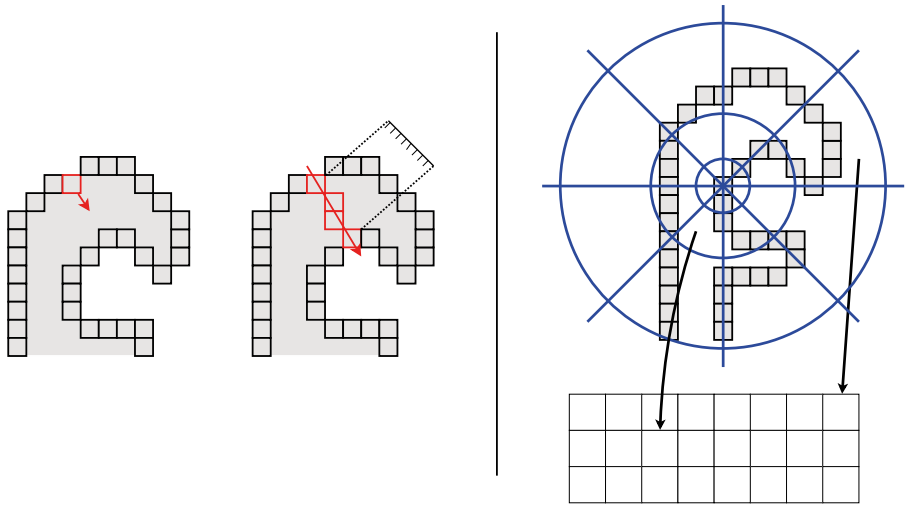

Figure 4.1. **Left:** An example of the Stroke Width Transform, where a line is traced over the image from one edge to another, approximately opposite, edge. **Right:** An example of collecting stroke edges into a log-polar shape context histogram, used for comparing patches.

$H \in \mathbb{R}^{p \times q}$, weighted by I_i and using bi-linear interpolation. Here, q is the number of angle bins p log radius bins and N the patch side length. Flattening this histogram gives a feature vector, making a shape sensitive comparison between patches possible. This procedure is illustrated in figure 4.1. For the implementation used in papers VII and VIII, the intensity I_i is an edge magnitude. To make the patch distance rotationally invariant, the gradient direction (from the original image) of the patch's centre pixel was subtracted from each θ_i in S^*. This is very convenient when all images in a collection can not be guaranteed to have the same orientation. Note that the logarithm makes the histogram have more weight for pixels closer to the centre of the patch, giving the shape sensitive distance metric some scale insensitivity.

With this patch based approach so far, each document/page can be represented using a $p \times q$ dimensional point cloud. Not a very convenient, and memory intensive format. The bag-of-features representation is a way of aggregating such point clouds of feature descriptors by clustering and only saving cluster membership. First used in computational linguistics as bag-of-words and later adapted to image data by Li and Perona (2005), only the proportions of cluster memberships are stored. Hence, for geometrically unstructured patches the positions of the extracted points in the original image are forgotten. This has proven to be a very successful approach, though unintuitive (why would location not be important). A common way of achieving a bag-of-features is the following:

1. **Extract a set of features** from each image (or image patch). This can be any feature relevant to the problem.
2. **Cluster** the feature vectors, e.g. using k-means, agglomerative clustering etc.
3. **Calculate the proportions** of feature vectors belonging to each cluster (called a bag). The probability of finding a vector belonging to each cluster center is used as the new bag-of-feature type feature vector.

A crude but effective way is to use k-means clustering, proposed by Macqueen (1967)[3], for clustering and then have a set of feature vectors split into crisp partitions belonging to the clusters[4]. A more soft way can be to fit a Gaussian mixture model (GMM) to the feature set and then use the mixture weights as the new feature. In this way, a single item in the set of feature vectors can belong to multiple clusters. Variants of this GMM are sometimes called GMM super-vectors, e.g. by Campbell et al. (2006) and by You et al. (2009) for speaker recognition and verification.

As a clustering algorithm for bag-of-feature representations, the k-means is simple but usually gives good results. To be able to do a clustering fast we used a mini-batched version of the k-means algorithm proposed by Sculley (2010). This made it possible to perform multiple experiments, varying the number of clusters with reasonable time complexity. In paper VII, we show that using an ensemble of estimators, varying the number of clusters in the bag-of-features, improved results. Using several clusterings makes it possible to create a more rich codebook where combinations of entries represent parts of clusters in the original clustering.

4.1.3 Text features for transcribed documents

In paper VIII, text based features were used. Inspired by the mixture of modalities used by Li et al. (2015), the work on historical sources by Pettersson et al. (2014) and by Piotrowski (2012) and, lastly, that the SDHK (section 2.3) collection had 5000+ charters transcribed, adding a modality of data seemed as an interesting challenge. The hope was that the estimators based on transcribed text and the image estimators, would compensate for each others shortcomings. This turned out to be the case, a short comparison of dating methods can be found in section 4.3. However, the dating estimators, based on text data and an ensemble of Gaussian processes, was good enough to stand on its own. In the paper, we speculate that the sounds of old Swedish were changing a lot during

[3]The k-means algorithm is often confused, as is well known by now in the document analysis community, with the algorithm formulation used by Lloyd (1982).

[4]This approach is very similar to the "shapeme" proposed by Mori et al. (2005), later used by Rusiñol and Lladós (2010) for a fast way of searching a set of company logos.

the time period our data comes from, making a case for that spelling changes in the Latin alphabet were reasonable to expect. However, the performance on church Latin is very similar to the performance on old Swedish, as shown in paper VIII.

The features for dating based on text were bag-of-features of n-grams. In the work by Cavnar et al. (1994), a bag-of-features representation of n-grams is used for classifying texts in a heterogeneous corpus. An n-gram is sequence of n number of characters. The n-grams of a text are all the possible, unique and overlapping, sequences of characters, of length n, that can be found. Making statistics of how many n-grams and normalizing the sum to 1, with a predefined n, was used as the final feature vector. In paper VIII, $n = 1, 2, 3$ were used. For a common reference frame, all possible n-grams were extracted from the full collection of transcriptions, instead of using all theoretically possible n-grams (a huge space). To reduce the dimensionality of 3-grams, from about 40000, two dimensionality reductions using principal component analysis were run with 500 and 1000 components, respectively, for details on such a procedure see Jolliffe (2002). In the end, the text based ensemble of estimators performed on par with the image based ensemble estimators, where the text based ensemble consisted of four estimators with different design parameters (1-gram, 2-gram, 3-gram with a 500 component PCA, 3-gram with a 1000 component PCA).

4.1.4 Divergence and distance

What is distance? Usually we think of Euclidean distance, as in equation 4.4. In a flat world, this is the distance a bird flies or the length of a straight line from point A to point B. This notion of distance, however, can be a bit misleading. In the real world, a bird can not fly in a "straight" line for very long. The length of a line drawn for the bird's flight must, for longer distances, have the same curvature as the earth's surface. Analogously, the concept of distance can be extended to statistics collected from manuscripts on the scribal hands. Then the technical term is *divergence* between discrete statistical distributions (e.g. quill features, bag-of-features vectors). Here, however, the equations are less intuitive compared to the Euclidean distance as it is presented in equation 4.4. In the following equations in this section, distance/divergence will be calculated between vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$

$$D_{\text{Euclidean}}(\mathbf{v}, \mathbf{u}) = \sum_{i=1}^{n} (\mathbf{v}_i - \mathbf{u}_i)^2 \tag{4.4}$$

In paper VI, we compared several divergence metrics for finding the "distance" between manuscript pages. We experimentally verified that Euclidean distance did not make a good metric for the examined distributions. In the work by Brink et al. (2012) and by Schomaker and Bulacu (2004), the χ^2 met-

ric (equation 4.5) was used because of its property to amplify the effect of areas with small differences. This is likely the reason for its relative success compared to Euclidean distance where smaller differences are "drowned out" by the larger. In effect, Euclidean focuses on large differences and χ^2 on the subtle ones.

$$D_{\chi^2}(\mathbf{v}, \mathbf{u}) = \sum_{i=1}^{n} \frac{(\mathbf{v}_i - \mathbf{u}_i)^2}{\mathbf{v}_i + \mathbf{u}_i} \tag{4.5}$$

In our paper, we evaluated two more metrics between discrete distributions. With inspiration from the excellent survey by Cha (2007), we looked at an additive symmetric χ^2 (equation 4.6). The name "additive symmetric" comes from the way two Person χ^2 non-symmetric divergence functions are added together to create a symmetric function. If the same procedure is performed on two Kullback-Leibler divergences ($D(a, b) + D(b, a)$), the result is the Jeffrey's divergence (equation 4.7).

$$D_{\text{AddSum}\chi^2}(\mathbf{v}, \mathbf{u}) = \sum_{i=1}^{n} \frac{(\mathbf{v}_i - \mathbf{u}_i)^2(\mathbf{v}_i + \mathbf{u}_i)}{\mathbf{v}_i \mathbf{u}_i} \tag{4.6}$$

$$D_{\text{Jeffreys}}(\mathbf{v}, \mathbf{u}) = \sum_{i=1}^{n} (\mathbf{v}_i - \mathbf{u}_i) \ln \frac{\mathbf{v}_i}{\mathbf{u}_i} \tag{4.7}$$

The Euclidean distance consistently showed lower evaluation results than the χ^2 and Jeffrey's divergence metrics, on the problem explored in paper VI. This is somewhat surprising since Euclidean distance is so commonly used. This might have been the most important point in the paper, that the methods we were using had a built in assumption of Euclidean space, even though that design choice could be detrimental to the task.

4.2 Assigning labels to the data

Most learning approaches utilize an unsupervised or supervised strategy. Supervised learning uses labelled data as input for finding a mapping between a data space and some target space. The target space will be categorical if the labels are such, or it can be any other domain as long as it is of the same type as the labels themselves. This can be performed for both classification and regression tasks. By contrast, unsupervised learning aims to find some structure in the data without labels revealing any of this structure. This is usually performed by assuming some abstract structure in the data and then trying to find a model of the data that fits into these assumptions. For example, a Gaussian mixture model (GMM) for clustering assumes that the data can be seen as generated from a finite number of Gaussian distributions. The number of mixtures

and constraints on the covariance matrix (e.g. diagonal) are design choices. A GMM might not be the best model for the data, but after training the GMM will fit the data with as high a likelihood as possible (assuming successful training).

A mix of supervised and unsupervised learning is called semi-supervised learning, and as the name implies, only some of the data points come with labels. This makes it possible to mix the approaches and find only the subspace of solutions allowed by both the supervised and unsupervised sub-problems. An advantage being that the supervised problem can take advantage of structure that is in the data without explicitly knowing the training labels. The many types of semi-supervised learning available in the literature is excellently presented by Zhu (2005b) and Zhu (2005a). For the task of writer identification, semi-supervised learning have successfully been employed by Ball and Srihari (2009) and by Porwal and Govindaraju (2013) for writer identification. For an overview of other types of successful writer identification techniques, see the conference competition reports by (Louloudis et al., 2012) and (Louloudis et al., 2013).

4.2.1 Semi-supervised learning on graphs

In paper VI, a semi-supervised label propagation approach was used on a graph for writer clustering inspired by Zhu (2005a). Every node in the graph was connected to a page in a medieval book (C61, the revelations of Saint Bridget, section 2.2). All edge weights was defined as the divergence between the respective nodes' feature histograms, weighted by a Gaussian PDF[5]. On this graph, labelled samples could propagate their labels. Distance/cost on the graph was defined as additive edge weights, giving a label propagation with clear parallels to Dijkstra's algorithm for shortest path. Each unlabelled node was assigned the label of the labelled node with the minimum accumulated divergence over the graph.

In algorithm 3, the label propagation procedure for paper VI is shown. Each page in C61 was modelled as a vertex with a feature histogram associated with it. In the paper, the quill feature and the extended quill curvature feature was used to characterize the manuscript page. Several divergence metric were also examined. Some extra non-linearity was introduced in the form of a weighting of the divergences (i.e. edge weights), depending on the design parameter σ.

[5]The parallels to kernel methods are ever present.

This parameters was estimated using leave one out cross validation, on the training set.

Algorithm 3: A slightly modified Dijkstra's algorithm for propagating class membership, from some seed vertices, on a graph (section 4.2.1). This algorithm was used in paper VI for semi-supervised labelling of writers.

Data: V: A set of vertices (each with an associated feature vector),
S: A set of labelled (as categorical class memberships) vertices,
σ: A parameter for the Gaussian weighting of feature differences,
$d(\cdot, \cdot)$: A distance function between feature vectors
Result: C: The set of categorical class membership for each vertex

```
/* Initialize the sets making up the graph G =
   (V, E, C, D)                                    */
```
for $v_1 \in V$ **do**
    $D[v_1] \longleftarrow \infty$ `/* Seed dissimilarity cost */`
    $C[v_1] \longleftarrow None$ `/* Class memberships */`
    for $v_2 \in V$ **do**
        if $v_1 \neq v_2$ **then**
            `/* Edge costs */`
            $E[v_1, v_2] \longleftarrow 1 - \exp\left(-\frac{d(featureOf(v_1), featureOf(v_2))^2}{2\sigma^2}\right)$

```
/* Set up the seed points and add all vertices
   to Q                                            */
```
$Q \longleftarrow copy(V)$
for $v \in S$ **do**
    $D[v] \longleftarrow 0$
    $C[v] \longleftarrow S[v]$

while $|Q| > 0$ **do**
    $u = \underset{v \in Q}{\text{argmin}}\, D[v]$
    for $v \in neighboursOf(u)$ **do**
        $n \longleftarrow D[u] + E(u, v)$
        if $n < D[v]$ **then**
            $D[v] \longleftarrow n$
            $C[v] \longleftarrow C[u]$
    $Q.removeVertex(u)$

A comparison of the classification procedures can be seen in figures 4.2 (nearest neighbour) and 4.3 (semi-supervised learning). In the case of nearest neighbour classification, the distance to the labelled data points is measured for every unlabelled data points. The label of the nearest labelled data point is

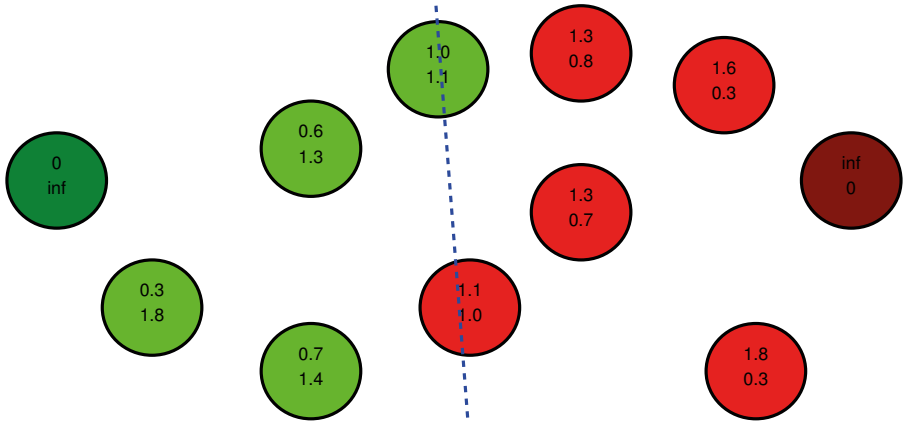

Figure 4.2. An example of label attribution (classification) using nearest neighbour. The darker nodes on each side are the labelled samples for the red and green classes. Distances to a labelled node is shown as numbers inside each node. The unlabelled samples get the label of the closest labelled sample. In this model, the structure of elongated clusters can not be taken into account, such as pages with small successive changes. Compare to figure 4.3 for a semi-supervised learning classification.

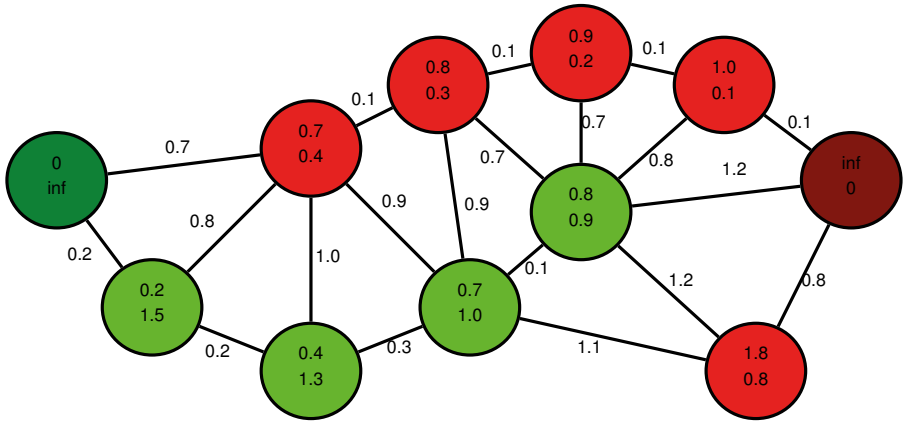

Figure 4.3. An example of label propagation using semi-supervised learning. The darker nodes on each side are the labelled samples for the red and green classes. Distances to a labelled node is shown as numbers inside each node. The distance between nodes are defined along the graph, with the edge weights shown beside each connection between nodes (i.e. the edges). Neighbouring nodes with very small changes are classified as one cluster, indifferent of their pair-wise distances from the labelled nodes. Compare to figure 4.2 for a nearest neighbour classification.

	Quill		Quill-Curvature	
	Original	Registration	Original	Registration
1-top	100%	100%	100%	100%
Hard 5-top	98%	99%	99%	99%
Hard 10-top	94%	97%	97%	99%
AUC kNN	98.5	99.0	98.9	99.2
AUC SSL	98.7	99.1	98.9	99.2
kNN data for 99% precision	**19.4%**	9.3%	12.1%	6.8%
SSL data for 99% precision	17.3%	6.8%	11.3%	**6.0%**

Table 4.1. *Selected results comparing nearest neighbour (kNN) classification and semi-supervised (SSL) classification from paper VI. Quill feature is the original proposed by Brink et al. (2012) and quill-curvature is proposed in paper VI. Also, the registration scheme (”registration”), proposed in paper VI, is compared to not using registration (”original”). Hard n-top results when using a nearest neighbour classifier and χ^2-dissimilarity. An explanation for the AUC evaluation metric can be found above in section 3.2.2. In the last two rows, the ratio of training data needed for a precision of 99% is shown (note figures in bold font). The proposed improvements from paper VI, primarily reduces the need for labelled data. For historical sources, labelled data is expensive since it demands a very high level of domain knowledge.*

chosen as the class of each unlabelled point. The distance metric is arbitrary and should be chosen as one that fits the data type. In figure 4.2, the distance to the two labelled data points , shown in a darker shade of red and green, are written inside the lighter coloured circles. A problem with this methods for writer identification is that the style of a writer will have small changes between every data point, which in this case are pages. The style of a scribal hand will depend most on who the writer is but also in fatigue, light in the scriptorium, degradation of the pen tip etc. In the feature space, this translates to that distance to a labelled sample is not the only criterion for classification that is relevant. Some way of letting data point very close to each other be considered together is needed. In figure 4.3, the labels of the labelled data points (i.e. the training set) are propagated to the surrounding data points by algorithm 3. The distance to the closest labelled data point is defined as the accumulative cost of the lowest cost path to the data point. The numbers inside the circles are these distances and each connection between circles has the cost of propagating through that edge. The colouring of the circles in the figure now show how two clusters of writers take closeness of unlabelled data points into account. This approach is an example of semi-supervised learning, as described above. Some results from the evaluation in paper VI is shown in table 4.1. The major improvement, from using a semi-supervised approach with curvature added to the quill feature, was that the ratio of labelled data could be lowered by a factor of three. Labelling data is expensive, hence, for the same labour cost, three times as many books can be precessed.

4.2.2 Gaussian process regression

For the dating application, some way of mapping points in a feature space to a time-line was needed. This is a classic regression problem. In the literature, support vector regression (SVR) has been used for age estimation of humans by Guo et al. (2008), with method development by Zhang et al. (2006), and adapted to documents by He et al. (2014b). Training an SVR estimator is fast, allows for a large number of training data points and is non-linear through the use of kernel functions. However, SVR does not easily allow for quantifying the uncertainty of the estimation. The distance from a regression line has no obvious interpretation. In a Gaussian Process (GP), where the target space consists of Gaussian variables, this distance from a decision boundary has a likelihood interpretation. The maximum a posteriori (MAP) estimate gives the same information as using an SVR, but the standard deviation of the output variable can have years as its unit.

When training an estimator, certain care has to go into the design methodology to not create an estimator suffering from over learning. This happens when the estimator becomes too specialized and loses its capacity to generalize to other data in the same problem domain. How the generalization properties should look to best suit the given problem is in most cases unknown. Hence, the capacity to generalize needs to be tested on available data. A common approach is to split the data into three non-overlapping sets for training, validation and testing. The training set is used to train or tune the estimator, the validation set for model selection and the test set for estimating the expected performance on an unseen set of data. Note that this procedure assumes that the data set catches the variance of the problem, which is not always the case as illustrated by Torralba and Efros (2011). GP regression is no exception to concerns about over learning. However, estimating a distribution over the model parameters mitigates this problem somewhat. In fact, a GP does not necessarily use any parameters in a traditional sense but uses all the training data as a part of the model. This approach stands in contrast to for example SVM and CNN approaches that use the data for training parameters and then forgets it.

Definition, and a prediction

A Gaussian Process (GP) (Rasmussen, 2006; Bishop, 2006) is a set of stochastic variables $\{X_s : s \in S\}$, indexed by some set S, that are jointly Gaussian (i.e. a vector $(X_{s_1}, ..., X_{s_n})^{\mathbb{T}}$ is a multi-variate Gaussian distribution). This can be written as $X \sim \mathcal{GP}(\mathrm{m}(\cdot), \mathrm{k}(\cdot, \cdot))$, where the mean function $\mathrm{m}(\cdot)$ and covariance function $\mathrm{k}(\cdot, \cdot)$ define the behaviour of the GP. Often, the mean of all variables X_s is assumed to be zero, leading to the very convenient property that the GP is completely defined only by the covariance function $\mathrm{k}(\cdot, \cdot)$. However, it is not obvious from this definition how to use a GP for predictions. Given some training set $\mathcal{D} = \{(\mathbf{x}_i, t_i)\}_1^N$, a partitioned Gaussian distribution can be constructed, where an unknown partition (the points to predict) can be

conditioned on a known part (the training data). The relations between the feature space points of training data and the points to estimate, is given by the covariance function $\mathrm{k}(\cdot,\cdot)$. The values of the know partition are given by the target variables of $\mathcal{D}$. Note that $\mathrm{k}(\cdot,\cdot)$ can, but does not have to, be the "standard" covariance function $\mathrm{k}(\mathbf{x},\mathbf{x}') = \mathbf{x}^{\mathrm{T}}\mathbf{x}'$. For this example, let the partition look like:

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix} \tag{4.8}$$

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ba} \\ \boldsymbol{\Sigma}_{ab} & \boldsymbol{\Sigma}_{bb} \end{pmatrix} \tag{4.9}$$

In this partition, $\boldsymbol{\Sigma}_{bb}$ is given by $k(\mathbf{x}_i,\mathbf{x}_j)$ and $\boldsymbol{\mu}_b = \mathbf{t}$, where $\mathbf{x}_i$ and $\mathbf{t}$ is from the training data $\mathcal{D}$. A new set of feature space points x^* are defined for where in the feature space to predict, giving $\boldsymbol{\Sigma}_{aa}$, $\boldsymbol{\Sigma}_{ab}$ and $\boldsymbol{\Sigma}_{ba}$. The relations between the partitioned Gaussian above and the predictions for the conditional Gaussian, $\boldsymbol{\mu}_{a|b}$ and $\boldsymbol{\Sigma}_{a|b}$, are given by:

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\left(\boldsymbol{\Sigma}_{bb} + \sigma^2\boldsymbol{I}\right)^{-1}(\mathbf{t} - \boldsymbol{\mu}_b) \tag{4.10}$$

$$\boldsymbol{\Sigma}_{a|b} = \left(\boldsymbol{\Sigma}_{aa} + \sigma^2\boldsymbol{I}\right) - \boldsymbol{\Sigma}_{ab}\left(\boldsymbol{\Sigma}_{bb} + \sigma^2\boldsymbol{I}\right)\boldsymbol{\Sigma}_{ba} \tag{4.11}$$

The derivations of these equations, 4.10 and 4.11, follow from the quadratic form of a Gaussian (using the given partitions). The quantity σ comes from a prior on the model noise (it can be thought of as the standard deviation of the measurement noise). In a GP, the variables are jointly Gaussian. Hence, a prediction can be seen as a draw from a Gaussian with zero mean, a covariance matrix given by $k(\cdot,\cdot)$, conditioned on the given data. Note that $\boldsymbol{\mu}_{a|b}$ and $\boldsymbol{\Sigma}_{a|b}$ describe the conditional distributions, not a draw.

Relation to linear regression

Another way of showing the relevance of a GP for the date estimation application is the following. Equation 4.12 shows a basic regression problem [6]. It is a mapping from some domain to a continuous surface (or line) in some other domain. In the case of manuscript dating, this is what is needed to map some vector of feature values to a time line. Some noise ϵ is added, to model the errors in the regression. In equation 4.13 , the regression function $f : \mathbb{R}^n \to \mathbb{R}^m$ is what we want to estimate (a mapping from an n dimensional space to an m dimensional space). The training data is defined as above, $\mathcal{D} = \{(\mathbf{x}_i, t_i)\}_1^N$, where $\mathbf{x}_i$ is the input data and t_i is the target (i.e. the desired return data from f given $\mathbf{x}_i$ as input). In the following derivation, $n > 1$ and $m = 1$. Also, the function ϕ is added in f as an arbitrary feature transformation on the input

[6] Gaussian process regression is sometimes called Kriging.

data. This function will later be used to show how kernel functions can be used with Gaussian Processes.

$$t_i = f(\mathbf{x}_i) + \epsilon \tag{4.12}$$

$$f(\mathbf{x}_i) = \phi(\mathbf{x}_i)^T \mathbf{w} \tag{4.13}$$

If the values of $\mathbf{w}$ would be determined by least squares and Gaussian noise is assumed for ϵ, this would be ordinary linear regression. The key step for GP regression, following the reasoning by Rasmussen (2006), is to assume that both ϵ and $\mathbf{w}$ are drawn from Gaussian distributions, where $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ and $\sigma \in \mathbb{R}$.

$$\epsilon \sim \mathcal{N}(0, \sigma^2) \tag{4.14}$$

$$\mathbf{w} \sim \mathcal{N}(0, \mathbf{\Sigma}) \tag{4.15}$$

This makes it possible to set up Bayes' law to get a probability distribution over $\mathbf{w}$, given the data. In the following derivation, $\mathbf{\Phi}$ will be used as the design matrix for all $\phi(\mathbf{x}_i)$, $\boldsymbol{\phi}_i$ as short hand for $\phi(\mathbf{x}_i)$ and $\mathbf{t}$ for all t_i. The expression for $p(\mathbf{w}|\mathbf{\Phi}, \mathbf{t})$ looks as follows:

$$p(\mathbf{w}|\mathbf{\Phi}, \mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{\Phi}, \mathbf{w}) p(\mathbf{w})}{p(\mathbf{t}|\mathbf{\Phi})} \tag{4.16}$$

The expression for the prior on $p(\mathbf{w})$ (follows from equation 4.15) is given by:

$$p(\mathbf{w}) = \frac{1}{\sqrt{(2\pi)^n |\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2} \mathbf{w}^{\mathrm{T}} \mathbf{\Sigma}^{-1} \mathbf{w}\right) \tag{4.17}$$

Since the data is assumed to be independent, it can be evaluated point wise. Note that σ is the standard deviation of the measurement noise (from equation 4.14). The likelihood of $\mathbf{t}$, given $\mathbf{\Phi}$ and $\mathbf{w}$, is given by (where N is the number of points in $\mathcal{D}$):

$$\begin{aligned} p(\mathbf{t}|\mathbf{\Phi}, \mathbf{w}) &= \prod_{i=0}^{N} p(t_i|\boldsymbol{\phi}_i, \mathbf{w}) \\ &= \prod_{i=0}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{(t_i - \boldsymbol{\phi}_i^{\mathrm{T}} \mathbf{w})^2}{2\sigma^2}\right) \\ &= \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left(-\frac{\left|\mathbf{t} - \mathbf{\Phi}^{\mathrm{T}} \mathbf{w}\right|^2}{2\sigma^2}\right) \\ &= \mathcal{N}(\mathbf{\Phi}^{\mathrm{T}} \mathbf{w}, \mathbf{I}\sigma^2) \end{aligned} \tag{4.18}$$

Since the numerator is a multiplication between two Gaussian distributions, only the quadratic forms need to be combined. Also, the denominator will only add a normalization for the expression in the numerator. Working from this, the following equation only contains the relevant parts.

$$\begin{aligned} p(\mathbf{w}|\mathbf{\Phi}, \mathbf{t}) &\propto \exp\left(-\frac{1}{2\sigma^2}\left(\mathbf{t}-\mathbf{\Phi}^{\mathrm{T}}\mathbf{w}\right)\left(\mathbf{t}-\mathbf{\Phi}^{\mathrm{T}}\mathbf{w}\right)\right)\exp\left(-\frac{1}{2}\mathbf{w}^{T}\Sigma^{-1}\mathbf{w}\right) \\ &\propto \exp\left(-\frac{1}{2}\left(\mathbf{w}-\boldsymbol{\mu}\right)\mathbf{\Lambda}\left(\mathbf{w}-\boldsymbol{\mu}\right)\right) \end{aligned} \tag{4.19}$$

, where:

$$\boldsymbol{\mu} = \sigma^{-2}\mathbf{\Lambda}^{-1}\mathbf{\Phi}\mathbf{t} \tag{4.20}$$

$$\mathbf{\Lambda} = \sigma^{-2}\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{T}} + \mathbf{\Sigma}^{-1} \tag{4.21}$$

Above, $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\mathbf{\Lambda} \in \mathbb{R}^{n\times n}$ are the sufficient statistics. Now the distribution of $\mathbf{w}$ given $\mathbf{\Phi}$ and $\mathbf{t}$ is:

$$\mathbf{w}|\mathbf{\Phi}, \mathbf{t} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Lambda}^{-1}) \tag{4.22}$$

For some new point $\boldsymbol{\phi}_*$, the predictive distribution can be found by marginalizing the weight vector $\mathbf{w}$ as $\int p(\mathbf{f}_*|\boldsymbol{\phi}_*, \mathbf{w})p(\mathbf{w}|\mathbf{\Phi}, \mathbf{t})d\mathbf{w}$, giving the following:

$$f_*|\boldsymbol{\phi}_*, \mathbf{\Phi}, \mathbf{t} \sim \mathcal{N}(\boldsymbol{\phi}_*^{\mathrm{T}}\boldsymbol{\mu}, \boldsymbol{\phi}_*^{\mathrm{T}}\mathbf{\Lambda}^{-1}\boldsymbol{\phi}_*) \tag{4.23}$$

The inverse in the above expression is of a $n \times n$ matrix. If $n < N$ (i.e. the feature space dimensionality is lower than the number of data points), a common re-parametrization is:

$$\begin{aligned} f_*|\boldsymbol{\phi}_*, \mathbf{\Phi}, \mathbf{t} \sim \mathcal{N}(&\boldsymbol{\phi}_*^{\mathrm{T}}\mathbf{\Sigma}\mathbf{\Phi}\left(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{\Phi} + \sigma\mathbf{I}\right)^{-1}\mathbf{t}, \\ &\boldsymbol{\phi}_*^{\mathrm{T}}\mathbf{\Sigma}\boldsymbol{\phi}_* + \boldsymbol{\phi}_*^{\mathrm{T}}\mathbf{\Sigma}\mathbf{\Phi}\left(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{\Phi} + \sigma\mathbf{I}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Sigma}\boldsymbol{\phi}_*) \end{aligned} \tag{4.24}$$

Note the similarities between equation 4.24 and equations 4.10 and 4.11 from the initial example of prediction (remember that the mean is assumed to be zero).

Non-linearity through kernels

In equation 4.24, the feature data only enters into the equation by scalar products (this is even more clear in equations 4.10 and 4.11) . Hence, the "kernel trick" can be used for non-linear feature transformations. Using a kernel can be seen as a projection of the data to a high (even infinite) dimensional feature space, where linear separations/relations can be found. Projecting the data

back brings the linear relations back into the current data space. What kind of non-linearities that can be found in the high dimensional space is a property of the chosen kernel function (i.e. getting non-linearity into a model brings out a new model parameter for kernel selection).

One of the most common kernels is the Gaussian radial basis function (RBF). It is sometimes referred to as a Gaussian kernel, because of the similarities to a Gaussian PDF with the standard deviation σ, excluding the normalization constant:

$$\mathrm{k}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2\sigma^2}\right) \tag{4.25}$$

In the RBF above, there is no reason (other than simplicity) why the scale parameter σ should be the same for all data dimensions (Neal, 1996). Having a different noise magnitude for each data dimension is equivalent to scaling/stretching of the data space. To some extent, this can even be seen as feature selection, since a very high noise parameter can almost completely reduce the influence of some data dimension.

$$\mathrm{k}(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(-\sum_{k=0}^{n} \frac{\left(x_i^{(k)} - x_i^{(k)}\right)^2}{2\nu_k^2}\right) \tag{4.26}$$

The free variables of the kernel function are called hyper-parameters. They are often denoted as the hyper-parameter vector θ, a concatenation of all model parameters.

Model evidence and selection

Going back to the reasoning that gave rise to equation 4.10 and 4.11, the likelihood of some set of hyper-parameters is given by the likelihood of drawing the target variables from the GP model. That is:

$$\mathbf{t} \sim \mathcal{N}(0, \mathbf{K}) \tag{4.27}$$

In this expression, the elements of the covariance matrix $\mathbf{K}$ are given by $\mathbf{K}_{ij} = \mathrm{k}(\mathbf{x}_i, \mathbf{x}_j)$. The covariance function is here exemplified by a RBF-ARD kernel with a noise parameter added to the diagonal using the Kronecker delta function. In the last subsection, the kernel parameters (i.e. the model hyper-parameters) had different names while here they are all concatenated into a vector $\boldsymbol{\theta}$. The dimensionality of the data is n and $\mathbf{x}_i^{(k)}$ is the k^{th} element of the i^{th} data vector.

$$\mathrm{k}(\mathbf{x}_i, \mathbf{x}_j) = \theta_n \exp\left(-\frac{1}{2}\sum_{k=0}^{n} \theta_k \left(\mathbf{x}_i^{(k)} - \mathbf{x}_j^{(k)}\right)^2\right) + \delta_{ij}\theta_{(n+1)} \tag{4.28}$$

A restriction on the hyper-parameters is enforced as $\forall i : \theta_i > 0$ for this to be a valid kernel. Working from this, inference in the model entails finding the hyper-parameter vector that maximizes the likelihood of drawing the target values from the GP. That is:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\mathbf{t}|\boldsymbol{\theta}, \mathbf{X}) \tag{4.29}$$

This maximization is usually achieved by gradient based optimization. The problem is non-convex, which can be somewhat mitigated by random restarts. This adds to the danger of over learning, so elegantly avoided by having a distribution instead of a fixed regression weight vector above. The objective function for the optimization is the likelihood of drawing $\mathbf{t}$. The likelihood function and its logarithm for equation 4.27 are given by:

$$p(\mathbf{t}|\boldsymbol{\theta}, \mathbf{X}) = \frac{1}{(2\pi|\mathbf{K}|)^{N/2}} \exp\left(-\frac{1}{2}\mathbf{t}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{t}\right) \tag{4.30}$$

$$\log(p(\mathbf{t}|\boldsymbol{\theta}, \mathbf{X})) = -\frac{1}{2}\log(|\mathbf{K}|) - \frac{N}{2}\log(2\pi) - \frac{1}{2}\mathbf{t}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{t}. \tag{4.31}$$

The derivative, with respect to the hyper-parameters, of the log likelihood function is then given by:

$$\frac{\partial}{\partial\theta}\log\left(p(\mathbf{t}|\boldsymbol{\theta}, \boldsymbol{X})\right) = -\frac{1}{2}\mathrm{Tr}\left(\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta_i}\right) + \frac{1}{2}\mathbf{t}^{\mathrm{T}}\mathbf{K}^{-1}\frac{\partial\mathbf{K}}{\partial\theta_i}\mathbf{K}^{-1}\mathbf{t} \tag{4.32}$$

Everything needed for a gradient based optimization method is now in place. Note that using an ARD kernel can lead to some dimensions of the data being suppressed (assuming this suppression helps maximizing the likelihood). If the effect of some data dimension on the regression result is negligible after maximization, this can be viewed as a form of feature selection. This is a great strength of this method of regression, since feature weighting/selection can be performed as a part of the inference, and not as a separate step before the regression.

Larger data sets are becoming the norm, even for historical material. In paper IX, our training set was in the thousands. This becomes a problem with the inference described above, since that equation 4.32 contains an inverse of the covariance matrix. The inverse comes from evaluating the likelihood, shown in equation 4.31, so there seems to be no way around a cubic complexity of the inference algorithm. Considering that any inference algorithm might require hundreds of iterations, a computational complexity of $O(n^3)$ (where n is the number of data points in the training set) is way too much. A way of getting around this problem is using a so called sparse formulation of the GP. For many types of sparse formulations of GPs see the survey by Quinonero-Candela and Rasmussen (2005). In paper IX, a formulation using inducing

points were used. The inducing points are points in the same space as spanned by the training data that acts as replacements for the actual training data, in some aspects. All training data is included in the inference, but the full calculations are only performed for the inducing points. This leads to a computational complexity of $O(nm^2)$, where m is the number of inducing points. An added bonus is that this formulation also impacts the memory complexity, reducing it from $O(n^2)$ to $O(mn)$.

4.2.3 Deep learning

Artificial neural networks is not a new idea and have enjoyed periods of popularity in the past. However, it is only lately that the algorithms and computing power for a higher number of layers in the networks have been available, realizing some of the promises of earlier days[7]. State-of-the-art results are achieved in tasks like image classification (e.g. by Krizhevsky et al. (2012)) object detection (e.g. by Ren et al. (2015) and He et al. (2015a)), semantic segmentation (e.g. by Lin et al. (2015) together with conditional random fields) and word spotting (e.g. Wilkinson and Brun (2016)).

The basic idea of how a neural network is implemented is to stack many differentiable (at least approximately so) linear and non-linear transformations on top of each other. The differentiability of this pipeline (using the chain rule) makes it possible to optimize the function parameters to minimize some differentiable loss function. A simplified way of illustrating the process is shown in equation 4.33, where the loss function L encapsulates all other functions and f_1 is fed the image data.

$$L \circ f_n \circ f_{n-1} \circ f_{n-2} \circ \ldots \circ f_3 \circ f_2 \circ f_1 \tag{4.33}$$

The function L is the loss function. This function has to be a differentiable error metric, for comparing target variables with data. For the rest of the pipeline, functions performing linear transformations are paired with non-linear functions. The order and types of the functions used, is called the network architecture. Each transformation function is called a layer, due to their sequential nature. It is important to note that the data flow can split and merge between the input and the loss function, allowing for parallel specializations instead of one-type-fits-all architectures.

Common building blocks include:

Linear transformation Simple transformations, often formulated as matrix multiplications (e.g. fully connected layers) or convolutions. This type of layer often rescales the data. If the input is a grey scale image of 128×128, the output when using 16 convolution masks and a stride of

[7]Deep neural networks have really taken off during the time of my PhD-studies and at present, there is no real excuse for any image analysis group to not look at the very impressive results.

2 is $64 \times 64 \times 16$. The stride of a convolution is the number of steps between samples.

Activation function Step functions introducing non-linearity through thresholding or squishing input values (e.g. sigmoid, tanh, ReLU etc). The non-smoothness of the functions can make differentiation a problem. Either, the function itself is smooth and approximates a step function or the derivative is approximated.

Pooling An operation combining several elements of the input data giving a reduced output dimensionality, e.g. the maximum or average of small neighbourhoods of the input. Pooling can also be used to change the dimensionality of the output arbitrarily compared to the input, or allowing arbitrary input dimensionality as proposed by He et al. (2014a).

Normalization To either force the layer output to some form (e.g. softmax for categorical data) or batch normalization (to mitigate problems with vanishing gradients in very deep architectures, as proposed by Ioffe and Szegedy (2015))

Loss function A function introducing an objective to minimize. This is the error measuring function encapsulating the full pipeline of layers. The loss can be categorical (e.g. for classification through the use of softmax) or continuous (e.g. mean square error, used in paper IX).

The standard way of training a network is the backpropagation algorithm . In this, the data is fed to the network, resulting in some error (as measured by the loss function). Gradient information is then propagated back through the network, enabling the calculation of the gradient for all parameters in the network. However, this method is subject to practical problems like a vanishing or exploding gradient, e.g. due to the difficulty in differentiating the activation/step functions. Significant work go into handcrafting a good neural network pipeline. Though methods are being developed to learn architectures, as well as the network parameters, e.g. see Baker et al. (2016). A very important innovation for deeper models has been better training algorithms. For paper IX, we used a methods called ADAM, proposed by Kingma and Ba (2014) , for improving the training beyond the backpropagation with gradient descent algorithm.

At the time of writing, deep architectures are becoming the norm in document analysis applications[8]. For estimating the production date of historical material using neural networks, the only work so far is proposed in paper IX for handwritten material, and the work by Li et al. (2015) for printed material (post 16^{th} century). Our model for dating was pre-trained on the ImageNet dataset (120000 iterations) as an initialization. We did this for the network to take on characteristics important for image classification and reduce the need for taking training data out of our data set. The network used in paper IX for

[8] With the exception of forensic writer identification on small data sets, so far.

Layer type	Kernel size/stride	Output size
Convolution	7×7 / 2	$112 \times 112 \times 64$
Max pool	3×3 / 2	$56 \times 56 \times 64$
Convolution	3×3 / 1	$56 \times 56 \times 192$
Max pool	3×3 / 2	$28 \times 28 \times 192$
Inception(3a)		$28 \times 28 \times 256$
Inception(3b)		$28 \times 28 \times 320$
Inception(3c)	stride 2	$28 \times 28 \times 576$
Inception(4a)		$14 \times 14 \times 576$
Inception(4b)		$14 \times 14 \times 576$
Inception(4c)		$14 \times 14 \times 576$
Inception(4d)		$14 \times 14 \times 576$
Inception(4e)	stride 2	$14 \times 14 \times 1024$
Inception(5a)		$7 \times 7 \times 1024$
Inception(5b)		$7 \times 7 \times 1024$
Average pool	7×7 / 1	$1 \times 1 \times 1024$
Fully connected		1

Table 4.2. *An overview of the modified GoogLeNet architecture, originally proposed by Szegedy et al. (2014), for performing production date estimation in paper IX on the manuscript collection SDHK (sections 2.3 and 4.2.3).*

dating was based on the GoogLeNet architecture, proposed by Szegedy et al. (2014). The architecture from our paper is detailed in table 4.2.

Besides using a modified GoogLeNet for dating, we also evaluated a slightly older architecture called AlexNet, proposed by Krizhevsky et al. (2012), with reasonable results. However, switching to GoogLeNet cut the MSE evaluation score in half.In papers by Huang and LeCun (2006) and by Tang (2013), the authors used support vector machines together with convolutional neural networks. This work inspired us to use deep networks for feature learning together with both support vector regression (SVR) and Gaussian process regression in paper IX. The SVR did not increase performance compared to using only the neural network. The Gaussian process, however, could make improvements on the production date estimation when using smaller sets of training data. Some results from paper IX, with performance numbers, are presented in section 4.3.

4.3 Production date estimation performance

A comparison of the production date estimation results from papers VII–IX, can be found in table 4.3. This is a sample of results from their respective papers. The Gaussian process (GP) based estimator with a bag-of-features from paper VII is the first, showing that this task can be performed. After its presentation, inspired by the work at Google for dating modern printed text by Li et al. (2015), the method was extended to include transcription data. Also, to mitigate the effect of a particular choice of design parameters, an ensamble

of estimators was used. It was observed that the set of estimation outliers for one GP did not necessarily overlap with those of another GP estimator. In paper VIII, the ensemble of GPs was trained on different clustering for the bag-of-features code book to mitigate the effect of hyper-parameter and design parameter settings.

From the results presented in table 4.3, it is clear that the deep neural network (DNN) approach is successful, but needs more training data than the hand-crafted features with a GP does. However, there is room for a more thorough study, since several techniques for reducing the need for training data have not been systematically explored, e.g. synthetic data, augmentation etc. An important results is how using different network architectures changed evaluation results, with our reduced GoogLeNet reaching half the MSE as we did with AlexNet. The improvement from using more training data for the DNN leveled out when using more than 20% of the collection, while the GP leveled out slightly below 10%. The strength of the DNN is the flexible feature learning, while the GP excels in generalizing from small sets of data. A logical next step would be to integrate the two approaches more as it is unlikely that the best possible feature model is found by hand-crafting[9].

During the last couple of years, we have been two groups working on production data estimation. The work proposed in papers (He and Schomaker, 2014; He et al., 2015b; He and Schomaker, 2016b; He et al., 2016b,e; He and Schomaker, 2016a; He et al., 2016a; He and Schomaker, 2017b,a), have been very successful at estimating the production dates of a medieval charter collection from several cities in the Netherlands. Their data set Medieval Palaeographic Scale[10] (MPS), was recently released publicly. Since neither we nor them have been able to release our data sets publicly earlier, there is no fair comparison between the methods proposed in our production of papers, yet.

[9]Though in some ways, writer identification has proven to be an easy problem. Several hand-crafted feature sets can be found in the literature that are both descriptive and linearly separable in a standard euclidean space. However, there is little works, so far, on big data collections for writer identification or dating. So far, the study in paper IX is probably the one with the largest handwriting collection used for evaluation.

[10]http://www.ai.rug.nl/ lambert/MPS/

Estimator	Paper	Data type	Training set ratio	P25	P50	P75	MSE
GP	VII	Low	5%	7.9	18.3	36.8	1389
GP ensemble	VIII	Low	6%	8	17	30	810
GP ensemble	VIII	Low+text	7%	6	12	22	**462**
DNN	IX	High	10%	5.9	12.7	23.3	645
DNN + GP	IX	High	10%	5.9	13.0	23.7	640
DNN + Sparse GP	IX	High	10%	5.1	11.1	20.7	**550**
DNN + Global SVR	IX	High	10%	8.2	17.9	31.8	905
DNN + Local SVR	IX	High	10%	6.7	16.4	33.0	940
DNN	IX	High	20%	4.6	10.0	18.5	**469**
DNN	IX	High	30%	4.7	10.3	19.0	494
DNN	IX	High	40%	4.5	10.2	19.0	485
DNN	IX	High	50%	4.4	10.0	18.5	469
DNN	IX	High	60%	4.9	10.7	20.1	505
DNN (AlexNet)	IX	High	60%	-	-	-	840

Table 4.3. *Selected evaluation results for the production date estimation in papers VII-IX, sorted by training set size. Noteworthy better results are in bold. The metrics presented are the* $25th$ *(P25),* $50th$ *(P50) and* $75th$ *(P75) percentiles of the absolute error together with the mean square error (MSE). The size of the training set is given as percent of the full set (i.e. 10992 charters). The label DNN (deep neural network) are results using a modified GoogleNet, with the exception of one entry where AlexNet is specified. The data set was SDHK (section 2.3) for all evaluations. The collection comes in two versions, low resolution (1,5 Mpixel) and high resolution (4 Mpixel), as specified in the table. Note that the approaches from the earlier papers are using the low resolution images, making a direct comparison to the DNN approach unfair. However, the results show that all methods perform well on the given data.*

5. Looking forward

5.1 Paper overview

Here follows an overview of the papers included in this dissertation. Each paper is summarized, with their respective major contributions to the field in italics. The Roman numerals match those used in the "list of papers" and throughout the text. Explanations for the data set abbreviations are found in chapter 2. For a summary of who contributed to the different parts of the papers, see the "Summary of contributions" at the top of the dissertation.

Paper I

In this paper, a *full pipeline for query-by-example word spotting* was presented. The input was images of book spreads, using a *automatic full segmentation* and the possibility to do *fast searching* using dynamic time warping for a template on *full text lines*, without word segmentation. The word spotting was *evaluated on the new C64 and C61* data sets.

Paper II

In this paper, the *novel line segmentation algorithm* (splitting a page image into smaller crops, one for each text line) from paper I was presented in detail and evaluated. The algorithm estimated the line segments using a projection based method and refined the estimates in several steps. The last step was based on a lowest cost path finding, given a penalty function, giving cut-outs of irregular shapes. The method was compared to a similar state-of-the-art approach using hand segmented pages from the manuscripts C61 and C64 (20 pages each). The proposed method showed improved capabilities of cutting apart lines with ambiguous overlaps. Previous methods had a tendency to arbitrarily cut apart connected components spanning several text lines.

Paper III

In this paper, an evaluation of the *relative importance of the Marti-Bunke features* was performed for word spotting, on the Washington database. Dynamic time warping was used for the experiment since the standard HMM approaches learn a weighting of the features. In conclusion, *vertical projection of the foreground pixels* was shown to be the most important, while lower gradient showed very little importance. Recognition accuracy increased by *removing some of the 11 features* in the standard set. Also, the heuristic pruning presented in the cited literature was shown to be unnecessary, increasing errors.

Paper IV
The standard features explored in paper III were shown to be noisy, based on a qualitative analysis. Inspiration form signal processing led to the hypothesis that *feature sequence noise could be filtered out*. The main contribution of this paper was to evaluate *Gaussian, median, bilateral and non-local means filtering* on the sequence of feature vectors extracted form a text line for recognition. The non-local means filter increased performance the most.

Paper V
In this paper, the work of the preceding papers was presented *for humanist researchers*. As a case study of the power of word spotting, words like ”och” and ”att” were searched for. These words are very common and show the variability of the scribe's hand even within a single page.

Paper VI
This paper presented an examination of computerized writer identification in comparison to human estimation of scribal hands. The writer feature was based on the quill feature (used as a comparison) but *extended with curvature* as a third dimension to the feature histogram. We presented a simple and *fast classification approach based on semi-supervised learning* to propagate labels in a point cloud of discrete probability densities. *Several distance/divergence metrics* were evaluated (euclidean, χ^2, additive-χ^2 and Jeffreys) for use in the relation graph for classification. Also, ways for extending the feature extraction scheme for *scale and rotation invariance* were developed both for lines and whole pages. The *evaluation on C61* showed a result very similar to that of a human, indicating that the feature approach was reliable from a palaeographic perspective. With novel feature extensions, the needed training data (always very expensive and hard to get) was reduced by approximately 70%.

Paper VII
In this paper, a *novel approach to manuscript dating* was proposed based on clouds of image patches. Instead of explicitly deciding what was interesting along an ink stroke, patches were samples along the edges. Using the shape context descriptor as a difference metric between patches, clusters were formed and a bag-of-features representation was created. The clusters were created using *unsupervised training*, opening for using more data in the feature learning than in the ensuing regression.

In the document analysis community, Bayesian non-parametric methods have, so far, rarely been used. The regression in this paper for mapping the feature vectors onto a *time-line is performed using a Gaussian process*. The new *data set SDHK*, containing close to 11000 manuscript images, was used for performance evaluation, showing a *final estimation on-par with a human expert*.

Paper VIII
This paper proposes extensions to the dating techniques presented in paper VII, with improved performance and ability to use text data. The estimation accuracy (in terms of years) was improved by training multiple Gaussian process regression estimators using a *varying number of clusters* in the bag-of-features representation of the underlying ink stroke shapes. Since the output of the regressions are Gaussian distributions over the time-line they already have information of uncertainty included and are therefore easy to combine. For the transcriptions, an *n-gram based feature set for text* was presented. The final results was an *ensemble of estimators* than halved the MSE, compared to paper VII.

Paper IX
For this paper, a *deep neural network was used for estimating the production dates of manuscripts*. In the literature, the performance of neural networks have been improved using support vector machines, for some tasks. In the paper, the network was trained to do date estimations on its own but also used as a *neural network feature extractor*, with a Gaussian process or support vector machine performing the production date estimation. The more big data friendly sparse Gaussian process formulation with a smaller set of points as stand-ins for the data was also examined. In the end, the neural network achieved state-of-the-art performance for higher numbers of training examples while the Gaussian process could improve the performance for limited amounts of training data.

5.2 Research potential in mapping SDHK

Having a metric of similarity between documents would be of tremendous help in interpreting the traces of the scribes in any collection. For the case of the SDHK charter collection (section 2.3), it could give us a map of the places of work and movements of the scribes in medieval Sweden. Today, we only know a few of the scribes by name, and can safely conclude (on palaeographic criteria) that small sets of manuscripts belong to single scribes. What we can not do, using only human labour, is to form reasonable hypothesises on influences between places or groups of scribes. The exciting research question touched upon in the second part of the dissertation is: how can we look "behind" the surface of the manuscript and see the scribe?

The next step for this line of research is to identify writer hands on a larger scale. The data available can be used for training, however, that is unlikely to be sufficient for good accuracy. For writer identification, the production dates could be used for weak learning, where the labels are not fully known. The inference can be constrained by defining charters separated sufficiently in time or space as different, on a sliding scale. Adding semi-supervised learning have

been beneficial in this type of problem before, like in paper VI. Another type of research that could be performed is exemplified by Thorpe and Alty (2015), where neurological disorders are studied using handwriting. The variations of the handwriting have a lot of information encoded. The problem is to decode it.

5.3 Writer identification

A natural extension to the work presented here concerning dating of manuscripts, is to change the regression to a classification and do writer identification. The supervised learning writer identification problem, is in essence trying to find a clustering corresponding to writing styles. However, it can also concern finding a distance metric between writer hands. This metric is learned on one set of writer hands and generalized to a disjunct set of writers hands. One such database is the CVL database, presented by Kleber et al. (2013), consisting of 311 writers contribution 5 or 7 pages each.

A kernel can be interpreted as a projection to a higher dimensional feature space where a linear separation between classes can be found. Working from this intuition, initial results indicate that a kernel can be learned for a classification task, where that covariance between points in the feature space can be interpreted as similarities in writing style (given some feature set). By adding a step function to the regression output of the Gaussian process (GP) as $t_i = \sigma(f_{\text{GP}}(\mathbf{x_i}))$ (following the notation in section 4.2.2) , the GP can be extended to solve this classification task.

Setting up a GP classification for a multi-class problem required some tweaking of the inference algorithm, but that is a story for a future paper. This approach looks promising and preliminary accuracy is around 80% for the 1-top measure commonly used in writer identification. This accuracy is achieved without tuning the feature extraction or attention mechanism to modern material. It is also, like our earlier work on dating, insensitive to scale differences and rotational invariant, properties that modern writer identification feature should have. It is not unlikely that the performance could be pushed higher if well-known features, like character slant, could be taken into account.

5.4 Big data and data driven research in digital humanities

With small and large libraries around the world taking digital photographing to heart, the accessible data will likely continue to increase exponentially. A lot of the work in document analysis have so far been restricted to smaller collections. This begs the question: How will the methods scale? This is of course an empirical question and something I can only speculate about. However, for

document analysis in the future, I think bigger collections should be the focus in method development (except for studies in very special circumstances, like very important texts, e.g. the Dead Sea scrolls, or uniquely obscure material).

As of yet, the reason why neural networks are so successful is an open question. Is is because of the number of parameters? Capability to learn from huge numbers of data? The GPU friendly formulation, making more and better training possible? Likely it is all three. In the future, we will need ways of encoding structure that is known to us into whatever machine learning model that is used in the digital humanities. The fields where DH can make an impact are far from new and a lot is already known. Methods need to be developed where the knowledge of a domain expert can be encoded while still retaining sufficient flexibility.

The reason why statistical models learned from data are rarely used in the humanities is not a lack of interest, it is that these methods have not proven useful yet due to the complexity of the tasks. With the developments in data collection and machine learning, we have an opportunity to see if this can be a thing of the past, or if it is still too hard to do.

The techniques and studies presented in this dissertation is an argument for that there is a lot left to be done. However, the future is bright when it comes to doing new, previously impossible, research in many fields of the humanities. I envision a future where applying for money to build a computer cluster will be as uncontroversial in literary science as it is in computer science. However, a word of warning, if fantastic promises never reach realization funding will disappear, indifferent of how good the actual results were.

References

Andersson-Schmitt, M. and Hedlund, M. (1989). *Mittelalterliche Handschriften der Universitätsbibliothek Uppsala: Katalog über die C-Sammlung: Bd. 2. C 51–200*. Almqvist & Wiksell International, Stockholm.

Arivazhagan, M., Srinivasan, H., and Srihari, S. (2007). A statistical approach to line segmentation in handwritten documents. In Lin, X. and Yanikoglu, B. A., editors, *Document Recognition and Retrieval XIV*, volume 6500 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*. SPIE-Intl Soc Optical Eng.

Asi, A., Saabni, R., and El-Sana, J. (2011). Text line segmentation for gray scale historical document images. In *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, HIP '11, pages 120–126, New York, NY, USA. ACM.

Astola, J., Haavisto, P., Heinonen, P., and Neuvo, Y. (1988). Median type filters for color signals. In *Proc. IEEE Int 1988. Symp. Circuits and Systems*, pages 1753–1756 vol.2.

Astola, J., Haavisto, P., and Neuvo, Y. (1990). Vector median filters. *Proceedings of the IEEE*, 78(4):678–689.

Åström, P. (2003). *Senmedeltida svenska lagböcker 136 lands- och stadslagshandskrifter, dateringar och dateringsproblem*. Acta Universitatis Stockholmiensis.

Avidan, S. and Shamir, A. (2007). Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3).

Ayyalasomayajula, K. R. and Brun, A. (2014). Document binarization using topological clustering guided laplacian energy segmentation. In *Proc. 14th Int Frontiers in Handwriting Recognition (ICFHR) Conf*, pages 523–528.

Ayyalasomayajula, K. R., Nettelblad, C., and Brun, A. (2016). *Feature Evaluation for Handwritten Character Recognition with Regressive and Generative Hidden Markov Models*, pages 278–287. Springer International Publishing, Cham.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Baker, B., Gupta, O., Naik, N., and Raskar, R. (2016). Designing neural network architectures using reinforcement learning. *CoRR*, abs/1611.02167.

Ball, G. and Srihari, S. (2009). Semi-supervised learning for handwriting recognition. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 26–30. Institute of Electrical & Electronics Engineers (IEEE).

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Blei, D. M. (2012). Probabilistic topic models. *Commun. ACM*, 55(4):77–84.
Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.
Blinchikoff, H. and Zverev, A. (1976). *Filtering in the Time and Frequency Domains*. Classic series. Institution of Engineering and Technology.
Bradley, A. P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159.
Brink, A., Smit, J., Bulacu, M., and Schomaker, L. (2012). Writer identification using directional ink-trace width measurements. *Pattern Recognition*, 45(1):162 – 171.
Brink, A. A. (2011). *Robust and applicable handwriting biometrics*. PhD thesis, University of Groningen.
Buades, A., Coll, B., and Morel, J.-M. (05). A non-local algorithm for image denoising. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02*, CVPR '05, pages 60–65, Washington, DC, USA. IEEE Computer Society.
Bulacu, M. and Schomaker, L. (2007). Text-independent writer identification and verification using textural and allographic features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):701–717.
Campbell, W. M., Sturim, D. E., and Reynolds, D. A. (2006). Support vector machines using gmm supervectors for speaker verification. *IEEE signal processing letters*, 13(5):308–311.
Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698.
Cappelli, A. (1928). *Lexicon Abbreviaturarum*. Verlagsbuchhandlung von J. J. Weber, Leipzig.
Cavnar, W. B., Trenkle, J. M., et al. (1994). N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175.
Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, 1(4)(4):300–307.
Christlein, V., Bernecker, D., and Angelopoulou, E. (2015). Writer identification using vlad encoded contour-zernike moments. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 906–910. IEEE.
Ciula, A. (2005). Digital palaeography: using the digital representation of medieval script to support palaeographic analysis. *Digital Medievalist*.
Derolez, A. (2003). The palaeography of gothic manuscript books : from the twelfth to the early sixteenth century. *Cambridge*.
Doetsch, P., Kozielski, M., and Ney, H. (2014). Fast and robust training of recurrent neural networks for offline handwriting recognition. In *2014 14th International Conference on Frontiers in Handwriting Recognition*, pages 279–284.
Eldridge, M., Nimmo-Smith, I., Wing, A., and Totty, R. (1984). The variability of selected features in cursive handwriting: Categorical measures. *Journal of the Forensic Science Society*, 24(3):179 – 219.
Epshtein, B., Ofek, E., and Wexler, Y. (2010). Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963–2970. Institute of Electrical & Electronics Engineers (IEEE).

Espana-Boquera, S., Castro-Bleda, M. J., Gorbe-Moya, J., and Zamora-Martinez, F. (2011). Improving offline handwritten text recognition with hybrid hmm/ann models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):767–779.

Fernández-Mota, D., Lladós, J., and Fornés, A. (2014). A graph-based approach for segmenting touching lines in historical handwritten documents. *International Journal on Document Analysis and Recognition (IJDAR)*, 17(3):293–312.

Fischer, A., Frinken, V., Fornés, A., and Bunke, H. (2011). Transcription alignment of latin manuscripts using hidden markov models. In *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*, HIP '11, pages 29–36, New York, NY, USA. ACM.

Fischer, A., Keller, A., Frinken, V., and Bunke, H. (2012). Lexicon-free handwritten word spotting using character hmms. *Pattern Recogn. Lett.*, 33(7):934–942.

Franke, K. and Srihari, S. N. (2008). *Computational Forensics: An Overview*, pages 1–10. Springer Berlin Heidelberg, Berlin, Heidelberg.

Frinken, V., Fischer, A., Manmatha, R., and Bunke, H. (2012). A novel word spotting method based on recurrent neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 34(2):211–224.

Gatos, B., Stamatopoulos, N., and Louloudis, G. (2010). Icfhr 2010 handwriting segmentation contest. In *Frontiers in Handwriting Recognition (ICFHR), 2010 International Conference on*, pages 737 –742.

Gatos, B., Stamatopoulos, N., and Louloudis, G. (2011). Icdar2009 handwriting segmentation contest. *International Journal on Document Analysis and Recognition*, 14:25–33. 10.1007/s10032-010-0122-8.

Gilissen, L. (1973). L'expertise des écritures médiévales: Recherche d'une méthode avec application à un manuscrit du xie siècle: le lectionnaire de lobbes, codex bruxellensis 18018.

Graves, A. and Schmidhuber, J. (2009). Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552.

Greening, C. M., Sagar, V. K., and Leedham, C. G. (1995). Automatic feature extraction for forensic purposes. In *Fifth International Conference on Image Processing and its Applications, 1995.*, pages 409–414.

Guo, G., Fu, Y., Dyer, C., and Huang, T. (2008). Image-based human age estimation by manifold learning and locally adjusted robust regression. *Image Processing, IEEE Transactions on*, 17(7):1178–1188.

Haji, M. (2012). *Arbitrary Keyword Spotting in Handwritten Documents*. PhD thesis, Concordia University.

He, K., Zhang, X., Ren, S., and Sun, J. (2014a). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 346–361. Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2015a). Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

He, S., Samara, P., Burgers, J., and Schomaker, L. (2014b). Towards style-based dating of historical documents. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 265–270. IEEE.

He, S., Samara, P., Burgers, J., and Schomaker, L. (2016a). Discovering visual element evolutions for historical document dating. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 7–12. IEEE.
He, S., Samara, P., Burgers, J., and Schomaker, L. (2016b). Historical document dating using unsupervised attribute learning. In *Document Analysis Systems (DAS), 2016 12th IAPR Workshop on*, pages 36–41. IEEE.
He, S., Samara, P., Burgers, J., and Schomaker, L. (2016c). Historical manuscript dating based on temporal pattern codebook. *Computer Vision and Image Understanding*, 152:167 – 175.
He, S., Samara, P., Burgers, J., and Schomaker, L. (2016d). Image-based historical manuscript dating using contour and stroke fragments. *Pattern Recognition*, 58:159 – 171.
He, S., Samara, P., Burgers, J., and Schomaker, L. (2016e). A multiple-label guided clustering algorithm for historical document dating and localization. *IEEE Transactions on Image Processing*, 25(11):5252–5265.
He, S. and Schomaker, L. (2014). Delta-n hinge: rotation-invariant features for writer identification. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 2023–2028. IEEE.
He, S. and Schomaker, L. (2015). A polar stroke descriptor for classification of historical documents. In *International Conference on Document Analysis and Recognition*. IEEE.
He, S. and Schomaker, L. (2016a). Co-occurrence features for writer identification. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 78–83. IEEE.
He, S. and Schomaker, L. (2016b). General pattern run-length transform for writer identification. In *Document Analysis Systems (DAS), 2016 12th IAPR Workshop on*, pages 60–65. IEEE.
He, S. and Schomaker, L. (2017a). Beyond ocr: Multi-faceted understanding of handwritten document characteristics. *Pattern Recognition*, 63:321–333.
He, S. and Schomaker, L. (2017b). Writer identification using curvature-free features. *Pattern Recognition*, 63:451–464.
He, S., Wiering, M., and Schomaker, L. (2015b). Junction detection in handwritten documents and its application to writer identification. *Pattern Recognition*, 48(12):4036–4048.
Howe, N. R. (2012). Document binarization with automatic parameter tuning. *International Journal on Documant Analysis and Recognition*. to appear; DOI: 10.1007/s10032-012-0192-x.
Huang, F. J. and LeCun, Y. (2006). Large-scale learning with svm and convolutional for generic object categorization. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 284–291. IEEE.
Huang, X., Acero, A., and Hon, H.-W. (2001). *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition.
Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.
Jolliffe, I. (2002). *Principal Component Analysis*. Springer Series in Statistics. Springer.

Jurafsky, D. and Martin, J. H. (2000). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Kane, S., Lehman, A., and Partridge, E. (2001). Indexing george washington's handwritten manuscripts. *Center for Intelligent Information Retrieval, Computer Science Department, University of Massachusetts, Amherst, MA*, 1003.

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kleber, F., Fiel, S., Diem, M., and Sablatnig, R. (2013). CVL-DataBase: An off-line database for writer retrieval, writer identification and word spotting. In *2013 12th International Conference on Document Analysis and Recognition*, pages 560–564. IEEE, Institute of Electrical & Electronics Engineers (IEEE).

Klemming, G. (1883). Heliga birgittas uppenbarelser. *Samlingar utgivna av Svenska Fornskriftsällskapet 14:5. Stockholm. Svenska fornskriftsällskapet*.

Kovalchuk, A., Wolf, L., and Dershowitz, N. (2014). A simple and fast word spotting method. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 3–8. IEEE.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

Levenshtein, V. I. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707.

Li, F.-F. and Perona, P. (2005). A bayesian hierarchical model for learning natural scene categories. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 524–531 vol. 2. Institute of Electrical & Electronics Engineers (IEEE).

Li, Y., Genzel, D., Fujii, Y., and Popat C., A. (2015). Publication date estimation for printed historical documents using convolutional neural networks. In *The 3rd International Workshop on Historical Document Imaging and Processing (HIP'15)*.

Likforman-Sulem, L., Zahour, A., and Taconet, B. (2007). Text line segmentation of historical documents: a survey. *Int. J. Doc. Anal. Recognit.*, 9:123–138.

Lin, G., Shen, C., Reid, I., et al. (2015). Efficient piecewise training of deep structured models for semantic segmentation. *arXiv preprint arXiv:1504.01013*.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Louloudis, G., Gatos, B., and Stamatopoulos, N. (2012). Icfhr 2012 competition on writer identification challenge 1: Latin/greek documents. In *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*, pages 829–834. Institute of Electrical & Electronics Engineers (IEEE).

Louloudis, G., Gatos, B., Stamatopoulos, N., and Papandreou, A. (2013). Icdar 2013 competition on writer identification. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1397–1401. Institute of Electrical & Electronics Engineers (IEEE).

Macqueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

Manmatha, R. and Croft, W. B. (1997). Word spotting: Indexing handwritten archives. *Intelligent multimedia information retrieval*.

Mårtensson, L. (2011). Studier i am 557 4to. kodikologisk, grafonomisk och ortografisk undersökning av en isländsk sammelhandskrift från 1400-talet. *Reykjavík*.

Mårtensson, L., Wahlberg, F., and Brun, A. (2015). Digital palaeography and the old swedish script: The quill feature method as a tool for scribal attribution. *Arkiv för nordisk filologi*.

Marti, U.-V. and Bunke, H. (2001). Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(01):65–90.

Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., Pinker, S., Nowak, M. A., and Aiden, E. L. (2011). Quantitative Analysis of Culture Using Millions of Digitized Books. *Science*, 331(6014):176–182.

Moore, G. E. et al. (1998). Cramming more components onto integrated circuits. *Proceedings of the IEEE*, 86(1):82–85.

Mori, G., Belongie, S., and Malik, J. (2005). Efficient shape matching using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(11):1832–1837.

Nain, D., Haker, S., Grimson, W. E. L., Cosman, Jr., E. R., Wells, III, W. W., Ji, H., Kikinis, R., and Westin, C.-F. (2002). Intra-patient prone to supine colon registration for synchronized virtual colonoscopy. In *Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention-Part II*, MICCAI '02, pages 573–580, London, UK, UK. Springer-Verlag.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer New York.

Ntirogiannis, K., Gatos, B., and Pratikakis, I. (2014). Icfhr2014 competition on handwritten document image binarization (h-dibco 2014). In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 809–813. IEEE.

Otsu, N. (1975). A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27.

Pal, U. and Datta, S. (2003). Segmentation of bangla unconstrained handwritten text. *Document Analysis and Recognition, International Conference on*, 2:1128.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pettersson, E., Megyesi, B., and Nivre, J. (2014). A multilingual evaluation of three spelling normalisation methods for historical text. *Proceedings of LaTeCH*, pages 32–41.

Piotrowski, M. (2012). Natural language processing for historical texts. *Synthesis Lectures on Human Language Technologies*, 5(2):1–157.

Plötz, T. and Fink, G. (2009). Markov models for offline handwriting recognition: a survey. *International Journal on Document Analysis and Recognition (IJDAR)*, 12(4):269–298.

Porwal, U. and Govindaraju, V. (2013). Semi-supervised framework for writer identification using structural learning. *Biometrics, IET*, 2(4):208–215.

Pratikakis, I., Gatos, B., and Ntirogiannis, K. (2013). Icdar 2013 document image binarization contest (dibco 2013). In *2013 12th International Conference on Document Analysis and Recognition*, pages 1471–1476. IEEE.

Quinonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Rasmussen (2006). *Gaussian Processes for Machine Learning*. MIT University Press Group Ltd.

Rath, T. M. and Manmatha, R. (2003a). Features for Word Spotting in Historical Manuscripts. In *International Conference on Document Analysis and Recognition*, pages 218–222.

Rath, T. M. and Manmatha, R. (2003b). Word image matching using dynamic time warping. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–521–II–527 vol.2.

Razaka, Z., Zulkiflee, K., Idris, M. Y. I., Tamil, E. M., Noor, M. N. M., Salleh, R., Yaakob, M., Yusof, Z. M., and Yaacob, M. (2008). Off-line handwriting text line segmentation : A review. *IJCSNS International Journal of Computer Science and Network Security*, Vol. 8 No. 7:12–20.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99.

Rothacker, L., Rusinol, M., and Fink, G. A. (2013). Bag-of-features hmms for segmentation-free word spotting in handwritten documents. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1305–1309. IEEE.

Rusiñol, M., Aldavert, D., Toledo, R., and Lladós, J. (2015). Efficient segmentation-free keyword spotting in historical document collections. *Pattern Recognition*, 48(2):545–555.

Rusiñol, M. and Lladós, J. (2010). Efficient logo retrieval through hashing shape context descriptors. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, pages 215–222, New York, NY, USA. ACM.

Saabni, R. and El-Sana, J. (2011). Language-independent text lines extraction using seam carving. *Document Analysis and Recognition, International Conference on*, 0:563–568.

Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49.

Schomaker, L. and Bulacu, M. (2004). Automatic writer identification using connected-component contours and edge-based features of uppercase western script. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):787–798.

Sculley, D. (2010). Web-scale k-means clustering. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 1177–1178, New York, NY, USA. ACM.

Shafait, F., Keysers, D., and Breuel, T. M. (2008). Efficient implementation of local adaptive thresholding techniques using integral images. In *Electronic Imaging 2008*, pages 681510–681510. International Society for Optics and Photonics.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions. *CoRR*, abs/1409.4842.

Tang, Y. (2013). Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239*.

The GPy authors (since 2012). GPy: A gaussian process framework in python. `http://github.com/SheffieldML/GPy`.

Thorpe, D. E. and Alty, J. E. (2015). What type of tremor did the medieval 'tremulous hand of worcester'have? *Brain*, page awv232.

Tibell, K., Spies, H., and Borga, M. (2009). Fast prototype based noise reduction. In Salberg, A.-B., Hardeberg, J., and Jenssen, R., editors, *Image Analysis*, volume 5575 of *Lecture Notes in Computer Science*, pages 159–168. Springer Berlin Heidelberg.

Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on*, pages 839–846. IEEE.

Torralba, A. and Efros, A. A. (2011). Unbiased look at dataset bias. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 1521–1528.

Trentin, E. and Gori, M. (2001). A survey of hybrid ann/hmm models for automatic speech recognition. *Neurocomputing*, 37(1–4):91 – 126.

van der Zant, T., Schomaker, L., and Haak, K. (2008). Handwritten-word spotting using biologically inspired features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1945–1957.

Van Oosten, J.-P. and Schomaker, L. (2014). A reevaluation and benchmark of hidden markov models. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 531–536. IEEE.

Voigtlaender, P., Doetsch, P., and Ney, H. (2016). Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In *15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016, Shenzhen, China, October 23-26, 2016*, pages 228–233.

Wahlberg, F. and Brun, A. (2012). Graph based line segmentation on cluttered handwritten manuscripts. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1570–1573. IEEE.

Wahlberg, F. and Brun, A. (2013a). Feature space denoising improves word spotting. In *Proceedings of the 2nd International Workshop on Historical Document Imaging and Processing - HIP '13*, pages 59–66. ACM, Association for Computing Machinery (ACM).

Wahlberg, F. and Brun, A. (2013b). Feature weight optimization and pruning in historical text recognition. In *International Symposium on Visual Computing*, pages 98–107. Springer, Springer Nature.

Wahlberg, F., Dahllöf, M., Mårtensson, L., and Brun, A. (2011). Data mining medieval documents by word spotting. In *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing - HIP '11*, pages 75–82. ACM, Association for Computing Machinery (ACM).

Wahlberg, F., Dahllöf, M., Mårtensson, L., and Brun, A. (2014a). Spotting words in medieval manuscripts. *Studia Neophilologica*, 86(sup1):171–186.

Wahlberg, F., Mårtensson, L., and Brun, A. (2014b). Scribal attribution using a novel 3-d quill-curvature feature histogram. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 732–737. IEEE, Institute of Electrical and Electronics Engineers (IEEE).

Wahlberg, F., Mårtensson, L., and Brun, A. (2015). Large scale style based dating of medieval manuscripts. In *Proceedings of the 3rd International Workshop on Historical Document Imaging and Processing*, pages 107–114. ACM.

Wahlberg, F., Mårtensson, L., and Brun, A. (2016a). Large scale continuous dating of medieval scribes using a combined image and language model. In *Document Analysis Systems (DAS), 2016 12th IAPR Workshop on*, pages 48–53. IEEE.

Wahlberg, F., Tomas, W., and Anders, B. (2016b). Historical manuscript production date estimation using deep convolutional neural networks. In *International Conference on Frontiers in Handwriting Recognition (ICFHR), October 23-26, 2016, Shenzhen, China.*

Wiktorsson, P.-A. (2015). *Skrivare i det medeltida Sverige*. Skara stiftshistoriska sällskap.

Wilkinson, T. and Brun, A. (2015). A novel word segmentation method based on object detection and deep learning. In Bebis, G., Boyle, R., Parvin, B., Koracin, D., Pavlidis, I., Feris, R., McGraw, T., Elendt, M., Kopper, R., Ragan, E., Ye, Z., and Weber, G., editors, *Advances in Visual Computing: 11th International Symposium, ISVC 2015, Las Vegas, NV, USA, December 14-16, 2015, Proceedings, Part I*, pages 231–240. Springer Science, Cham.

Wilkinson, T. and Brun, A. (2016). Semantic and verbatim word spotting using deep neural networks. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 307–312.

You, C. H., Lee, K. A., and Li, H. (2009). An svm kernel with gmm-supervector based on the bhattacharyya distance for speaker recognition. *IEEE Signal Processing Letters*, 16(1):49–52.

Zhang, H., Berg, A., Maire, M., and Malik, J. (2006). Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2126–2136. Institute of Electrical & Electronics Engineers (IEEE).

Zhu, X. (2005a). Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.

Zhu, X. (2005b). *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University.

Acta Universitatis Upsaliensis

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 1475

Editor: The Dean of the Faculty of Science and Technology

A doctoral dissertation from the Faculty of Science and Technology, Uppsala University, is usually a summary of a number of papers. A few copies of the complete dissertation are kept at major Swedish research libraries, while the summary alone is distributed internationally through the series Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology. (Prior to January, 2005, the series was published under the title "Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology".)

Distribution: publications.uu.se
urn:nbn:se:uu:diva-314211

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2017