INTRODUCTION

1 FROM MONKS TO THE MONK SYSTEM

In the current computer-age, a large part of our communication is born digital. However, there are still large paper-based archives that contain handwritten materials such as letters, royal decrees, diaries and much more. These materials are usually historically relevant. For example, the important Qumran collection, also known as the Dead Sea scrolls, is a collection of handwritten texts that are among the oldest manuscripts included in the bible. A more recent, but also historically relevant example is a collection from the Dutch National Archive, the Cabinet of the King (van der Zant et al., 2009, 2008b): This collection contains Dutch royal decrees such as appointments of judges, rulings and laws from around the 1900s.

Archives that maintain collections such as the Cabinet of the King or the Qumran scrolls often have the important task of providing access to their collections. The historical relevance of the materials attracts many scholars and people from the general public interested in the contents of the documents. Because these archives are vast (the Cabinet of the King fills about 3km of shelf-space with handwritten documents), it is cumbersome to find information: An interested researcher usually has to consult multiple indices to find documents that are relevant to his or her query. The subject of this thesis is how to build a search engine for historical handwritten document collections and its building blocks. Using a search engine for handwritten pages reduces
the amount of work needed to find a page containing relevant information. Moreover, the knowledge about the contents and how to read the material is preserved in the search engine itself—Knowledge that until now is mainly stored in the minds of historians and archivists.

Another important reason to study handwriting recognition is the task itself of reading historical documents. This is a challenge for human readers, let alone for machines: Unfamiliar scripts and unknown abbreviations and shorthand make it difficult to decipher the information that a document contains. The learning of reading skills is interesting to Artificial Intelligence researchers from two perspectives: First, the human intelligence perspective is interesting because it teaches about learning and pattern recognition in biology. Secondly, the machine intelligence perspective provides the challenge of getting a machine to read.

To study handwriting recognition from the machine intelligence perspective, the Monk system (Schomaker, 2016) was developed. At its core, Monk is a search engine for handwritten documents (van der Zant et al., 2009) that makes many collections in many different scripts, from many historical periods by many writers accessible: From the Cabinet of the King, to personal communication from the interbellum, medieval collections of jury verdicts,
Figure 1.2: An example of a page from the Leuven alderman scroll collection (1421 A.D.). The alderman scrolls document the proceedings of law and the fines that citizens had to pay. When the fines were paid, and the case was closed, a mark was made through the record, indicating that the debt was settled. This example features some challenges a handwriting researcher has to face: abbreviations, strike troughs, writing between lines, ink smudges and old paper.

Chinese and Arabic letters as well as a collection of fragments from the Dead Sea scrolls. See Figure 1.1 for an illustration of the amount of scans available in Monk. These collections can be quite difficult to read properly, due to, e.g., degraded ink, abbreviations that are no longer in use or strike troughs. Figure 1.2 shows an example of such difficult material.

Unfortunately, handwriting recognition techniques are often tested on relatively clean datasets, such as the isolated digit collection MNIST (Collobert et al., 2006; Bhowmik et al., 2011)

Figure 1.3: An example from the well-studied Washington collection. Please notice the crisp letters, consistent, clear writing style and quality of the background, which are in contrast with Figure 1.2
and the neatly written letters by president George Washington (see Figure 1.3 and Fischer et al., 2012; Wei et al., 2013). The application of techniques that work well for these academic datasets on the collections in Monk proved difficult. It was not straight-forward to reproduce the results that were reported in the literature. Also, for most methods an extensive amount of labelled data is necessary, which is unavailable for newly scanned historical documents. The problem of starting from scratch is called the bootstrapping problem. Solving this problem was one of the design goals of Monk. We will study bootstrapping and a number of other issues that we encountered while trying to reproduce the results from the literature and during the development of Monk.

A big issue that prevented a rapid growth of labelled data in the early stages of the Monk system was the slow progress of annotation. At first, some experiments were performed for line retrieval (Schomaker, 2007), but a text line is not a natural object for search. For these experiments, a line-based web-interface was used to annotate line by line, starting at the first page. This means that pages that have been completely annotated are also fully searchable. However, unseen pages cannot be used as input for the machine learning methods. This can be an issue for instance when writing styles change over time. Furthermore, this does not use the full potential that the computer has to offer. A switch was therefore made to an approach based on data-mining, where segmented words, presented in a hit-list interface, can be annotated throughout the entire collection.

By engaging the human annotators differently—by annotating through a hit list instead of transcribing text line by line—we realised that humans can be involved in machine-based handwriting recognition in different ways: 1. By developing the techniques that learn from observations (machine learning), 2. by designing feature extraction methods that transform the written words into mathematical vectors to be used by the machine learning, and 3. by providing the labels: the knowledge of which word is depicted on an image—the ground truth.
Figure 1.4: A common way of representing the handwriting recognition process. A handwritten document is segmented into lines and words. Each word is then pre-processed and transformed into a feature vector using a feature extraction method. Machine learning can be applied to these feature vectors. The ground truth for the machine learning methods is provided by the human labelling process. The segmentation and pre-processing steps are outside of the scope of this thesis.

In this thesis, we argue that the machine learning methods should not get the singular focus of the handwriting recognition research community. There is also a need to further develop the fields of labelling and feature design. The goal of this thesis is to study the assumptions in all three aspects of involvement such that we can improve the handwriting recognition process. The next section will give an overview of the topics that are studied in this thesis and show how the different aspects of human involvement are related to each other.

2 HUMAN INVOLVEMENT IN THE HANDWRITING RECOGNITION PIPELINE

Figure 1.4 shows the handwriting recognition process as it is frequently presented: As a pipeline. The first step is to pre-process a document and segment it into individual word images. These are then transformed into a feature vector: a robust representation suitable for numeric computation. Together with a dataset of
manually labelled word images, a machine learning method can be applied to learn how to classify unseen word images.

The first step of the pipeline, the pre-processing and segmentation, is important. Because this is the first step, all other steps can be affected by errors here (this is related to the concept “garbage in, garbage out”). There have been numerous studies on segmentation-free methods (Rothacker and Fink, 2015; Almazán et al., 2014; Lorigo and Govindaraju, 2006) to prevent errors in segmentation to have such a cascading effect. Recurrent neural networks such as LSTMs can operate without segmentation beforehand. However, even if the input can be processed without segmentation in the first step, a post-processing step that does the segmentation—usually in ASCII-space by specifying codes for blanks, line endings and paragraphs or by introducing blank tokens between repeated characters (Bluche et al., 2015; Hannun, 2017)—is still required.

A common argument against segmentation is overcommitment: The errors made in the segmentation step can not be corrected in the later stages of the pipeline. For image retrieval, overcommitment is not a big issue because of over-segmentation. This means that a word-image can have many overlapping word zone candidates, as shown in Figure 1.5. The assumption is that the correct word zone candidate will be ranked higher in a hit list because it is more prototypical (see also Chapter 4). Oversegmentation looks expensive, but the number of word zone candidates, typically in the dozens or a few hundreds, is much smaller than the total number of horizontal pixel positions along the x-axis, which is currently typically in the order of several thousands.

The main focus of this thesis will not be on segmentation and pre-processing, but on the other three steps in the pipeline: Machine learning, feature extraction and labelling. Specifically, we will discuss how we, as researchers, can be involved in the handwriting recognition process. In the design and development of a large, trainable retrieval engine for handwriting such as Monk, we stumbled upon several common assumptions that either proved to be misleading or required a twist in order to be useful for
Figure 1.5: Monk uses over-segmentation for finding word zone candidates. Each line below the words indicates the width of a number of word zone candidates. The word zone candidate corresponding to the second line then shows the word “Jan”, the abbreviation of “January”. This word zone candidate will rank very high in the “Jan” hit list.

obtaining effective retrieval and recognition performances. These assumptions will be treated in the following sections.
2.1 Assumptions in machine learning algorithms

The first part of the pipeline that we will examine in this thesis is the machine learning process. Specifically, we will look at common assumptions in hidden Markov models (HMMs) and support vector machines (SVMs). Even though they have gained considerable attention, artificial neural networks and deep learning are not studied in depth. HMMs and SVMs are interesting because they are strong classifiers and have been very popular in the handwriting recognition field.

Training HMMs is a non-trivial problem because observations only give indirect evidence of the hidden states. This means there are some—well-known—issues with training HMMs. Most notably, the algorithm will optimize towards a local optimum (a sub-optimal solution), not necessarily to the global optimum. Unfortunately, the phenomenon of local optima itself is not the subject of many studies, while this is such an important property of the Baum-Welch training method.

We study the local optima to get a better understanding of how and when they are reached. It is interesting to get a global perspective on the training of HMMs because this will give more insights into how models behave. We compare models with known parameters to trained models. The assumptions that are examined are related to this issue: Are models that are close to the global optimum better than models that are further away?

2.2 Assumptions on features

Hidden Markov models learn both the structure and the observations that provide the partial evidence of the structure. The observations are usually features that are extracted from, in our case, images of handwritten words. The assumption related to features that we test in this thesis is that the underlying structure is considered to be more important than the observations.

We test this assumption by answering two questions. First, can we find the structure we know is present in the data by only using observations? And secondly, can we still perform classification
when we remove temporal structure from a model? Again we use models and data with known parameters for comparison with trained models and calculation of the classification accuracy of the trained models.

We are interested in seeing the influence the transition probabilities have on the classification accuracy, and how important the observed features are. This insight is useful in the discussion of where to focus our engineering effort: On the part that learns the hidden parameters or the observations, i.e., on the Machine Learning part of the pipeline, or on the feature extraction.

2.3 Assumptions on the origin and availability of labels

The final assumptions that we will study are related to labels. Labels provide the ground truth that allows the machine learning to update the parameters and generalise to unseen data. There are two assumptions to be studied related to the labels. First, there is the assumption that there is a dataset available that is properly labelled. Generally, researchers use existing, academic datasets or collect labels outside of the recognition process. Secondly, there is the assumption that the handwriting recognition process is static, as described in Figure 1.4. Instead, we consider the process to be a loop like in Figure 1.6 instead, incorporating all elements in a continuous learning cycle.

This loop is facilitated by using a hit list interface. Word images are divided into different classes and then ranked such that the images at the top are most likely to be correct and useful for the labelling process. A human annotator can then easily select the word images that are correctly labelled and update the label store with many labels at once. The classifier and ranking method are then retrained which updates the hit lists for the annotator, allowing for even more labels to be added to the system, and yielding a snowball effect.

The final assumption discussed in this thesis is then the assumption that both classifying and ranking should be done by the same mechanism. We believe that there are actually two func-
Figure 1.6: Overview of how we consider the process to be a loop instead of a static pipeline. The hit list is an important concept that contains a list of retrieved images for a certain class that can be easily and quickly labelled by a human annotator. Each update is used by the retrieval engine to update the classification and ranking methods, which provides better hit lists. This way, a snowball effect can occur.
3 Research Questions

The main focus of this thesis is the relationship between the three aspects of human involvement discussed above. Because the handwriting recognition community has a strong focus on the machine learning aspect of handwriting recognition, it is argued in this thesis that especially the labelling part of the handwriting recognition loop has been neglected. The main, general research question is related to this argument.

General Research Question

Where can we have the most impact on the results of a search engine for historical handwritten documents? Should the attention be focused on improving the machine learning methods, the feature engineering methods or the labelling methods? To answer this question, we will look at the individual parts of the handwriting recognition process. For each aspect we consider the more concrete questions.

Research questions related to Machine Learning

Since handwriting concerns patterns of variable width (analogous to variable duration in speech) a predominant model consisted of hidden Markov models, that were well studied in the literature since 1988. However, there are a few assumptions to be aware of when using HMMs. The models are trained using the Baum-Welch algorithm, which is a type of Expectation Maximisation (EM) algorithm. These types of algorithms are used when certain parameters of a model are not directly observable. This is exactly the case in HMMs: The observations only give a partial evidence of the underlying structure. Of course, this is a difficult problem to solve: How can we model a set of parameters for which we only have indirect evidence?

The Baum-Welch training algorithm works by randomly creating a model (the initial model) and updating it iteratively. After a number of iterations, or until the updates have become too small, this process stops. The resulting model (the learned model)
can be stuck in a local optimum. A relatively straightforward method to deal with local optima is to train multiple models and choose the highest performing one. Even though there have been some studies on how to converge to the global optimum (for example, Lee and Park, 2006; Siddiqi et al., 2007, but see Chapter 2 for a more in-depth discussion), this “restarting” scheme is still a popular method to deal with local optima.

The following questions are intended to study the phenomenon of local optima:

- Is there a relation between the distance from a learned model to the global optimum and the final performance of the learned model?

- Is there a relation between the distance of the initial, random model and the distance of the final, trained model?

The assumptions that are questioned here are (a) that the closer a model is to the global optimum, the better a model should perform, and (b) that the closer the initial, random model is to the global optimum, the closer to the global optimum that model will end up after training.

**Research questions related to Feature Extraction**

Since hidden Markov models are aimed at modelling time series, an important component of such models is the transition matrix. This matrix defines the probability for switching from one state to another and is therefore the temporal part of the model. However, typically, the underlying temporal information is not directly observable: The observations only give partial evidence of the underlying state.

The general structure of the transition probabilities is called the topology and indicates which state transitions are allowed and which are not. For example, the Bakis topology only allows transitions from a state $S_j$ to state $S_{j+1}$ or to itself, as shown by the topology diagram in Figure 1.7. Of course, from the observations we cannot directly determine whether the structure
is organised in a Bakis topology or something else, but it is very common to force the models in handwriting recognition to have such a topology (Zimmermann and Bunke, 2002; Bunke et al., 1995; Britto et al., 2001).

The observation probability distributions provide the other important part of the models. They model the probabilities that a certain feature will be observed in a certain state. The observations are features that are extracted from the raw image pixels—usually an abstraction such that they are robust representations of the (sub-) characters or words.

The questions related to Feature Extraction are intended to study the relation between transition and observation probabilities:

- Since the observations only provide partial evidence for the underlying states, can we learn the topology (the general structure) of the transition matrix from observations alone?
- Since HMMs model time series, and the temporal information is such a central part of the models, is classification performance reduced when temporal information is removed from hidden Markov models?

**Research questions related to Labelling**

The Monk system implements both classification and searching methods to provide access to historical documents. However, the use of the support vector machine (SVM), a strong classifier, for ranking images in search results, showed unintuitive results in the top of the search results. This was surprising because
the SVM generally reports high accuracy on image classification tasks. Since we also use these lists to solicit valuable feedback from the users, the top of the list is very important: With one press of a button, the top results can easily be confirmed to be correct.

This hit-list based approach to labelling enables the snowball effect. By providing more labels, Monk increases its performance, which in turn generates a better hit list that makes it easier to label more images. This effect creates jumps in the number of labelled images, which remind us of phase transitions in physical systems. Figure 1.8 shows an example of a collection in the Monk system with these jumps in the number of labels. This effect is not possible in a left-to-right annotation process, because there is no mechanism to use feedback to speed up the labelling process.
The following questions were raised while observing the counter-intuitive results in the top of hit lists, and are related to how to effectively gather good quality labels and achieve a snowball effect:

- Why are SVMs not suited for ranking handwritten word images?
- How do we get intuitive hit lists?

4 OUTLINE OF THE THESIS

This thesis has three chapters related to answering the research questions posed in the previous section. In Chapter 2 we address the machine learning questions by studying HMMs, while Chapter 3 discusses HMMs from a feature perspective. Finally, Chapter 4 brings these subjects together and adds the labelling perspective to answer the question of how to get intuitive hit lists.

The last chapter, Chapter 5, concludes the thesis with a summary of all the findings and a discussion of the main subject of the thesis: loops instead of pipelines. We also discuss the connection to methods such as active learning and deep learning.