The snowball principle for handwritten word-image retrieval
van Oosten, Jean-Paul

DOI:
10.33612/diss.160750597

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2021

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):

Copyright
Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license. More information can be found on the University of Groningen website: https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment.

Take-down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): http://www.rug.nl/research/portal. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Download date: 03-03-2024
The subject of this dissertation is the development of search engines for historical handwritten document collections. Techniques such as machine learning and image representation are often being studied in the field of handwriting recognition to improve the accuracy of handwriting recognition systems. Usually, in order to have fair comparisons, standard datasets are used for the performance evaluation. While most techniques work very well on datasets such as MNIST or the letters by president George Washington, their application is much less straight-forward on collections with more difficult material such as those of the alderman scrolls of the city of Leuven, as discussed in Chapter 1.

The Monk system (Schomaker, 2016) is a search engine and data mining tool for historical document collections. Many different collections are available in Monk, ranging from rather neat collections, written by a single writer, to very difficult-to-read collections from medieval times. These collections are usually interesting to humanities researchers or even the general public. Due to the nature of these collections, we have observed some notable differences between applying machine learning techniques on the neat, academic datasets, and the raw datasets that are being made public by archives worldwide (see for example Figure 1.2 on page 3 and Figure 4.6 on page 67).
One of the techniques we observed with different results between academic benchmarks and historical material is hidden Markov modelling (Chapters 2 and 3), which has a long history in handwriting recognition systems as well as other applications. HMMs model time series using a hidden state transition probability matrix and per state a model of the observation probabilities. The application of HMMs on the difficult datasets in Monk has not resulted in accuracy scores that are expected based on the literature. We performed two studies to understand the differences in performance. The first study deals with the training method while the second study concerns the understanding of the different components of these models.

1.1 Baum-Welch training of HMMs

The process of training HMMs with Baum-Welch is well known for its tendency to get stuck in local optima, resulting in less-than optimal models. The idea that sparked the study in Chapter 2 is that the initial model, which is randomly selected to start the training, has a big impact on the final performance. We therefore studied whether models converge to the global, rather than a local, optimum if the starting point is already closer than other, non-converging models. To study the training procedure, we generated an artificial data set: A randomly chosen model with known properties generated data sequences on which new HMMs can be trained. The model that is used to create the dataset is then considered the global optimum for this particular set and can be compared to the trained models.

The main conclusion from the experiments in Chapter 2 is that the $\chi^2$ distance of a trained model to the global optimum is not a good predictor of likelihood. At first glance, this is surprising because one would expect that models that are similar to the global optimum would also show a performance similar to the model at the global optimum. Furthermore, the experiments show that it is very hard for the Baum-Welch training algorithm to converge to a point close to the global optimum without guidance. However, when we know either the state transition probabilities or the observation probabilities beforehand, the
resulting models are very close to the global optimum. This is relevant because it shines some light on the conditions in which HMMs will perform well.

1.2 The relative importance of transition vs. observation probabilities

When we know either the state transition probabilities or the observation probabilities, we can “clamp” these parameters, which means that they are fixed while still training the other properties. This greatly reduces the number of parameters to be learned. The fact that clamping has a big positive effect on the ability of the training algorithm to find the global optimum, raises an interesting question: What is the most critical design element of an HMM? The transition probability matrix $A$ or the observation probabilities $B$? This is the main question in Chapter 3, and is also studied by using generated data from a known model. This time, the model has a very specific topology: The Bakis topology dictates a diagonal structure of the transition matrix, leaving most probabilities in the matrix $P_{ij} = 0$. It is shown that the Baum-Welch training algorithm has difficulty finding this clear diagonal structure. Furthermore, using “real” data, extracted from handwritten word images, it has been shown that removing all temporal information (and clamping the transition matrix to a uniform distribution) from the models, does not result in as drastic a drop in performance as one might expect.

The main conclusion from Chapter 3, that the observation probabilities seem to have a larger impact on the model performance than the transition probabilities, has some interesting implications for a search engine for handwritten documents. It means that special attention should be given to the representation of word images. One of the main challenges is the bootstrapping phase of a collection, where there are very few labelled instances. In these cases, it is very hard to automatically learn the best representation of a word image, let alone train both representation and classification at the same time like in deep learning. It makes sense to create the feature representation and selection methods by hand in cases where there are very few labelled images.
The discussion in the first two chapters has been around machine learning and representation. We have discussed why it is not straightforward to learn the parameters of a Markovian process and that the representation of handwritten word images is important for HMMs. However, in order to learn from known examples and to generalize to unseen examples, it is important to have a large amount of labelled images, i.e., tuples of \((\text{class label}, \text{word image})\). Collecting these tuples can be quite labour-intensive. Frequently, machine learning experts presume the existence of labelled data and only focus on either the (deep) machine learning or the feature representation of the word images. Labelled data has not been a big problem in the studies on HMMs in Chapters 2 and 3 of this dissertation because the data was mostly generated artificially, and the classes were therefore known. However, the extensive datasets of handwritten word images had to be labelled by hand to create a proper ground truth.

In a system such as Monk, new datasets are added on a regular basis. These new datasets do not contain labels yet and are very diverse in script-type and picture quality, making it difficult to build off other document collections. Transfer learning (Pan et al., 2010) would generally be useful in these cases, as long as the source and target domains are related, i.e., the feature spaces are not too different. However, the differences between collections in the Monk system are quite large. This means that it is required to ‘bootstrap’ new collections: starting with a small number of labels and quickly building the necessary body of knowledge about a collection. By aiding the human annotator, it should be possible to gain momentum in the labelling process. Quick accumulation of new labels, often in a group of related classes, also known as a snowball effect, is something that is implemented in the core processes of the Monk system. The snowball effect is usually marked by sudden jumps in the number of added labels (see Figure 1.8, page 14.)

In Chapter 4, a solution for the problem of bootstrapping and quickly building a large database of labelled data is explored.
that uses hit lists. This works by generating a list of images a classifier determines to belong to a particular class. The system then ensures that the images that are most likely to be correct are ranked at the top. The annotator uses this fact in the hit-list interface to label yet unknown images quickly by marking the top $n$ images to be correct. This method has been shown to allow the sudden jumps in number of added labels. Correcting a label provides a large amount of new knowledge to the system as well: Correcting mistakes either suggests new classes or exposes a confusion between two existing classes.

Another method for selecting instances that should be labelled for the largest impact on the classification performance is active learning (Settles, 2009; Baum and Lang, 1992). The selection process in active learning is based on the idea of finding the images for which the classifier has the least evidence of belonging to a certain class (i.e., has the most confusion). This works well for discriminative learning, because at the decision boundaries, new knowledge has the biggest impact. However, a different approach might be better suited for Bayesian or generative methods, where each class is represented by its own model and the classification is solved by a ‘winner takes all’ principle (usually by applying the argmax function on the probabilities per class). In this case, the better a single model represents samples of that class, the higher this model will end up in the final ranking. See also Figure 5.1. This is related to the concept of density weighting in active learning (Settles and Craven, 2008) where the instances to be queried are weighted by their distances to all other known instances, but this does not necessarily take the class into consideration.

Hit lists have been effective in labelling new word images. A few guidelines can be established for building an efficient snowball effect. First, the images should be ranked from most likely to least likely belonging to the class of the hit list. This ensures that the top $n$ instances can quickly be assessed on their correctness. Secondly, the interface should allow for quickly labelling a large number of images at once, preferably by selecting the first $n$ images and accepting the label the classifier has assigned to these
Discriminative models (a) are separated by a decision boundary. The objective is to get this boundary as accurate as possible. Active learning works well here because it samples around the current decision boundary. Generative models (b) consist of multiple models for each class, in this case represented by a Gaussian. Applying active learning here would mean sampling around the area between the classes, skewing the distribution per class.

Figure 5.1: Discriminative models (a) are separated by a decision boundary. The objective is to get this boundary as accurate as possible. Active learning works well here because it samples around the current decision boundary. Generative models (b) consist of multiple models for each class, in this case represented by a Gaussian. Applying active learning here would mean sampling around the area between the classes, skewing the distribution per class.

images. Thirdly the images in the hit list that are not of the correct class, should be ‘intuitive’, such that they are not totally different from the correct instances of that class. Apart from the difficulty in explaining very obvious mistakes to the user, labelling mistakes have a much smaller impact when the differences are small. Furthermore, classes that are related to each other can then be spotted and labelled as such more easily. Using a hit list-based approach is preferable to a linear process, where word images are annotated from left to right, top to bottom, because a hit list allows the user to inspect many occurrences of a word in a single screen. This aids the recognition of misclassifications and allows the annotator to scan instead of typing in every label by hand.

In order to get hit lists with quality results in the top ranks, a two-step process is needed. This process alternates between classification and ranking. Each phase can use their own machine learning and representation methods. The two-step process, together with an increasing amount of labelled data, will improve the hit lists over time and yield even more useful labels. This way, we can ensure that the right method for the right job is used, but also that once one method does not yield enough
new labels any more, a different method can be used instead. This interplay between different feature extraction methods and iterated application of building hit lists can be compared to the Fahrkunst elevator\(^1\) in ancient mines, which uses an alternation of steps to get at a higher level.

\section*{3 \textsc{Loops and Snowballs, Not Pipelines}}

The observation of the snowball effect, marked by sudden jumps in number of added labels, leads to the idea that handwriting recognition should not be a simple, one-directional pipeline, but a loop. This loop has been described in Chapter 1 (see Figure 1.6 on page 10). By defining the handwriting recognition process as a pipeline, the larger feedback loop is ignored, and it will be harder to bootstrap new, challenging manuscripts with a new script type or different vocabulary. This means that even though Machine Learning and the representation of the word images is very important, all elements, including the ground truth, are important and should be considered in any system that uses handwriting recognition techniques for retrieval or classification.

In this dissertation, we have studied the different aspects of the handwriting recognition loop. We can consider the effect of human feedback on these aspects. The first aspect we studied is the algorithmic level in Machine Learning. Human feedback here improves the algorithm, which can be applied on many different problems. However, the improvement is usually tested on specific use-cases or on standard academic datasets (e.g., the letters by President Washington or the MNIST handwritten digit dataset). Furthermore, the results in Chapter 3 suggest that an improvement to the overall system performance may be achieved more efficiently by focussing on representation, rather than learning the underlying process parameters. Currently, a lot of studies are focussing on deep learning methods, that are especially interesting because these methods train both classifiers as well as representation. These methods will be discussed in Section 4 below.

\footnote{\textit{1} Also known as the man engine.}
The representation of the data is the second aspect that we studied where human feedback can be applied. Improvements in this area are often tightly coupled to the dataset that they are developed for. This frequently leads to the phenomenon of ‘One PhD, One book’ where the efforts of, e.g., a graduate student leads to accuracy improvements for only a single book or at best a single collection. However, human effort on this aspect does have a large impact on model performance, especially because it is sensitive to the book or collection that it is applied to. We believe there is still a need of handcrafted features: The two-step process discussed in Chapter 4 allows changing the feature extraction method to apply the best method for a collection or even a class, in order to ‘harvest’ the most new labels. Transfer learning (Pan et al., 2010) is a very interesting technique that might be very relevant for achieving high accuracy on new datasets, however the datasets in Monk can be very different from one another, in handwriting style as well as the material (such as the paper, ink deposition method, etc.), and may warrant a different representation method.

The third and final aspect of human feedback that we discussed is that of providing labels. Labels are what drives the handwriting recognition loop. Any effort to improve labels, improves the knowledge of the system as a whole. The more knowledge is embedded in a system, even more and better algorithms can be applied. This means that different techniques, both at machine learning as well as the representation level, should be used for new collections than for collections that have many more labels, classes and writing styles available. This knowledge is even more tied to the collection than representation is, even though some efforts have been made to use the knowledge of one collection for bootstrapping another. The general principle, as well as any techniques to apply these ideas, can be applied to any collection.

4 DEEP LEARNING

The topic of Deep Learning, or techniques related to it, was outside the initial scope of this dissertation. However, since a couple of years Deep Learning has gathered much support in the
community of AI and Machine Learning. Rightfully so, given the breakthroughs and number of competitions that have been won by Deep Learning methods. For example, in (Sanchez et al., 2016) all the participants used a deep learning method in one shape or form. The first major competition where Deep Learning made a significant jump in performance was in the ImageNet competition (Krizhevsky et al., 2012). Given the popularity of Deep Learning, it is a subject that should be addressed in this dissertation as well.

The core topic of this dissertation is the general feedback loop in a search engine for handwritten document collections. In Chapter 4 and this chapter, we have made the case for a method that is relatively independent of the specific machine learning or representation method. Deep learning seems like it would fit in this framework, but it also seems that it is an opposing framework: Deep Learning trains both classifier and representation simultaneously.

There is much excitement surrounding deep learning since the results and quality of the models are very impressive. However, there are also reasons to be cautious. The downside is that you need a large amount of labelled data. Furthermore, the fact that you do not have to create a separate representation method does not imply that there is no “manual labour” needed anymore. Designing the network architecture and tuning all the hyperparameters is still largely done by hand and can be very labour intensive. Also, training a character-based classifier is susceptible to the same pitfalls as those mentioned in Chapter 4 (see for example Figure 4.6 on Page 67).

The increasing interest in Deep Learning models however seems to underscore the conclusions from Chapters 2 and 3, that using HMMs for handwriting recognition purposes is not straightforward and may have fundamental issues. Figure 5.2(a) shows the relative interest in HMMs versus LSTMs on Google: Since 2016 the number of search queries for LSTMs has overtaken the number of queries for HMMs. Both are methods for modelling time series, where the main advantage of LSTMs is that they allow for a longer history to be used in calculating the probability of the
next observation. Furthermore, the Deep Learning community is very effective in transferring new ideas in one field of study to others. For example, the attention mechanism for LSTM networks (Bahdanau et al., 2014; Doetsch et al., 2016) allows even more precision in aligning the inputs to each part of the output sequence, even though it is not necessarily developed for the handwriting recognition community.

When there is enough training data to train a combined classifier and representation method, the results tend to exceed the more traditional methods where representation is hand-crafted and classifiers such as SVMs or HMMs are used. However, the feedback loop will be a lot slower: The large training set and large number of model parameters make the training of an LSTM or Convolutional Neural Network for example, computationally expensive. The flexibility of creating separate representation and classification methods is especially helpful to gain “phase transitions” in the feedback loop.

Finally, the introspection methods described in Chapters 2 and 3 can, with some adjustments, also be applied to Deep Learning methods. Opening black boxes of classifiers is gaining popularity, especially to build trust in the methods. The LIME method (Ribeiro et al., 2016) aims to be a classifier-agnostic method. There are a number of introspection methods for neural networks as well (e.g., Olah et al., 2018; Barratt, 2017). It still seems interesting to look at deep learning methods from a global perspective, similar to the way we have looked at the HMMs. This would help with a better understanding of strengths and weaknesses of the backpropagation training method.

5 Conclusion

The discussions in this chapter on the different aspects of handwriting recognition do not necessarily only apply to the handwriting recognition field; The conclusions can have implications across disciplines and methods. A focus of this dissertation is the inclusion of the labelling of images in the entire process, and to consider this process to be a loop instead of a fixed pipeline.
Labelled data is important—for both scientific research as well as industry—because it drives the feedback loop and therefore allows for improvements in both accuracy and the methods, either machine learning or representation.

One thing that we have noticed while working on the Monk system is that exploration is very useful. The Monk system uses this idea to trigger phase transitions by switching ranking and classification methods. This is especially useful when a certain combination of these methods is no longer effective in getting new labels from annotators. At the same time, we have seen that the diversity in scientific research has declined: Deep Learning, or neural networks in general, has been gaining popularity. A quick study of abstracts and keywords of the 2018 edition of the ICFHR conference shows that 55% of the papers mention something related to deep learning or LSTM, up from 17% in 2014, see Figure 5.2(b). However, we believe that the field in general benefits from exploration as well.

Exploration can take many forms. From the Monk system, we have seen the idea of exploring different classification and ranking methods and Chapter 4 shows that it is useful to separate these two functions. Another parameter to explore is the representation of the images. A deep learning method is usually not ideal for these separate stages since the representation and classification method are usually deeply interconnected, and because retraining takes a lot of time. However, the Monk system still allows a user to select different methods to use in the sorting of hit lists. Finally, inspiration can also be taken from biology. While the idea of a neural network is biologically inspired, the implementation is usually very engineering oriented.

An interesting biologically inspired approach can be found in Hawkins and Blakeslee (2007); Hawkins et al. (2018). Hawkins is working on a dual mission to both “reverse engineer” the neocortex, as well as creating software based on the current level of understanding of the neocortex. The approach is interesting because it is an exploration of unsupervised learning of hierarchical representations and predictions. It would be very interesting to see these theories applied to handwriting recognition, as a way
Figure 5.2: (a) Comparison of the number of searches on Google on “Hidden Markov Model” versus “LSTM”. The y-axis represents the number of searches relative to the highest number of searches in a given month—in this case: November 2018 with the highest number of searches for “LSTM”. Data by Google Trends. (b) Percentage of abstracts and keywords in ICFHR articles that mention either a) LSTM, CNN, Deep Learning or Neural Networks, b) HMMs, or c) SVMs.
of exploring a different direction from gradient descent-based methods.

Besides a focus on methods and tools, there can be feedback on other aspects of the writing as well. An interesting future research area would be to focus on giving feedback on a sentence, page or even book level. Feedback on these levels would help with identifying relations between words and their position. Feedback on semantics would help in understanding rather than just classification.

The main ideas of this dissertation are a) that human feedback can be injected in several aspects of the pipeline, b) that we should consider handwriting recognition to be a loop instead of a pipeline and finally, c) that by taking advantage of the loop, a snowball effect can be achieved. Therefore, the advice is to invest in a system of getting more and better labels in order to increase the gain in the feedback loop, which has a positive effect on both machine learning and (learned) representations as well.

While this dissertation focussed on techniques such as HMMs and SVMs, using the described framework, it is possible to collect enough labels to further study “label-hungry” methods such as deep learning because this framework is method agnostic. We even believe that this advice is not limited to the handwriting recognition field. Having proper, labelled data is crucial to many machine learning and pattern recognition applications across many fields and industries.