Active Learning for Historic Handwritten Text Recognition

Konstantinos Tsakalis
St. Edmund's College

UNIVERSITY OF CAMBRIDGE

A dissertation submitted to the University of Cambridge in partial fulfilment of the requirements for the degree of Master of Philosophy in Machine Learning, Speech and Language Technology

University of Cambridge
Department of Engineering
Trumpington Street
Cambridge CB2 1PZ
United Kingdom

Email: kt430@cam.ac.uk

August 10, 2016

Declaration

I, Konstantinos Tsakalis of St. Edmund's College, being a candidate for the M.Phil in Machine Learning, Speech and Language Technology, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

Total word count: 10,049

Signed: [illegible]

Date: 12 August 2016

Abstract

This thesis examines the use of active learning for the task of handwritten text recognition in historical documents. Active learning is a machine learning paradigm which enables the learner to select the data that is being trained on. In domains where procuring annotated data is expensive but there are large amounts of unlabelled data available, active learning can lead to better models with the same annotation effort. We apply different active learning algorithms to the problem and note notable improvement over random sampling. We discuss suitable distance measures for handwritten lines and use them to improve active learning performance. We report annotation cost saving up to 18% even with the simpler application of active learning.

Acknowledgements

I would like to thank my supervisor, Professor Bill Byrne for his guidance throughout this task. I am also obliged to Felix Stahlberg for providing the baseline recognition system and his fruitful advice.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recognition Systems for handwritten historic text documents are an emerging research topic. Currently, large amounts of these documents are being published by on-line digital libraries worldwide in the form of raw digital scans. Having transcriptions of these scanned images significantly increases the usefulness of them since it allows indexing, easier access and faster reading of them.

There are many challenges associated with the task of obtaining such transcriptions, as the language is archaic, the optical input is distorted and the writing styles are unusual and diverse. Traditional Optical Character Recognition (OCR) techniques are not applicable since characters can not be isolated automatically in these images. Instead holistic, segmentation-free Handwritten Text Recognition (HTR) techniques have been developed. These HTR algorithms share methods and concepts from the area of Automatic Speech Recognition (ASR): such as Hidden Markov Models - Gaussian Mixture Models (HMM-GMM), neural networks and N-Gram language models. These machine learning algorithms require a significant amount of human annotated data to be trained on and their performance typically improves with increasing amount of training data. Usually, there is a large amount of unlabelled images, and a subset of those are selected randomly to be manually annotated and learn from. This methodology is called passive learning:

a passive learner receives a randomly sampled data set from the underlying population distribution and then learns a model. However, acquiring highly quality annotated data is an expensive task in terms of time and labour and is often the limiting factor for attaining better recognition performance. Hence, it is particularly valuable to determine ways to make the best use of the annotation budget available. Since there is a large collection of unlabelled documents available, instead of randomly selecting them to be manually labelled we can choose elaborately those that we believe will be most useful for the model to learn from. For example, some lines that are very similar to ones that the model has already "seen" and is good at recognising them, will not be as useful as new lines that the model is uncertain about.

Active learning is a machine learning paradigm that attempts to achieve this objective, of selecting the most informative examples from a pool of unlabelled data for annotation. The ultimate motive of using an active learner is to either reduce the annotation cost by achieving the same performance as random sampling with fewer data, or, for a fixed annotation budget attain better recognition performance.

Active learning, in general, can be valuable in domains where unlabelled data are readily available, but obtaining training labels is expensive. The domain for historical document HTR fits that description and thus we believe is important to investigate its applicability and performance. Ultimately we believe active learning can lead to better recognition models for historical documents with the same amount of effort and thus increase the uselessness of such models for this task. Furthermore since to the best of our knowledge, there has not been any work for active learning in HTR in any context, there is an additional value in investigating some aspects of active learning particular to this task.

The thesis is structured in five Chapters. Chapter 2 describes the baseline HTR system we are using and the details of the particular dataset. It also provides the necessary background for active learning in general. Chapter 3 discusses some of the available literature that is particularly relevant to this work. In Chapter 4 we present the algorithms and procedures that we choose to adapt to the task of historical documents HTR which are then being used to run experiments. Chapter 5 presents the results of these experiments and some discussion on the results is being done. Finally in Chapter 6, we summarise the findings, draws some conclusion, point some limitations and propose areas for future work.

Chapter 2

Background

In his section, we are presenting the necessary background that this work builds upon. We first discuss some details about the particular dataset we are using. We then describe the baseline handwritten text recognition system and finally we are doing an overview of some general concepts in active learning.

2.1 The Handwritten Text Recognition Task

2.1.1 The dataset

The dataset we are using for this task is the one provided for the English language in the tranScriptorium project [2] and in particular for the ICFHR2014 Competition on Handwritten Text Recognition [3]. It consists of a large set of manuscripts written by the renowned English philosopher Jeremy Bentham (1748-1832). The Bentham collection has more than 80,000 documents, most of them digitised.

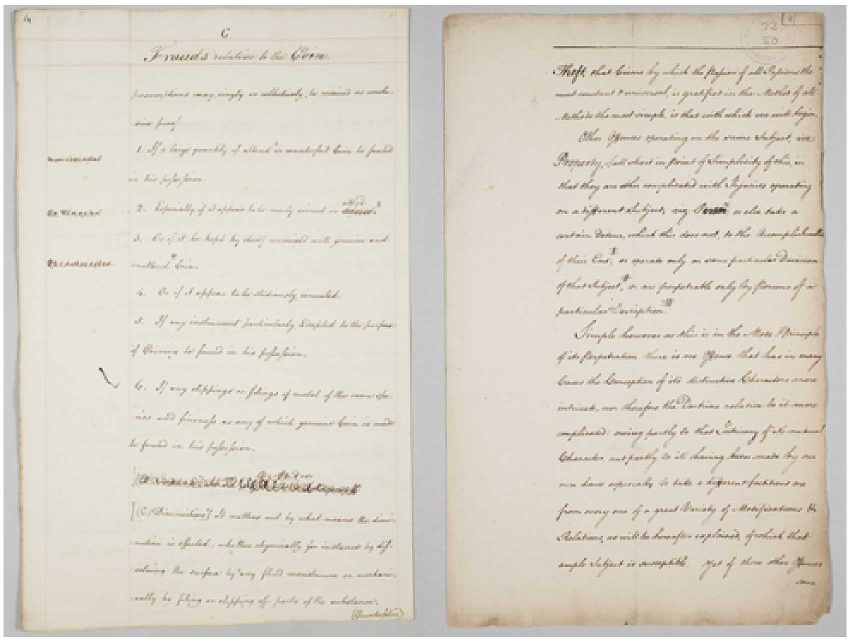
From the digitised documents, only around 6,000 have been transcribed by humans, which demonstrates the common phenomenon in tasks as these, of the abundance of un-transcribed documents. The documents have been written by various writers: Bentham himself and his secretaries. Figure 2.1 shows a sample of two pages of documents from this collection.

The provided dataset comprises of 433-page images however the transcription and recognition is done line-by-line. Thus each line has been extracted and paired with its corresponding transcription. The line extraction was done by a semi-automatic procedure by the dataset providers. Figure 2.2 shows an example of three extracted lines in the provided format.

The 433 pages of the corpus contain in total 11,537 lines with nearly 110,000 words and a vocabulary of more than 9,500 distinct words. These are split into a training, development, and test set, comprising of 350, 50 and 33-page images respectively, which corresponds to 9,198 1,415 and 860 lines in each set.

One of the peculiarities of this dataset is that the lines vary wildly in length. Figure 2.3 shows the histogram of the lengths of the transcriptions of each line. A significant number of lines have just one word. These are usually numbers, section headers, or inserted words. As we will discuss later this abnormal distribution of lengths has a particular importance in the context of active learning.

As the writing varies significantly across different lines, before training the recognition system, a number of preprocessing steps on the line images are necessary. The preprocessing performed by the baseline system is the same as the one performed in [4]. For each line, baseline estimation is performed so that deviations from the horizontal writing line can be corrected. Furthermore writing styles usually have different slants which is also estimated and normalised for all lines. All lines are rescaled to have the same height of 78 pixels while trying to preserve the aspect ratio. Finally, the variation in the writing thickness due to different pens is removed. Some sample lines from the dataset along with their corresponding normalised binary images are shown in Figure 2.2.



C

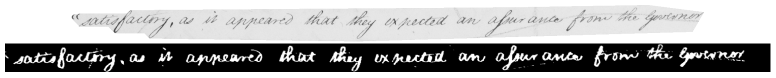
Frauds relative to the Coin.

[illegible]

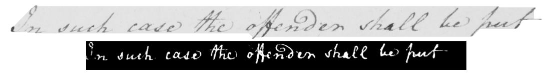
Theft [illegible]

[illegible]

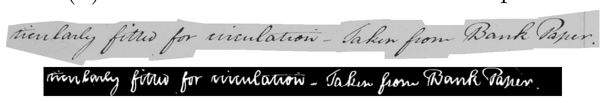
Figure 2.1: Two sample documents from the Bentham collection dataset



(a) "satisfactory, as it appeared that they expected an assurance from the governor



(b) In such case the offender shall be put



(c) ticularly fitted for circulation _ Taken from Bank Paper.

Figure 2.2: Sample line images and their binary normalisation, along with their transcriptions

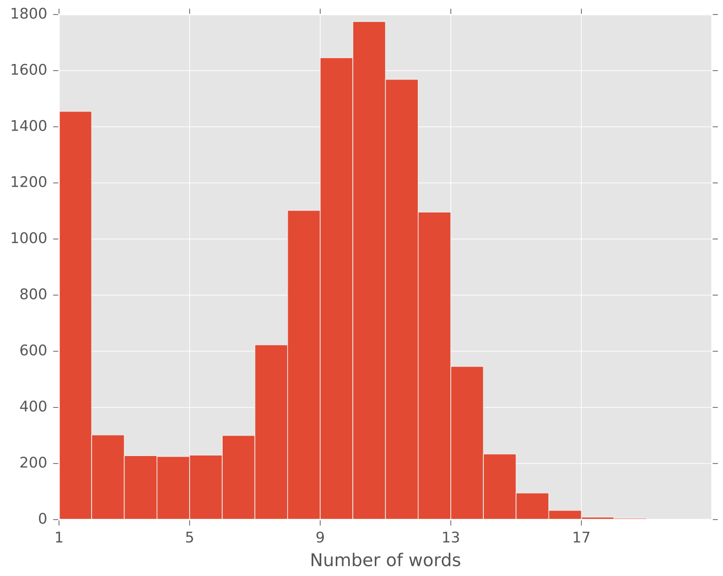


Figure 2.3: Sentence length histogram

2.1.2 The recognition system

The baseline handwritten text recognition system we use in this project is based on the speech recognition toolkit Kaldi [5]. Kaldi is based on recipes, i.e pipelines of algorithms, that here have been adapted to work with image sentences instead of speech utterances. In particular, for ASR the audio waveform is split into 10-15 ms chunks known as frames and a fixed dimension feature vector is extracted from each one of these. For the HTR task the lines which have been normalised to fixed height are split into windows with three pixel width, sequentially in the direction of reading order. The raw pixel values of the normalised image are used as features, resulting to a feature vector dimensionality of 234 (3*78) per window. The windows overlap by 1 pixel, i.e the feature extraction window is being shifted by 2 pixels at a time. The length of the sequence of the feature vectors is dictated by the width of the line image which due to the normalisation is almost directly proportional to the length of the sentence in characters.

The other parallel with ASR is regarding the recognition units. In ASR the statistical models aim to predict a sequence of phonetic units known as phones, where they can be context-dependent: different recognition targets according to the preceding and following phone. In the HTR task the basic units are characters and there is also the notion of context dependency in terms of the previous and next character as we discuss next.

For the particular handwritten recognition task there are 91 character units (alphanumerics and symbols). Since the way the characters are written changes depending on the preceding and the following characters the recognition is based on sequences of tricharacters. However there are 91^3 different such context-dependent characters which would make the training infeasible. Instead, the tricharacters are assigned to clusters based on their similarity. Of course, tricharacters belonging to the same cluster have the same central character. In this task the clustering usually results to around 350 context-dependent characters and the task for the optical model is to output a sequence of probability distributions over these for each line image. The optical model in the baseline system is a Long short-term memory (LSTM) neural network. The LSTM is trained on the aligned tricharacter transcriptions obtained from an HMM-GMM model.

The second part of the recognition system is the language model which is trained on the corpus consisting of the training lines and assigns probabilities to sequences of words. The particular system uses a 3-gram language model which models sequences of words including the two preceding ones.

The decoder is then provided with a lexicon: the words that occur in the training corpus and their character composition, the language model and the probabilities of the optical model over tricharacters for each line-image column. The decoder then maps the sequence of characters to a sequence of word hypotheses from the lexicon and assigns two scores to each word: the probability of the word's characters under the optical model and the probability of the word sequence under the language model. Even though Only the highest scoring sentences are considered but, the number of different hypothesised sentences is enormous. An efficient way to represent a great number of alternative theories is in a graph, know as lattice. A lattice provides a wider searching space than storing the N most likely sentences (N-best list) and also avoids the drawback of representing many very similar hypotheses in competing strings.

The recognition system training pipeline can be summarised in the following stages:

- The raw images are **binarized** and **normalized**.
- From each sentence a sequence of **windows with width 3px and stride 2px** are extracted.
- The linear transformed (PCA) raw features are used to train **character** based **HMM-GMM** systems.
- We gradually add **left and right context** and **delta features** and use LDA and MLLT for **per book adaptation**.
- The HMM-GMM model is used to train an **LSTM** that maps from the sequence of **raw pixels** to the sequence of distributions over the **clustered 'tricharacters' states**
- A language model is trained on the corpus consisting of the training lines
- The **decoder** gets scores from the **LSTM optical model** and from **the language model** and produces **lattices** over different hypotheses

The recognition system is trained and the decoding is run in parallel on 32 cores. The training of the LSTM is done on a NVIDIA GTX 980 GPU. Table 2.1 show some indicative running times with the available resources. The decoding times are linear with the number of lines being decoded and the LSTM training time is correlated with the number of training samples. Runtime of the training and prediction (decoding) pipelines is of particular importance in the context of active learning as we will discuss later.

Pipeline Stage	Duration
Data preparation	12 min.
Training stage 1 (PCA features, context-dependent GMM training)	38 min.
Training stage 2 (LDA+MLLT features, context-dependent GMM training)	15 min.
Decoding (LDA+MLLT, GMM)	54 min.
Training stage 3 (Speaker adaptive training)	8 min.
Decoding (SAT, GMM)	114 min.
Training stage 4 (LSTM training full dataset)	**440 min.**
Decoding 2000 lines (LSTM)	**102 min.**
Total	$\boxed{\textbf{783 min.}}$

Table 2.1: Training and prediction pipeline timings

2.2 Active Learning

In the previous sections, we presented the dataset that we are working with and the recognition system that is available. The aim of this project is to investigate how active learning can be used in this framework to utilise the large amount of un-transcribed data in task like this and maximise the return from the human annotation efforts. In this section we give a general overview of this general machine learning framework.

Actively learning, also know as "query learning", is a machine learning paradigm where the learner is allowed to choose the data on which it is trained on with the purpose of performing better with a smaller amount of labelled, training data. In contrast with passive learning where the performance is purely determined by the model, active learning emphasizes the role of the input selection.

In the most common setting, active learning is used to reduce the cost of manual annotation for producing training data. As demonstrated in [6] an increasing amount of training data almost invariably results to better generalisation performance. However, not all potential training samples are of the same usefulness toward that goal. The objective of an active learner is to select the most useful samples for annotation, such that for a fixed annotation cost attains better performance than if the samples were chosen randomly.

In a related manner, active learning can be useful for dataset compression, whereby a large training set with already available annotations is reduced in size in order to improve the training time [7].

In the most common setup of active learning know as pool-based, there is small set of labelled data $\mathcal{L}$ and a large pool of unlabelled data $\mathcal{U}$ available. The model is initially trained on the available labelled set, and we then select one or more informative samples from the unlabelled pool for an oracle to provide labels for and add them to $\mathcal{L}$. The model is then retrained on the new labelled pool, and the process is repeated until either we achieve a target generalisation performance, or a fixed annotation budget is reached.

Ultimately we want to select those samples that once labelled and added to the training data, minimise the expected generalisation error of the model on future test samples. In [8], Cohn et. al propose a statistically optimal framework to tackle this problem for the regression case. They construct queries that maximise the error reduction by minimising the learners variance exploiting the fact that for an unbiased learner they are equivalent. They obtain a closed form solution for the case of Locally Weighted Regression and Mixture of Gaussians Regression. However, a closed-form solution for the expected variance for more complex models is intractable to compute.

In the light of these limitations, in [9] they describe a Monte-Carlo approximation of the expected future error. In particular for classification tasks they estimate the expected cross-entropy loss, by sampling examples from the unlabelled pool. However this approach, requires retraining the model for each possible label of each sample. Although that can be tractable for some classes of models like Support Vector Machines that support incremental retraining, and some classification tasks with a small number of classes, it is a daunting task in the general case.

Since the optimal selection is intractable in practice, a plethora of active learning methods that optimise alternative, sub-optimal criteria have been developed and widely used. In these methods the selection of which samples to query for labels, is done based on metrics that assess how informative each sample x will be for the model via some measure $\phi(x)$.

One of the most common general algorithms for quantifying informativeness is uncertainty sampling, where the samples being queried are those that the model is most uncertain (least confident) about. The model that has been trained on the labelled pool is used to make predictions for the samples in the unlabelled pool. Then for the classification task the uncertainty measures stem from the resulting probability distributions over the classes.

Such measures can be the least confidence (LC):

$$\phi(x)^{LC} = 1 - P(y = c^*|x)$$

where c^* is the class wih the highest probability.

Another is the difference between the posteriors of the most probable and the second most probable labellings, know as margin (M):

$$\phi(x)^{M} = P(y = c^2 2|x) - P(y = c^1|x)$$

Finally the Shannon entropy can be used to quantify uncertainty on the prediction:

$$\phi(x)^{E} = -\sum_{i=1}^{M} P(y = c_i|x) \log P(y = c_i|x)$$

The entropy of a random variable represents the information needed to encode the distribution of its outcomes. It is effectively a measure of the flatness of the distribution and is widely used as an uncertainty measure in machine learning.

For sequence labelling tasks such as HTR where we only have scores for a subset of possible transcriptions, the least confidence and the margin can be calculated directly by using the scores of the most likely hypothesised sentences instead for the probability of classes. The fact that the recognition scores are not probabilities will not affect the ranking. However measuring the entropy of distributions over sequences can have different interpretations. One can define the token entropy which is the average over the image columns (or time frames) of the entropy of distributions over the characters (or phones):

$$\phi(x)^{TE} = \frac{1}{T} - \sum_{t=1}^{L} \sum_{m=1}^{M} P(y_t = m|x) \log P(y_t = m|x)$$

Taking the mean will discourage the selection of longer sequences which naturally have more entropy. However it might be beneficial to promote longer sequences and thus the total token entropy may considered:

$$\phi(x)^{TTE} = T\phi(x)^{TE}$$

Alternatively, we can measure the entropy of the sequence by considering only the N best hypothesised whole sequences. We first have to normalise the scores of the best hypotheses which only occupy a portion of the probability mass, so that they sum to one.

In [10], Settles et al. compare the performance of the aforementioned uncertainty measures and they conclude that the sequence performs best in most of the considered tasks.

Another active learning framework that targets to select informative samples is the query-by-committee approach. We have a committee of different models where each one them is trained independently on the same dataset. Each member of the committee is then used to do predictions for the unlabelled set. We consider the most informative query, to be the instance over which the committee is in most disagreement about how to label. There are again various ways to measure disagreement, such as vote entropy, Kullback-Leibler divergence and others as described in [10].

Other criteria to compute informativeness in a very principled manner such as Fisher Information, exist although they are intractable to compute for larger models.

It has been argued that uncertainty sampling and QBC tend to select outliers or noisy samples for labelling. Although the model may be very uncertain about such samples is unlikely that the transcription of them will be helpful in general. Thus along with informativeness some algorithms are also considering how representative of the data distribution the sample is. We will describe such work in Section 3.

The conventional active learning techniques over-viewed above nominally select only a single data instance at a time for manual labelling and retrain the model after every individual query. This can be prohibitive if the training time of the model is high, and also requires an oracle to provide the labels on-line, one at a time after each retraining. This is of limited use in a practical scenario which utilises services like the Amazon Mechanical Turk where a lot of oracles can provide labels simultaneously [11].

Since single instance selection strategies require expensive retraining for each instance labelled and cannot benefit from parallel labelling systems, many approaches have been proposed to do multi-instance selection, commonly known as batch mode active learning. Some researchers [12, 13] have simply used single instance selection strategies like the ones describes above to select more than one unlabelled instance at a time. For the case of uncertainty sampling that would entail selecting the top k most uncertain samples each time. However, this approach will not take into account the information overlap between the multiple instances selected. For example if the model is very uncertain about two samples that are almost identical, we will select both for labelling although we could do better if we select only one of these two, and a different one more diverse, even though we might be less uncertain for it.

To account and adjust for this information overlap a lot of the algorithms in the literature [14, 11, 15] introduce heuristic selection schemes based on some distance measure between the samples. For this reason a significant part of this project is to investigate such distance measure suitable for handwritten text.

I this section we gave an overview of some prominent approaches to active learning in general. In the next Chapter we describe in more detail some work which we believe is of particular relevance and applicability for the HTR task, on top of which we build the rest of the work in this thesis.

Chapter 3

Related Work

Although there have been some work on active learning for handwritten character recognition [18, 19] to the best of our knowledge there are no results for active learning in the task of handwritten text recognition. However as we described in Section 2.1.2 HTR is remarkably similar to ASR. For ASR there has been significant amount of research on active learners motivated by the fact that obtaining labelling data is also expensive for speech, while there is an abundance of un-transcribed audio.

Therefore in this section we review work that been done for speech recognition and could be adapted to our task, along with some general active learning work that we deem relevant and applicable.

In one of the first applications of active learning in ASR, Hakkani et. al [12], implement a simple form of uncertainty sampling based on confusion networks. They obtain confusion networks over words from the lattices and use the posteriors as a confidence score for each word. Then they propose different ways to combine the confidence scores for each word in the sentence such as arithmetic averaging, or multiplying them to obtain a per utterance uncertainty measure. Finally they select the k most uncertain utterances for annotation and retrain. Their selected batch size k is 4000 which quite large, and they do 12 iterations of incremental retraining. They report minor improvements in terms of WER against random sampling.

In [20] a very similar approach is followed but they use a different strategy to combine the per word confidence scores in a per utterance score. In particular a voting scheme with a threshold is introduced where only the words with confidence score above the threshold are contributing with score 1. Furthermore, in their experiments they use a much small batch size of queries per iteration, $k = 400$. They note a significant improvement over the results in [12].

Yu et. al in [15] proposed a unifying framework for both active and semi-supervised learning based on the concept of maximizing the lattice entropy reduction over the whole dataset. Heuristically, providing the transcription for the least confident utterances can provide the most information to the system and thus most of the standard active learning approaches select the least confident sentences. However they argue that selecting the utterances based solely on the confidence of the model can lead to the inclusion of outliers and noise corrupted utterances. The inclusion of those will not be useful for learning to transcribe other utterances rather than the included. However if an utterance that is similar to many others is included even thought we are less uncertain about it, it will be more beneficial. The core idea is to select the utterances that can provide the most benefit for the recognition of the utterances coming from the true data distribution which they consider to be represented by the unlabelled pool of samples. So for each utterance they estimate the total reduction in entropy over the whole pool, that would ensue if a particular utterance was to be annotated. For this a distance measure is necessarily to be defined between that utterance and all the others in the pool. In this work they used the KullbackLeibler divergence (KLD) between the two hypothesized lattices. The selection process however is a difficult optimisation problem as the inclusion of one sentence will affect the selection of others based on how similar they are. To accommodate for that after each is sentence is included they adjust (reduce) the entropies of all the other sentences in the pool according to how similar they are to the one selected. In effect this promotes diversity within the batch of selected utterances. They results they report are state of the are for active learning in ASR.

In [13] is being argued that the full lattice entropy is dependent on the utterance length. They propose instead the entropy of the distribution over the N-Best list of sentence hypotheses produced by a baseline ASR system, to be used as the uncertainty measure. Furthermore for the representative score of a sentence with respect to the data pool they use the average term-frequency (TF) - inverse term frequency (IDF) similarity of the utterance against all others in the pool. The informativeness and representativeness scores are combined through multiplication. The selection of the sentences to be annotated is then being done greedily without accounting for information overlap within the batch. In the results presented the N-Best entropy on its own shows an advantage against the full lattice entropy for pure Uncertainty Sampling, however the incorporation of the representatives measure resulted to only marginal improvement. Nevertheless this scheme is simpler and computationally cheaper to implement than [15].

In general, recent state of the art active learning algorithms are comprised of a distance measure, an uncertainty measure and a search scheme. In [11] the general active learning problem is casted as an optimisation problem by using a heuristically build objective, which can be optimised efficiently via sub-modular optimisation. Unlabelled instances are selected greedily by maximising a heuristically constructed objective. The objective is comprised from the uncertainty measure (entropy), the mean kernelized distance from all the labelled points in the training set, and the minimum distance of the candidate from the already selected points which quantifies the similarity of an unlabelled point from the already selected set. Although the present results for simple classification tasks we believe that this approach can scale to be applicable to HTR and ASR.

The approaches described above aim to extend the well established single-instance uncertainty sampling to batch mode active learning by using heuristics. Such heuristics aim to select samples according to some of these criteria: The samples should be informative for the recognition model [12, 20, 11, 13, 15] while being diverse enough so that their information overap within the batch is minimized [15, 11], and representative of the training set so outliers and noise are avoided [15, 13].

Some work has been done to formalise batch mode active learning and avoid use of heuristic measures. In [16] following a standard paradigm in semi-supervised learning, a good classifier is defined as one that achieves high expected log likelihood of the labelled data and low entropy on the prediction on unseen data (unlabelled pool). The former assures that the predictions are correct, while the latter that the classifier can generalise. The proposed instances at each iteration are obtained as the solution of an optimisation problem. However, this approach requires searching over all possible label configurations of the unlabelled set which is prohibitively expensive for problems with more than a few classes.

In the next chapter we take the ideas primarily from the most recent work in active learning for ASR [13] and [15], and describe how they are applicable in HTR. We also discuss some distance measures particularly for the HTR task.

Chapter 4

Design and Implementation

In this section we describe in detail the algorithms and the procedures we use in this work for the task of active learning in HTR. We are looking at the simpler uncertainty sampling with the N-Best list entropy, and then global entropy reduction. As the later requires some sort of a kernelised distance measure between handwritten lines, we discuss two such measures which are deemed appropriate for it.

4.1 N-Best List Entropy

As mentioned in Sections 2.2 and 3, one of the best uncertainty measures for the purpose of active learning is entropy. In the context of handwriting recognition where there is a huge number of hypotheses, there are different ways to measure entropy. Here we focus on the entropy derived from the N-best hypotheses and describe in detail how to obtain it. From the lattice L_i of each sentence we obtain the N-Best paths. Each path is a candidate transcription T_j and is accompanied with a score from the language model l_j and a score from the optical LSTM model a_j. The N-Best list is thus a list of tuples $\{(T_1, l_1, a_1) \ldots (T_N, l_N, a_N)\}$. A combined score for each path is simply the weighted sum of these two. In this work we only use a weight for the optical model to get the overall score as : $s_j = wa_j + l_j$. As the weights are used to select the best path for calculation of the WER, we can tune it by minimizing the error rate on the development set.

To form a valid probability distribution the scores of all N paths are normalised to sum to one : $p_j = \frac{s_j}{\sum_{n=1}^{N} s_n}$. Then the entropy of the N-Best list of each line is defined as:

$$H_i = -\sum_{i=1}^{N} p_i \log p_i$$

The active learner selects for labelling the k lines with the highest entropy.

4.2 Global Entropy Reduction

As we mentioned in Section 3, a more state of the art algorithm which in ASR tasks [15] showed very good performance is the global entropy reduction. It is again based on entropy but it also uses distance information. This algorithm approximates the total entropy reduction that is expected to occur by the transcription of the utterance x_i as:

$$E\left[\Delta H_i^t\right] \cong \sum_{j=1}^{N_u^t} H_j^t \exp(-\beta d(x_i, x_j)) \qquad (4.1)$$

The summation is over all the samples in the in the unlabelled pool. For utterances in the unlabelled pool that are close to x_i according to the distance metric, labelling of x_i will result to a significant reduction in their entropy. On the other hand, uncertainty for distant utterances will not be affected significantly.

Samples are selected in a greedy fashion, one at a time. The line with the highest estimated entropy reduction, that has not been selected already, is deemed the most informative and selected for transcription.

After selecting the highest scoring x_*, all the entropies are modified as:

$$H_j^{t+1} \cong H_j^t - H_j^t \exp(-\beta d(x_*, x_j)) \tag{4.2}$$

and we repeat the calculation of the entropy reduction to select the next best one. The purpose of the reduction step is to approximate the entropies that would result if we actually trained a new model with the inclusion of x_*, by reducing the uncertainty of the samples in the pool proportionally to how similar each one is to the transcribed. In [15] they adjust only a subset of the entropies for those utterances that are sufficiently close to the selected one. The distance threshold is selected such as the maximum change in entropy is 10%. The process is repeated until k distinct samples have been selected. Algorithm 1 summarizes the procedure.

Algorithm 1 Global Entropy Reduction algorithm

```
1: procedure GER
2:     H[M] ← Array of entropies of unlabbelled samples
3:     k ← Batch size                          ▷ The number of samples to be selected
4:     U ← {$\mathcal{U}$}                     ▷ The set of all available unlabelled samples
5:     S ← {}                                  ▷ The set of selected samples
6:     for i=0 to k do
7:         Calculate GER for all samples in U (Eq. 4.1)
8:         $x_*$ ← the sample with maximum GER not in S
9:         Add $x_*$ to S
10:        Adjust all entropies in H[M] according to Eq. 4.2
```

The global entropy reduction requires two parameters to be set: β, the scale of the distance for the calculation of the entropy reduction and the threshold on distance below which to reduce the entropies. β controls how important is the entropy of the considered sample vs selecting a sample that is on average similar to others. Furthermore in the reduction stage it affects the importance of the distance in reducing the entropies of the neighbours of the selected sample. For β going to infinity the algorithm it is equivalent to uncertainty sampling as it will select samples based on their entropy only. Conversely small values of β are going to lead in selection of samples that are most similar to others in the pool, downplaying the importance of the uncertainty. During the reduction step it also controls the relative importance of selecting high entropy samples vs selecting samples with no information overlap within the same batch. The right value of β depends on the distribution of the pairwise distances, the size of the batch of selected samples and the nature of the task.

The second parameter, the threshold below the entropies of the neighbours of the selected sample are reduced it is of a smaller importance. It is devised to counteract the effect of small values of beta by not allowing entropy reductions to all of the samples, to prevent the entropies from becoming degenerate in the selection.

In [15] they used the full lattice entropy with GER. However in this work we use it with the N-Best list entropy as in [13] they argue it is a most effective measure of uncertainty.

In order to use global entropy we must first calculate the pairwise distance between the unlabelled samples in the pool with under some suitable distance measure. In the next two section thus we develop two such distance measures which we believe they are suitable for the HTR task.

4.3 Dynamic Time Wrapping Image distance

Calculating distances between the unlabelled lines can be done on the predictions (lattices) of the recogniser as they do for example in [15] where they use the KL divergence or on the feature space. In this discuss a distance metric acting on the feature space and in the next one on the lattices. As we discussed in Section 2.1.2 the recognition system treats the line images as a sequence of feature vectors. Calculating distances between sequences of varying length and speed cannot be done trivially. Simply summing the distances between each ith point in one sequence from the ith point in the other it's not well defined if the series are of different length and can be produce counter-intuitive results if for example the sequences are identical but one of them is slightly shifted.

A common way to solve this problem and produce intuitive distances is Dynamic time Warping. It finds an optimal alignment between points of the two time series by warping one of them, i.e stretching or shrinking the time axis. The total warped path distance is the sum of the distances between the corresponding samples. We seek the alignment path that minimizes it.

In particular, given two sequences $x_1 \dots x_N$ and $y_1 \dots y_M$ and a distance metric between two samples of the sequences : $d(x_i, y_j)$ the DTW-distance between them is $D(N, M)$ and is obtained as the solution to the following recurrence relation.

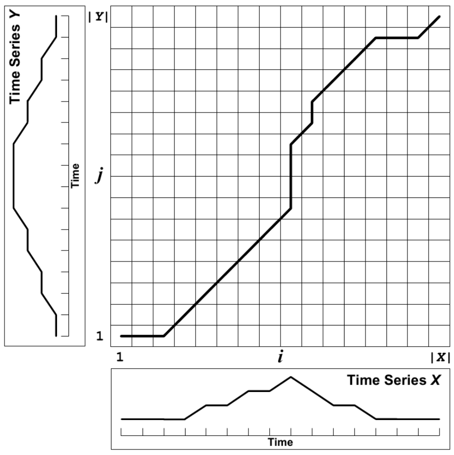
$$D(i,j) = \min\{D(i,j-1), D(i-1,j), D(i-1,j-1)\} + d(x_i, y_j)$$

The complexity of the dynamic programming solution is $\mathcal{O}(NM)$. As part of the solution we obtain the alignment path of the two sequences: a sequence of tuples: $((i_1, j_1), \dots, (i_K, j_K))$ where K is the length of the alignment. The determined distance can thus be as:

$$D(X,Y) = \sum_{k=1}^{K} d(x_{i_k}, y_{j_k})$$

Is clear the the magnitude of the distance is affected directly by the length of the warping path and thus indirectly by the length of the sequences. This is not always a desired property and thus to debias the distance we can normalise by the length of the path K.

In order to avoid excessive compression or expansion of the time scales, a so-called global continuity constraint is sometimes enforced [21]. This is done by restricting the alignment path to be within a specific region as shown in Figure 4.1b. This restriction has the the added benefit of speeding up the computation of the alignment.



(a) Alignment path of two time series. The DTW distance is the sum of the matrix entries along the path.

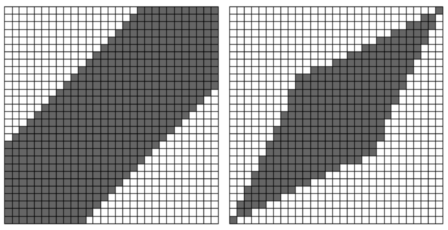

(b) Examples of global constraints: (a) Sakoe-Chiba band; (b) Itakura parallelogram.

Figure 4.1: The figures are extracted from [22]

In [21] they use DTW to do spotting of word images in historical documents in the absence of a transcription. Given a template word image they calculate the DTW distance with each word in the target corpus to find matches. To do this they first do baseline offset correction, slant and skew normalisation in each word to minimise intra word variability. Then for each word image they a extract a series of 4-dimensional feature vectors for each column of pixels. An image of M columns is thus represented as sequence $\mathbf{x}_1 \dots \mathbf{x}_M$ where $\mathbf{x_i} = [x_{i1}, x_{i2}, x_{i3}, x_{i4}]^T$. The distance between two samples (columns) of two images is defined as the squared Euclidean distance of the their feature vectors:

$$d(\mathbf{x_i}, \mathbf{y_j}) = \sum_{k=1}^{4} (x_{ik} - y_{jk})^2$$

The features extracted from each image column are widely used in the handwriting recognition literature and aim to retain as much information as possible to allow reconstruction of the written word.

The first feature is known as projection profile and aims to capture the distribution of ink along the vertical dimension. Is defined as the sum of the intensity values in each column:

$$x_{i1} = \sum_{r=1}^{r=78} I_x(r,i) - 255$$

where the image I_x is binary with ink segments corresponding to maximum intensity (255).

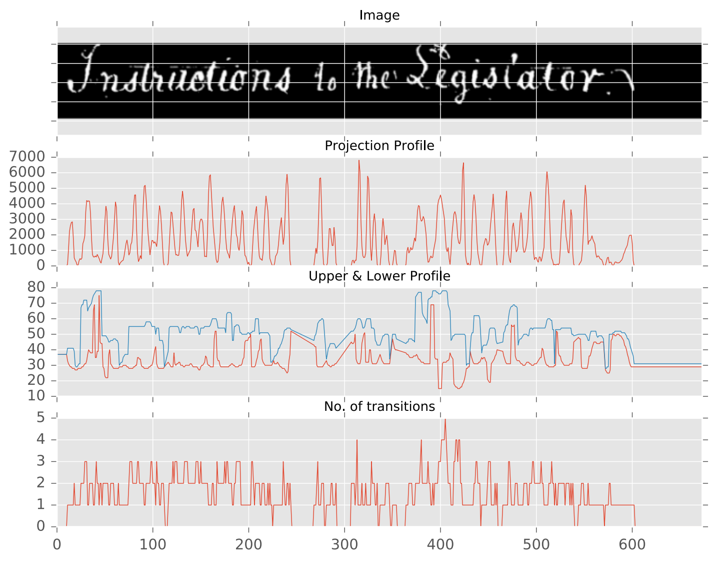
The second and third features capture the word profile. These features also capture information about the ink distribution and they respectively record in each column the location of the uppermost and lowermost pixel with ink. To smoothen the word profiles for columns where there no ink pixels at all, linear interpolation is being used: the two closest points where the word profile feature values could be reliably determined are used to fill these gaps.

$$x_{i2} = \min\{r : I_x(r,i) > 0\}$$
$$x_{i3} = \max\{r : I_x(r,i) > 0\}$$

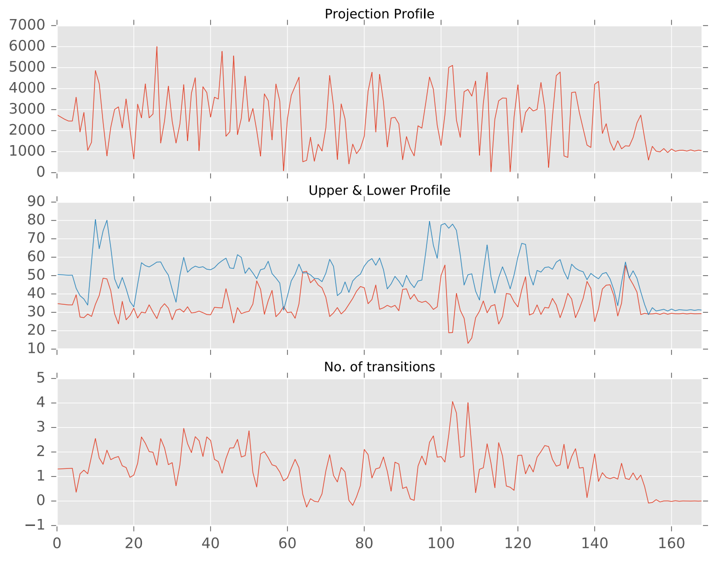
Lastly the number of transitions from ink to no ink sections in each column constitutes the fourth feature. This aims to capture information about the structure of the characters.

Figure 4.2 shows the described extracted feature sequences for a sample line image to demonstrate how they capture the handwriting profile. The raw features are normalized in the $[0,1]$ range by using the minimum maximum values across all columns of all lines before the distance calculation.

For the purposes of active learning we need to calculate all the pairwise distances between the unlabelled sentences. For the present experimental setup described in the next section, there are 9198 such line images and thus we need to perform approximately 43 million distance computations. Even



(a) Before decimation



(b) After decimation

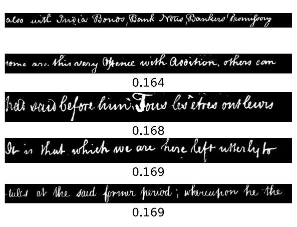
Figure 4.2: Extracted feature sequences for a sample line image

though the computations can be trivially parallelised, with the resources available we cannot compute full DTW distances in a reasonable time-frame. Even restricting the alignment path as mentioned above doesn't give enough improvement. We thus consider an approximate implementation of the DTW proposed in [22]. This algorithm start from a coarse approximation of the th wrap path and recursively projects it to a higher resolution and refines it. It achieves finding the approximate solution in linear time which make it very completive against exact DTW for longer sequences.

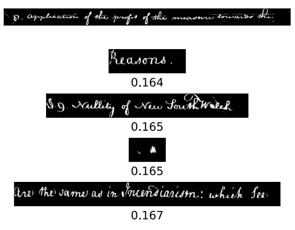
Furthermore, for a more drastic and direct speed-up we reduce the length of the feature vector series by decimation. Each feature sequence is treated as a time series and passed through an order 8 Chebyshev type I filter before keeping one out out 4 consecutive samples. These reduces the length of each sequence by a factor of 4. Unfortunately this comes with significant loss of information particularly for the first feature (projection profile) which has high variance. This is demonstrated in Figure 4.2.

Using the procedure described, the pairwise distances are computed in parallel using 42 CPUs which takes approximately 37 hours. The resulting distances of some sentences are shown in Figure 4.3. In particular we show six sample lines and their four nearest neighbours as calculated by the DTW distance. We notice that for longer sentences (Figures 4.3a, 4.3b, 4.3d) is not completely clear why the distances with the neighbours depicted are low, the similarities are not completely obvious. For lines comprising of 1 word the similarities are more apparent (Figures 4.3e, 4.3f) but still not without shortcomings (Figure 4.3c). Furthermore we note that even with the path-length normalisation, the shorter lines have significant smaller distances with their nearest neighbours than the longer ones. The path-length normalisation also seems to introduce artefacts as demonstrated in Figure 4.3b where a line sentence is assigned a very low distance from a one word line. Figure 4.4 shows the histogram of the pairwise distances between all lines. Of course, the distances are confined in $[0, 1]$ as the features have been normalized and the DTW distance is normalized by the length of the alignment path. However the distribution is relatively sharp with most of the distances falling between

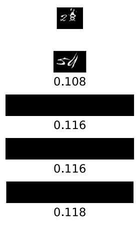
0.18 and 0.4.



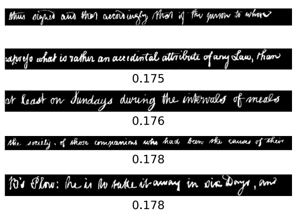
(a) Image 1



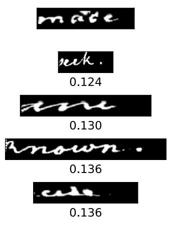
(b) Image 2



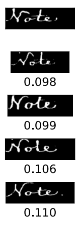
(c) Image 3



(d) Image 4



(e) Image 5



(f) Image 6

Figure 4.3: Six sample line line images (top of each subfigure), their corresponding four nearest neighbours and their DTW distances

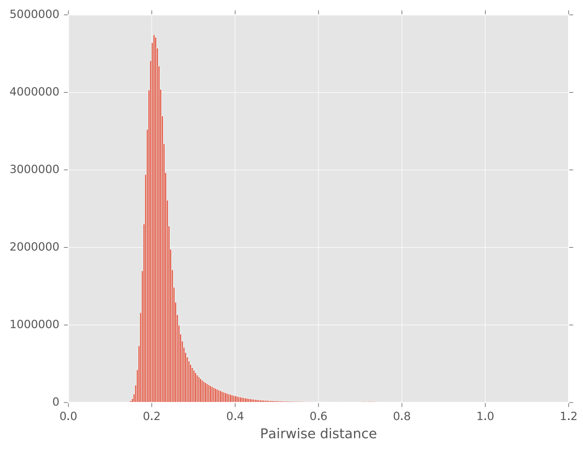


Figure 4.4: Histogram of all calculated pairwise DTW distances

4.4 TF-IDF transcription distance

The DTW distance described in the section above acts on the image space. However the uncertainty is being derived from the N-Best list hypotheses and thus it is sensible to consider distance metrics based on these. Inspired by the work in [13] we use the term frequency - inverse document frequency distance (TF-IDF) by treating each N-Best list of transcriptions as a document. In particular from each sentence's lattice we extract the 1000 best scoring hypotheses in terms of clustered tri-characters and we consider them as a bag of tokens. As discussed in Section 2.1.2 there are around 350 different such context dependent character tokens which we treat as terms. Each N-Best list is thus described by a 350 dimensional vector where each entry has the TF-IDF score of the particular term (tri-character). The term frequency (tf) is the normalized occurrence count of each term in the document, and the inverse document frequency (idf) is a metric of the relative importance of that term in terms of how common the term is in the whole corpus. The N-Best list transcriptions of all the unlabelled lines comprise the corpus D. The tf-idf score of each *term* is then defined as the product of two terms:

$$\begin{aligned} tf(term, d) &= \frac{\#\text{occurences of } term \text{ in } d}{\text{length of } d} \\ idf(term) &= \frac{|D|}{1 + |\{d : term \in d\}|} \end{aligned}$$

where $|D|$ is the total number of lines in unlabelled pool and $|\{d : term \in d\}|$ is he number of N-Best lists in which the term appears.

Having a vector representation of the lines transcriptions we can use any suitable distance measure. Typically for tf-ifd representations the cosine distance is being used:

$$d(\mathbf{x_i}, \mathbf{y_j}) = 1 - \frac{\mathbf{x_i}^T \mathbf{y_j}}{|\mathbf{x_i}||\mathbf{y_j}|}$$

As before to quantify the workings of the distance we are looking at the

	Distance	Transcription
0	0.000	thus signed and that accordingly that if the person to whose
1	0.250	ever signed by any other person than he who name it is that is
2	0.254	the time and that the amputation cured, and that a cure
3	0.257	be that it should amount to half a day's Wages according to the ordi=

Table 4.1: Nearest neighbours of a sentence, when using the TF-IDF distance

closest neighbours of a few samples. Now only the transcriptions are being considered and the neighbours are more intuitive as there is a significant overlap in words even though we only take into account the context dependent characters.

In the next chapter we will present the results from the procedures described here. We first run uncertainty sampling with the N-Best List entropy and then the global entropy reduction with two different distance metrics: DTW and TF-IDF.

Chapter 5

Experiments

5.1 Experimental Setup

To evaluate the performance of different active learning algorithms we use the available training data to create two pools: one of labelled data and one of unlabelled. The labelled pool consists of 10% of the data (920 lines) which are randomly selected from the total and we use them to train an initial model. The predictions (lattices) of the trained model for the unlabelled pool along with the calculated pairwise distances are used to propose the 920 most informative samples within the unlabelled pool. A new model is then trained on the original 10% lines plus the additional 10% proposed by the active learner. In the next iteration the active learner proposes the best 920 lines samples out the remaining 80% and adds them to the previous 20% of labelled data. This is repeated until we have a model trained on 90% of the available data; at that stage there is not a choice of data as there are exactly 920 samples left for the model trained on the full available dataset.

At each stage we calculate the WER that the current trained model achieves on the development and test held out data. This is compared with the WER that is achieved by adding the training data again in 10% increments but selecting them randomly. This emulates the scenario where unlabelled data are incrementally in batches of 920 are given to an oracle for annotation without any selection procedure; passively.

5.2 Uncertainty Sampling with N-Best list Entropy

The first algorithm we consider is Uncertainty sampling with the Entropy calculated from the N-Best lists. At each iteration we train the model on the current labelled pool and we use it to obtain the lattices for the lines in the unlabelled pool. We then extract from them the N-Best lists and calculate the entropy as described in Section 4.1. The unlabelled lines are then ranked by the highest entropy and the top 920 of them are added to the labelled pool for the next iteration. The size of the N-Best list we use was selected to be a 1000 after a very coarse tuning. The optimal model scale for the scores was set to 0.1 without tuning.

Figure 5.1 compares uncertainty sampling with the random sampling approach in terms of WER achieved in the test and development sets. It's evident that uncertainty sampling performs quite favourably. The annotated dashed lines in the plots correspond to the gains in two different use cases of active learning. The horizontal lines show the percentage of data that we save to achieve a particular target WER when using active learning. This translates to savings in annotation cost. Similarly, the vertical lines show the improvement in WER that uncertainty sampling achieved for a fixed annotation budget. The maximum savings in annotation data occurs when targeting for 29% WER : with uncertainty sampling it is achieved with 29% less training data. Similarly the maximum improvement in WER is observed when we use 50% of the data where we note a 4.4 percentage points reduction.

Although it is standard in the active learning literature for ASR to show the WER as function of the number of training utterances used for the purposes of comparing it with random sampling or other approaches, we believe that for the handwritten recognition task viewing it as a function of the total number of words is more fair. This is mainly due to the fact that annotation costs of handwritten texts are proportional to the number of words rather than the number of lines. In particular for this dataset where as discussed in Section 2.1.1 the length of the lines vary wildly such normalisation is important. For example we saw that uncertainty sampling achieves 29% WER with 29% less lines than random sampling. However that will not necessarily translate in 29% annotation cost savings since as we shall see the higher entropy lines which are selected by uncertainty sampling have more words.

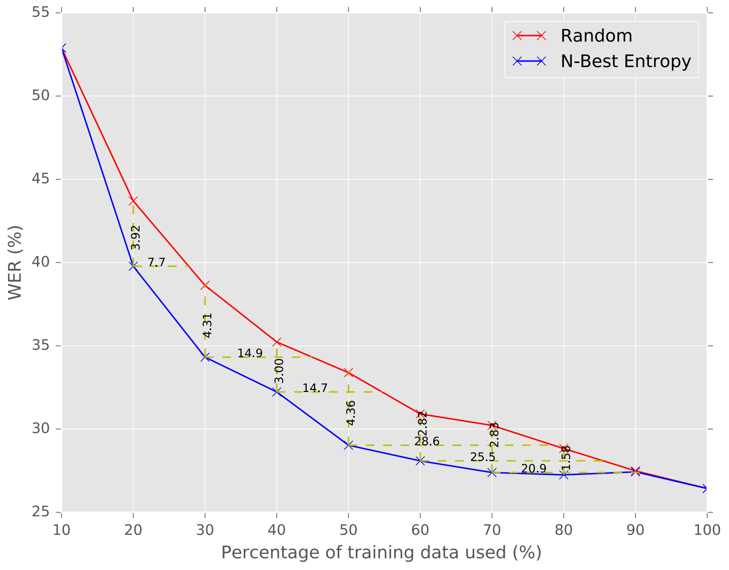
Motivated by this discussion, Figure 5.2 shows the WER for random sampling and uncertainty sampling as a function of the percentage of the total words contained in the training set that was used in each iteration. We notice that although uncertainty sampling still performs clearly better the difference is less pronounced. In particular the maximum annotation savings costs are now 16% where as the maximum reduction in WER for a fixed annotation budget is 2.2 percentage points. Looking at the distribution of entropies in Figure 5.3 is clear that in particular one-word sentences and to a smaller extend the 2 word ones can have much lower entropies. Thus uncertainty sampling won't select most of the very short sentences until very late, which leads to an increased number of words for the same number of lines against random sampling. The entropies presented in that Figure 5.3 come from the lattices generated by the model trained on 10% data. In theory the entropies should change as we train with more data and obtain a better model. It is interesting to investigate how an increasing amount of training data affects the entropies estimated and thus the ranking of the proposals. This is of particular importance since it can affect the batch size of the queries and lead to computational savings. As we mentioned in each iteration we rank all samples in the unlabelled pool, select the top 920 and move them to the labelled pool. So for example if with the entropies calculated on the first iteration the samples ranking in 921-1840 are similar to those ranking 1-920 on the second iteration we could save one retraining and query the top 1840 samples from the first iteration directly. In the next section and in particular in Figure 5.8 we show that in fact the entropies change significantly and there is a loss in performance of active learning with larger batch sizes.

It is reasonable to expect that especially in the early iterations where the amount of data is limited, active learning will select sentences for transcription that will expand the vocabulary. This is because out of vocabulary words will never be recognised correctly and so it is reasonable to be presented by new words rather than get more instances of the same word. Figure 5.4 shows how the size of vocabulary grows with the amount of training data. Indeed it is evident that active learning adds new words much faster than random sampling. In particular, 60% of the training data contains 90% of the vocabulary when selected by uncertainty sampling whereas when randomly selected only 71% of the vocabulary has been seen.

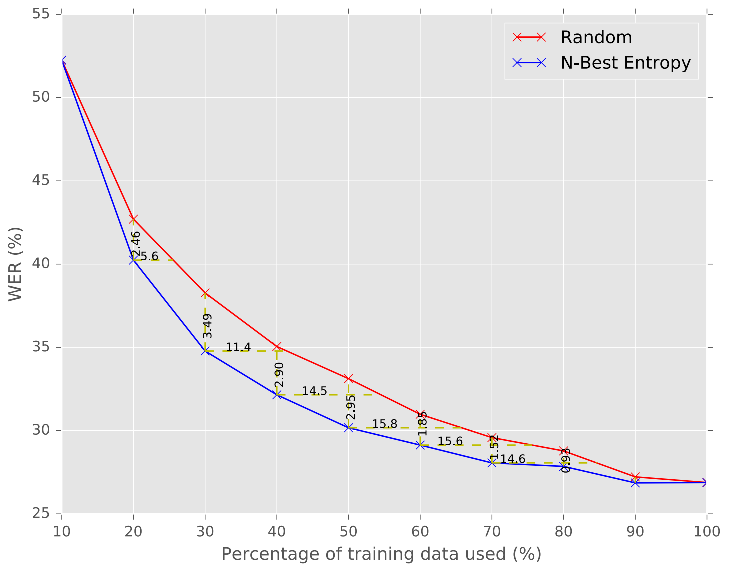
One point to note is, that it is not easy to do a direct comparison with the performance curves reported in the literature for active learning within the ASR tasks. The reason being is that in these tasks they have used enough training data to reach a plateau in the WER. The amount of data needed to reach that plateau with active learning vs with random sampling is used as a metric. Even in that case though the results can vary depending on how many additional data they used past the plateau point. However in our case is clear that the error rate improves significantly for random sampling even with the last addition of data at 90%. This indicates that there is significant room for more training data before we reach the plateau. The fact that the increase in the vocabulary size in Figure 5.4 continues to grow linearly up to the last stages, corroborates this fact. New words are being encountered with same rate even though we have added almost all data. For this reason a direct comparison with other work is different domains it would lack consistency.

Using our limited size training set we notice that the effectiveness of adding labelled data proposed by the active learner depends on what "training stage" the model is. Figure 5.5 shows the reduction on the WER achieved by the addition of each batch of 920 training samples as proposed by uncertainty sampling against the random sampling. We observe that active learning gains its advantage in the early rounds where it achieves significant reduction over random sampling, which is probably expected as the most informative samples are added early. After around 40% of the data have been added the reductions are similar until the final round where the active learning proposal become worse as only less informative samples are still in the pool. What's interesting however is that the final addition of the last 10% of data, results to a significantly higher improvement than the previous proposals of the active learning. This signifies that the last 10% or so of samples with the lowest entropy that are added last, were actually more informative than the previous 20% with higher entropies. As mentioned above those very low entropy lines are actually those consisting of one word. Although in the entropy sense thy are less informative, transcribing them lowers the WER quite significantly, presumably because the test sets also contain such lines.

We believe that the observed behaviour of diminishing returns of active learning towards the last few additions of data is related to our experimental setup. In particular the unlabelled pool of candidates shrinks too quickly because we emulate it with the labelled training set. In a real scenario the amount of unlabelled lines would be significantly larger than the total annotation budget and thus the total number of candidate queries would be much larger even in the latter stages.

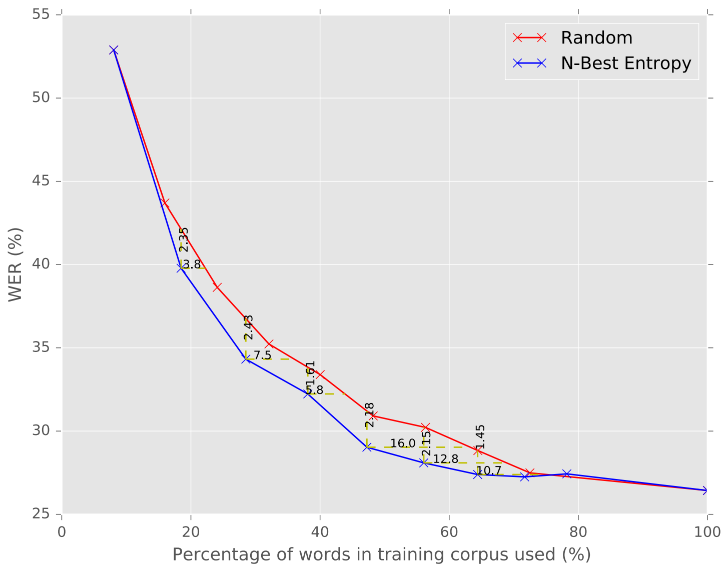


(a) Test Set

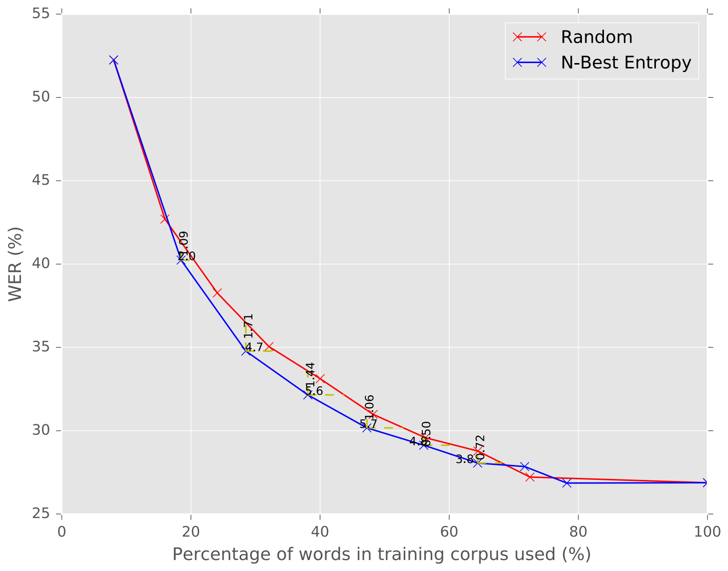


(b) Development Set

Figure 5.1: Performance curves of uncertainty sampling with N-Best List Entropy by considering the number of training lines



(a) Test Set



(b) Development Set

Figure 5.2: Performance curves of uncertainty sampling with N-Best List Entropy by considering the number of training words

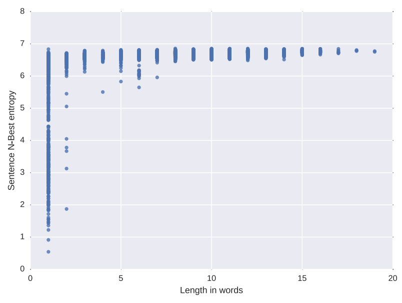


Figure 5.3: N-Best List Entropies of the sentences in the unlabelled pool. Evaluated from the lattices produced from the model trained on 10% of the data.

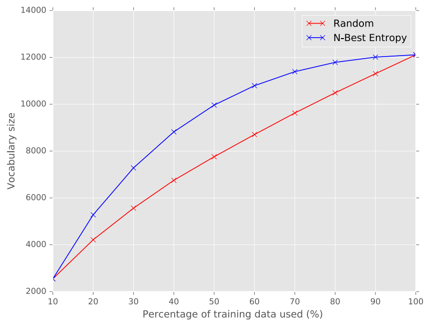


Figure 5.4: Size of vocabulary as a function of the percentage of the training lines used

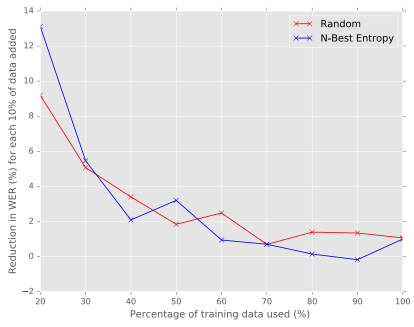


Figure 5.5: Reduction on the WER achieved by the addition of each batch of 920 training samples.

5.2.1 Global Entropy Reduction

In this section we investigate the results produced by an adaptation of the more sophisticated active learning algorithm proposed in [15] and over-viewed in Section 4.2. Global Entropy Reduction (GER), along with the entropy derived from the predictions from the trained model, uses distance information, in an attempt to avoid querying outliers and noise and also to avoid information overlap within the queried samples. The distance metric we are use using for these experiments is the DTW distance on image features developed in Section 4.3. The experiments performed are the same as for uncertainty sampling previous section and we aim to directly compare the two active learning algorithms.

As discussed in Section 4.2, global entropy reduction is sensitive to two parameters that need tuning. We first tune β, the scaling of the distance in the exponential. However since the runtime of training the recognition system is high, we do a first coarse tuning by looking at the first iteration, how similar the queried samples are with those of uncertainty sampling. As mentioned we want large enough β so the entropy is not being ignored but small enough for the queries to be sufficiently different. The table below shows the percentages of overlap in queries from global entropy reduction with uncertainty sampling as β is varied.

Value of β	20	30	40	50	60	70	80	90
Overlap with Unc. Samp. (%)	5.6	5.6	5.6	8.0	23.3	65.5	93.4	98.8

Following the reasoning mentioned above it is clear that the value of β should be between 60 and 70. We thus run the global entropy reduction with β equal to 60, 62.5, 65, 67.5 and 70 to propose 920 unlabelled samples. We then train the recognition system with each of these proposed sets added to the initial 10% of the data and select the beta which resulted to the minimum WER. The value of β that resulted to the minimum WER was 67.5 which we use for all the subsequent experiments.

The global entropy reduction algorithm with the DTW distance is compared against the random sampling approach and uncertainty sampling for the test and development sets, following the same procedures as before. As before we can consider the number of lines that the training set contains (Figure 5.6) or the number of words (Figure 5.7).

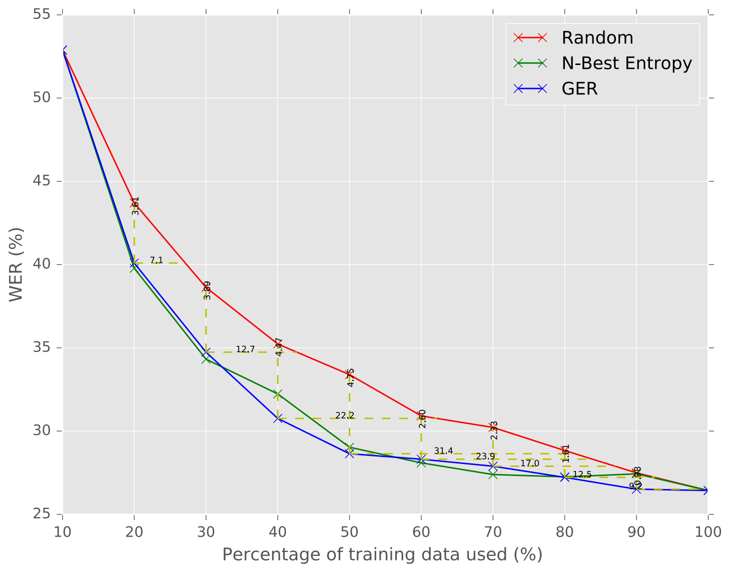
We notice that GER with DTW distance performs only slightly better than the simple uncertainty sampling and the overall curves are very similar. The advantage of GER is more evident in the development set. This could be due to the fact that the tuning of β is done based on the development set, although the tuning is so coarse that the discrepancy may well be due to differences in the two set-sets. This difference in the two sets is corroborated by the fact that in all our experiments (Figures 5.1, 5.2, 5.6, 5.7) the advantage of both active learning algorithms over random sampling is more pronounced in the test set. Although the WERs for random sampling are approximately the same, active learning performs better in the test set. This difference in improvements between the two sets can be as high as 1 percentage point. It is not apparent to us what causes this discrepancy between these two sets.

Since GER performs so similarly to N-Best list is interesting to look at how the lines being added on each iteration compare between the two active learning algorithms. The table below show how much the 920 proposed lines overlap between the two algorithms as well as how similar the total training sets are in each iteration. We notice that although the queries from the GER and uncertainty sampling are significantly disjoint, the difference in the overall training sets is fairly small especially for the later iterations. This signifies that although the top 920 selected examples might be disjoint the overall ranks that the two algorithms assign are not that different. This is exacerbated by the fact that we use a small training set to emulate the large pool of unlabelled candidates.

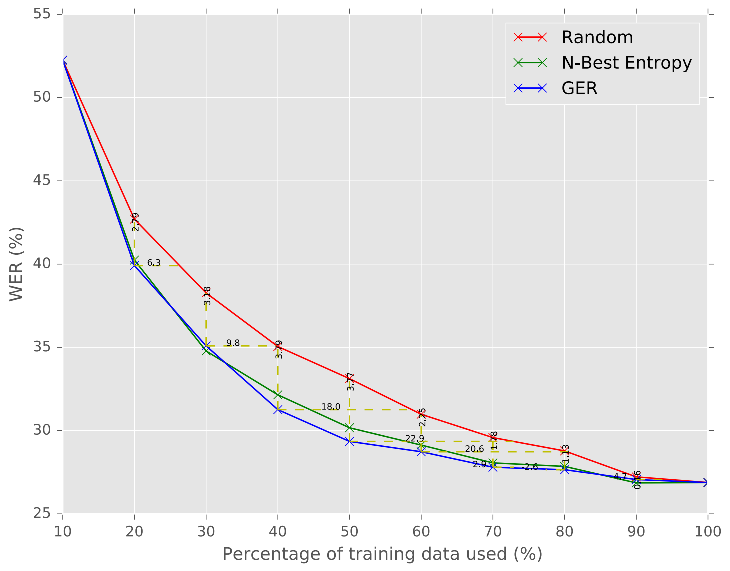
Training data (%)	10	20	30	40	50	60	70	80	90
Overlap in batch (%)	100.0	53.9	35.5	23.8	26.1	28.6	30.2	36.0	48.0
Overlap in total (%)	100.0	77.0	72.6	78.4	84.0	88.1	92.0	93.9	97.0

Lastly we look at the effect of the batch size in the performance of the two algorithms. Instead of selecting the best 920 candidates on each iteration we select the best 2760 (30% of the available data). The curves in Figure 5.8 are with respect to the number of words contained in the data. We see now that the performance of both active learning algorithms deteriorates and they struggle to compete against random sampling particularly in the first iteration. This is mostly because the initial model trained on only 10% of the data is too weak to produce entropies accurately reflecting the uncertainty. Also, with a larger batch size in theory GER should benefit against uncertainty sampling as it uses the distances to account for information overlap within the batch which is more important for larger batches. However the difference between the two is again too small to accurately to reliably make any conclusive statements about that.

We also tried to use GER with the TF-IDF distances described in Section 4.4 but the initials results were disheartening so we didn't pursue it further. It could be that better tuning was needed but in the time available we could not find any configuration that worked better than than uncertainty sapling with N-Best list entropy.

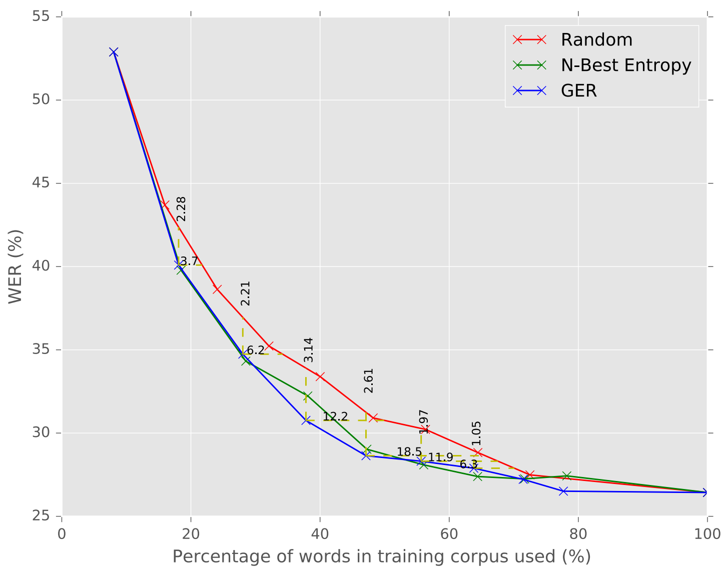


(a) Test Set

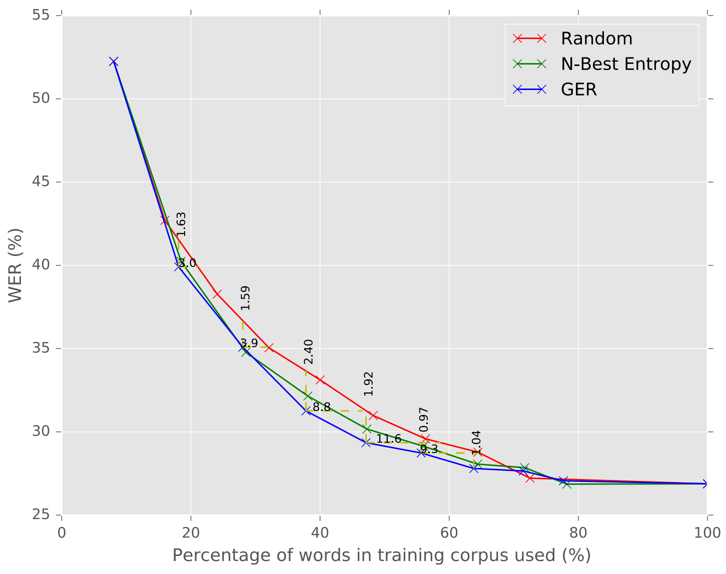


(b) Development Set

Figure 5.6: Performance curves of active learning with Global Entropy Reduction by considering the number of training lines

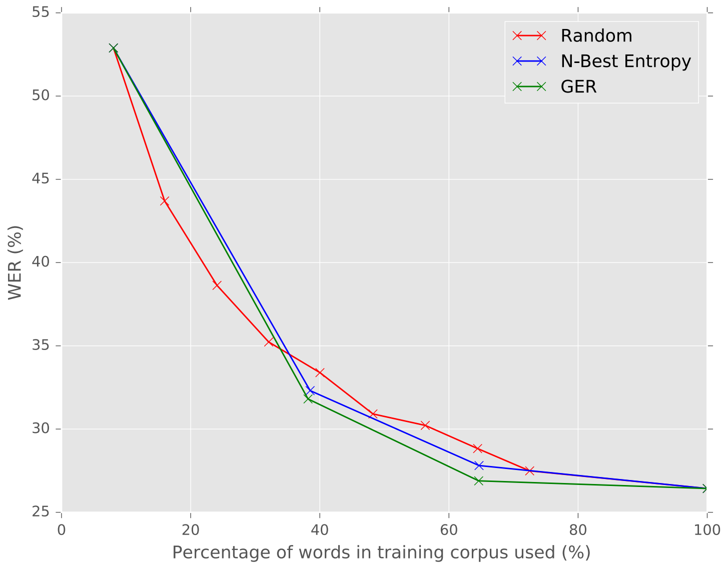


(a) Test Set

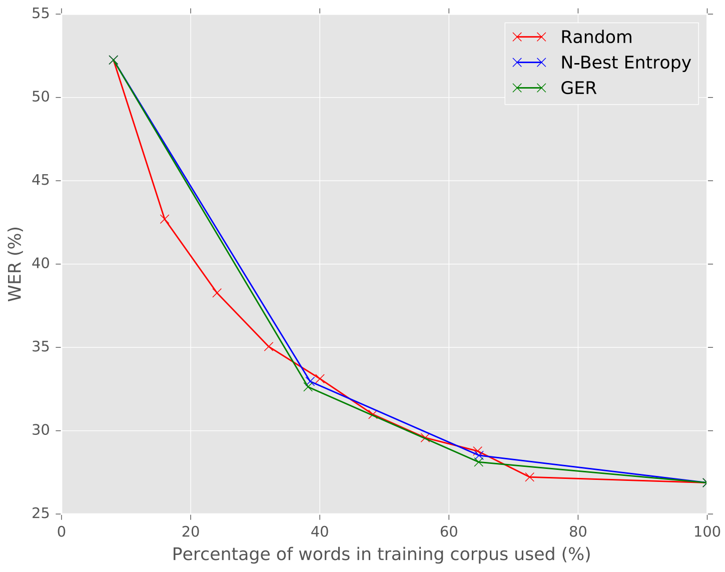


(b) Development Set

Figure 5.7: Performance curves of active learning with Global Entropy Reduction by considering the number of training words



(a) Test Set



(b) Development Set

Figure 5.8: Performance curves of active learning with an increased batch size of the proposals by considering the number of training words

Chapter 6

Summary and Future Work

This thesis explored the use of active learning for the HTR task in historical documents.

We presented the particular dataset we are working with and the recognition system we are using. Different active learning approaches were discussed along with the fact that only a few of those are applicable to HTR primarily due to the long training times and the complex nature of sequence classification models. Although there has been no previous work on active learning for HTR we note the similarities with ASR and draw from research in that area. Most prominent active learning algorithms require an uncertainty measure as a minimum and more sophisticated ones introduce a pairwise distance metric. For the uncertainty measure we deemed N-Best list entropy as the most appropriate and investigated its performance in the context of uncertainty sampling. We noted noticeable improvement in terms of annotation expenditure savings for a target WER and achieving better performance with a fixed annotation budget. We proposed looking at the total number of words used for training as a more subjective means of evaluating the active learning performance in the context of HTR tasks.

The results show that albeit its simplicity, uncertainty sampling could have a real impact on the field of historical document recognition leading to improved recognition performance with the same resources. Further tuning, particularly of the size of the N-Best List used for the entropy calculation could improve the results further. One of the limitations of our experimental setup, stems from the fact that we use the labelled set to emulate the unlabelled pool. As we mentioned the size of the available training set is small for this task and thus emulating active learning with it potentially underestimates the benefits. It would be nice to test this approach either in an actual use-case-like setup, where selected lines are coming from a large pool of unlabelled instances which we incrementally query until we reach satisfactory performance, either use a significantly larger train set that saturates the WER performance to emulate this, to the same effect.

We then investigated a more sophisticated approach for active learning know as global entropy reduction. We proposed a DTW based distance metric acting on the the image space to be used and also experimented with a TF-IDF distance based on the transcription output. We found that with the DTW distance GER managed to improve over uncertainty sampling, albeit only by a small margin. We believe it would be beneficial to investigate finer tuning of this algorithm, as well as calculation of the DTW distance without the decimation approximation and more features, given enough time and computation resources. Different, more computationally expensive distance metrics that have been used in the ASR literature such as the KL lattice divergence could also be worth exploring.

Bibliography

[1] Carmelo Cassisi, Placido Montalto, Marco Aliotta, Andrea Cannata, and Alfredo Pulvirenti. Similarity measures and dimensionality reduction techniques for time series data mining. *Advances in data mining knowledge discovery and applications*, 2012.

[2] Joan Andreu Sánchez, Günter Mühlberger, Basilis Gatos, Philip Schofield, Katrien Depuydt, Richard M Davis, Enrique Vidal, and Jesse De Does. transcriptorium: a european project on handwritten text recognition. In *Proceedings of the 2013 ACM symposium on Document engineering*, pages 227–228. ACM, 2013.

[3] J.A. Snchez, V. Romero, A.H. Toselli, and E. Vidal. ICFHR2014 competition on handwritten text recognition on transcriptorium datasets (HTRtS). In *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 181–186, 2014.

[4] F. Stahlberg and S. Vogel. The QCRI Recognition System for Handwritten Arabic. In *ICIAP*, 2015.

[5] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[6] Michele Banko and Eric Brill. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th annual meeting on association for computational linguistics*, pages 26–33. Association for Computational Linguistics, 2001.

[7] Ibrahim M Alabdulmohsin, Xin Gao, and Xiangliang Zhang. Efficient active learning of halfspaces via query synthesis. In *AAAI*, pages 2483–2489, 2015.

[8] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 1996.

[9] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, pages 441–448, 2001.

[10] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics, 2008.

[11] Shayok Chakraborty, Vineeth Balasubramanian, and Sethuraman Panchanathan. Adaptive batch mode active learning. *Neural Networks and Learning Systems, IEEE Transactions on*, 26(8):1747–1760, 2015.

[12] Dilek Hakkani-Tür, Giuseppe Riccardi, and Allen Gorin. Active learning for automatic speech recognition. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 4, pages IV–3904. IEEE, 2002.

[13] Nobuyasu Itoh, Tara N Sainath, Dan Ning Jiang, Jie Zhou, and Bhuvana Ramabhadran. N-best entropy based data selection for acoustic modeling. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4133–4136. IEEE, 2012.

[14] Yuhong Guo. Active instance sampling via matrix partition. In *Advances in Neural Information Processing Systems*, pages 802–810, 2010.

[15] Dong Yu, Balakrishnan Varadarajan, Li Deng, and Alex Acero. Active learning and semi-supervised learning for speech recognition: A unified framework using the global entropy reduction maximization criterion. *Computer Speech & Language*, 24(3):433–444, 2010.

[16] Yuhong Guo and Dale Schuurmans. Discriminative batch mode active learning. In *Advances in neural information processing systems*, pages 593–600, 2008.

[17] Yuxin Chen and Andreas Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *ICML (1)*, pages 160–168, 2013.

[18] Leonidas Lefakis and Marco A Wiering. Semi-supervised methods for handwritten character recognition using active learning. 2007.

[19] Jan Richarz, Szilard Vajda, Rene Grzeszick, and Gernot A Fink. Semi-supervised learning for character recognition in historical archive documents. *Pattern Recognition*, 47(3):1011–1020, 2014.

[20] Giuseppe Riccardi and Dilek Hakkani-Tur. Active learning: Theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, 13(4):504–511, 2005.

[21] Toni M Rath and Raghavan Manmatha. 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. IEEE, 2003.

[22] Stan Salvador and Philip Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.