Offline Recognition of Large Vocabulary Cursive Handwritten Text

A.Vinciarelli and S.Bengio
Dalle Molle Institute for Perceptual Artificial Intelligence
Rue du Simplon 4
CH1920 - Switzerland
{vincia,bengio}@idiap.ch

H.Bunke
University of Bern
Neubrückstrasse 10
CH3012 - Switzerland
bunke@iam.unibe.ch

Abstract

This paper presents a system for the offline recognition of cursive handwritten lines of text. The system is based on continuous density HMMs and Statistical Language Models. The system recognizes data produced by a single writer. No a-priori knowledge is used about the content of the text to be recognized. Changes in the experimental setup with respect to the recognition of single words are highlighted. The results show a recognition rate of ∼85% with a lexicon containing 50'000 words. The experiments were performed over a publicly available database.

1. Introduction

The offline cursive recognition systems presented in the literature deal, with almost no exception, with single words. Few works proposed the recognition of word sequences, but only in the context of heavily constrained applications or using simplifying conditions. In [1], the problem of postal address lines transcription is considered. The variability of the lines is not high. Moreover, the lexicon is relatively small (350 entries) and no Out Of Vocabulary words (OOVs) are expected. The recognition of common texts (news, reports, tales, etc.) is addressed in [4], but the problem of OOVs is avoided by extracting the lexicon from the test set. This corresponds to a strong constraint since it fits the system to a specific set of texts and makes it not robust with respect to a change of data.
This work presents an offline cursive handwriting recognition system dealing with unconstrained texts segmented into lines. No simplifying assumptions are made and no constraint is imposed to the data to be recognized (except for the fact of being written in English). The pages contain documents belonging to a corpus assumed to reproduce the statistics of average English. This allows us to apply Statistical Language Models (SLM) in order to improve the performance of our system [3]. We used N-gram models (the most successful SLM applied until now [3]) of order 1, 2 and 3. The shift from single word to text line recognition involves several changes in the experimental setup. The most important one concerns the selection of the lexicon. In all of the cursive word recognition experiments, the lexicon has a full coverage of the data to be recognized (it is sure that one of the entries of the dictionary is the actual transcription of the handwritten sample). This is no longer true for the recognition of unconstrained texts.
When no constraints can be imposed on the text to be recognized, the selection of the vocabulary entries can rely only on linguistic and statistical criteria. This means that especially for small vocabulary sizes (less than 20'000 words) it is probable to have a low coverage of the data.
Another important difference is in the way the performance of the system is measured. In single word recognition, a sample is correctly or uncorrectly recognized (there is a single source of error). In text recognition, there are several kinds of error. A word can be not only misclassified, but also deleted. Moreover, words not appearing in the text can be inserted during decoding.
Several experiments (changing the size of the lexicon from 1'000 to 50'000 and the order of the language model from 1 to 3) were performed over single writer data. The rest of the paper is organized as follows: Section 2 provides the statistical foundations of our approach, Section 3 presents SLMs and N-gram models, Section 4 describes the recognition system used in our work, Section 5 reports experiments and results obtained and the final Section 6 draws some conclusions.

2. Statistical Foundations

This section describes the problem of handwritten text recognition from a statistical point of view. The image is converted into a sequence $O = (o_1, o_2, \ldots, o_m)$ of observation vectors and the recognition task can be thought of as finding a word sequence $\hat{W}$ fulfilling the following condition:

$$\hat{W} = \arg\max_{W} p(O|W)p(W). \tag{1}$$

where $W = (w_1, w_2, \ldots, w_n)$ is a sequence of words belonging to a fixed vocabulary V. Equation 1 is obtained by applying a Maximum A Posteriori (MAP) approach and its right side shows the role of the different sources of information in the recognition problem. The term $p(O|W)$ is the probability of the observation sequence O being generated given the sentence W. Such probability is estimated with HMMs. The term $p(W)$ provides an a priori probability of the word sequence W being written and it is estimated using a Statistical Language Model. A good SLM can significantly constrain the search space so that all the sentences that are unlikely to be written have a low probability.

3 Statistical Language Modeling

Statistical Language Modeling involves attempts to capture regularities of natural languages in order to improve the performance of various natural language applications, e.g. Information Retrieval, Machine Translation and Document Classification [3].

This section is focused on the use of SLMs in our specific problem, i.e. the decoding of handwritten texts. As shown in Equation 1, the SLM is supposed to give the a priori probability of a certain sentence being written [3].

If W contains n words, $p(W)$ can be decomposed as follows:

$$p(W) = \prod_{i=1}^{n} p(w_i|h_i) \tag{2}$$

where $h_i = w_1^{i-1} = (w_1, w_2, \ldots, w_{i-1})$ is referred to as *history* of word i.

However, Equation 2 has a fundamental problem: the number of possible histories is very high. Most of them appear too few times to allow a statistical approach. The solution to this problem is to group the histories in a reasonable number of equivalence classes. Equation 2 can then be rewritten as follows:

$$p(W) = \prod_{i=1}^{n} p(w_i|h_i) = \prod_{i=1}^{n} p(w_i|\Phi(h_i)) \tag{3}$$

where $\Phi : \{h\} \rightarrow C$ associates an equivalence class belonging to a finite set C to a history h. The nature of $\Phi(h)$ allows one to distinguish between different SLM techniques presented in the literature (see [3] for an extensive survey).

Until now, the staple of language modeling is represented by the N-gram models. A N-gram model makes an equivalence class out of all the histories ending with the same $N-1$ words:

$$p(W) = \prod_{i=1}^{n} p(w_i|w_{i-N+1}^{i-1}). \tag{4}$$

In the next subsection, N-gram models are analyzed in more detail.

3.1 N-gram Language Models

Although N-grams are based on a simple idea and do not take any linguistic knowledge into account, they are still the most successful SLM until now [3]. The probabilities of the words being written (given their histories) are obtained by simply counting the relative frequencies of the word sequences appearing in a text corpus:

$$p(w_i|w_{i-N+1}^{i-1}) = \frac{C(w_{i-N+1}^{i})}{C(w_{i-N+1}^{i-1})} \tag{5}$$

where $C(.)$ is the number of times a certain event appears. This corresponds to a Maximum Likelihood (ML) estimation: the estimated probabilities $p(w_i|h_i)$ maximize the likelihood of the training text. However, this approach gives rise to a serious problem: the model is fitted to the training set and the probability of any N-gram not represented in the training corpus is estimated to be zero.

The method implied by Equation 5 is a weak approximation since no text can contain all possible N-grams, hence ML estimations must be *smoothed*. This means that the probability mass must be redistributed across all possible N-grams in order to give a nonzero probability to all sequences of N words.

Smoothing allows the extension of an N-gram model trained on a certain text to any other text, but it gives a non-zero probability to N-grams that are impossible from a linguistic point of view. This is the main limit of the N-gram models.

Among the smoothing techniques available in the literature, we selected the so-called Good-Turing method. This technique has the advantage of being based on the Zipf law (a natural law concerning the distribution of event frequencies in many natural phenomena) and is then more robust with respect to a change of data.

The performance of a language model is measured in terms of *Perplexity* (PP). The PP is estimated as follows:

$$PP = 2^H \tag{6}$$

where $H = \frac{1}{n}\sum_i p(w_i|h_i)$ is an estimation of the entropy of the model (measured over a text). The PP is the average branching factor of the model, i.e. the average number of words having a probability significantly higher than zero at each step of the decoding [3]. In a problem where all the words have the same probability of being written, the PP corresponds to the size of the lexicon. For this reason, the PP is often interpreted as the dictionary size in case of a single word recognition problem.

The relationship between the PP of the SLM and the recognition rate of the system using it cannot be modeled clearly [3]. A decrease of the PP, which corresponds to an improvement of the SLM, does not necessarily lead to better recognition rate for the system (it can even have negative effects) and vice versa.

4 The Recognition System

The recognition system used in this paper was originally developed to work on single words (for a full description, see [6]), but no modifications were necessary to use it for handwritten texts .

The system is based on a sliding window approach: after a normalization step, a fixed width window shifts column by column from left to right and, at each position, a feature vector is extracted. The sequence of vectors so obtained is modeled with continuous density Hidden Markov Models using diagonal Gaussian Mixture Models as emission probabilities.

A different HMM is trained for each letter and several approximations are applied: the first one is that a single model is used for both upper and lower case versions of the same letter. The capital letters available in the database are in fact not sufficient for a reliable training. The second one is that all the models have the same number of states S and Gaussians G in the mixtures. The third approximation is that only letters (as well as the "blank") are modeled. The other symbols (punctuation marks, digits, parentheses, etc.) are treated like noise. This is due to the fact that not enough symbols are available for a reliable training.

The training is performed with the Baum-Welch algorithm and it is embedded, i.e. the Baum-Welch is applied to concatenations of letter models corresponding to the transcriptions of the handwritten lines. This has two advantages, the first one is that no segmentation of the training data into letters is necessary. The second one is that the letters are modeled as parts of words as they actually are in cursive handwriting.

The recognition is performed with the Viterbi algorithm which finds the alignment with the highest likelihood between a state sequence and the observation vectors. This is a good approximation of the most probable word being generated by the model given the observations.

5 Experiments and Results

The experiments were performed using texts written by a single person. The data set is publicly available on the web [1] and will be referred to as *Cambridge* database.

The Cambridge database was originally presented in [5] and contains 353 handwritten text lines split into training (153 lines), validation (83 lines) and test set (117 lines). The partition is not performed randomly. The lines are kept in the same order as they were written in order to reproduce a realistic situation where the data already written by a person is used to obtain a system able to recognize the data that he or she will write in the future.

The language models were obtained using the TDT Corpus [2], a collection of news transcriptions obtained from several journals and broadcasting companies. The corpus is completely independent from the texts in the handwriting data sets. In this way, the language models are not fitted to the specific texts they have to model and the experimental setup reflects the realistic condition of unconstrained text recognition.

In the next subsections we show in more detail how the lexicon was selected (Section 5.1), how the language models were trained and tested (Section 5.2) and the obtained results (Section 5.3).

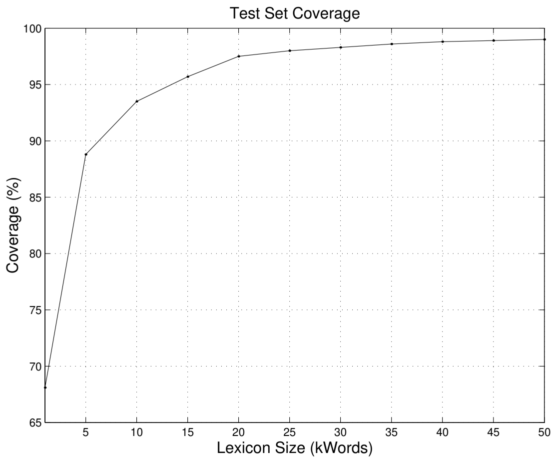

Figure 1. Test set coverage. The plot shows the coverage of the test set as a function of the lexicon size.

5.1 Lexicon Selection

In single word recognition, the lexicon is implicitly assumed to always cover 100% of the data to be recognized. The lexicon is typically determined by information coming from the application environment (e.g. the zip code in postal address recognition).

This is not the case for the recognition of unconstrained texts. The presence of proper names, technical terms and morphological variations of a single stem makes it impossible to define a *universal* lexicon.

The only source of information we assume to have at the

[1] The data can be downloaded at the following ftp address: `ftp.eng.cam.ac.uk/pub/data`.

linguistic level is the text corpus we use to train the Language Models. The lexicon must then be extracted from it. The TDT corpus is composed of 20'407'827 words. The size of its dictionary is 196'209. In order to build a lexicon of size M, we selected the M most frequent words in the corpus.
This is based on the hypothesis that many words appearing in a text are the so called *functional words* (prepositions, articles, conjunctions, etc.) and that certain words are of common use in the language. An important advantage of this approach is that, using the most frequent words in the corpus, it is possible to obtain a more reliable model. The N-gram probabilities are in fact better estimated for the most frequent words.
The plot in Figure 1 shows the Cambridge test set coverage of the lexica obtained with the above mentioned criterion in function of their size. The coverage (the percentage of words in the text actually represented in the lexicon) is, at a given lexicon size, the upper limit of the recognition rate.

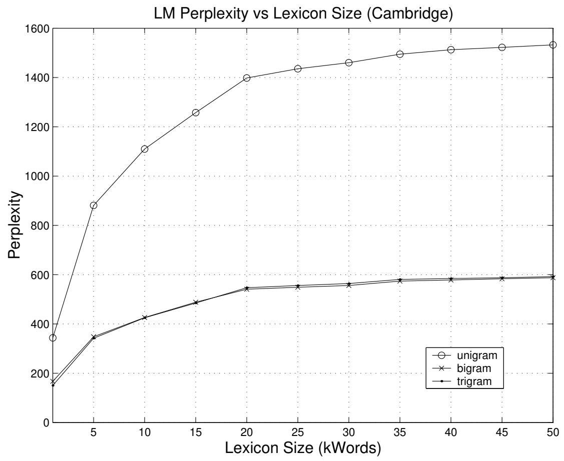

Figure 2. Perplexity. The plot shows the PP of the different models (unigram, bigram and trigram) over the test set.

5.2 N-gram Models Training

The N-gram models used in this work were trained over the TDT-2 corpus [2], a collection of transcriptions from several broadcast and newswire sources (ABC, CNN, NBC, MSNBC, Associated Press, New York Times, Voice of America, Public Radio International).
The transcriptions of the training set of the handwriting database was added to the corpus in order to increase the alignment of the corpus with the texts to be recognized.
For each lexicon described in Section 5.1, three models (based on unigrams, bigrams and trigrams respectively) were created. The plots in Figure 2 show the perplexities of the SLMs as a function of the lexicon size. The perplexity is estimated over the part of the test set covered by the lexicon, without taking into account the Out-Of-Vocabulary words.
From Figure 2 we can conclude that a significant improvement is obtained when passing from unigrams to bigrams, but no further improvement is obtained when applying trigrams. This happens for several reasons. The first one is that the handwritten text is split into lines and only the words after the third one can take some advantages from the trigram model. Since a line contains on average 10 words, this means that only 80% of the data can actually benefit of the trigram model (while 90% of the data can be modeled with bigrams).
The second problem is that the percentage of trigrams covered by the corpus in the test set is only ~40%. This further reduces the number of words for which the trigram model can have a positive effect. The coverage in terms of bigrams is much higher (around 70%) and the percentage of words over which the model can have an effect is around 90%. For this reason, the bigram and trigram models have a similar perplexity.

5.3 Recognition Results

The performance of a text recognition system can be measured in different ways. This depends on the fact that there are three sources of error: substitution (when a word is incorrectly classified), deletion (when a word is lost because it is attached to another one during the decoding), insertion (when a non existing word is added by the decoder). The most realistic measure of the performance is the *recognition rate* which takes into account the three kinds of errors. The recognition rate is given by $100 - s - d - i$ where s, d and i are the substitution, deletion and insertion rate respectively. Note that this can be less than 0 (there is no contraint on the number of insertions). Another performance measure is the *accuracy*, which corresponds to $100 - s - d$. The accuracy accounts for the percentage of words correctly classified, but does not take into account how close the transcription to the actual content of the handwritten text. In the remaining of this work, all the performances are measured in terms of recognition rate.
First, a baseline system (not using SLMs) is obtained. Models with $10 \leq S \leq 14$ and $10 \leq G \leq 15$ are trained over the training set and tested over the validation set (S and G are the number of states and Gaussians in the models respectively). The system having the best recognition rate (over the validation set) is the one having $S = 12$ and $G = 12$. This system is retrained over the union of training and validation set and is used in the actual recognition experiments. For each lexicon, we tested four versions of the system: the

baseline version (without SLMs) and three versions using unigram, bigram and trigram models respectively. The performances of the systems on the test set are reported in Figure 3, where the recognition rate is plotted as a function of the lexicon size. The performance is the result of a tradeoff between the positive effect due to the improvement of the test set coverage and the negative effect due to the increase of the lexicon size.
The use of the N-gram models is shown not only to improve the recognition rate (once the lexicon is big enough), but also to improve the robustness with respect to the increase of the size of the lexicon.
The main source of error is the substitution (around 10%) followed by the insertion (around 5%). The reasons of substitution errors are the same as in single word recognition: short words are more difficult to classify and similar words (e.g. *yield* and *yields*) are confused. The insertion is due to two reasons: the first one (accounting for $\sim$2%) is due to scratchs. They are classified as words that do not exist in the actual transcription of the text. The second one (accounting for a $\sim$3%) is due to the fact that some words are split into two (e.g. *widest* is transcribed as *wide st*).
No deletion error is observed. The separation between following words is in fact too evident to attach them by missing the inter-word blank.

6 Conclusions

This work presented a system for the offline recognition of cursive handwritten lines. The experimental setup has been designed to reproduce the conditions of unconstrained text recognition. The only hypothesis about the text to be transcribed is that it is written in English. This allows the application of Statistical Language Models trained over a corpus supposed to reproduce the average statistics of English.
The SLMs are shown to have a two-fold positive effect: they improve the recognition rate at a given dictionary size and they make the system more robust with respect to an increase of the lexicon size. This is important because the performance is the result of a tradeoff between coverage and dictionary size.
Although a significant difference in perplexity between unigram on one side and bigram and trigram on the other side can be observed, the different language models give more or less the same results in terms of recognition rate. This effect is often observed in the literature [3] and the correlation of perplexity and recognition rate cannot be modeled very clearly. This seems to suggest that it is not useful to increase the order of the model from 1 to 3. On the other hand, in order to have a definitive answer about this problem, more experiments are needed: further work should be done soon on the recognition of multiple writer data using bigger databases. The possibility of a better alignment between SLM and text to be recognized through an adaptation process will be also explored and lines following each other will be concatenated to get more advantage from bigrams and trigrams.

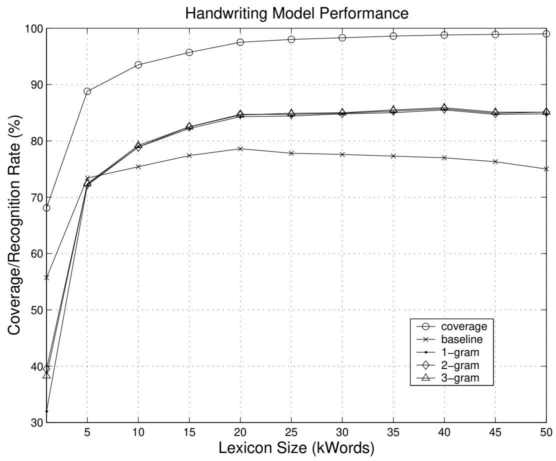

Figure 3. Recognition rate. The plot shows the recognition rate of the baseline system and of the systems using different SLMs.

Acknowledgements This work was done under the grant 21-55733.98 issued by the Swiss National Science Foundation. The authors acknowledge also financial support provided by the Swiss National Center of Competence in Research on Interactive Multimodal Information Management (IM)2.

References

[1] A. El Yacoubi, J. Bertille, and M. Gilloux. Conjoined location of street names within a postal address delivery line. In *Proc. of ICDAR*, volume 2, pages 1024–1027, 1995.
[2] D. Graff, C. Cieri, S. Strassel, and N. Martey. The TDT-3 text and speech corpus. In *Proceedings of Topic Detection and Tracking Workshop*, 2000.
[3] F. Jelinek. *Statistical Aspects of Speech Recognition*. MIT Press, 1998.
[4] U. Marti and H. Bunke. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(1):65–90, 2001.
[5] A. W. Senior and A. J. Robinson. An off-line cursive handwriting recognition system. *IEEE Trans. on Patt. An. and Mach. Int.*, 20(3):309–321, March 1998.
[6] A. Vinciarelli and J. Lüttin. A new normalization technique for cursive handwritten words. *Patt. Rec. Lett.*, 22(9):1043–1050, 2001.