A COMPARISON OF LIGATURE AND CONTEXTUAL MODELS FOR HIDDEN MARKOV MODEL BASED ON-LINE HANDWRITING RECOGNITION

J.G.A. Dolfing

Philips GmbH Forschungslaboratorien
Weißhausstr. 2
D-52066 Aachen, Germany
Email: dolfing@pfa.research.philips.com

ABSTRACT

This paper addresses the problem of on-line, writer-independent, unconstrained handwriting recognition. Based on hidden Markov models (HMM), we focus on the construction and use of word models which are robust towards contextual character shape variations and variations due to ligatures and diacriticals with the objective of an improved word error rate. We compare the performance and complexity of contextual hidden Markov models with a 'pause' model for ligatures. While the common contextual models lead to a word error rate reduction of 12.7%-38% at the cost of almost six times more character models, the pause model improves the word error rate by 15%-25% and adds only a single model to the recognition system. The results for a mixed-style word recognition task on two test sets with vocabularies of 200 (up to 98% correct words) and 20,000 words (up to 88.6% correct words) are given.

1. INTRODUCTION

In this paper, we address the problem of handwriting modeling in the context of an on-line handwriting recognition system based on hidden Markov models. This approach to handwriting is well-established now [2, 4, 6, 9]. The writer-independent system processes characters, words, and sentences. Each character is modeled by a hidden Markov model and a word or sentence is modeled as a sequence of hidden Markov models. We focus on the recognition of mixed-style, handwritten words and exploit knowledge about the handwriting structure to adapt the word models to minimize the character shape variation due to ligatures, to train more accurate models and to minimize the word recognition error rate.

In general, handwriting exhibits a natural variation where the character shape depends on the neighboring characters and the ligatures between the characters. Other handwriting properties include delayed strokes or diacriticals, e.g., the dot on the 'i'. As an example, consider the shape of the 'n' in the cursive, handwritten text 'on' and 'in'. It is obvious that the shape of the 'n' depends on the neighbor character, i.e., the character shape is context-dependent. In this example, the shape variation is mainly caused by the ligature connecting the two characters. Even discrete handwriting without written ligatures exhibits the contextual shape variations of neighboring characters.

Because the recognition depends on the character shapes, and not on the ligature shapes, we attempt to minimize the effect of the ligatures on the character shapes with either contextual or ligature models. First, we employ context-dependent models, which are called trigraphs or bigraphs in handwriting recognition. This approach is very common in speech recognition [5, 7] and is also applied in handwriting recognition [4, 9]. The context-dependent models automatically model the connection with the previous and next character.

Second, we introduce a ligature or 'pause' model similar to [1] which explicitly models the transition between two characters. Therefore, the data describing the character shape itself exhibits less variation. This approach also merges the difference between discrete and cursive style handwriting, i.e., pen-up or pen-down ligatures, in one model. In contrast, contextual models have to include both the discrete and cursive variation of the trigraph. Optionally, the 'pause' model is also used to model delayed strokes, e.g, a dot on the 'i' in the word 'in' which is placed on the 'i' after writing the 'n' and not after the body of the 'i'. Such a model at the end of a word is called a 'backspace' model and was introduced by [9]. The advantage of the approach is that there is no need for explicit detection and preprocessing of the delayed strokes as in [8].

In Section 2, we introduce the properties of our handwriting recognition system and the handwriting data used for training and testing. In Section 3, we discuss the pause model while Section 4 explores the context-dependent models. We conclude with Section 5 where we compare the approaches by means of experimental results.

2. SYSTEM OUTLINE

The platform for capturing handwriting is a Philips proprietary tablet called Philips Advanced Interactive Display (PAID) consisting of an LCD plus orthogonal sensors for pen and finger input sampling a stream of (x, y) coordinates with up to 200 pps. This tablet is connected to a PC with pen-enhanced Unix or (Pen)Windows.

The preprocessing converts the sampled handwriting to the size-independent segment representation and computes 19 components per feature vector [2]. Each lower-case character is modeled by a left-to-right hidden Markov model containing five states with loop, forward and skip transitions probabilities which are estimated during the training process. The observation probabilities are continuous mixtures of Gaussian densities with density-specific diagonal covariance matrices. Up to 32 densities per mixture are allowed. Training of the hidden Markov model parameters is done by using the Maximum Likelihood criterion and applying the Viterbi approximation [7]. Recognition is based on a one-stage beam search algorithm using a tree-organized dictionary and vocabularies of 200 and 20,000 words.

For the experiments reported here the training data consisted of more than 10,000 handwritten words from about 60 writers of several nationalities from on-site collected data and Unipen training data [3]. Two test sets are used written by the same 10 writers. The dataset *Mixed1* contains 500 words while the dataset *Mixed2* contains 2483 words hand-segmented from sentences. The dataset *Mixed2* contains a lot of short words with only two or three characters. A number of samples are shown in Figure 1.

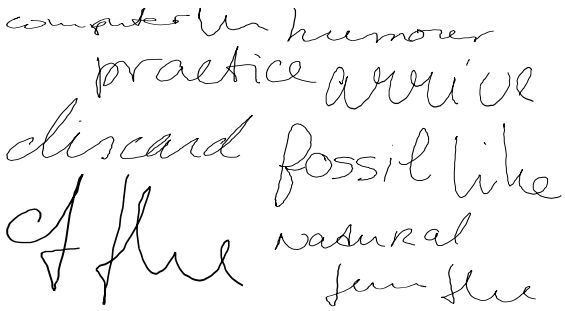

Figure 1: Test set data samples.

3. PAUSE MODEL

Like trigraph modeling, a pause model is an attempt to isolate the variability of character transitions. Consequently, the context-free character models will contain fewer variable shapes. Examples of other approaches where the structure of the hidden Markov model is adapted in order to better match the handwritten input are state-duration modeling, the 'backspace' state [9], and the ligature model [1].

We assume that λ_c is the hidden Markov model of a character c and contains five states. To minimize the character shape variations due to character transitions and ligatures, we introduce a single 'pause' model λ_p which consists of one state. Based on this model, we extend the word model of word w_i with $l(w_i)$ characters from $\lambda_{w_i} = \lambda_{c_1}, \lambda_{c_2}, \ldots, \lambda_{c_{l(w_i)}}$ to $\lambda_{w_i} = \lambda_{c_1}, \lambda_p, \lambda_{c_2}, \lambda_p, \ldots, \lambda_p, \lambda_{c_{l(w_i)}}$ in order to include pauses between subsequent characters. This is graphically shown in Figure 2. The transition probabilities of both the character and 'pause' models are reestimated during the training process.

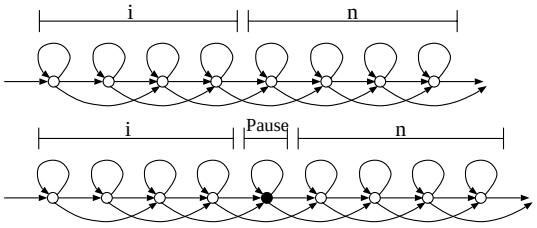

Figure 2: A hidden Markov model of the word 'in' with (bottom) and without (top) an intermediate 'pause' model which is colored black.

In [1], ligature models in an off-line word recognition task are used resulting in an improvement of the recognition accuracy by 20% to 45%, depending on the dictionary size. That approach uses a number of pause models describing categories of character transitions depending on the neighboring characters. In contrast, we model *all* ligatures with the same pause model. This is based on the assumption that the number of ligature types is limited. The number of observations is one for a pen-up ligature and variable for a pen-down ligature.

We experimentally investigated the effect of the 'pause' model on the word error rate. The hidden Markov models are trained based on λ'_{w_i} instead of λ_{w_i} using an otherwise unchanged training procedure. The results of the experiment are summarized in Table 1. The models with 'pause' always produce the best results. The relative reduction in word error rate compared to the 'no-pause' models is 20%-25% for a 200 word vocabulary and around 15% for a 20,000 word vocabulary. An additional experiment where we used the 'pause' model also as 'backspace' model did not result in an improved word error rate.

The interpretation of these results is that the ligature shapes have effect of the recognition performance. If the ligatures are trained together with the character models, the ligatures add an extra source of variation to the character shapes, i.e., the character models include the ligature data as extra noise. The 'pause' model separates the ligature shape variations from the character shape variations. This leads to

Table 1: Recognition results in word error rate [%] using models with and without a 'pause' model and two test sets. The mean error rate per writer is indicated as μ while the standard deviation is indicated as σ.

Model type	**Vocabulary size**	
	Mixed1	
	200	20,000
	μ (σ=2.3)	μ (σ=11.5)
No pause	2.7	16.5
Pause	2.0	13.9

Model type	**Vocabulary size**	
	Mixed2	
	200	20,000
	μ (σ=6.6)	μ (σ=12.1)
No pause	11.0	26.9
Pause	8.7	23.2

more accurate character models and a more accurate recognition.

Finally, we observe that the recognition performance of the dataset *Mixed2* is generally inferior to *Mixed1*. This is due to the average word lengths in the datasets. A short word simply provides less context compared to longer words and the more context, the better the recognition result. A similar effect was observed by [8].

4. CONTEXTUAL MODELS

It is known that character-based, contextual hidden Markov models, such as trigraphs, improve the recognition performance of a writer-dependent handwriting recognition system [4, 9]. However, [9] does not compare contextual with context-free models and therefore, the benefit due to contextual models remains unclear. More recently, [4] applied trigraphs to a writer-dependent, on-line handwritten word recognition task and achieved an error-rate reduction of 50% for a 1000 word vocabulary and 35% for a 30,000 word vocabulary.

A trigraph models a character given both a left and right contextual character. We denote such a trigraph model of a character c as $\lambda_{\{l\}c\{r\}}$. A bigraph is a contextual character model with either left or right context, i.e., $\lambda_{\{l\}c}$ or $\lambda_{c\{r\}}$.

In contrast to the previous section where a word w_i with $l(w_i)$ characters is modeled as $\lambda_{w_i} = \lambda_{c_1}, \lambda_{c_2}, \ldots, \lambda_{c_{l(w_i)}}$, the contextual model of the word w_i is $\lambda''_{w_i} = \lambda_{\{\#\}c_1\{c_2\}}, \lambda_{\{c_1\}c_2\{c_3\}}, \ldots, \lambda_{\{c_{l(w_i)-1}\}c_{l(w_i)}\{\$\}}$ where $\#$ and $\$$ are the markers for word start and end, respectively. Such a word model implicitly models the ligatures between the characters and does not include 'pause' models.

The number of possible trigraphs given 26 lower-case characters is 26^3. In spite of the fact that some character combinations do not occur in English, the number of possible trigraphs is too large to reliably train each hidden Markov model. Therefore, we have to balance the available training data and the number and kind of models which was referred to as 'trainability versus specificity' in [5].

The algorithm to select appropriate trigraphs and bigraphs is simple. We parse all the training text and calculate the number of occurrences of characters, bigraphs and trigraphs. We add a given trigraph to the set of models if its count exceeds a threshold. The threshold has to be high enough, e.g., 30 as a minimum, to guarantee enough training data for each model. Bigraphs are added if their count exceeds the same threshold and if the bigraph is not part of an accepted trigraph. For example, the set of 147 contextual models with at least 200 training samples in Table 2 includes 18 trigraphs, 100 bigraphs, and 26 unigraphs or context-free models.

The training procedure for contextual models is based on the training of the context-free models, without 'pause' model, in the previous section. We determine the set of contextual models as described above. After that, we initialize all models $\lambda_{\{l\}c\{r\}}$ with the parameters of λ_c and continue with a number of additional Viterbi training iterations based on the time-alignment of the last, context-free training iteration.

The results of the writer-independent recognition experiment with vocabularies of 200 and 20,000 words on the basis of *Mixed1* and *Mixed2* with an increasing number of contextual models are summarized in Table 2.

Table 2: Effect of an increasing number of contextual models on the mean word error rate [%]. Results are obtained with a 200 and 20,000 word vocabulary and two data sets. A contextual model is created if at least the number of samples in parentheses is observed in the training material.

Models		**Dataset and vocabulary size**			
#models	**#densities**	**Mixed1**		**Mixed2**	
		200	20K	200	20K
26 (-)	3009	2.7	16.5	11.0	26.9
26 (-)	3946	2.9	13.7	8.1	23.7
66 (300)	8979	2.0	13.5	5.8	21.3
147 (200)	17394	1.8	11.4	6.5	20.7
284 (100)	25763	2.2	13.5	6.8	21.1
476 (60)	31964	3.1	18.8	7.3	23.0
878 (30)	35314	4.3	21.7	8.7	27.8

First, it is observed that the additional training iterations with a constant number of models and based on the com-

puted time-alignment reduce the word error rate. This is shown in Table 2 as the difference between the first and second line of results with 26 models.

Second, the table shows that the word error rate does not decrease monotonously with the increase in the number of models but instead reaches a minimum word error rate and increases again. This effect is expected and caused due to the trade-off between robust and detailed models in combination with the available training data per model [4, 5].

The best results are reached with the use of 147 contextual models, which is a word error rate of 11.4% ($\sigma = 7.7$) and 20.7% ($\sigma = 9.3$) with a 20,000 word vocabulary for the datasets *Mixed1* and *Mixed2*, respectively. The error reduction in the experiment with 147 contextual models and a 20,000 word vocabulary is 16.8% and 12.7% for the data sets *Mixed1* and *Mixed2*, respectively. The experiment with a 200 word vocabulary achieves an error-rate reduction of 38% and 19.8% with *Mixed1* and *Mixed2*, respectively.

5. CONCLUSION

We have compared the effect of pause and contextual models on the word error rate given two different test sets. The pause and contextual models decrease the word error rate by 15%-25% and 12.7%-38%, respectively. Overall, the best word error rates for *Mixed1* and *Mixed2* are 11.4% and 20.7%, respectively.

Table 3: Complexity comparison of contextual and 'pause' models where $l(w_i)$ is the number of characters in word w_i and the word length is given in number of states.

Model type	#Models	#Densities	Word length
Contextual	26	3946	$5 \cdot l(w_i)$
Contextual	147	17394	$5 \cdot l(w_i)$
Pause	26 + 1	3009 + 32	$6 \cdot l(w_i)$ -1

Compared to the pause model, the use of the contextual models leads to a larger word error rate improvement at the cost of a large increase of the number of models. The recognition system using the 'pause' model contains 26 character models plus one additional 'pause' model containing a maximum of 32 densities. The best performing recognition system with contextual models contains 147 character models. Although the number of models increases by almost a factor of six, the number of densities increases only by a factor 4.4, as presented in Table 3, because the trigraphs and bigraphs are often modeled with less densities compared to context-free models. The change in the total number of distance calculations depends on the combined effect of pruning threshold and the more specific contextual models which affect the beam width. The addition of the pause model increases the number of states per word model as presented in Table 3. This means that the search space is enlarged by approximately a factor of $6/5$.

To summarize, the contextual models yield a larger word error rate improvement but increase the number of densities in the recognition system. The use of the pause model improves the word error rate based on only a few additional densities at the cost of a slightly larger search space.

6. REFERENCES

[1] W. Cho, S.W. Lee, and J.H. Kim. Modeling and Recognition of Cursive Words with Hidden Markov Models. *Pattern recognition*, 28(12):1941–1953, 1995.

[2] J.G.A. Dolfing and R. Haeb-Umbach. Signal Representations for Hidden Markov Model based on-line Handwriting Recognition. In *International Conference of Acoustics, Speech and Signal processing*, pages 3377–3380, April 1997.

[3] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. In *12th International Conference on Pattern Recognition*, pages 29–33, 1994.

[4] A. Kosmala, J. Rottland, and G. Rigoll. An Investigation of the Use of Trigraphs for Large Vocabulary Cursive Handwriting Recognition. In *International Conference of Acoustics, Speech and Signal processing*, pages 3373–3376, 1997.

[5] K.F. Lee. *Automatic Speech recognition*. Kluwer Academic Publishers, 1989.

[6] K.S. Nathan, H.S.M. Beigi, J. Subrahmonia, G.J. Clary, and H. Maruyama. Real-time on-line unconstrained handwriting recognition using statistical methods. In *International Conference of Acoustics, Speech and Signal processing*, pages 2619–2622, 1995.

[7] L.R. Rabiner and B.H. Juang. *Fundamentals of speech recognition*. Prentice hall, first edition, 1993.

[8] M. Schenkel, I. Guyon, and D. Henderson. On-line Cursive Script Recognition using Time-delay Neural Networks and Hidden Markov Models. *Machine Vision and Applications*, 8:215–223, 1995.

[9] T. Starner, J. Makhoul, R. Schwartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *International Conference of Acoustics, Speech and Signal processing*, pages 125–128, 1994.