Use of a Confusion Network to Detect and Correct Errors in an On-Line Handwritten Sentence Recognition System

Solen Quiniou, Eric Anquetil

Use of a Confusion Network to Detect and Correct Errors in an On-Line Handwritten Sentence Recognition System

Solen Quiniou Eric Anquetil
IRISA - INSA
Campus de Beaulieu
35042 Rennes Cedex, France
{Solen.Quiniou, Eric.Anquetil}@irisa.fr

Abstract

In this paper we investigate the integration of a confusion network into an on-line handwritten sentence recognition system. The word posterior probabilities from the confusion network are used as confidence scored to detect potential errors in the output sentence from the Maximum A Posteriori decoding on a word graph. Dedicated classifiers (here, SVMs) are then trained to correct these errors and combine the word posterior probabilities with other sources of knowledge. A rejection phase is also introduced in the detection process. Experiments on handwritten sentences show a 28.5 % relative reduction of the word error rate.

1. Introduction

Most handwritten sentence recognition systems use word graphs to represent alternative sentence hypotheses. The standard Maximum A Posteriori (MAP) decoding is then performed to find the sentence with the highest probability given information from the recognition system and a language model. Unlike this approach which aims at minimizing the sentence error rate, another recent approach from speech recognition tries to minimize the word error rate by using *confusion networks* [6, 10].

Confusion networks where introduced in [6]. These networks are used to represent a set of alternative sentence hypotheses and rely on word posterior probabilities. The posterior probability of a word is the sum of the probabilities of all graph paths this word belongs to. These posterior probabilities can then be used to retrieve the best sentence hypothesis or as a confidence measure on words [2]. This confidence measure can also be combined to other knowledge sources in a neural network [5] or a SVM [4].

In this paper, we integrate confusion networks in our handwriting recognition system to improve its performance. To our knowledge, confusion networks have not been used yet in the field of handwritten sentence recognition. This confusion network is jointly used with the MAP decoding, already present in our recognition system [7], to detect potential recognition errors. The word posterior probabilities as well as other sources of knowledge are then used to correct these errors. Moreover, a rejection strategy is introduced to reject words that cannot be corrected by the presented approach. This rejection strategy allows the highlight of misrecognized words in an input interface or enables an additional recognition step on these rejected words.

The remaining parts of this article are as follows. In section 2, an overview of the recognition system is given. Confusion networks are then introduced in section 3 while our proposed detection and correction approach is presented in section 4. Finally, experimental results are discussed in section 5 whereas section 6 draws some conclusions.

2. Overview of the on-line handwritten sentence recognition system

In our sentence recognition system presented in figure 1, each handwritten word of the segmented input sentence $W = w_1 \dots w_n = w_1^n$ is first given to our word recognition system RESIFMot [1]. This system outputs a list of candidate words which are ordered according to their *lexicon score* (depending on edit operations during the lexical post-processing step). A *graphic score* is also added to each candidate word and combines adequation measures between each character and its letter model as well as spatial and statistical information between characters.

A word graph is built using these lists (see part (a) of figure 1) where each node represents a $(n-1)$-gram w_{i-n+1}^{i-1} (and is valuated by the scores of word w_{i-1}) and each edge corresponds to the n-gram w_{i-n+1}^{i} (and is valuated by a language model probability).

The Viterbi algorithm is then performed to find the likeliest path in the word graph (each path corresponding to a sentence), by combining graphic and linguistic information as given by equation 1:

$$\hat{W} = \arg\max_{W}\ score(S|W) + \gamma \log\left[p(W)\right] \quad (1)$$

where $score(S|W)$ is the score of the signal S for the given sentence W and is estimated by the recognition system (combining graphic and lexicon scores); $p(W)$ is the a priori probability of the sequence W, given by a statistical language model. A language weight γ is used to balance the influence of the language model against the score from the recognition system [8].

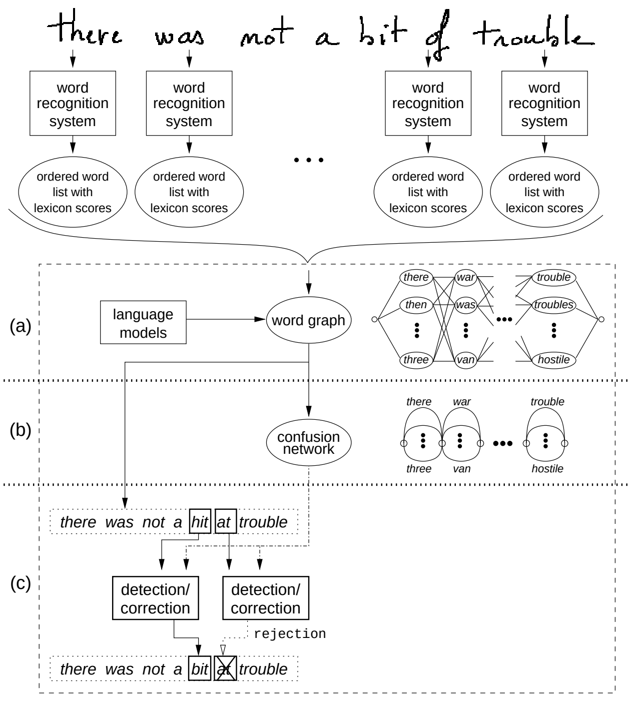


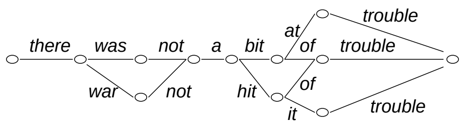
Figure 1. Sentence recognition system.

After describing our recognition system, we will present the different steps of the confusion network construction.

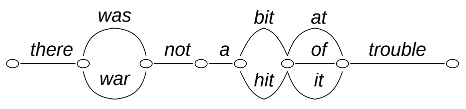
3. Confusion networks

3.1. Motivation

The standard approach to sentence recognition is the MAP approach which finds the sentence with the highest posterior probability. However, the most commonly used metric to evaluate the performance of a recognition system is the word error rate. There is thus a mismatch between the recognition approach and the performance metric. Unlike the MAP approach, confusion networks aim at maximizing word posterior probabilities. The nodes in a confusion network represent equivalence classes (*confusion sets*) *i.e.* confusions between word hypotheses for a given position in the output sentence and adjacent nodes are linked by as many edges as word hypotheses (see figure 2).



(a) Word graph



(b) Confusion network

Figure 2. Word graph and corresponding confusion network.

3.2. Algorithm

A confusion network is built by first aligning hypotheses from a word graph where each edge stands for a word hypothesis, as can be seen in figure 2(a). Posterior probabilities are then computed for every edge of the graph. Equivalence classes are then initialized such that each class consists of the links with the same starting and ending times as well as the same word label (an edge pruning step can be introduce beforehand to remove links whose posterior probability is too low). Afterward, the intra-word clustering aims at merging classes with the same word label and with overlapping starting and/or ending times. The inter-word clustering is finally done to group together classes corresponding to different words, based on their graphic similarity essentially. This leads to a confusion network as the one from figure 2(b). The hypothesis with the lowest word error rate can then be extracted by picking the word with the highest posterior probability in each confusion set, at each position in the sentence. This hypothesis is called the *consensus hypothesis*.

In order to focus only on the impact of the confusion network, handwritten sentences, in this paper, are manually segmented to introduce no bias due to incorrect segmentation. The construction of the confusion sets is then straightforward and only word posterior probabilities have to be computed, from the link posterior probabilities.

3.3. Word posterior probabilities

The posterior probability of a word is the sum of all link posterior probabilities whose labels are the considering word. The computation of the probability of the k^{th} link of word $w_{i,j}$ can be done efficiently with the forward-backward algorithm and is given by equation 2:

$$p(w_{i,j}^{(k)}) = \frac{\alpha(n_{i-1,k})\, p(w_{i,j}|w_{i-1,k})^{\gamma}\, score(w_{i+1,k})^{\delta}\, \beta(n_{i,j})}{\sum_l \sum_m \sum_r p(w_{l,m}^{(r)})} \quad (2)$$

where $w_{i,j}^{(k)}$ is the link corresponding to k^{th} hypothesis of word $w_{i,j}$, $\alpha(n_{i-1,k})$ is the forward probability of node $n_{i-1,k}$, $\beta(n_{i,j})$ is the backward probability of node $n_{i,j}$, $p(w_{i,j}|w_{i-1,k})$ is the probability of bigram $w_{i-1,k}w_{i,j}$ given by the language model and $score(w_{i+1,k})$ is the score given by our word recognition system to word $w_{i+1,k}$ and combines the lexicon and graphic scores presented in section 2 (since our recognition system is non-probabilistic, this score is brought back to range [0:1] and can be assumed to be a likelihood). γ and δ are weights to scale the language model probability and the recognition score respectively and are optimized on a validation set.

The word posterior probability is given by equation 3:

$$p(w_{i,j}) = \sum_k p(w_{i,j}^{(k)}). \quad (3)$$

After presenting confusion networks and the computation of word posterior probabilities, we describe our approach to use a confusion network in conjunction with the word graph, to detect and correct decoding errors.

4. Error detection/correction using posterior probabilities of recognized words

The detection and eventually correction steps are illustrated in part (c) of figure 1. The confusion network is used to measure the confidence in each word of the best sentence hypothesis given by the Viterbi decoding on the word graph. When an error is detected, a correction step is performed to retrieve the correct word, using this word posterior probability as well as different sources of knowledge.

In the following subsections, we describe our approach to detect and correct errors on the output words. Two types of errors are considered and, for each one, a dedicated classifier is trained to correct the associated error type. An additional rejection strategy is presented in the last subsection.

4.1. Mismatch with word whose posterior probability is the highest

In this first type of error detection, the considered output word is detected as potentially misrecognized if its word posterior probability is not the highest one in its corresponding confusion set. In other words, the words from the sentence given by the Viterbi algorithm and from the consensus hypothesis are not the same. To correct the potential error, a dedicated classifier is trained to chose between two words: the recognized word and the one with the highest posterior probability. Three features are considered for each word, namely its posterior probability as well as its normalized lexicon score and its normalized graphic score (see section 2 for a description of these latter scores).

4.2. Word with unreliable posterior probability

The second type of detected errors concerns recognized words whose posterior probability is the highest in its confusion set but this probability is below a certain threshold (here, set to 0.8) and is thus considered as unreliable. For the corresponding correction phase, a dedicated classifier is also trained and aims at discriminating the top 2 words of the confusion set (ranked by their decreasing posterior probabilities). The same features as previously are used for each of these two words.

4.3. Rejection

In the two correction strategies, we assume that the correct word is one of the two words given as inputs of each dedicated classifier. In fact, the correct word can be either in the remaining words of the confusion set or even not in the confusion set. This simplification was done because most correct words are one of those two words. Indeed, 93.3 % of the words to recognize which appear in the confusion network and which are detected by the strategy presented in section 4.1 are one of the two words considered as inputs of the correction classifier. In the same way, 88.4 % of present words detected by the second strategy (see section 4.2) are one of the two input words. Moreover, there is a problem of data sparsity since by considering more words as inputs of the dedicated classifiers, more training data will be needed to obtain reliable classifiers.

This rejection strategy thus aims at discriminating words where the correct word is one of the two considered words from those where the correct one is not one of them. A new classifier with 6 inputs (corresponding to the 3 previous features for each of the two words) and 2 outputs (acceptation and rejection) is then trained for each detection strategy.

We will compare these strategies in the next section, after describing our data.

5. Experiments and results

5.1. Linguistic and handwritten data

The language model is a bigram model extracted from the Brown corpus [3], with the SRILM toolkit [9]. This corpus contains 52,954 sentences (1,002,675 words) where

46,836 sentences (900,108 words) were used to learn the model. The associated lexicon includes 13,748 words.

The handwritten material consists of 118 different sentences from the remaining part of the Brown corpus. These sentences were segmented manually to introduce no bias due to incorrect segmentations.

The training set includes 398 sentences (6,417 words) written by 17 writers (this set is used to learn the classifiers as well as to tune parameters for both the Consensus and the Viterbi algorithms) whereas the test set includes 320 sentences (5,068 words) written by 10 writers. The writers of the test set are different from those of the training set.

5.2. Confusion networks

Table 1 compares the results on using the Viterbi algorithm on the word graph or the Consensus algorithm on the confusion network to find the optimal decoding sentence. We remind the achieved rate of the baseline system which uses no language model. The improvement with the Viterbi algorithm is higher than the one with the Consensus algorithm, compared to the baseline system, leading to a 45.7 % reduction of the word error rate on the test set. The fact that the Consensus algorithm performs slightly worse than the Viterbi algorithm can be explained by the already high recognition rate obtained with the Viterbi algorithm. Indeed, the correlation between word error rate and sentence error rate was shown to be much stronger when the word error rate is low [2] so the benefit of switching from the Viterbi algorithm to the Consensus algorithm is reduced. Furthermore, since our word recognition system is not a probabilistic one, the current combination of the graphic model and the linguistic model may not be optimal.

Table 1. Word recognition rates with the Viterbi and the Consensus algorithms.

	training set	test set	w.e.r. rel. decrease
Baseline	75.9 %	81.3 %	-
Viterbi	86.0 %	89.9 %	45.7 %
Consensus	85.7 %	89.5 %	44.0 %

In the following experiments, the word posterior probabilities computed on the confusion network are used to evaluate the reliability of the output sentence from the Viterbi algorithm. The quality of word posterior probabilities as confidence scores is evaluated by the normalized cross entropy (NCE) [4]. The computed NCE is then 0.2. The classifiers used to correct the so-detected errors are *Support Vector Machines* with gaussian kernels. We chose SVMs because of their capacity to deal with unbalanced classes (in terms of training data) and because of their efficiency.

5.3. Decision based on mismatched words

This first detection strategy (see section 4.1) allows the detection of 3.0 % of the words of the test set, which represents 18.1 % of the word errors made by using the Viterbi algorithm. Table 2 gives the word recognition rate over both selected words (*ie* words selected by this type of decision) and present words (*ie* selected words where the correct word is one of the two words to classify), for different strategies. The proposed correction strategy leads to a decrease of 43.9 % of the errors among the present words compared to the Viterbi algorithm alone.

Table 2. Word recognition rates for the decision based on mismatched words.

Strategy	over sel. words	over pres. words
Viterbi	38.0 %	58.2 %
Detection/correction	50.0 %	76.5 %
Det./corr. & rejection M	49.3 %	75.5 %

Figure 3 plots the ROC curve for different rejection classifiers, for each of the two detection strategies. ROC curves shows the compromise between unrejected correct words (TAR) and words to reject and not effectively rejected (FAR). The point M corresponds to the rejection classifier chosen for the current approach. This classifier allows the rejection of 12.0 % of the selected words with a TAR of 99.0 %. The recognition rate is then 49.3 %.

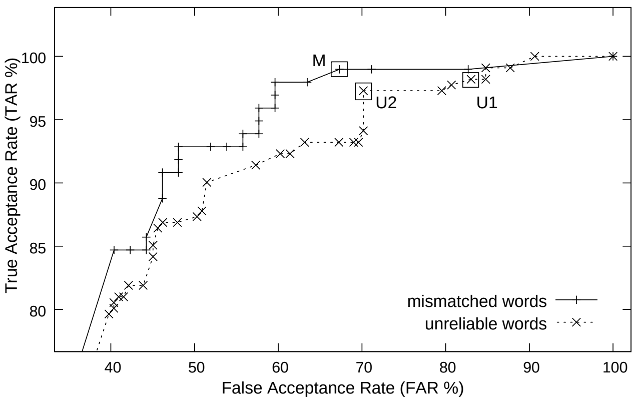


Figure 3. ROC curves to reject words in each decision strategy.

5.4. Decision based on unreliable words

This second detection strategy (see section 4.2) allows the detection of 7.7 % of the words of the test set, representing 44.4 % of the errors made by using the Viterbi algorithm. Table 3 shows word recognition rate over both selected and present words, for the different recognition strategies. This correction strategy on unreliable words reduces the error rate over present words by 17.5 %.

Among the rejection classifiers plotted in figure 3, for this detection strategy, we consider two of them (denoted by points U1 and U2 on the figure). This allows us to consider two rejection rates on the words. Thus, with the first rejection classifier, 8.4 % of the selected words are rejected with a TAR of 98.6 % and the recognition rate is then 43.6 %. The second rejection classifier leads to a higher rejection rate of 14.5 % with a TAR of 98.2 %. The corresponding recognition rate decreases a little to reach 43.4 %.

Table 3. Word recognition rates for the decision based on unreliable words.

Strategy	over sel. words	over pres. words
Viterbi	41.8 %	74.2 %
Detection/correction	44.4 %	78.7 %
Det./corr. & rejection U1	43.6 %	77.4 %
Det./corr. & rejection U2	43.4 %	76.9 %

5.5. Global decision combination

The combination of the two previous detection strategies enables the selection of 10.7 % of the words in the test set. These selected words represents 62.5 % of the errors made by using the Viterbi algorithm. Table 4 shows the recognition rate over the selected words, the present words as well as over all the words of the test set. The global correction strategy leads to a 90.4 % recognition rate over all the words. In fact, the achievable recognition rate is 91.8 % *ie* all the present words for each of the two strategies are correctly recognized. Thus, the reduction of the word error rate towards this achievable error rate is 28.5 %.

Table 4. Word recognition rates for the global decision.

Strategy	over all words	over sel. words	over pres. words
Viterbi	89.9 %	40.8 %	69.3 %
Detection/correction	90.4 %	45.9 %	78.7 %
Det./corr. & rej. M+U1	90.3 %	45.2 %	76.8 %
Det./corr. & rej. M+U2	90.3 %	45.0 %	76.5 %

When incorporating the rejection strategies into the global decision (by using rejection classifier M and either classifier U1 or U2), the decrease in the word recognition rate is small. However, the first rejection strategy allows a 13.6 % reduction of all errors (*ie* word errors as well as correctly rejected words) whereas the second rejection strategy leads to a 17.7 % decrease of all the errors.

6. Conclusion

In this paper, we have presented the integration of a confusion network in a sentence recognition system where a word graph is already used. The word posterior probabilities computed on this confusion network were then used as a confidence measure on the words of the sentence recognized by the Viterbi algorithm performed on the word graph. Two types of potential errors were thus highlighted and considered. A correction step was also added to correct each type of detected errors, by using SVMs. This correction step allowed the reduction of errors on the words of the sentence from the Viterbi algorithm. Finally, a rejection step was integrated into each of the two decision types to identify words detected as potentially misrecognized but which cannot be corrected by the corresponding correction step. These words can then be highlighted to a user in an input interface or submitted to another recognition step.

Future work will also investigate improvements on the correction of already detected errors as well as a finest detection of potential errors. The sentence segmentation will also be introduced in the confusion network, leading to considering the first steps of the construction of the network.

References

[1] S. Carbonnel and E. Anquetil. Lexicon Organization and String Edit Distance Learning for Lexical Post-Processing in Handwriting Recognition. In *9th IWFHR*, pp 462–467, 2004.

[2] G. Evermann and P. Woodland. Large Vocabulary Decoding and Confidence Estimation Using Word Posterior Probabilities. In *25th ICASSP*, pp 1655–1658, 2000.

[3] W. Francis and H. Kucera. *Brown Corpus Manual*. Brown University, 1979.

[4] D. Hillard and M. Ostendorf. Compensating for Word Posterior Estimation Bias in Confusion Networks. In *31st ICASSP*, pp 1153–1156, 2006.

[5] T. Kemp and T. Schaaf. Estimating Confidence Using Word Lattices. In *5th Eurospeech*, pp 827–830, 1997.

[6] L. Mangu. *Finding Consensus in Speech Recognition*. PhD thesis, Johns Hopkins University, 2000.

[7] S. Quiniou and E. Anquetil. A Priori and A Posteriori Integration and Combination of Language Models in an On-line Handwritten Sentence Recognition System. In *10th IWFHR*, pp 403–408, 2006.

[8] S. Quiniou, E. Anquetil, and S. Carbonnel. Statistical language models for on-line handwritten sentence recognition. In *8th ICDAR*, pp 516–520, 2005.

[9] A. Stolcke. SRILM - An Extensible Language Modeling Toolkit. In *7th ICSLP*, pp 901–904, 2002.

[10] J. Xue and Y. Zhao. Improved Confusion Network Algorithm and Shortest Path Search from Word Lattice. In *30th ICASSP*, pp 853–856, 2005.