IMPROVED PERFORMANCE AND GENERALIZATION OF MINIMUM CLASSIFICATION ERROR TRAINING FOR CONTINUOUS SPEECH RECOGNITION

Darryl W Purnell and Elizabeth C Botha
Department of Electrical and Electronic Engineering
University of Pretoria, Pretoria, 0002
South Africa
Email: darryl@ee.up.ac.za, botha@ee.up.ac.za

ABSTRACT

Discriminative training of hidden Markov models (HMMs) using segmental minimum classification error (MCE) training has been shown to work extremely well for certain speech recognition applications. It is, however, somewhat prone to overspecialization. This study investigates various techniques which improve performance and generalization of the MCE algorithm. Improvements of up to 7% in relative error rate on the test set are achieved.

Keywords — speech recognition, discriminative training, minimum classification error, overspecialization, overtraining

1. INTRODUCTION

Discriminative training of hidden Markov models (HMMs) using minimum classification error (MCE) has been used in several speech recognition tasks with much success. Tasks where MCE training has been used to improve recognition performance include: connected digit recognition [1, 2, 3], the English “E”-set {b,c,d,e,g,p,t,v,z} [2], speaker adaptation [4] and continuous speech [3, 5, 6]. The MCE algorithm, however, suffers from overspecialization, as most discriminative training algorithms do.

We focus on continuous speech recognition. This study therefore uses string-level MCE [5, 6, 1], which applies the MCE criterion function at a string level, as opposed to phoneme level. We use the TIMIT acoustic-phonetic database [7] to experimentally determine the effectiveness of our work.

In this study, we propose and test several modifications to the MCE algorithm which improve performance and generalization. Firstly, we investigate the effect of breaking long utterances up into words and performing MCE training using those words. Secondly, we penalize small variances (a result of overspecialization) by adding a penalty term to the criterion (loss) function. We also propose adding a weighted likelihood term to the loss function. These modifications to MCE are shown to improve performance markedly, with up to 7% relative improvement in error rate over standard MCE.

The organization of this paper is as follows. The MCE procedure is introduced in Section 2. The extension of MCE to strings, as opposed to phonemes, is also described, as well as the concept of overspecialization. Section 3 describes the proposed modifications to MCE. The modifications are experimentally compared in Section 4. Finally, we summarize the results and conclude in Section 5.

2. MCE TRAINING

The aim of minimum classification error training is to correctly discriminate the observations of an HMM for best recognition results and not to fit the distributions to the data. An optimization criterion is therefore defined, which will give a reasonable estimate of the error probability. The optimization criterion is defined in terms of a misclassification measure, which provides a measure of how a class was classified.

Given that \( g_i(O, \lambda_j) \) is the log-likelihood of the input utterance or observation sequence \( O = \{o_1, o_2, \ldots, o_n\} \) for the \( j \)-th model \( \lambda_j \), the misclassification measure is defined [1] as:

\[
 d(O) = -g_i(O; \lambda_i) + \log \left( \frac{1}{N} \sum_{j \neq i} \exp \left( g_j(O; \lambda_j) \right) \right), \tag{1}
\]

where \( \eta \) is some positive number, and \( N \) is the number of \( N \)-best incorrect classes which are used in the misclassification measure. The misclassification measure is a continuous function of the classifier parameters. A value of \( d(O) > 0 \) implies a misclassification and \( d(O) < 0 \) means that the correct model was recognized. The value of \( \eta \) influences the behavior of the right-hand term in Equation 1; for \( \eta = \infty \) the term becomes \( \max_j g_j(O; \lambda_j) \) (where \( i \) is the correct class). A large \( \eta \) will result in the misclassification measure only incorporating the closest incorrect class, whereas a small \( \eta \) will include contributions from all of the incorrect classes.

A loss function ([1] based on \( d(O) \) must be defined [1] (a smoothed zero-one function), to be used as the optimization criterion. A popular choice for the loss function is the sigmoid function, given by:

\[
 l(d) = \frac{1}{1 + e^{-\delta d}}, \tag{2}
\]
where \( \theta \) is typically set to zero (or slightly smaller than zero) and \( \gamma \) is set to be greater than zero. A loss value greater than 0.5 indicates a misclassification has occurred (assuming \( \theta = 0 \)). Whether a zero-one loss function is truly necessary is questionable, and we experimentally compare the utility of a sigmoid loss function in Section 4, as opposed to using \( d(O) \) directly as optimization criterion.

Minimizing the loss function will result in the number of
misclassifications being minimized. The generalized proba-

The derivation of the derivatives for the loss function with
respect to the various HMM parameters is beyond the
scope of this article and the reader is referred to [1, 8]
for a detailed description of the MCE procedure.

2.1. String MCE

MCE can be applied at the level of various speech units,
such as phonemes, words and sentences. The application
of MCE to long strings of phonemes or words is known as
string-level MCE [1, 6]. This is as opposed to label-based
MCE, where MCE is applied individually to the phoneme
or word labels. String-level MCE attempts to increase the
recognition performance of the entire string, and therefore
indirectly improves the continuous phoneme recognition.
MCE has the unfortunate tendency to overtrain (overfit),
even for string-level MCE which is less prone to overtrain-
ing than label-based MCE. We use the N-best search pro-
based on a small set of training examples [9] to generate the N-best alignments.

2.2. Overspecialization

Overspecialization (or overtraining) occurs in most training
algorithms where a finite number of examples are available for training. If the training data set was perfectly representative of the test set (this would only truly occur when an infinite number of training examples were available), there would be no difference between training set and testing set performance. However, in practice data sets are limited and the test set performance tends to be worse than the training set performance. This is a result of the model becoming too specialized and not generalizing well.

Another result of overspecialization, is a decrease in testing set performance after maximum performance has been attained (in terms of training time). Minimum classification error training suffers from both these problems; see Figure 2 in Section 4.

Figure 1 shows the typical form of results obtained for the training and testing sets (a). Limiting specialization of the classifier would result in reducing the difference between training and testing set error rates. We would, for example, prefer result (b) in Figure 1 to result (a). We also wish to limit the degradation in performance after maximum performance has been reached. An algorithm which had the characteristics of (c) in Figure 1 would be advantageous in that a cross-validation set would not be required to choose the best model in an unbiased way.

3. IMPROVEMENTS TO MCE

3.1. Word MCE

Presenting arbitrarily long strings to the string-level MCE algorithm is not optimal. Errors occurring earlier during recognition of a string undoubtedly influence the recognition for the rest of the string. Our confidence in the accuracy of segmentation and classification after an error has occurred will therefore tend to be low. As the N-best string outputs from the recognizer are used as discriminative training examples, the number of incorrect strings are limited. Most of these “incorrect” strings differ only in a few places, resulting in only a few potential errors being addressed during discriminative training.

To improve the above, we investigate presenting smaller word-based strings to the string-level MCE algorithm. This is particularly appropriate when training speech recognizers on speech databases which have long sentences.

3.2. Variance penalty

To reduce overfitting, we propose a penalty term propor-
tional to the sum of the square (or power) of the inverse of
the variances (precisions) of the pdf’s of the HMM states. This is as expressed in Equation 3, which is added to the loss function of MCE (Eq. 2). We add \( 1 + \rho \sigma_{ji}^2 \) to ensure that the gradient is finite for \( \sigma_{ji} = 0 \).

\[
\alpha \sum_{\sigma} (1 + \rho \sigma_{ji}^2)^{-\xi}
\]  

\( (3) \)
This results in what could be called “precision decay”, thereby ensuring that variances do not become too small. This has the indirect consequence of reducing overfitting. We empirically found that $\xi = 3$, $\rho = 1000$ and $\alpha = 0.1$ produced particularly good results.

### 3.3. Weighted likelihood

One potential problem with string-level MCE is that parts of the training string that do not result in errors are effectively ignored. Focusing solely on errors will result in overspecialization. We therefore propose adding a weighted likelihood term (of the correct class) to the MCE loss function. This will tend to reinforce correct substrings, while still penalizing errors. The misclassification measure in Equation 1 then becomes

$$d(O) = -(1 + \kappa)g(O; \lambda) + \log \frac{1}{N} \sum_{j \neq i} \phi^j(O; \lambda)^1 / \eta,$$

where $\kappa$ is the weighting of the additional likelihood term ($g(O; \lambda)$). Although we cannot mathematically justify this modification when using a smoothed zero-one loss function, it can, however, be implemented as a simple heuristic where the gradient for the correct class is simply multiplied by a weighting factor $(1 + \kappa)$.

### 4. EXPERIMENTAL EVALUATION

The TIMIT database [7] was used to evaluate the modifications discussed above. The full TIMIT training and test set were used throughout. Following convention, we recognize the standard 39-phone set. The speech signal is blocked into frames of length 16ms, with overlap of 6ms. Thirteen Mel-frequency cepstral coefficients (MFCCs), along with their first and second order differentials are used. Each phone is represented by a simple left-to-right, 3 state, 5 mixture HMM. The accuracy of the system is reported, where accuracy is defined as $(\text{correct} - \text{substitutions} - \text{insertions} - \text{deletions}) / \text{total}$.

We use the online descent algorithm, as opposed to deterministic (batch) gradient descent, as we have found it to be considerably faster in terms of training time. The values of the sigmoid function parameters used in this study are $\theta = 0$ and $\gamma = 0.01$. When using a sigmoid function in the MCE algorithm a learning rate ($\epsilon$) of 1.0 was found to work well. When used without a sigmoid function, a learning rate of 0.001 was used. Note that the maximum of the derivative of the sigmoid function is equal to 0.25$\gamma$, which explains why the learning rate is much higher when using a sigmoid function.

Figure 2 presents a comparison between MCE with a sigmoid loss function and MCE without a sigmoid loss function. We have found, as seen in Figure 2, that using a sigmoid loss function holds little or no advantage for our application.

The improvement in test set performance when using word strings is marked (more than 6% reduction in error rate), as can be seen in Figure 3, where comparative results for sentence- and word-level MCE are presented. This confirms the assumptions made in Section 3.1. The results are similar, but slightly worse when using a sigmoid loss function.

Figure 4 shows the advantage of using the penalty term defined in Section 3.2. Here we have used $\xi = 3$, $\rho = 1000$ and $\alpha = 0.1$, and no sigmoid loss function. The results are similar when using a sigmoid loss function. The test set results are considerably better (2.4% relative improvement in error rate), while there is little difference in the training set results.
Fig. 4. Training and testing set performance for standard string MCE and modified MCE with variance penalty (MCE+VPEN), no sigmoid loss

The weighted likelihood term, discussed in Section 3.3, too has a role to play. It, unlike the variance penalty term, does not improve performance to any significant degree. It does, however, limit degradation of performance after reaching a maximum. Figure 5 shows how the addition of the weighted likelihood term slightly improves maximum performance, but more significantly, stops degradation of performance after peaking at 3-4 epochs. Results when not using a sigmoid loss function are similar with a slightly better error rate being achieved.

Table I gives a summary of the results when using the different modifications. MCE alone produces a 17.7% relative reduction in error rate over baseline maximum likelihood (ML). However, employing the modifications results in a relative reduction in error of up to 23.3% being attained.

5. CONCLUSION

Significant improvements in performance on the testing sets are obtained using the modifications to MCE as proposed. The variance penalty-based approach, and word-level MCE in particular resulted in significantly better test set performance, with limited or no difference to the training set performance. The weighted likelihood approach proposed, stopped degradation of performance after a maximum had been reached. The modifications proposed are relatively simple to implement and limit overspecialization to a large degree.

6. REFERENCES