Boosting with Anti-models for Automatic Language Identification

Xi Yang\textsuperscript{1}, Man-hung Siu\textsuperscript{1}, Herbert Gish\textsuperscript{2}, Brian Mak\textsuperscript{1}

\textsuperscript{1}Hong Kong University of Science and Technology \textsuperscript{2}BBN Technologies
yangxii2004@gmail.com, msiu@ieee.org, hgish@bbn.com, mak@cse.ust.hk

Abstract

In this paper, we adopt the boosting framework to improve the performance of acoustic-based Gaussian mixture model (GMM) Language Identification (LID) systems. We introduce a set of low-complexity, boosted target and anti-models that are estimated from training data to improve class separation, and these models are integrated during the LID backend process. This results in a fast estimation process. Experiments were performed on the 12-language NIST 2003 language recognition evaluation classification task using a GMM-acoustic-score-only LID system, as well as the one that combines GMM acoustic scores with language model scores from GMM tokenization. Classification errors were reduced from 18.8\% to 10.5\% on the acoustic-score-only system, and from 11.3\% to 7.8\% on the combined acoustic and tokenization system.

Index Terms: Language identification, Boosting, Discriminative training

1. Introduction

Combining classifiers to improve classification has been widely used in both the machine learning community and speech processing community. Examples include ROVER \cite{1} and consensus networks \cite{2} in speech recognition and ensemble approaches, such as bagging and boosting \cite{3} in machine learning. One widely used boosting technique is ada-boost \cite{4}. It combines a sequence of weak-learners, which are identical in structure, and are trained sequentially. Mis-classified training samples at one stage are weighted more heavily when training the next learner. This emphasis of the mis-classified training adds a new learner in the space near the decision boundary that has not been well separated by the classifier yet. In \cite{5}, it is shown that ada-boost can be viewed as a stage-wise optimization of the additive logistic regression, which is one type of discriminative classifiers.

In this paper, we apply boosting to language identification (LID) by augmenting the traditional Gaussian Mixture Model (GMM) classifier with two sets of boosted models: language-specific \textbf{boosted target models}, and language-specific \textbf{boosted anti-models}. Anti-models can separate near competitors, and are used to improve class separation, and these models are integrated during the LID backend process. This results in a fast estimation process. Experiments were performed on the 12-language NIST 2003 language recognition evaluation set. With less than 5 iterations, the proposed algorithm significantly improved LID performance. On the acoustic only GMM system, the classification error and verification equal error rate (EER) were reduced from 18.8\% error and 7.3\% EER to 10.5\% error and 3.4\% EER, respectively. These represent a relative improvement of 40-50\%.

The rest of the paper is organized as follows. In Section 2, we describe our proposed boosted target and anti-model algorithm. In Section 3, we describe our baseline LID systems, as well as experimental results of applying the boosted target and anti-models. We conclude the paper in Section 4.

2. Proposed Approach

Boosting has been applied to different speech processing problems \cite{9, 10, 11, 12}. One advantage of applying boosting is the minimal modification needed for classifier training. It can also be applied to LID to create a discriminatively trained classifier.

The acoustic LID system was first proposed in \cite{13} where language-specific acoustic models are represented by Gaussian mixture models; each can have thousands of mixture components. The GMM parameters are typically learned via ML estimation, or via Bayes adaptation from a set of language-independent GMMs.

In \cite{6, 7}, anti-models were proposed. In a typical LID system, each target language is represented by a set of target GMMs, and classification decisions are made by the comparison of the likelihoods of these target GMMs. While competitor information can generally be captured by models from other classes, target-language specific anti-models can be used to capture data that are close to the decision boundary. One anti-model is created for each target language and is trained with data not belonging to the target language. During classification, they generate a normalizing log likelihood. Specifically, denote \( f_{t}(l, x_t) \) to be the target log likelihood of data \( x_t \) for language...
l, and \( f^{-}(l, x_l) \) to be log likelihood of the anti-model of language \( l \). The classification score would now becomes
\[
s(l, x_l) = f^{+}(l, x_l) - w_l f^{-}(l, x_l),
\]
where \( w_l \) is a weighting factor to balance the relative importance of the two models.

The underlying motivation for the boosting approach is to increase the contributions of mis-classified training data to improve classification performance. With ada-boost, it is common to run hundreds of iterations and each training iteration involves the same amount of data (although weighted differently) as in the original classifier. Directly applying this to LID can be computationally expensive both for training and test. There are two ways to reduce the computation complexity of the component classifier. One is to use smaller models such that the computation per iteration can be reduced. Alternatively, one can use less data by removing training samples that are reliably classified [5]. In our work, both approaches are adopted.

Both the target and anti-models can also be “boosted” under the boosting framework, in which a sequence of boosted target and anti-models are created, and their contributions combined. When there is a classification error on \( x_i \), say, class \( j \) mis-classified as \( j' \), the boosted target model approach would improve the likelihood of model \( l \) by putting \( x_i \) into the training of boosted target model \( l \) in the next iteration. To reduce the chance of \( x_i \) being classified as class \( j \), one boosts the anti-model \( j \) by putting \( x_i \) into its training in the next iteration. Note that the training for the anti-model \( j \) comes from training data of other classes that were mis-classified as class \( j \).

The anti-model can be used independently from or together with boosted target model. In our work, they are used together. The ensemble classifier is formed by the combination function \( L() \), which denotes the score combination function with LDA+Gaussian backend as described in [8]. Denote \( y_t \) to be the true class for data \( x_i \). The procedure for building the boosted target and anti-models is as follows.

1. Use the ML-trained, language specific GMMs as the initial base classifier, \( f_0(x) \).
2. Build the initial ensemble classifier, \( F_0(x) = L(f_0(x)) \).
3. For \( m = 1 \) to \( M \),
   (a) Classify all the training samples, \( x_i \)'s, to form the classification decision
   \( l_m(x_i) = L(f_0(x_i), \ldots, f_{m-1}(x_i)) \).
   (b) Define two sets of weighting function, \( \{ w_{l,t,m}^{(a)} \} \) and \( \{ w_{j,t,m}^{(a)} \} \). For the boosted target model, if \( x_i \) is mis-classified, set the sample weight to 1. That is,
   \[
   w_{l,t,m}^{(a)} = \begin{cases} 1 & \text{if } l_m(x_i) \neq y_t \\ 0 & \text{otherwise} \end{cases}
   \]
   For the boosted anti-model, if \( x_i \) is mis-classified as class \( j \) \( l_m(x_i) \neq y_j \), set the sample weight for training anti-model \( j \) to 1. That is,
   \[
   w_{j,t,m}^{(a)} = \begin{cases} 1 & \text{if } j = l_m(x_i) \neq y_j \\ 0 & \text{otherwise} \end{cases}
   \]
   (c) Train the new boosted GMMs, \( f_m(x) \) with the boosted target model using the weight set \( \{ w_{l,t,m}^{(a)} \} \), and the boosted anti-model with the weight set \( \{ w_{j,t,m}^{(a)} \} \).
   (d) Build the new ensemble classifier
   \[
   F_m(x) = L(f_0(x), \ldots, f_m(x))
   \]
   by estimating the LDA transform and Gaussian classifier. Scores from different component classifiers are concatenated as a super-vector before multiplying with the LDA transform. Note that in general, the anti-models will produce a negative contribution to the target class likelihood but the exact weights are produced via the LDA transformation.

4. Build the final classifier as \( F(x) = F_M(x) \).

Similar to the ada-boost, the weights on the training samples are determined by whether \( x_i \) is correctly classified by the most recent ensemble classifier, \( F_{m-1}(x_i) \). Parameters in \( L() \) can either be estimated from the same training set or from a set of held-out data. Note that the boosted target and anti-models can be used separately by using either set of weights in step 3b.

3. Experiments

Experiments were performed on the NIST 2003 language recognition evaluation (LRE) data. The NIST 2003 evaluation set contains data from twelve languages, including: American English, Arabic, Farsi, Canadian French, Mandarin, German, Hindi, Japanese, Spanish, Korean, Tamil and Vietnamese. Most of the training, the development and evaluation data come from the CallFriend Corpus. The training for most languages consists of 20 two-sided conversations per language, each 30 minutes long. While separate test-sets on 30s, 10s and 3s are available, our experiments were performed only on the 30-second test-set. The NIST 2003 evaluation set (eval-set) consists of 1280 utterances of which 960 come from CallFriend. Furthermore, 80 of the non-CallFriend utterances are Russian and were excluded in our experiments. A separate development set, again with 12 languages are also defined that were used for learning the backend parameters.

Our baseline LID system is GMM-based that includes both acoustic models and GMM indexes as tokens for a GMM tokenization system [14]. Language specific, 2048-component gender-independent GMMs were trained using 55-dimensional feature vectors, including the SDC with configuration 7-1-3-7 plus the original 6 MFCC coefficients [15]. During training, automatic speech detection was performed using a two-state HMM, combined with the acoustics from both sides of the conversation. To maintain the consistence with training, speech-detection was also applied on test even though the test contains very little silence. On average, more than 25% of the test observations are classified as silence and were excluded. While this appeared to be too aggressive, tuning the speech detector in training would require re-training of all the GMMs, which is computationally intensive for us; thus, this setting was kept.

Instead of building bigram language models with the token sequences, support vector machines (SVMs) were used because of its better performance [16]. While higher order n-gram, such as bigram, is usually better at capturing sequence information, the large number of possible bigrams makes it difficult to train them with SVM, and can hurt performance. In this paper, our tokenized system with SVMs used only the unigram count features with IDF weighting [17]. Our SVMs were implemented using the SVMlight [18]. In all our SVM experiments, the one-
against-all strategy was used to make multi-class classification decisions and to obtain the target language scores.

3.1. Baseline Experiments

Two baseline experiments were carried out. The first one used GMM of 2048 mixtures with only acoustic scores. The other one combined the acoustic scores and SVM-based LM scores (denoted as GMM-SVM). Scores from either experiment were transformed by the LDA+Gaussian backend, whose parameters are estimated from the development data. Close-set classification and language verification were both performed. In order to report classification results, Russian data in the evaluation set are excluded. The results are shown in Table 1.

<table>
<thead>
<tr>
<th>Exp.</th>
<th>ID Err (%)</th>
<th>EER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>acoustic only</td>
<td>18.8</td>
<td>7.3</td>
</tr>
<tr>
<td>GMM-SVM LM</td>
<td>11.3</td>
<td>4.2</td>
</tr>
</tbody>
</table>

Table 1: Baseline performance with or without SVM-LM scores for 12 language classification error and EER in language verification.

For non-discriminatively trained GMMs, verification EERs reported in the literature [19, 20, 15, 21] range from 4-7%. Our result of 7.3% EER is at the high-end. Note that we did not use gender dependent models, vocal tract normalization or another score transformation techniques [15, 19]. Our speech detection is also more aggressive in removing more than 25% of data from test which can reduce the effective length of test utterance and can hurt performance. After combining with the GMM-SVM, our baseline of 4.2% EER is comparable with other reported results with similar settings.

3.2. Effect of Boosted Target and Anti-Model

Boosting experiments were performed using 64-component GMM as the component classifiers. In order to have sufficient training errors, 3-second long training utterances were used. During the course of boosted model estimation, the combination function \( L() \) was trained with the training data such that it optimizes the classification on training, which is 3 seconds long. When forming the final classifier, which will be used for testing 30-second long test utterances, \( L() \) was trained with 30s long development utterances. Our preliminary experiments showed that applying boosted target and anti-models at the same time gave better performance than target or anti-models alone. The results are shown in Figure 1. The solid curve shows the training error and the dashed curve shows the error in evaluation set. We notice that the model actually converges fairly quickly, with very little change in both training errors and evaluation error after the 4-th iteration. “Dev error” performed better because the development data was used in learning the parameters of \( L() \). With the acoustic GMM system, the 12-language classification results improved from the baseline of 18.9% error to 10.5% error, a 44% relative error reduction. The acoustic LID system can be combined with the GMM-SVM scores during the fusion step. In this case, the GMM-SVM system is treated like another component classifier such that the \( L() \) transformation can be applied. The results are shown as the lower dotted curve in Figure 1. We notice that after adding the GMM-SVM LM scores, the baseline, without boosted target and anti-models, is at 11.3% error which is approximately the same level as the best result using only acoustic GMM with boosted target and anti-models. With the improved acoustic scores, the classification error is reduced to 7.8% after 3 iterations, a 30% relative error reduction. The acoustic LID system can be combined with the GMM-SVM scores during the fusion step. In this case, the GMM-SVM system is treated like another component classifier such that the \( L() \) transformation can be applied. The results are shown as the lower dotted curve in Figure 1. We notice that after adding the GMM-SVM LM scores, the baseline, without boosted target and anti-models, is at 11.3% error which is approximately the same level as the best result using only acoustic GMM with boosted target and anti-models. With the improved acoustic scores, the classification error is reduced to 7.8% after 3 iterations, a 30% relative error reduction. The gain is smaller compared to the acoustic GMM system probably because this is a combined system with the GMM-SVM LM scores which remain unchanged such that not all the gains from the improved acoustic GMM are carried over. It is also possible that some of the errors corrected by the boosted target and anti-models in fact are also those corrected by the GMM-SVM LM scores.

3.3. Language Verification

So far, all our results are reported as classification errors. In order to compare performance with results in the LID literatures, language verification results, in terms of EER, are tabulated in...
Table 2: Summary of LID classification (in % error) and verification in EER on the NIST 2003 evaluation set without Russian

<table>
<thead>
<tr>
<th>Systems</th>
<th>ID Err (%)</th>
<th>EER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline Aco. only</td>
<td>10.5</td>
<td>3.4</td>
</tr>
<tr>
<td>Target+Anti Aco. only</td>
<td>10.5</td>
<td>3.4</td>
</tr>
<tr>
<td>Baseline Aco. + GMM-SVM LM</td>
<td>11.3</td>
<td>4.2</td>
</tr>
<tr>
<td>Target+Anti Aco.+GMM-SVM LM</td>
<td>7.8</td>
<td>2.9</td>
</tr>
</tbody>
</table>

Table 2, and their DET curves are plotted in Figure 2. The top line is the acoustic GMM baseline performance. Adding the boosted target and anti-models significantly improves the verification performance over all operating points such that its performance is close to the performance of adding GMM-SVM LM scores, which is shown as a more solid dotted line. The final dotted line combines the boosted acoustic system with the GMM-SVM LM scores.

All the results in Table 2 and Fig. 2 are selected after the third iteration. We note that consistent with the gains in classification, the EER for the acoustic only system reduced from 7.3% to 3.4%, which is more than 50% relative reduction, and the EER for the acoustic plus GMM-SVM LM scores reduced from 4.2% to 2.9% which is more than 30% relative reduction. This shows that while the boosted models are learned to optimize classification results, its gain is more or less carried over to verification.

4. Conclusions

In this paper, we proposed to improve the acoustic modeling of GMM-based LID system by boosting that requires little implementation effort and can create discriminative trained acoustic models. Rather than using component models identical to the base classifier models, and combined linearly as in ada-boost, we proposed the use of smaller boosting models that can be combined seamlessly in the LID back-end process. This not only reduces computation but also improves performance. We introduced the use of boosted target models to improve classification likelihood to reduce “false reject”, and anti-models to reduce “false accept” errors. Experimental results on the 12-language the NIST 2003 evaluation set for both classification and verification have shown significant improvement. This approach may be applicable to other speech processing tasks, such as speaker verification or even speech recognition.

5. References