SAMPLE SELECTION FOR AUTOMATIC LANGUAGE IDENTIFICATION

[+]*David Farris,* [+*]*Chris White,* [+]*Sanjeev Khudanpur*

[+]Center for Language and Speech Processing
[*]Human Language Technology Center of Excellence
Johns Hopkins University, Baltimore, MD, 21218

ABSTRACT

Current approaches to automatic spoken language identification (LID) assume the availability of a large corpus of manually language-labeled speech samples for training statistical classifiers. We investigate two methods of active learning to significantly reduce the amount of labeled speech needed for training LID systems. Starting with a small training set, an automated method is used to select samples from a corpus of unlabeled speech, which are then labeled and added to the training pool — one selection method is based on a previously known entropy criterion, and another on a novel likelihood-ratio criterion. We demonstrate LID performance comparable to a large training corpus using only a tenth of the training data. A further 40% improvement in LID performance is obtained using a third of the training data. Finally, we show that our novel selection method is more robust to variance in the unlabeled pool than the entropy based method.

Index Terms— speech processing, unsupervised learning, natural languages

1. INTRODUCTION

Training an Automatic Spoken Language Identification (LID) system requires a large corpus of training data. Depending on the complexity of the LID algorithm, training can be computationally intensive. Also, the cost for labeling a large database covering many languages and dialects may be prohibitively high and there will always be the choice as to which data deserve labels. One would expect that increasing the amount of training data in general would improve LID accuracy. While this seems to be true for the most part, previous work (including ours, [1]) suggests that this may not always be the case. We compared the accuracy of a LID system trained on the full CallFriend *train* partition [2] to the accuracy of systems trained on random subsets of the partition. These random subsets were constructed by randomly selecting n speakers per language, where n was chosen such that the subsets are either 40% or 80% of the full *train* partition. There was a significant variance in the accuracy of the systems trained on the small subsets. One of these systems even achieved higher accuracy than the baseline, despite the fact that the system was trained using only 40% of the data. This suggests that certain subsets of the partition are more helpful for increasing LID accuracy than others due to factors such as the similarity of speakers, noise, etc. In this paper we develop an algorithm to discover these types of subsets. Other authors have referred to this kind of procedure as *selective sampling* [3]. A selective sampling algorithm automatically determines which samples ought to be used to train a classifier. Largely the literature suggests that samples should be selected if they are determined to exist in a region of uncertainty, where misclassification is possible.

Here we develop an experimental setup and a sample selection algorithm for a three-language LID task. Training samples are drawn from three languages in the CallFriend *train* partition. We evaluate on test segments from the NIST Language Recognition Evaluation (LRE) sets from 1996 and 2003. Using a simple iterative approach to select samples in regions of uncertainty, we reduce the Equal Error Rate (EER) by about 40% relative to the baseline sytem. We achieve this error rate by using only a third of the training data. To match the EER of the system trained on the full partition, we only need a tenth of the training data.

The selective sampling algorithm can also be cast as an instance of active learning. In this case one would start with a small amount of labeled data, and proceed to direct an annotator to the next set of segments that should be labeled. This way the same amount of data would be labeled given a fixed amount of the annotator's time, but the effect would be to have a better corpus in the sense of LID performance. Using our algorithm, we can reduce by 88.8% the number of samples that need to be labeled to maintain the same error rate given our current database. Cohn points out that gathering samples may be relatively inexpensive, whereas assigning labels may be costly [3]. If this is true for our LID task, then our approach can significantly reduce the cost of building a training corpus, or it could provide the community with a more effective training corpus for a given cost.

2. SELECTIVE SAMPLING PROCEDURE

This section describes the procedure for selective sampling. We use an iterative approach that is similar to the procedure described in [4]. The algorithm begins with a large set of un-

labeled data U, where each member of the set is a training sample. To initialize the algorithm, we request labels for a set L_0, where L_0 consists of randomly drawn samples from U. We use L_0 to train a model λ_0, which seeds our iterative algorithm. The algorithm uses a sorting criterion f to determine which samples $u \in (U \setminus L_i)$, $i = 0, 1, \ldots$, are likely to be misclassified by the current model λ_i. If we use an appropriate f, then samples that lie in the region of uncertainty should be at the head of our sorted list. These are the samples for which we will request labels at the i^{th} iteration. Table 1 shows the details of the algorithm.

1. Define L_0 to be a random subset of U. Request labels for L_0.
2. Set $i = 0$.
3. Train model λ_i using segments in L_i.
4. Use λ_i to sort the samples in $(U \setminus L_i)$ according to some criterion f.
5. Define $L^{'}$ to be the top θ segments in the sorted list. Request labels for $L^{'}$.
6. Set $L_{i+1} = L_i \cup L^{'}$
7. Set $i = i + 1$.
8. If stopping criterion is satisfied, stop. Else go to Step 3.

Table 1. Selective Sampling Algorithm

At each iteration i, we measure the error rate of our current model λ_i on a held-out evaluation set. The error-rate on this held-out set could be used to define a stopping criterion. Alternatively, we could simply define a maximum number of iterations M, after which the algorithm exits.

3. CORPUS

Training data were drawn from the *train* partition of the CallFriend corpus [2]. For these experiments, we use only the English, Mandarin, and Spanish conversations. We use this subset of CallFriend, referred to here as *CF-train-sm*, in order to manage the computation time for experiments and better understand how the sample selection procedure extends to a LID task.

The CallFriend corpus consists of unscripted 30 minute conversations between two speakers. There are 40 conversations between 80 unique speakers for each language l, for a total of 240 speakers. If we define each conversation side to be a training sample, then we constrain our sample selection criterion f to selecting particular speakers that reside near the decision boundary rather than segments of speech. This approach would also limit us to only 240 total samples in U. We overcome these constraints by creating many samples from a single conversation side such that each sample (denoted $\mathbf{x}$) consists of a 15 second segment of contiguous speech frames. Non-speech frames are rejected using a speech activity detector. This procedure increases the total number of samples in U to 12851, which corresponds to 53.5 hours of speech. Table 2 shows the number of samples for each language.

Language	Number of Segments
English	4404
Mandarin	4080
Spanish	4367
Total	12851

Table 2. *CF-train-sm*: A subset of CallFriend *train* partition

For evaluation, we use the English, Mandarin, and Spanish test segments from the 1996 and 2003 NIST Language Recognition Evaluation (LRE) sets, referred to as *eval96* and *eval03* respectively. We show results for 30 second test segments.

4. EXPERIMENTAL SETUP

4.1. Gaussian Mixture Model LID System

The baseline LID system, which is similar to the one described in [5], uses Gaussian Mixture Models (GMM). The GMM uses Shifted Delta Cepstral feature vectors, which are a concatenation of k delta cepstrum vectors of length N, each shifted by P frames. As in [5], the N, d, P, k parameters 7,1,3,7 were selected. There are 1024 Gaussian mixture components with a diagonal covariance matrix. Language-specific models are adapted from a Universal Background Model (UBM) as described in [6]. For a given test segment $\mathbf{x}$, the GMM for language l produces a log posterior probability $\log Pr\{l|\mathbf{x}\}$, which is proportional to the log likelihood $\log Pr\{\mathbf{x}|l\}$ of our MAP adapted models under an assumption of a uniform prior on the language. We use ratios of the log likelihoods for decoding and compute the EER according to the guidelines in the NIST LRE.

4.2. Baseline Training Sets

The traditional GMM system building approach does not subsample the training corpus; rather, every labeled example is used to train the acoustic models. This is equivalent to requesting labels for every sample in U. To simulate different sizes of U, we choose j segments from k speakers in *CF-train-sm* to create a set $U_{j,k}$, such that $U_{j,k}$ has $j * k$ total labeled segments. Segments are drawn randomly from *CF-train-sm*, with the constraint that $U_{j,k}$ be balanced with respect to gender, dialect, and language. This closely resembles a realistic approach to build a LID training corpus of a fixed-size. For these experiments, we set j to 30 and vary k from

12 to 240. For each choice of (j, k), we build 10 random sets $U_{j,k}$ to ensure reliability of our results. We define U_{all} to be the set of every segment in *CF-train-sm*.

5. EXPERIMENTS

5.1. Baseline Results

The performance of the GMM LID system trained on the baseline training sets $U_{j,k}$ can be seen in Figure 1. EER for *eval96* is shown as a function of the size of $U_{j,k}$, where the size of $U_{j,k}$ is the proportion $\frac{|U_{j,k}|}{|U_{all}|}$. As expected, the EER tends to decrease as the size of $U_{j,k}$ grows. There is a significant variance in EER when the size of $U_{j,k}$ is small, which is consistent with the observation in [1]. This confirms the hypothesis that for sets of a given size, some sets $U_{j,k}$ are better than others for training a LID system and further motivates this work of finding a good subset for training. Our selective sampling algorithm discovers one such set automatically for a given initialization.

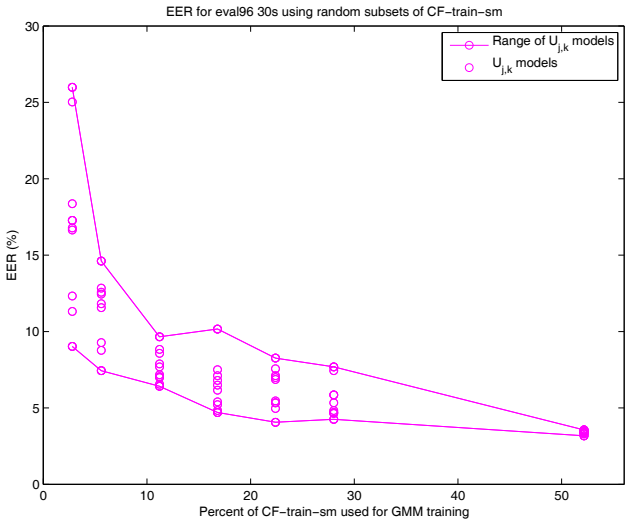

Fig. 1. EER for random subsets of CallFriend.

5.2. Selective Sampling using Entropy

Next, we measure the error rate achieved using the selective sampling algorithm described in Section 2. We seed the algorithm with 10 different choices of L_0, where L_0 has 360 segments. L_0 was chosen to have 360 segments so that the initial model could be trained quickly with minimal annotation and yet was large enough that the model would still produce somewhat reasonable LID results. At each iteration i we request labels for $\theta = 360$ additional segments, which corresponds to 2.8% of U_{all}. For our stopping criterion, we define a maximum number of iterations $M = 12$. The sorting criterion f is Entropy. An entropy-based measure was shown to be a suitable sorting criterion for a related effort in semi-supervised learning [7]. We define our Entropy measure f_H to be:

$$f_H(\mathbf{x}) = -\sum_l Pr\{l \mid \mathbf{x}\} \log Pr\{l \mid \mathbf{x}\} \quad (1)$$

Figure 2 shows the EER of the sample selection algorithm using Entropy. The EER is plotted as a function of the size of L_i, where the size of L_i is the proportion $\frac{|L_i|}{|U_{all}|}$.

Our approach outperforms systems trained on the baseline sets $U_{j,k}$ for every size of $U_{j,k}$. After just three iterations of the algorithm (which corresponds to 11.2% of U_{all}), we can match the EER for the system trained on U_{all}. After eleven iterations (33.6% of U_{all}), the mean EER reaches 2.02% for *eval96* and 1.19% for *eval03*. The system trained on U_{all} only achieves 3.65% for *eval96* and 3.0% for *eval03*. These results show that our selective sampling approach is effective for reducing EER for LID.

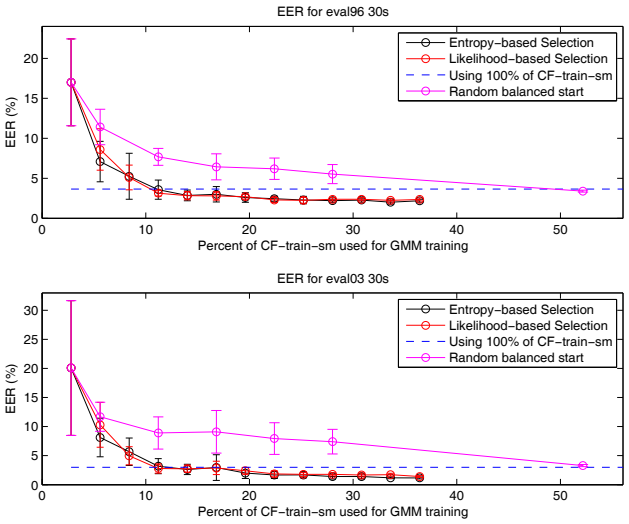

Fig. 2. Selective Sampling results for *eval96* and *eval03*.

5.3. Selection using relative likelihood

In this section, we study a likelihood-based criterion f for sample selection. Recall that the goal of our criterion is to discover which samples lie in the region of uncertainty. Whereas Entropy measures uncertainty amongst our language-specific GMM's, our likelihood criterion measures likelihood relative to the Universal Background Model (UBM). We define our likelihood measure f_{rL} to be:

$$f_{rL}(\mathbf{x}) = \log Pr\{\mathbf{x} \mid UBM\} - \max_l (\log Pr\{\mathbf{x} \mid l\}) \quad (2)$$

Experiments from Section 5.2 were rerun using f_{rL}. Results from these experiments are shown in Figure 2. The EER

for systems trained using f_{rL} consistently outperform the systems trained using random sets $U_{j,k}$. This suggests that f_{rL} is also a suitable criterion for selective sampling. On average, f_{rL} performs about as well as f_H, though it is interesting to note that at each iteration the two criteria request labels for different samples. We also observe that for $i < 7$, there is less variance in EER for f_{rL}, reflected by the smaller error bars. This suggests that if we are attempting to minimize the number of iterations M, we may get more predictable results by using f_{rL} rather than f_H.

6. SUMMARY AND DISCUSSION

We develop an iterative algorithm that automatically selects which samples to use to train models. For a three-language LID task, our sample selection scheme consistently achieves lower error rates than a balanced random selection method. We also show lower error rates than a system trained on the full *CF-train-sm* partition. After eleven iterations of the algorithm, we reduce the EER from 3.0% to 1.19% for the *eval03* test set. This is accomplished using only 33.6% of the samples in *CF-train-sm*. In order to match the performance of the system trained on the full partition, the sample selection algorithm only requires 11.2% of the samples in *CF-train-sm*.

We show that two different sorting criteria - Entropy and relative likelihood - are suitable for performing sample selection. With Entropy the procedure selects the most confusable samples, which helps determine class boundaries. Under the relative likelihood criterion, we select samples that are *typical* under the UBM and yet do not fit any of the language specific models, suggesting that these samples contain novel information for whichever language they truly belong to, and hence merit being labeled. Both criteria provide useful information about potential training samples, and in the future we want to consider how to compare the samples that each select.

The selective sampling algorithm described can be easily explained in terms of active learning. If we assume that we start with a small set of labeled data and a large set of unlabeled data, we can use selective sampling to determine which samples to label on an iterative basis. This differs from a traditional approach which would either label all the samples in the large set or a random subset. Our selective sampling approach consistently outperforms both these techniques for a three-language LID task. We reduced by 88.8% the number of samples that need to be labeled in order to match the error rate of the system trained on the full *CF-train-sm* partition. Assuming that the cost to label examples is high, our selective sampling algorithm could be employed to significantly reduce the cost of building training corpora for LID.

In the future, we would like to test the sample selection algorithm using all the languages in the full CallFriend training partition. We would like to extend this to large selections of partially labeled data gathered from the web, or labels that include a varied source or microphone. It may also be of value to investigate how to select which speakers to label when initializing the algorithm.

7. ACKNOWLEDGMENTS

The authors would like to thank MIT Lincoln Labs for permission to use parts of their language identification code to perform experiments.

The work of Chris White was supported by a U.S. Department of Homeland Security (DHS) Fellowship under the DHS Scholarship and Fellowship Program, a program administered by the Oak Ridge Institute for Science and Education (ORISE) for DHS through an interagency agreement with the U.S Department of Energy (DOE). ORISE is managed by Oak Ridge Associated Universities under DOE contract number DE-AC05-06OR23100. All opinions expressed in this paper are the authors and do not necessarily reflect the policies and views of DHS, DOE, or ORISE.

8. REFERENCES

[1] David Farris, "An evaluation of automatic language identification techniques," M.S. thesis, Johns Hopkins University, Baltimore, MD, May 2007.

[2] Alexandra Canavan and George Zipperlen, "The CallFriend corpus," Linguistic Data Consortium, 1996.

[3] David Cohn, Les Atlas, and Richard Ladner, "Improving generalization with active learning," *Machine Learning*, vol. 15, no. 2, pp. 201 – 221, May 1994.

[4] Teresa Kamm and Gerard Meyer, "Automatic selection of transcribed training material," in *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding*, 2001, pp. 417 – 420.

[5] E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, and D.A. Reynolds, "Acoustic, phonetic, and discriminative approaches to automatic language identification," in *Proc. Eurospeech*, 2003, pp. 1345 – 1348.

[6] Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, pp. 19 – 41, 2000.

[7] Rong Zhang and Alexander I. Rudnicky, "A new data selection principle for semi-supervised incremental learning," *International Conference on Pattern Recognition*, vol. 2, pp. 780–783, 2006.