A GENERATIVE-DISCRIMINATIVE FRAMEWORK USING ENSEMBLE METHODS FOR TEXT-DEPENDENT SPEAKER VERIFICATION

Amarnag Subramanya[1,*], *Zhengyou Zhang*[2], *Arun C. Surendran*[2],
Patrick Nguyen[2], *Mukund Narasimhan*[2], *Alex Acero*[2]

[1]SSLI Lab, University of Washington, Seattle, WA 98195, USA
[2]One Microsoft Way, Microsoft Research, Redmond, WA 98052, USA

ABSTRACT

Speaker Verification can be treated as a statistical hypothesis testing problem. The most commonly used approach is the likelihood ratio test (LRT), which can be shown to be optimal using the Neymann-Pearson lemma. However, in most practical situations the Neymann-Pearson lemma does not apply. In this paper, we present a more robust approach that makes use of a hybrid generative-discriminative framework for text-dependent speaker verification. Our algorithm makes use of a generative models to learn the characteristics of a speaker and then discriminative models to discriminate between a speaker and an impostor. One of the advantages of the proposed algorithm is that it does not require us to retrain the generative model. The proposed model, on an average, yields 36.41% relative improvement in EER over a LRT.

Index Terms— Speaker Verification, Discriminative Models, Boosting.

1. INTRODUCTION

Often speaker verification (SV) is formulated as a statistical hypothesis test. The most commonly used approach is the likelihood ratio test (LRT) (shown by the Neyman-Pearson lemma to be optimal in certain cases [1]), where the likelihood ratio test statistic is compared to a threshold. The likelihoods of the data are computed using two competing generative models, one describing the null hypothesis (in our case, the targeted speaker) and the other describing the alternate hypothesis (everything except the targeted speaker). Such a test is often treated as being uniformly most powerful (UMP), assuming that the threshold is independent of the testing data and the alternate hypothesis. This assumption fails because, among other things, we do not know the exact form of the underlying distribution and such distributions are estimated using limited training data. Further, the alternate hypothesis is composite, making it difficult to model. Instead of designing UMPTs, some methods have chosen to design locally most powerful tests (LMPTs) [1, 2], replacing the composite hypothesis with a set of simple hypotheses using locally competing models. Examples of such approach are cohort methods based on competing speakers [5, 3] or competing phone models [4]

One of the goals of this paper is to present a more robust approach to such a testing. Since existing theory is not applicable, we can adopt generalized tests [2], where the test statistic is a function of the computed likelihoods: $T = f(L(X|\Lambda^s), L(X|\Lambda))$, Λ^s and Λ are the speaker and alternative models respectively. These generalized test statistics can be discriminative. Sometimes the model themselves may be trained discriminatively to optimize some misclassification or mis-verification functions [11, 12] of these discriminative measures. Although such full-scale discriminative models tend to outperform generative models, they often need much more data to train. This can be a problem when amount of speaker training data is limited or when getting impostor data is difficult (e.g. where each person's password in unique). In such cases, a generative model may be preferable.

We adopt a third approach in this paper which strikes a balance between the alternatives. We choose to keep the speaker models generative while designing tests using discriminative classifiers. In text dependent speaker verification there is an additional dimension to this problem - when the underlying models are HMMs, we do not know how to design tests that incorporate temporal information [6, 7]. In such cases we can take from LMPTs the idea of using local scores or temporal scores as part of the decision making criterion, e.g. in text dependent speaker verification scenarios, while designing such test statistics, we can further consider likelihood information from the sub-parts of the HMM like word models or states. In this paper, we will consider a specific form of text dependent system called fixed-vocabulary systems. We specifically consider digit based word models. Hence our local scores will be based on the digit based models.

There have been some examples in literature that follow the approach that we are using, e.g. in [13] SVMs are used for classification. The minimum verification error based approach in [12] can be thought of as using logistic function on the model scores. Utterance and frame level scores have been used for verification in [15] and for confidence measures for speech recognition in [16]. One disadvantage with the above approaches is that the decision about which features to use for classification has to be done in advance, and the classifiers such as SVMs can be computationally expensive to train.

What we need is a discriminative classifier that is as simple as possible, and one that chooses the best set of features from a wide set of available features automatically. Boosting is a classification framework that provides such a flexibility, and we will use this approach in this paper. This approach provides additional flexibility in SV - for example if small changes are made to one part of the data or the model e.g. if more impostor data is available, or if the speaker model need be updated by adding a new digit model, the hypothesis test can be quickly updated, possibly by only adding another stage to boosting, without having to retrain the entire system. In contrast, systems like [17] where the GMMs themselves are boosted, require more extensive retraining every time small changes are needed.

2. BASELINE SYSTEM

In this section, we describe our baseline text-dependent SV system. During enrollment the system parameters are adjusted to better model the speaker (user). A user S is first asked to repeat her

*This work was done at Microsoft Research

password n times. These enrollment data are then used to adapt the speaker independent models (Λ) to yield speaker specific models (Λ^s) using maximum likelihood linear regression (MLLR) [9]. As the amount of adaptation data is limited, we make use of global adaptation, i.e. we estimate a single rotation and translation for all the means in the recognizer.

During verification there are two inputs: (a) a claim such as "user X claims to be user S", and (b) the input speech signal containing a pass-phrase as spoken by user X. The problem can be recast as accept/reject the hypothesis, H_0 : the given speech signal was spoken by user S and contains the users pass-phrase.

Let $O_{1:T}$ be the features vectors extracted from the speech signal. In this paper, unless otherwise stated, we assume that the user's pass-phrase consists of a sequence of n words, $\mathbf{W} = \{w_1, w_2, \ldots, w_n\}$. The verification step involves the following:

1. We run forced alignment using speaker independent models Λ on $O_{1:T}$. The score returned by recognizer in this step is denoted by $p(O_{1:T}|\Lambda, \mathbf{W})$,
2. We repeat the step above, but replacing Λ with Λ^s. Let the score here be $p(O_{1:T}|\Lambda^s, \mathbf{W})$,
3. Finally, we use $p(O_{1:T}|\Lambda, \mathbf{W})$ and $p(O_{1:T}|\Lambda^s, \mathbf{W})$ to either accept or reject the hypothesis,

In drawing a parallel with text-independent SV, the speaker independent model (Λ) plays the same role as the universal background model (UBM) [5]. A classical approach to hypothesis testing is to compute,

$$F = \frac{L(\Lambda^s, \mathbf{W}|O_{1:T})}{L(\Lambda, \mathbf{W}|O_{1:T})} = \frac{p(O_{1:T}|\Lambda^s, \mathbf{W})}{p(O_{1:T}|\Lambda, \mathbf{W})} \tag{1}$$

where $L(\Lambda^s, \mathbf{W}|O_{1:T})$ represents the likelihood of the model Λ^s and the word sequence $\mathbf{W}$ given the observations. A more familiar form is $f = \log F = \log p(O_{1:T}|\Lambda^s, \mathbf{W}) - \log p(O_{1:T}|\Lambda, \mathbf{W}) = l(\Lambda^s, \mathbf{W}|O_{1:T}) - l(\Lambda, \mathbf{W}|O_{1:T})$. The hypothesis is then accepted or rejected based on a simple thresholding on F (or f). This is the so-called likelihood ratio test (LRT). Neyman-Pearson lemma suggests that, if both the training and test sets are drawn from the same underlying distribution, then for a given significance level, there is no test more powerful than the LRT.

In practice though, the Neyman-Pearson lemma cannot always be applied. This is because (a) as the amount of training data is only finite, it is not possible to estimate the true underlying distribution that generated the data (training and test), and (b) it is also known that HMM based speech models are approximations of the actual speech process. As a result, we can no longer claim that the LRT is the most powerful hypothesis test.

3. WEIGHTED LIKELIHOOD RATIO TESTS

While the discussion above is applicable to LRTs in general, in this section we focus on some inherent shortcomings of LRT for SV. The final score that is used in the LRT, which is the score at the utterance (sentence) level is a function of the scores at a more sub-utterance level, for example, the state level, or phone level, or syllable level, or even the word level. The recognizer in essence maps these sub-unit scores into a score at the utterance level. Since the recognizer is not necessarily trained to optimize the SV performance, we cannot expect it to learn the optimal mapping (from the SV perspective) of the scores from the sub-unit level to the utterance level. Further, if it is the case that certain classes of words provide more speaker discriminability than others, then these set of words should in essence get a larger weight in the verification process in comparison to other classes. However, in the case of the LRT, all scores are given an equal weight. To illustrate the above point, we use a simple example: Let $\mathbf{W} = \{w_1, w_2, \ldots, w_n\}$ and that w_i generated $O^{w_i}_{t_{s,i}:t_{e,i}}$ i.e., if we ran forced alignment with $\mathbf{W}$, then $t_{s,i}$ and $t_{e,i}$ would be the start and end of the i^{th} word w_i. Thus (if we neglect language model probabilities) we have that,

$$\begin{aligned} f &= \log p(O_{1:t}|\Lambda^s) - \log p(O_{1:t}|\Lambda) \\ &\approx \sum_{i=1}^{n} \log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda^s) - \sum_{i=1}^{n} \log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda) \end{aligned} \tag{2}$$

As it can be seen every word gets an equal weight. Consider the objective function

$$f' \approx \sum_{i=1}^{n} a_i \ \log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda^s) - \sum_{i=1}^{n} b_i \ \log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda) \tag{3}$$

where the weights $\lambda = \{a_i, b_i\}, 1 \leq i \leq n$ are learnt to optimize overall SV performance. Intuitively, it would make sense to impose the constraint $a_i, b_i \geq 0 \, \forall \, i$. Further, the classical approach is only a special case of the weighted formulation, i.e., $f = f'$, if, $a_i = b_i = 1, \ \forall \ i$. The question now is whether we can find a principled way to learn the weights λ. For this consider a small modification of the above,

$$f'' \approx \sum_{i=1}^{n} a_i \ \left[\log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda^s) - \log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda)\right] \tag{4}$$

This has a special significance in the light a popular learning approach. We can think of each of the terms $\log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda^s) - \log p(O^{w_i}_{t_{s,i}:t_{e,i}}|\Lambda)$ as being a 'weak' classifier, and then the final classification is based on a weighted sum of these weak classifiers. In spirit, this is very similar to the approach of boosting wherein a number of weak classifiers are combined to produce a strong classifier. Note that, while in above discussion, we use a weighted sum at the word level, in theory the sum can be formed at other sub-utterance levels, such as state, phone, triphone, etc.

4. BOOSTING

Boosting is a technique for sequentially training and combining a collection of classifiers in such a way that the later classifiers make up for the deficiencies of the earlier ones [10]. In boosting literature each classifier is referred to as a weak learner, i.e., each classifier on its own is only capable of producing an output that is slightly better than chance, but when combined form a powerful classifier. The boosting algorithm that we used in this work is outlined in table 1.

In section 5.2 we discuss what features ($\mathbf{x}$) were used and how they were obtained from the recognizer. We make use of decision trees as weak learners in this work. Each node in the tree is essentially a decision stump operating on a single dimension of $\mathbf{x}$ (refer to step 1 of the boosting algorithm). In other words, at each iteration, the algorithm selects one element (dimension) from $\mathbf{x}$ and a corresponding threshold such that it minimizes the weighted training error. Note that the pair (dimension and threshold) are jointly chosen to minimize the weighted error. Intuitively, this is a discriminative feature selection strategy. Thus in our case, $h_p(\mathbf{x}) = \mathbf{I}\big(x^p > K_p\big)$, where x^p is some element of $\mathbf{x}$ that was chosen during the p^{th} iteration, K_p is its corresponding threshold and $\mathbf{I}$ is the indicator function, that returns 1 if the condition is true and 0 otherwise. Thus the final decision function is given by $H(\mathbf{x}) = \sum_{p=1}^{P} \alpha_p \mathbf{I}\big(x^p > K_p\big)$. Thus, we make use of boosting to learn the a_i (see equation 4) and an associated threshold in a discriminative fashion.

Given a training set $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i$ are the feature vectors derived from the recognizer (i.e. the generative model), and $y_i \in \{0, 1\}$ are the labels, initialize $\{D_i^1\}_{i=1}^N = \frac{1}{N}$. Here D_i represents the weight on the i^{th} sample in the training set.
For iterations $p = 1, \ldots, P$, do

1. Train a weak learner based on the weighted training error, let this classifier be h_p.
2. Compute $\epsilon_p = \sum_{i=1}^N D_i^p \mid y_i - h_p(\mathbf{x}_i) \mid$.
3. Set $\alpha_p = \frac{1}{2} ln\left(\frac{1-\epsilon_p}{1+\epsilon_p}\right)$.
4. Update $D_i^{p+1} = D_i^p e^{-\alpha_p f\left(y_i, h_p(\mathbf{x}_i)\right)}$, where $f(m, n)$ returns $+1$ when $m = n$ and, -1 otherwise.
5. Renormalize, $D_i^{p+1} = \frac{D_i^{p+1}}{Z_{p+1}}$, where $Z_{p+1} = \sum_{i=1}^N D_i^{p+1}$.

Final classifier is given by $H(\mathbf{x}) = \sum_{p=1}^P \alpha_p h_p(\mathbf{x})$

Table 1. Boosting Algorithm

5. EXPERIMENTAL SETUP

5.1. Corpus Description

In this section we describe the corpus that was created for this work. We started with the YOHO corpus [8] which was designed for digit based text-independent SV task. It consists of 144 speakers, each having an enrollment and verification section. Users are prompted to utter a randomly generated strings of six digits. For example, "26-81-56", in which case, the user says "twenty six sil eighty one sil fifty six sil" (sil refers to a short period of silence). As the prompts are randomly generated, the utterances in the enrollment and verification sets do not match, the corpus cannot be used for the test-dependent task. This required modifying the YOHO corpus to suit the problem at hand.

For all experiments in this paper we make use of the Microsoft telephony engine as the generative model. The recognizer is a standard HMM system, trained on about 2000 hours of data (digits and short commands) with 94k Gaussians. The frontend used MFCC-like features reduced to 36 coefficients with HLDA. During decoding, all possible words were allowed in a uniform loop grammar. We first ran the recognizer on the entire YOHO corpus. The word error rate was found to be 3.6%. We then removed the utterances that were mis-recognized, yeilding a corpus whose error rate was 0%. Next we ran forced alignment on these utterances using the references (as the WER is 0%, the viterbi hypothesis and reference are the same). The resulting segmentations were used to chop the sentences into three segments, each containing two digits. For example, an utterance "26-81-56" was chopped to form three utterances containing "26", "81" and "56". Further each of these files was labeled 'start', 'middle' or 'end' based on their origin. In the above example, "26" was labeled as start, "81" was labeled middle and "56" labeled end. We refer to the resulting database as the CUT YOHO corpus. The error rate on the CUT YOHO corpus was found to be 1.6%. We hypothesize that these errors might be due to faulty segmentation. The files that resulted in an error were removed from the corpus yeilding a CUT YOHO corpus with an effective an error rate of 0%.

In the resulting CUT YOHO corpus, we had a number of examples of various two digit sequences spoken by the users. It is straightforward to construct multiples of two digits (i.e. four, six, etc.) sequences by simply concatenating these two digits sequences. In the above process, care was taken to ensure that original dynamics of the speaker was preserved, i.e., an utterance labeled "start" was not concatenated at the end and so on. For the purposes of training the boosting algorithm, we split the corpus into a training, development and test set. While choosing these sets, we ensured that each speaker had a fairly equal representation in each set to overcome undue bias towards any speaker. The utterances in the training set were also used to produce Λ^s using MLLR.

Length of pass-phrase	LRT	Boosting
2 digits	3.35	2.12
4 digits	1.89	1.62
6 digits	0.63	0.26

Table 2. Equal Error Rate (in %)

5.2. Feature Sets

In this section we discuss the features that were used for the boosting stage. As explained in section 2, given an utterance from speaker S, we make two passes using the recognizer yielding scores $l(\Lambda^s, \mathbf{W}|O_{1:T})$ and $l(\Lambda, \mathbf{W}|O_{1:T})$. In addition, we also obtain the word level scores, $l(\Lambda^s, w_i|O^{w_i}_{t_{s,i}:t_{e,i}})$, $l(\Lambda, w_i|O^{w_i}_{t_{s,i}:t_{e,i}})$, $1 \leq i \leq n$. For each level (i.e. word and utterance) we use the following features,

1. the raw likelihoods resulting from each recognizer pass,
2. the difference between the raw likelihoods (LR), and
3. the durations.

We do not use utterance level durations, so the last point in the above only applies to the word level. Further, we also append the normalized likelihoods (normalized by number of frames) in all the above cases. Furthermore, the negative of all the likelihoods in the above cases are also added to the feature vector. This is because of the nature of our weak learner, i.e. the thresholding is always in one direction (see section 4). Thus, if the user's pass code consists of n words, we extract $13n + 12$ features from the recognizer outputs. Intuitively, while some of the features used above might seem to lack the speaker discrimination capabilities, the basic characteristics of boosting allow us to choose as many features as possible, and then let the algorithm pick the best features discriminatively.

6. RESULTS

Table 2 shows the equal error rate (EER) results of our SV system. We used the CUT YOHO corpus with varying lengths of pass-phrases. In the 2-digit case itself there were over a 100,000 positive examples and 400,000 negative examples in the test set. It is important to highlight the fact that in our test set all impostors know the users password, i.e., we are testing the system in the worst case scenario. We made use of 4 utterances to adapt the speaker independent models Λ to speaker dependent model Λ^s. The column labeled "LRT" in table 2 shows the results obtained using a simple likelihood ratio test as described in section 2. As expected the EER decreases with increasing length of the pass-phrase. The column marked "Boosting" shows the results of boosting the scores from the generative model. In each case, the optimal number of boosting iterations was determined on a held-out set. Note that we learn a

Feature	Relative Weight
Norm., LR Utterance	0.23
Norm., LR first word	0.06
Norm., LR second word	0.19
Norm., LR third word	0.16
Norm., LR fourth word	0.04
Un-norm., second word, Λ^s	0.10
Neg. Un-norm., second word, Λ	0.055
Neg. Norm., third word, Λ	0.034
Un-norm., third word, Λ^s	0.131

Table 3. Different Features that were chosen by the algorithm along with their relative weights in the 4-digit pass-phrase case. Here "Norm." refers to normalized by duration, "LR" is difference between likelihoods from Λ^s and Λ, "Neg." before a feature implies a sign change.

single boosted tree of classifiers for all the speakers in our corpus for a given length of pass-phrase. As it can be seen, the boosting stage is able to improve over the results of LRT, and on a average provides a relative 36.41% improvement in EER.

A close look at the features that were chosen and the weight for each of the features leads to more insights about speaker discrimination in general. In the following we take the 4-digit pass-phrase system as an example (similar trends were observed for other lengths as well). In the 4-digit pass-phrase case, we derive 64 features from the generative model. The optimal number of boosting iterations on the held-out set was found to be 9. Thus 9 out of the 64 features were picked by the boosting algorithm and table 3 shows these 9 features and their relative weights (α_p). As expected the likelihood ratio at the sentence level gets the highest weight. However as hypothesized in section 2, the likelihood ratios at the word level also exhibit discriminative capabilities and thus get relatively large weights. In particular, it can be seen that the algorithm places particular emphasis on the words occurring in the middle of the users' pass-phrase in comparison to the start and end. It is also interesting to note that some of the un-normalized likelihoods are chosen as well. While this might seem surprising, it might be the case that, they always follow a particular pattern. For example, $l(\Lambda^s, w_2|O^{w_2}_{t_s:t_e})$ is always positive, whereas $l(\Lambda, w_2|O^{w_2}_{t_s:t_e})$ is always negative for the 'true' speaker (in other words $-l(\Lambda, w_2|O^{w_2}_{t_s:t_e})$ is always positive).

7. CONCLUSIONS AND FUTURE WORK

In this paper we have shown how a discriminative algorithm can used in conjunction with a generative model for the text-dependent SV task. The algorithm can be motivated by considering situations where the Neyman-Pearson lemma does not apply. One of the advantages of the proposed approach is that it does not require us to retrain the generative model. Further, as the discriminative model has a small number of free parameters, it can be trained even with small amounts of training data. We get, on an average 36.41% relative improvement in EER as a result of the boosting stage compared to doing a LRT on the scores from the generative model.

In future, we plan on investigating other features that can derived from the generative model. For example, the means of the gaussian mixtures belonging to the most likely state sequence contain valuable information about the speaker. Another area where LRT's have been found to be wanting in performance is generalization to unseen/noisy environments, i.e., thresholds chosen to yield optimal performance in clean conditions do not always work for noisy conditions. Thus, we plan on testing the boosting approach in noisy conditions. As boosting trains the classifiers in a 'maximum margin' sense, we expect them to generalize better to noisy conditions.

8. REFERENCES

[1] E. L. Lehmann, "Testing Statistical Hypotheses", Wiley, New York, 1959.

[2] C.-H. Lee, "A Tutorial on Speaker and Speech Verification", In *Proceedings of NORSIG '98*, pp. 9-16, Vigo, Denmark, June 1998.

[3] A. E. Rosenberg, J. Delong, C.-H. Lee, B.-H. Juang, and F. K. Soong, "The Use of Cohort Normalized Scores for Speaker Recognition", In *Proc. ICSLP*, pp. 599-602, October 1992.

[4] R. Sukkar, C.-H. Lee, "Vocabulary Independent Discriminative Utterance Verification for Non-Keyword in Sub-word based Speech Recognition", In *IEEE Trans. Spech and Audio Processing*, Vol. 4, No. 6, pp. 420-429, Nov 1996.

[5] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker Verification Using Adapted Gaussian Mixture Models", In *Digital Signal Processing*, Academic Press, 2000.

[6] H. Schalk, H. Reininger, and S. Euler, "A System for Text Dependent Speaker Verification - Field Trail Evaluation and Simulation Results", In *EUROSPEECH*, 2001.

[7] K. Yu, J. Mason, and J. Oglesby, "Speaker recognitions using hidden Markov models", In *IEE Vision Image and Signal processing*, vol. 142, no. 2, pp. 291 - 298, Apr. 1994.

[8] J. P. Campbell Jr., "Testing with the YOHO CD-ROM Voice Verification Corpus", In *Proceedings of ICASSP*, 1995.

[9] M. Gales, "Maximum Likelihood Linear Transformations for HMM-based Speech Recognition", In *Technical Report, CUED/FINFENG/TR291*, Cambridge Univ., 1997

[10] Y. Freund and R. Schapire, "Experiments with a new boosting algorithms", In *Proceedings of Thirteenth International conference on Machine Learning*, Bari, Italy, July 1996.

[11] L. Heck and Y. Konig, "Discriminative training of minimum cost speaker verification systems", In *Proc. of RLA2ESCA* , pages 93–96, Avignon, France, 1998.

[12] Rosenberg, A. and Siohan, O. and Parthasarathy, S. "Speaker verification using minimum verification error training", In *Proc. of ICASSP'98*.

[13] S. Fine, J.Navr'atil, and R. Gopinath, "A Hybrid GMM/SVM approach to speaker identification", In *Proc. of ICASSP*, 2001.

[14] S. Nakaga and K. P. Markov, "Speaker Normalization using Frame and Utterance Level Likelihood Normalization", In *Proc. of ICASSP*, 1997.

[15] R. D. Zilca, "Text-Independent Speaker Verification Using Utterance Level Scoring and Covariance Modeling", In *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 6, September 2002.

[16] N. Moreau and D. Jouvet, "Use of a Confidence Measure Based on frame level Likelihood Ratios for the Rejection of incorrect data", In *Proc. of Eurospeech* 1999.

[17] S. Z. Li, D. Zhang, C. Ma, H. Shum, and E. Chang, "Learning to Boost GMM Based Speaker Verification", In *Proc. of Eurospeech*, Geneva, Switzerland. Sep 2003.