Word-Conditioned HMM Supervectors for Speaker Recognition

Howard Lei[1,2]*, Nikki Mirghafori*[2]

[1]The International Computer Science Institute, Berkeley, CA, USA
[2]The University of California, Berkeley, CA, USA
{hlei,nikki}@icsi.berkeley.edu

Abstract

We improve upon the current Hidden Markov Model (HMM) techniques for speaker recognition by using the means of Gaussian mixture components of keyword HMM states in a support vector machine (SVM) classifier. We achieve an 11% improvement over the traditional keyword HMM approach on SRE06 for the 8 conversation task, using the original set of keywords. Using an expanded set of keywords, we achieve a 4.3% EER standalone on SRE06, and a 2.6% EER in combination with a word-conditioned phone N-grams system, a GMM-based system, and the traditional keyword HMM system on SRE05+06. The latter result improves on our previous best.

Index Terms: speaker recognition, supervector, keyword HMM, keyword constraining, round-robin training.

1. Introduction

Speaker recognition has historically relied on low-level acoustic features with GMMs using a bag-of-frames approach for speaker discrimination [1]. These approaches typically treat the feature frames as being time-independent in their statistical processing. Moreover, these approaches ignore idiolect-based speaker information from word N-grams. Recently, a variety of approaches have been created to expand upon this idea, including systems relying on word and phone N-gram frequencies as features [2][3], and HMM-based approaches.

The keyword HMM system [4] is a recent development that uses HMMs instead of GMMs to capture time-dependent information among the frames. In this system, an HMM is trained using Mel-Frequency Cepstral Coefficient (MFCC) feature sequences within the boundaries of a set of keywords (keyword constraining) consisting of unigrams and bigrams. For each target speaker, an HMM is trained using each keyword via MAP adaptation from a speaker-independent (background) HMM for the corresponding keyword [4]. To classify test-utterances, the log-likelihoods of MFCC feature sequences from each keyword in a test-utterance are computed using the target speaker and background HMMs. The test utterance is more likely to be spoken by the target speaker if its target speaker model log-likelihoods are greater than the background model log-likelihoods.

In this paper, we introduce an alternative speaker recognition approach that uses the means of the Gaussian mixture components of each keyword HMM state as features for an SVM classifier. Our approach was inspired by Campbell et al.'s [5], which used the Gaussian mixture means from a GMM-based system in an SVM classifier. Unlike their approach, however, we used time-dependent acoustic feature information and applied keyword constraining.

This paper is organized as follows: Section 2 describes the database and preprocessing. Section 3 describes keyword HMM training, supervectors, and SVM training. Section 4 describes experiments and results. Section 5 provides a summary and conclusion of our findings.

2. Data and preprocessing

We used the Switchboard II and Fisher corpora for background model training, and the SRE05 and SRE06 corpora for target speaker model training and testing. Additionally, we used the Switchboard II and SRE04 corpora to train example impostor speakers [5] for SVM training purposes. SRE04-06 are subsets of the MIXER conversational speech corpus, where two unfamiliar speakers speak for roughly 5 minutes. A conversation side (roughly 2.5 minutes for non-Fisher and 5 minutes for Fisher) contains speech from one speaker only. 7,598 conversation sides were used from SRE06 (for target speaker model training and testing), 6,090 from SRE05 (a portion of which are the same as the 7,598 from SRE06 due to the overlap between SRE05 and SRE06), 1,792 from SRE04, 4,304 from Switchboard II, and 1,128 from Fisher. 1,553 Fisher and Switchboard II conversation sides were used as background conversation sides, where each speaker was represented by no more than one conversation side. There were 16,831 total trials for SRE06 with 2,010 true speaker trials.

For a given conversation side, segments containing speech were extracted using a speech/non-speech detector, and word recognition was performed using the DECIPHER recognizer [6], developed by SRI. MFCC features (C0-C19 plus deltas) were extracted every 10 ms from 25 ms frames using HTK [7].

3. Statistical processing and classification

3.1. HMM training

HMM training was done in the same manner as the traditional keyword HMM system [4], using a set of keywords. For each keyword, one background HMM was trained using MFCC features from the background conversation sides. For a given keyword, MFCC feature frames corresponding to all instances of the keyword were used to train the background keyword HMM. The observation distribution at each HMM state consisted of a mixture of Gaussian components. Ideally, there should be enough components to represent a wide range of distributions necessary to model the data, and not too many such as to pose a risk for over-training in addition to being computationally expensive. Eight Gaussian mixture components were experimentally chosen to satisfy both criteria. The HMMs were left-to-right with self-loops at each state and no skips [4]. The number of states for each keyword HMM was the following:

$$NumStates = min\left(3P, \frac{1}{4}D\right) \quad (1)$$

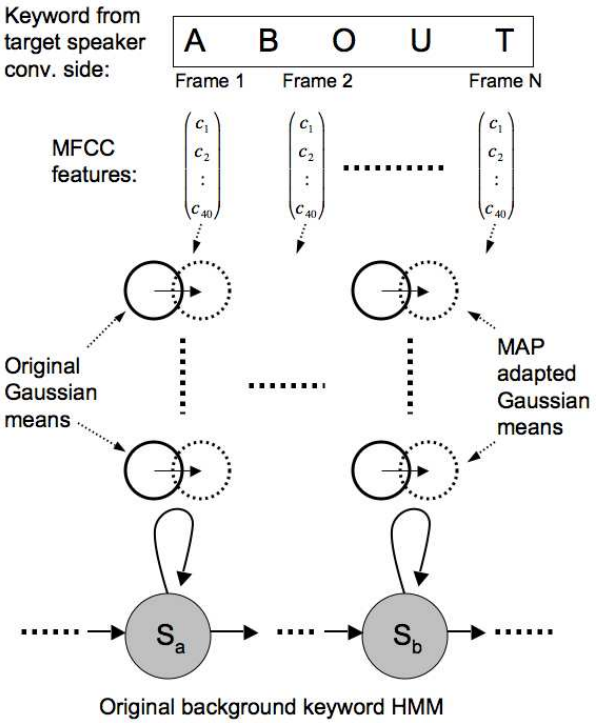

Figure 1: *MAP adaptation of background keyword HMM to create target speaker keyword HMM.*

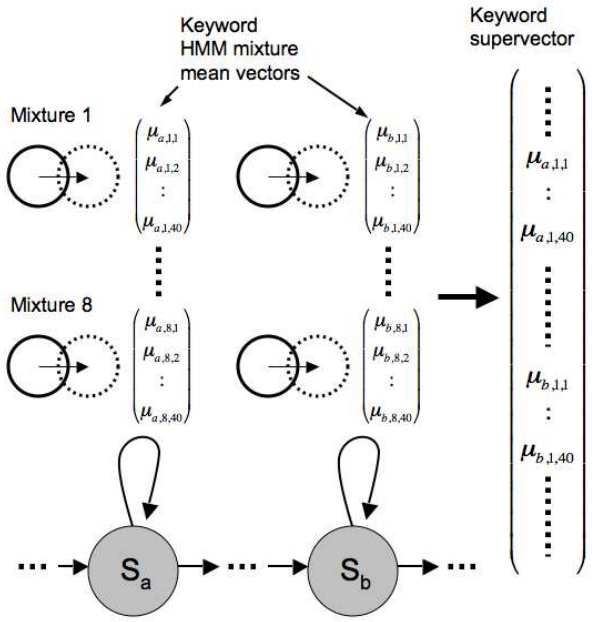

Figure 2: *Obtaining supervector from MAP adapted Gaussian mixture means.*

where P is the average number of phones comprising the keyword, and d is the median number of MFCC frames for the keyword [4].

For each keyword, background keyword HMM parameters were trained via the EM algorithm using HTK. For each target speaker, an HMM was trained for each keyword, using MFCC features constrained by the keyword instances from eight conversation sides of target speaker data. Eight conversation sides were used to provide sufficient keyword HMM training data, because not all keywords may exist in a single conversation side. Training was done via MAP adaptation from the background keyword HMMs, where target speaker keyword HMM parameters were adapted from the corresponding background keyword HMM. MAP adaptation ensured consistency between the background and target speaker keyword HMMs, such that if there were no data for a target speaker, its keyword HMM would be the same as the background keyword HMM.

Only the Gaussian mixture means were altered, via HTK, as follows: for state j and Gaussian mixture m [4][7]:

$$\hat{\mu} = \frac{N_{jm}}{N_{jm}+\tau}\bar{\mu}_{jm} + \frac{\tau}{N_{jm}+\tau}\mu_{jm} \tag{2}$$

where τ is the weight of a priori knowledge to the adaptation data, N_{jm} is the occupation likelihood of the adaptation data, $\bar{\mu}_{jm}$ is the Gaussian mean of the adaptation data, μ_{jm} is the Gaussian mean of the background keyword HMM, and $\hat{\mu}_{jm}$ is the updated Gaussian mean. Figure 1 illustrates MAP adaptation using MFCC features for a given keyword.

In the traditional keyword HMM approach, a sequence of feature vectors $(f_1, ..., f_N)$ belonging to a keyword instance in a test utterance conversation side was scored against target speaker models as follows:

$$Score = \log\left(\frac{p(f_1, ..., f_N|M_{TS})}{p(f_1, ...f_N|M_{BKG})}\right) \tag{3}$$

where M_{TS} is the target speaker keyword HMM, M_{BKG} is the background keyword HMM, and

$$\log(p(f_1, ..., f_N|M)) = \log(\sum_x p(f_1, ..., f_N|x, M)p(x|M)) \tag{4}$$

where x is the sequence of allowable states.

3.2. Supervectors and SVM training

Instead of computing the log-likelihoods and scoring each test utterance as in the traditional approach, we used the MAP adapted Gaussian mixture means of each target speaker keyword HMM as features in an SVM classifier. The 40-dimensional Gaussian mixture mean vectors of each component of each state (excluding the first and last states) were concatenated to form a high-dimensional supervector. The supervector concept was introduced by Campbell et al. [5] in a similar system using GMMs instead of HMMs as statistical models. Figure 2 illustrates this process.

As in [5], we trained an SVM classifier for each target speaker. All SVM training was done using the SVM^{light} software package [8]. An SVM with a linear kernel was used. For each target speaker, the supervector obtained from its keyword HMM served as the positive SVM training example, supervectors from keyword HMMs trained from example impostor speakers served as negative training examples, while those from keyword HMMs trained using data from single test utterances served as SVM test examples. The same MAP adaptation was used to train keyword HMMs for example impostor speakers (using eight conversation sides) and test utterances (using one conversation side). A total of 1,330 example impostor speakers (1,105 from Switchboard II and 225 from SRE04) were used.

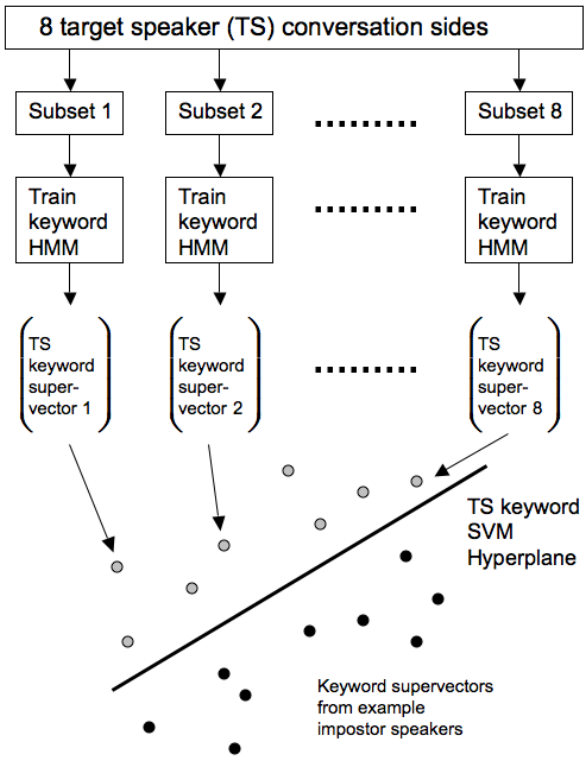

Figure 3: *Target speaker round-robin training for SVMs.*

Each target speaker keyword HMM was trained using eight conversation sides from the corresponding speaker. Because there was only one keyword HMM per target speaker, this approach implied that each target speaker SVM was trained with only one positive training example. To increase the number of positive training examples, different subsets of the eight conversation sides for a target speaker were used (in a round-robin) to train a keyword HMM per subset, and supervectors from keyword HMMs from all subsets were used as positive training examples. Hence, this round-robin training gave as many positive training examples as the number of subsets. Note that this was in contrast to the SVM training approach of [5], which used one target speaker supervector per conversation side (the possible absence of keyword instances in single conversation sides prevented us from doing the same). Figure 3 illustrates the round-robin training process.

Different weights were assigned to SVM training errors from positive and negative training examples. Because there were still many more negative training examples than positive training examples for each target speaker even after subset selection, giving the positive example training errors more weight compared to negative example training errors was desirable. Once an SVM was trained for each target speaker, they were used to classify supervectors from keyword HMMs trained from single test-utterances. If a keyword was missing in a test-utterance, the supervector from the corresponding background keyword HMM was used as a substitute.

The above approach trained one SVM for each target speaker from the corresponding keyword HMM, such that the speaker discriminative power of each keyword were determined separately. To combine the keywords, supervectors obtained from all keyword HMMs for a target speaker were concatenated into one higher-dimensional supervector for the target speaker (the same must be done for each example impostor speaker and test utterance). SVM training and testing, as previously described, were performed using the higher-dimensional supervectors to determine the combined speaker discriminative power of all keywords.

4. Experiments and results

We used the set of 19 keywords from the original keyword HMM system [4], excluding one which occurred infrequently. Our keyword list consisted of the following: *actually, anyway, i know, i mean, i see, i think, like, now, okay, right, see, uh, uhhuh, um, well, yeah, yep, you know*. Some were among the common discourse markers, back-channels, and filled pauses [4]. We then determined our system's performance using the original keywords plus 20 high-frequency keywords in the 1,553 background conversation sides: *about, all, because, but, have, just, know, mean, no, not, one, people, really, so, that, there, think, this, was, what*.

# keywords	# pos. train. examples	W	EER (%)
18	1	50	6.1
18	1	1	6.3
18	8	50	6.5
18	8	1	4.9
38	1	50	5.6
38	1	1	6.1
38	8	50	6.2
38	8	1	4.3

Table 1: *Keyword combined results.*

Several keyword combination experiments were performed using the lists of 18 and 38 keywords. For some, we applied round-robin training using subsets of 3, 5, and 7 target speaker conversation sides, giving eight positive training examples per target speaker. We experimented with weight ratios (W) of 1, 50, and 500 for positive to negative example training errors. All results were achieved on the SRE06 eight conversation task. Our best results involved weight ratios of 1 and 50, and using subsets of only 3 conversation sides for round-robin training. These are shown in table 1.

Note that the original keyword HMM system had a 5.5% EER using the list of 18 keywords. The optimal supervector HMM system, using the same list of keywords (4.9% EER), achieved a 10.9% improvement. Also, increasing the number of keywords from 18 to 38 decreased the EER from 4.9% to 4.3% (12.2% improvement). As seen in table 1, these optimal results were obtained using round-robin training and by weighting positive and negative example training errors equally. Thus, increasing the number of positive training examples from one to eight, while making positive training errors less harmful (using a weight ratio of 1 as opposed to 50), produced the best-trained target speaker SVM models.

Results were also obtained for each keyword separately. Because we were only interested in getting a general sense of how well each keyword performed, round-robin training (which was more computationally expensive) was not used to obtain multiple SVM positive training examples. The weight ratio given to positive example SVM training errors to negative example training errors was 50 to 1. Table 2 shows SRE06 eight conversation task results for the top 15 keywords along with

Keyword	EER (%)	# of occurrences
yeah	17.4	26530
you know	18.2	17349
um	19.3	11962
that	20.0	26277
like	20.5	18058
but	22.1	12766
uh	23.6	18065
right	23.6	8021
because	24.3	5164
i think	24.3	6288
have	24.8	9610
so	25.2	14291
not	25.6	6817
i mean	26.4	5470
uhhuh	27.0	8371

Table 2: *Results for top 15 keywords using one positive training example per target speaker and a 50 to 1 weight ratio for positive example training errors to negative example training errors.*

their number of occurrences in the background conversation sides. Only trials in which the keyword existed in the test utterance were used. Note that if a bigram keyword contained a unigram keyword, only the bigram keyword was counted. Comparing results in tables 1 and 2, individual keywords results were inferior to the keyword combination results. There appeared to be a correlation between keyword performance and its number of occurrences, implying that keyword performance was correlated with the amount of its training data.

Lastly, the optimal supervector HMM system (SVHMM), using 38 keywords, was fused with best results (in terms of EER) for other systems to determine the degree of orthogonality of the current approach and existing approaches. Other systems included the original keyword HMM system (HMM), a word-conditioned phone N-grams system w/ Tnorm [1] (WCPN), where phone N-gram frequencies constrained by a set of words were used as features in an SVM classifier [9], and a cepstral GMM system w/ Tnorm (GMM) [6]. Combination of results for the systems was achieved using a neural network with 2 hidden nodes and 1 hidden layer [10], trained on results from one split of SRE05+SRE06 data (7,312 trials with 1,029 true speaker trials) and tested on the other split (9,319 trials with 826 true speaker trials). Results are shown in table 3.

Fused systems	EER (%)
SVHMM	4.2
GMM	4.9
WCPN	4.6
HMM	5.5
SVHMM+GMM	4.0
SVHMM+WCPN	3.3
SVHMM+HMM	3.9
SVHMM+WCPN+GMM+HMM	2.6

Table 3: *System fusion results.*

The supervector HMM system (SVHMM) combined well with each of the previous systems, particularly with the word-conditioned phone N-grams system. The optimal result, with 2.6% EER, was achieved by combining all systems together. This result improved on our previous best.

5. Conclusion

We have extended the state of the art for speaker recognition by using keyword HMM supervectors in an SVM classifier. This approach provides a new method for using HMMs in speaker recognition, and helps achieve our best SRE06 eight conversation task result to date. We have also demonstrated that using more keywords can lead to better error-reduction in keyword-based speaker recognition systems.

6. Acknowledgements

The authors wish to thank Sachin Kajarekar for providing the GMM-based system results and Andreas Stolcke for providing word recognition decodings. This research was funded by University of California - Berkeley and NSF grant number 0329258.

7. References

[1] Bimbot, F., Bonastre, J.F., Fredouille, C., Gravier, G., Magrin-Chagnolleau, I., Meignier, S., Merlin, T., Ortega-Garcia, J., Petrovska-Delacrétaz, Reynolds, D.A., "A Tutorial on Text-independent Speaker Verification", in EURASIP Journal on Applied Signal Processing, Vol. 4, pp. 430-451, 2004.

[2] Doddington, G., "Speaker Recognition based on Idiolectal Differences between Speakers", in Proc. of Eurospeech, pp. 2521-2524, 2001.

[3] Hatch, A.O., Peskin, B., Stolcke, A., "Improved Phonetic Speaker Recognition Using Lattice Decoding", in Proc. of ICASSP, Vol. 1, pp. 169-172, 2005.

[4] Boakye, K., "Speaker Recognition in the Text-Independent Domain Using Keyword Hidden Markov Models", Masters Report, University of California at Berkeley, 2005.

[5] Campbell, W.D., Sturim, D.E., Reynolds, D.A., "Support vector machines using GMM Supervectors for Speaker Verification", in IEEE Signal Processing Letters, Vol. 13, pp. 308-311, 2006.

[6] Kajarekar, S., Ferrer, L., Venkataraman, A., Sonmez, K., Shriberg, E., Stolcke, A., Gadde, R.R., "Speaker Recognition using Prosodic and Lexical features", in Proc. of the IEEE Automatic Speech Recognition and Understanding Workshop, pp. 19-24, 2003.

[7] HMM Toolkit (HTK): http://htk.eng.cam.ac.uk

[8] Joachims, T., "Making Large Scale SVM Learning Practical", in Advances in kernel methods - support vector learning, B. Schoelkopf, C. Burges, and A. Smola, Eds. MIT-press, 1999.

[9] Lei, H., Mirghafori, N., "Word-Conditioned Phone N-grams for Speaker Recognition", in proceedings of ICASSP, 2007.

[10] Lippmann, R.P., Kukolich, L.C., Singer, E., "LNKnet: Neural Network, Machine Learning, and Statistical Software for Pattern Classification", in Lincoln Laboratory Journal, Vol. 6, pp 249-268, 1993.