A Hybrid HMM/DNN Approach to Keyword Spotting of Short Words

I-Fan Chen and Chin-Hui Lee

School of Electrical and Computer Engineering, Georgia Institute of Technology
Atlanta, GA 30332-0250, USA

ichen8@gatech.edu, chl@ece.gatech.edu

Abstract

An HMM/DNN framework is proposed to address the issues of short-word detection. The first-stage keyword hypothesizer is redesigned with a context-aware keyword model and a 9-state filler model to reduce the miss rate from 80% to 6% and increase the figure-of-merit (FOM) from 6.08% to 21.88% for short words. The hypothesizer is followed by a MLP-based second-stage keyword verifier to further reduce its putative hits. To enhance short word detection, three new techniques, including an HMM-based feature transformation for the MLPs, knowledge-based features, and deep neural networks, are incorporated into redesigning the verifier. With a set of nine short keywords from the TIMIT set the best FOM we had achieved for the proposed KWS system was 42.79%, which is comparable with that of 42.6% for long content words and much better than the FOM of 18.4% for short keywords reported in previous research [10].

Index Terms: keyword and filler modeling, keyword detection, utterance verification, deep neural networks, knowledge-based

1. Introduction

Keyword spotting (KWS) [1][2] is a task of detecting a set of preselected keywords in continuous speech. The technology has been used in various applications including speech surveillance [3], spoken term detection [4][5][6], spoken document indexing and retrieval [7], spoken message understanding [8], etc. In the past decades, a variety of KWS techniques have been proposed [1][[2][4][5][6][9][10][11][12]. In general, they can be categorized into two groups, namely: (i) large vocabulary continuous speech recognition (LVCSR) based KWS [4][5][6], either using vocabulary-dependent word lattices [4][5][6] or vocabulary-independent phone lattices [5]; and (ii) the classical keyword-filler based approach [1][2]. In LVCSR based methods, the language model plays a key role. Previous research showed that the word lattice based method has the best performance [9]. Nevertheless, the disadvantage of the word-based method is that the vocabulary and the language have to be fixed in advance. On the other hand, the phone lattice based method is more flexible and vocabulary-independent. However, its performance is usually worse than the keyword-filler network based method due to the low phone recognition accuracy [9].

Despite the vast literature, most KWS studies were not focused on detecting short-duration keywords. When compared with long words the difficulty in detecting short keywords arises due to the fact that there are usually fewer cues available in the speech segments containing the keywords for reliable detection and more possibilities of similar and confusable partial words in various parts of a spoken utterance that often trigger false alarms. In addition, the pronunciations of short words are easily affected by the neighboring speech segments. It has been clearly shown [10] that the figure-of-merit (FOM [13]) for content words (e.g., 42.6% [10]) is often much higher than that for function words (e.g., 18.4% [10]). Nonetheless, since short words account for the majority in a dictionary [14], a KWS system still needs to deal with the issue of detecting some short words. Furthermore, being able to detect short words correctly is often crucial in real-world applications because many of them carry key information in speech understanding. For example, a failure to detect the word "not" in an input utterance in dialog processing may completely reverse the meaning of a user.

To address the issue of short-word detection in the keyword-filler based KWS systems, a two-stage detection and verification framework, shown in Figure 1, is adopted in the current study. The first stage of the system is a keyword hypothesizer which generates a set of putative hits for the second-stage keyword verifier (KV) to accept or reject. Both the detection and verifications stages are redesigned especially for short keywords to enhance the detection rate and to reduce the false alarms. The best FOM we had achieved for the redesigned KWS system with a set of nine short keywords from the TIMIT data set was 42.8%.

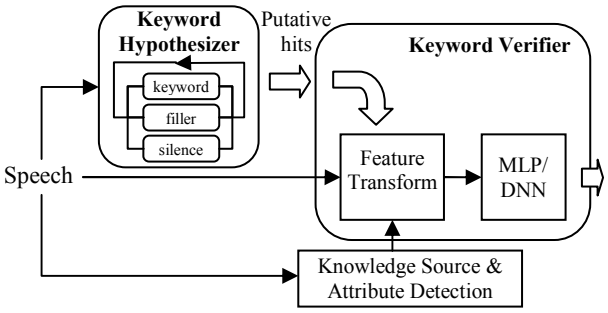

Figure 1. Proposed KWS system for short word detection.

2. Keyword hypothesizer for short words

A keyword hypothesizer shown in the upper left part of Figure 1 in a KWS system aims at detecting all possible occurrences of any given single keyword in continuous speech. Therefore the design of a good first-stage hypothesizer focuses on maintaining a high keyword detection rate while keeping the false alarms rate low, a major challenge for a system dealing with short words because they tend to be confusable with partial words in other speech segments in a spoken utterance and trigger false alarms.

2.1. Miss rate reduction for keyword hypothesizer

To reduce the miss rate, we redesign both the keyword and the filler models in our system. In the conventional keyword-filler based KWS systems, keyword models are either whole-word [1] or phone-concatenated [2] HMMs. Since monophone models do not take all contexts into account during training, the resulted keyword models do not well handle co-

articulation effect at word boundaries. For long words, this is not a major issue because the boundary phones are only a small portion of the whole word, i.e., the information provided by the center region of a keyword is enough to support good detection results. Conversely, for short words, the boundary phones occupy the most part of the word. If the keyword model does not well capture the pronunciation variation at the word boundaries, the short word may not be easily detected correctly in some cases. Therefore instead of constructing the keyword models with conventional approach for short words, we use a context-aware monoword model proposed in [15] which models keywords for the short words by concatenating the triphone HMM models. And for the boundary states, the state mixtures are the combination of the state mixtures of all possible crossword triphones, i.e., the mixture counts of the boundary states of the keyword will be more than that of the center states so that it could capture the pronunciation variation well.

In addition to using the new keyword models, filler models are redesigned in our system. Usually a filler model is made up of a phone-loop [2]. However since a keyword can also be represented as a sequence of phones, the phone-loop usually absorbs the short keywords and causes the high miss rate. To improve short-word detection, for each monophone we reduce the state mixture number to 2 and merge them into a single 3-state HMM model as our new filler model. The resulting filler model thus has 78 (2x39) Gaussian mixtures in each state.

2.2. False alarm reduction

Despite of a lower miss rate, the above system generates a great deal of false alarms. To reduce the unnecessary false alarms so that the severity of the training-data-unbalance problem for the second stage KV can be alleviated, we further increase the state number of our filler model from 3 to 9, which is selected using the development set, by repeating the original filler model 3 times, known as a garbage-collection filler [1]. This change increases the required minimum interval between two detected keywords and thus effectively reduces the number of false alarms. From another perspective, this can be seen as raising the filler model from a phone level to a word level and therefore it can be used as the word background model as well.

Maximum a posteriori probability (MAP) adaptation [16] will be applied to the keyword and filler models using the training data. This adaptation process learns the pronunciation of the keywords and non-keywords in more detail and thus improves the performance of the keyword hypothesizer.

Usually, a KWS system attempts to give each putative hit a confidence score so that by tuning a threshold we can adjust the sensitivity of each keyword detector. In our hypothesizer, the score is evaluated as the frame-normalized log-likelihood ratio (LLR) [12] for the keyword-filler over the background filler loops.

3. MLP-Based Keyword Verifier

Short words usually occur more frequently than long words [14]. For short words with plenty of training data, more complex models (e.g., MLP/DNN models) can be used for better detection performance. In the proposed system, the putative hits, containing a lot of false alarms, generated by the keyword hypothesizer will be processed by a second-stage MLP-based keyword verifier. Since the MLP models using acoustic features as input proved successful in various areas in automatic speech recognition [18][19][20], unlike conventional approaches which rescore the putative occurrences based on the likelihood outputs of the keyword hypothesizer [17], the proposed verifier extracts acoustic features from speech directly according to the word boundary information provided by the hypothesizer.

3.1. Generation of input features for MLP models

To equip the MLP model with the frame-based acoustic features, we need to transform the feature frames in each putative segment into a fixed-length feature vector. In previous research [11], this was done by dividing each hit into three equal duration segments and representing each segment by its average feature vector. It assumed the cue in each equally divided segment is stable and thus can be robustly represented by their mean. However, this is usually not true even for short words due to co-articulation.

In this study, we propose a HMM-state-alignment (HSA) [21] based feature transformation which converts the input feature frames of the putative hit into a fixed-length vector according to the state-level alignment of the keyword HMM. The acoustic feature frames of a putative hit are segmented by the state alignment; the mean of feature frames in each state is concatenated to form the input vector for the MLP model.

3.2. Knowledge-based features for MLPs

The acoustic feature used in this research is the general 12 Mel-frequency cepstral coefficients (MFCC) [22] plus energy and their first and second time derivatives [23]. Inspired by work on integrating additional acoustic-phonetic knowledge sources to assist pruning and rescoring in KWS [10], we select the Government Phonology (GP) feature set [24] as the articulatory knowledge source to further improve keyword verification.

The GP set is a feature representation derived by examining the spectral properties of speech sounds. In the GP set, sounds are divided into a collection of primes (attributes); and every phonological phenomenon can be represented by fusing the primes structurally. In this study, we use King and Taylor's modified version [25], containing 11 attributes of the GP feature set, as the additional articulatory knowledge sources for the keyword verifier. The GP features are detected at the frame level by a recurrent neural network detector proposed in [26].

To combine these detected articulatory evidences with the original MFCC features, the GP features are transformed into a fixed-length vector using the methods described in Section 3.1 and then appended them to the transformed MFCC feature vector to form a combined input feature vector for the MLP model. The MLP verifier fuses these two vectors in model training.

3.3. Deep neural network

Traditional MLPs use random seeds to initialize the model parameters. It makes the control of setting the model at a good start point difficult, sometimes even at a bad beginning area, especially for MLPs with more than one hidden layer. It made a verifier perform worse than a single-layer MLP [11].

On the other hand, deep neural network (DNN) [27], which uses MLP with more than one hidden layer for classification, is a state-of-the-art technique proved successful

in various machining learning tasks, such as computer vision [27] and speech recognition [20]. By learning a multilayer generative model as the seed, the MLP parameters are initialized at better starting areas with the help of the knowledge of data distribution. Thus, even with many hidden layers, the MLP model can still be trained well [20].

Furthermore, the variation on the input data can often be absorbed by the multiple layers, and result in robust target estimation. Since short words usually come with pronunciation variations due to the context, we adopt this DNN technique in our keyword verifier to improve the robustness of the system.

4. Experiments

4.1. Experiment setup

All the experiments are conducted on the TIMIT corpus. The TIMIT database is divided into three parts: a training set (3296 utterances, 2.79 hours), a development set (400 utterances, 0.34 hours), and a test set (1344 utterances, 1.14 hours)). The training and development sets are subsets of the standard TIMIT training set, while the test set is the standard TIMIT test set. Dialect utterances (SA1 and SA2) are not used in the experiment. The acoustic features used in the experiments are 12 mel-frequency cepstrum coefficients (MFCC) plus energy and their first and second time derivatives. The monophone HMM models, which are used for phone-loop filler model and the baseline keyword models, and the triphone HMM models, which are used for the construction of the monoword keyword model, are trained on the TIMIT training set.

4.2. Keywords

Nine short words, whose lengths are no longer than 4 phones, are selected from the original TIMIT dictionary. The words are selected based on their number of occurrence in the TIMIT corpus. Each selected word is required to have at least 50 samples in the training data. The short words used in this paper are: *was*, *with*, *his*, *this*, *from*, *not*, *but*, *every*, and *often*.

4.2.1. Performance measures

A keyword reference occurrence is considered correctly detected, or *hit*, if the mid-point of the reference occurrence falls within the time boundaries of the hypothesis. Three performance measures, the miss rate, the number of false alarm, and figure-of-merit (FOM) [13], are used in this paper. The miss rate and the false alarm rate are used to check if the keyword hypothesizer works well, namely it can successfully detect most of the keywords in the given speech with limited numbers of incorrect results. The miss rate is defined as the percentage of the undetected keywords in the total keyword occurrences in the testing speech materials. To evaluate the performance of the overall system, the FOM metric, which is an upper-bound estimate of word spotting accuracy averaged over 1 to 10 false alarms per hour, is used. An average FOM over all keywords is used as the overall performance measure.

4.2.2. MLP/DNN based keyword verifier configuration

All the MLP/DNN models of the keyword verifiers in this research have two output nodes, which correspond to the posterior probability of the input putative hit being a true hit and the probability of being a false alarm. The putative hits output of the keyword hypothesizer on the TIMIT training data are used to train the MLP/DNN models. Backpropagation algorithm with cross entropy cost function is used for training. The number of node for each hidden layer in the MLP/DNN models is set to 512. The learning rate is set to 0.001; each MLP model is trained by 1000 iteration.

For the DNN model, during pre-training the Gaussian-Bernoulli Restricted Boltzmann Machine (GB-RBM) is used as the input layer, while for the rest layers the Bernoulli Restricted Boltzmann Machine (RBM) is used. The learning rate for the GB-RBM is set to 0.002 with 75 iterations of training; while the learning rate and the number of training iteration for the RBM is 0.02 and 50, respectively. After converting the pre-trained model into an MLP model, we train the DNN model using the backpropagation algorithm as described above. Since the DNN models are trained on putative hits instead of the whole speech corpus, and the model structures are much simpler than DNNs used in the LVCSR tasks [20], the computational cost for model training is much less than the LVCSR DNN models. Here, the average training time for DNNs with 3 hidden layers is less than 6 minutes on a single i5-2400 CPU@3.10GHz personal computer. No GPU acceleration was used.

4.3. Keyword hypothesizer

The baseline keyword hypothesizer uses monophone HMM concatenated keyword model with the phone-loop filler model. Putative hits of each keyword are detected by 1-best Viterbi decoding with the keyword's corresponding keyword-filler loop grammar. The average miss rate and the average number of false alarm for the 9 short words are shown in the first row in Table 1. Due to the co-articulation effect for the short words, the keyword model constructed by concatenating monophone HMM models is not good enough to detect the keyword correctly. Most of the keywords are absorbed by the phone-loop filler and the average miss rate is as high as 80.26%. By changing the keyword model into the monoword model, which takes co-articulation into account, the miss rate is reduced to 53.55%, which means only about the half of the keywords are missing now. However, this miss rate is still too high for a keyword hypothesizer. If further replace the phone-loop filler by the proposed 3-state filler, the miss rate significantly dropped to 6.15%, which is a desired property for a good keyword hypothesizer.

Table 1. Average miss rate and average number of false alarm are taken over all the keywords for four systems.

Hypothesizer Systems	Average Miss Rate (%)	Average # False Alarm
Baseline system	80.26	15.22
Monoword model + phone-loop filler	53.55	51.22
Monoword model + 3-state filler	6.15	809.45
Monoword model + 9-state filler	6.08	616.77

One major remaining problem of the hypothesizer now is the huge increase of false alarms which often demand more careful examinations during operational situations. It also increases the computational cost for the second stage of the system and makes it hard to train the keyword verifier due to the unbalanced sizes of the keywords and non-keywords in the training data. This problem can be alleviated by increasing the filler model length to 9 states. After using the 9-state filler to

absorb extraneous speech, the number of false alarms was reduced from 809.45 to 616.77. We use this system as our final keyword hypothesizer for all the experiments throughout the remainder of the paper.

4.4. Keyword verifier (KV)

The results of the KWS systems are summarized in Table 2. The first two rows show the performance of the systems without the verifier. The baseline hypothesizer used the phone-concatenated keyword model and the phone-loop filler model. Due to the high miss rate, the baseline system has a very low FOM of 6.08% on these 9 short words. By adding the context-aware monoword model and the 9-state filler model, the proposed hypothesizer has a better FOM of 21.88% shown on the second row.

The third to seventh rows in Table 2 show the system FOM by imposing different second-stage keyword verifiers. Unlike what's reported in [11], the FOM decreased slightly after adding the keyword verifier with an equal-segment feature transform [11]. This change in results could come from different definitions of the keyword list. With the proposed HMM-based feature transform, the FOM improved from 21.88% to 27.23% as shown in the fourth row of Table 2. After introducing the detected GP attributes to the keyword verifier as the additional knowledge source, the overall system performance was further improved to 38.08%.

Table 2. Comparison of the KWS systems with different settings. MLP n=1: a single hidden layer. MLP n≥1: greater than or equal to 1 hidden layer, while the number n is selected empirically based on the performance on the development set.

KWS Systems	FOM (%)	Runtime (sec)	Real Time Factor
Baseline Keyword Hypothesizer	6.08	129	0.03
Proposed Keyword Hypothesizer	21.88	83.6	0.02
KV MLP n=1 (3-eq-seg transform) [MFCC]	19.01	83.6 + 0.012	0.02
KV MLP n=1 (HSA transform) [MFCC]	27.23	83.6 + 0.012	0.02
KV MLP n=1 (HSA transform) [MFCC+GP]	38.08	83.6 + 0.013	0.02
KV MLP n≥1 (HSA transform) [MFCC+GP]	37.68	83.6 + 0.037	0.02
KV DNN n≥1 (HSA transform) [MFCC+GP]	42.78	83.6 + 0.037	0.02

So far the MLPs used for the verifiers contain only one hidden layer. It would be interesting to know if more hidden layers and pre-training proposed in DNN help the system performance. The sixth and bottom rows in Table 2 display the FOMs when using multiple hidden layers. The number of the hidden layers was decided by the model performance on the development set. Without pre-training, the FOM of the verifiers with more than one hidden layers dropped slightly to 37.68%, which verified the finding in [11]. However, by employing pre-training, the FOM of the DNN-based keyword verifiers further improved to 42.78%.

Table 2 also compares runtime of the systems. The baseline keyword hypothesizer is slower than the proposed one because of the larger search space introduced by the phone-loop filler. The average runtime for the proposed keyword hypothesizer processing the 1.14-hour test data is 83.6 seconds, and the real time factor (RTF) is 0.02. Notice that the second-stage MLP/DNN KVs take less than 0.1 seconds to rescore the putative hits, and RTFs of the systems remain about 0.02.

Figure 2 shows the ROC curves of the three systems in this research. The conventional keyword spotting system is the lowest curve in Figure 2 due to the high miss rate. The proposed keyword hypothesizer represented as the blue line with cross marks is much higher than the conventional system. Further, the system with the proposed keyword verifier, which shown as the red line with triangle marks, has significant improvement over the other systems.

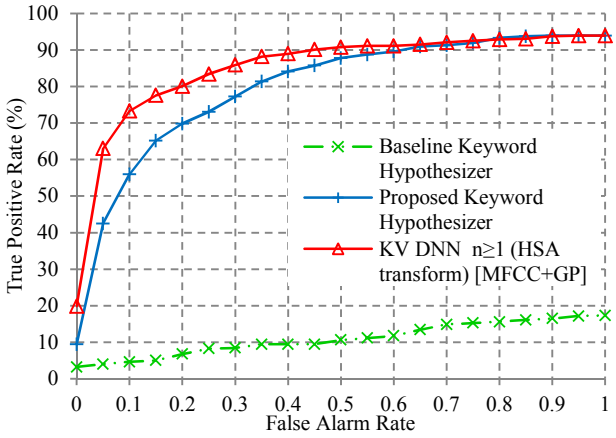

Figure 2. ROC curves for the baseline and proposed keyword hypothesizers and our final KWS system.

In summary, the proposed keyword hypothesizer has an FOM improvement of 15.80% over the baseline system on short words. Using the keyword verifier with HSA feature transform provided another 5.35% of the FOM improvement. The FOM can be further improved by introducing the GP knowledge source which brought another 10.85% of the FOM improvement. Finally by using the DNN techniques, another 4.70% of FOM improvement can be achieved.

5. Discussion and Future Work

A two-stage hybrid HMM/MLP framework for keyword spotting of short words is proposed. The first-stage keyword hypothesizer is redesigned to address key issues related to detecting short words, while the proposed knowledge-based features, feature transformation, and the DNN verifiers are integrated into redesigning the second-stage keyword verifiers. Experimental results show that the proposed KWS system achieves a sizable FOM improvement – from 6.08% for the baseline system to 42.78%.

Typically short words occur a lot in natural and spontaneous speech [28]. This makes the proposed approach quite appealing. For future work, we plan to test our approach on speech data like the Switchboard corpus to study the performance of the approach under circumstances of spontaneous speech and noises. Also, we believe our proposed KWS system can be further improved by adding a third stage, which does knowledge-based pruning and rescoring by utilizing speech attributes, such as manner and place, as described in [10], after the keyword verifier. Since our current framework requires the selected words to have enough occurrences in the training set, this requirement may not be easy to satisfy for some words, such as content words. A different strategy would be needed in a future study.

6. References

[1] J. G. Wilpon, L. R. Rabiner, C.-H. Lee, and E. R. Goldman, "Automatic Recognition of Keywords in Unconstrained Speech Using Hidden Markov Models," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 38, no. 11, Nov. 1990.

[2] R. C. Rose and D. B. Paul, "A Hidden Markov Model Based Keyword Recognition System," in *Proc. of ICASSP*, 1990.

[3] R. L. Warren. "BROADCAST SPEECH RECOGNITION SYSTEM FOR KEYWORD MONITORING." U.S. Patent 6332120 B1, Dec. 18, 2001.

[4] D. R. H. Miller, M. Kleber, C.-L Kao, O. Kimball, T. Colthurst, S. A. Lower, R. M. Schwartz, and H. Gish, "Rapid and Accurate Spoken Term Detection," in *Proc. of Interspeech*, 2007.

[5] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary Independent Spoken Term Detection," in *Proc. of SIGIR*, 2007, pp. 129-136

[6] D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang, "The SRI/OGI 2006 Spoken Term Detection System," in *Proc. of Interspeech*, 2007.

[7] J. S. Garofolo, C. G. P. Auzanne, and E. M. Voorhees, "The TREC Spoken Document Retrieval Track: A Success Story," in *Proc. of The Text Retrieval Conf. (TREC-8)*, 1999.

[8] B.-H Juang and S. Furui, "Automatic Recognition and Understanding of Spoken Language – A First Step Toward Natural Human-Machine Communication," in *Proc. of IEEE*, vol. 88, no. 8, pp. 1142-1165, Aug. 2000.

[9] I. Szoeke, P. Schwarz, P. Matejka, L. Burget, M. Karafiat, M. Fapso and J. Cernocky, "Comparison of Keyword Spotting Approaches for Informal Continuous Speech," in *Proc. of ICASSP*, 2005.

[10] C. Ma and C.-H. Lee, "A Study on Word Detector Design and Knowledge-based Pruning and Rescoring," in *Proc. of Interspeech*, 2007.

[11] R. P. Lippmann and E. Singer, "Hybrid Neural-Network/HMM Approaches to Wordspotting," in *Proc. of ICASSP*, 1993.

[12] M. Weintraub, "LVCSR LOG-LIKELIHOOD RATIO SCORING FOR KEYWORD SPOTTING," in *Proc. of ICASSP*, 1995

[13] NIST, "The Road Rally Word-spotting Corpora (RDRALLY1)," NIST Speech disc 6-1, 1991.

[14] B. Sigurd, M. Eeg-Olofsson, and J. van de Weijer, "Word length, sentence length and frequency – ZIPF revisited," Studia Linguistica 58(1), pp.37-52, 2004.

[15] I.-F. Chen and C.-H. Lee, "A Study on Using Word-Level HMMs to Improve ASR Performance over State-of-the-Art Phone-Level Acoustic Modeling for LVCSR," in *Proc. of Interspeech*, 2012.

[16] J.-L. Gauvain and C.-H. Lee, "Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains," *IEEE Trans. on Speech and Audio Processing*, vol. 2, no. 2, Apr. 1994.

[17] J. Ou, K. Chen, X. Wang, and Z. Li, "Utterance Verification of Short Keywords Using Hybrid Neural-Network/HMM Approach," in *Proc. of ICII*, 2001.

[18] H. Hermansky, D. P.W. Ellis, and S. Sharma, "Tandem Connectionist Feature Extraction for Conventional HMM Systems," in *Proc. of ICASSP*, 2000.

[19] S. M. Siniscalchi, "Combining speech attribute detection and penalized logistic regression for phoneme recognition," Neurocomputing , vol. 93, pp. 10-18, 2012.

[20] A.-R Mohamed, G. E. Dahl, and G. Hinton, "Acoustic Modeling using Deep Belief Networks," *IEEE Trans. on Audio, Speech, and Language Processing*, 2012.

[21] K.-Y Su and C.-H. Lee, "Speech Recognition using Weighted HMM and Subspace Projection Approaches," *IEEE Trans. on Speech and Audio Processing*, vol. 2, no. 1, pp 69-79, Jan. 1994.

[22] S. B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357-366, 1980.

[23] S. Furui, "Speaker Independent Isolated Word Recognition Using Dynamic Feature of Speech Spectrum," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 34, pp. 52-59, 1986.

[24] J. Harris, English Sound Structure, Blackwell, 1994.

[25] S. King and P. Taylor, "Detection of Phonological Features in Continuous Speech using Neural Networks," Computer Speech and Language, vol. 14, pp. 333-353, 2000.

[26] I.-F. Chen and H.-M. Wang, "Articulatory Feature Asynchrony Analysis and Compensation in Detection-Based ASR," in *Proc. of Interspeech*, 2009.

[27] G. E. Hinton, S. Osindero, and Y. W. The, "A fast learning algorithm for deep belief nets," Neural Computation, vol. 18, no. 7, pp. 1527-1554, 2006.

[28] P. Jeanrenaud, E. Eide, U. Chaudhari, J. McDonough, K. Ng, M. Siu, and H. Gish, "Reducing Word Error Rate on Conversational Speech from The SWITCHBOARD Corpus," in *Proc. of ICASSP*, 1995.