



LSTM based Attentive Fusion of Spectral and Prosodic Information for Keyword Spotting in Hindi Language

Laxmi Pandey¹ and Karan Nathwani²

¹Indian Institute of Technology, Kanpur, India

²Indian Institute of Technology, Jammu, India

laxmip@iitk.ac.in, karan.nathwani@iitjammu.ac.in

Abstract

In this paper, a DNN based keyword spotting framework, that utilizes both spectral as well as prosodic information present in the speech signal, is proposed. A DNN is first trained to learn a set of hierarchical non-linear transformation parameters that project the original spectral and prosodic feature vectors onto a feature space where the distance between similar syllable pairs is small and between dissimilar syllable pairs is large. These transformed features are then fused using an attention-based long short-term memory (LSTM) network. As a side result, a deep denoising autoencoder based fine-tuning technique is used to improve the performance of sequence predictions. A sequence matching method called the sliding syllable protocol is also developed for keyword spotting. Syllable recognition and keyword spotting (KWS) experiments are conducted specifically for the Hindi language which is one of the widely spoken languages across the globe but is not addressed significantly by the speech processing community. The proposed framework indicates reasonable improvements when compared to baseline methods available in the literature.

Index Terms - Discriminative deep metric learning, deep denoising autoencoder, long short term memory

1. Introduction

Keyword spotting (KWS) is a challenging task especially in the context of audio retrieval applications where the variability in the audio query is very high. The variability in an audio query is due to the regional variation in articulation rate and its manifestations across speaking situations. In order to differentiate among these variations, prosodic information is also used in combination with spectral information in this work. There is a considerable and significant work on KWS [1], [2], [3], [4] based on confidence measures and feature fusion methods. Keyword spotting using Deep Neural Networks (DNN) has also been studied in [5], [6], [7]. In a recent work [5], author used a 4 hidden layered DNN trained with context dependent triphones and has reported a significant improvements in KWS performance. In another work [6], authors proposed a 2-level context-aware keyword verifier, using Hidden Markov Model (HMM)-feature transformation for Multilayer Perceptrons (MLPs) and DNNs, for spotting keywords. The system with a CNN-RNN character Language Model (LM), and a KWS neural network is studied in [8].

In this work, a novel framework is proposed that uses an attention based deep network for the fusion of two mutually exclusive spectral and prosodic information belonging to same speech signal. Here, two exclusive information refers to two different types of features derived from short term power spectrum of the speech signal and from pitch contour (F_0). For

each frame, a network predict which information is the most relevant by learning the attention scores as a latent variable in the network. Frame level attention to the appropriate information leads to the improved sequence prediction for keyword spotting in Hindi language. The inputs to the attention network are spectral and prosodic features which are extracted from a pretrained DNN that projects the original feature vectors onto a feature space where the distance between similar syllable pairs is small and between dissimilar syllable pairs is large.

Further, motivated by the success of the long short-term memory (LSTM) [9], [10], [11] networks for the task of sequence prediction, the attention based transformed features is then fed to the LSTM to model the audio data as a sequence of frames. Subsequently an error modeling scheme is also developed to model errors in LSTMs. The errors in the syllabic output obtained from baseline LSTMs are corrected using Deep Denoising Autoencoders (DDA) [12], [13], [14]. The DDA predicts an improved syllabic sequence by minimizing the error between predicted sequence from the LSTM and the ground truth labels. To facilitate this, both the originally predicted sequence and the ground truth labels are represented as a one-hot encoded vectors prior to being fed to the denoising autoencoder. The final syllabic sequence obtained from DDA exhibits reasonable improvement when compared to the original syllabic sequence obtained from the baseline LSTM. The improved syllabic sequence thus obtained is then used in a sequence matching method called the sliding syllable protocol for keyword spotting giving significant gain in the keyword spotting performance for Hindi language.

2. Discriminative Deep Metric Learning based Feature Embeddings

The methodology for extracting spectral and prosodic information from a given speech signal is presented here. Further, a new strategy to transform the obtained features into a new feature space is presented, where the new space is such that the similar units lie together and dissimilar units lie far apart. This strategy utilizes deep neural networks and is broadly inspired from Discriminative Deep Metric Learning (DDML).

In order to describe a speech utterance, we use one of the most established method of feature extraction i.e. Mel frequency Cepstral Coefficients (MFCC) [15], [16] also referred to as spectral feature. We also introduce a new feature format taking into account the prosodic information of speech signal. In order to extract prosody features [17], the fundamental frequency contour (F_0) for each voiced segment of speech signal is calculated using auto correlation method. The mean, median and standard deviation of F_0 contour is then computed and short term energy is also obtained from energy contour for ev-

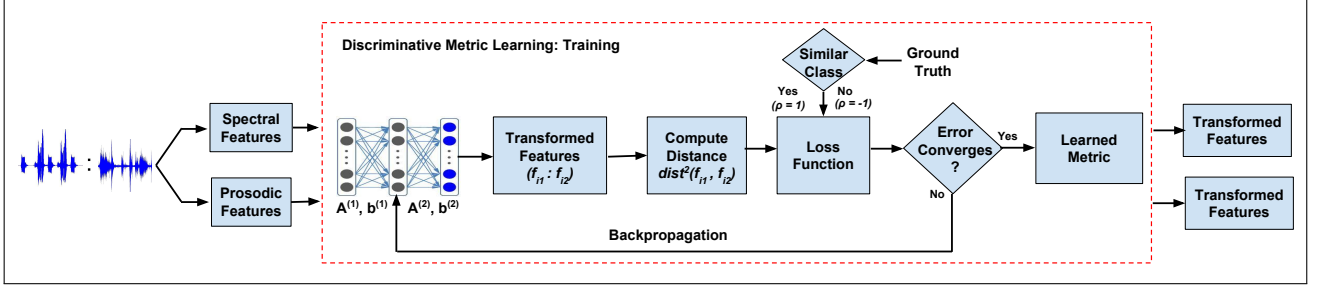


Figure 1: Metric Learning Architecture in a DNN Framework for Feature Transformation

ery voiced region of speech signal. The duration of the voiced segment for accounting speaking rate and the individual frame auto correlation measures are also added to the feature set. The 13-dimensional MFCC features and 6-dimensional prosody features can be illustrated as,

$$\mathbf{f}_m = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_{13}]^T, \quad \mathbf{f}_p = [\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_6]^T \quad (1)$$

2.1. Feature Transformation using DDML

Motivated by the application of metric learning [18], a DNN has been designed to learn a better representation of an audio signal. The constructed network consist of N hidden layers and $l^{(n)}$ units in the n^{th} layer, $\forall n = \{1, 2, \dots, N\}$. Assuming a fixed time length for every syllable, the input audio signal is broken down to get audio signal at syllable level. The fixed time window used for each syllable is 200ms, which corresponds to 20 frames. For each syllable, a 13-dimensional MFCC [15], [16] features and 6-dimensional prosody features are computed which is further feed-forwarded through the network to get the transformed features.

The proposed architecture for deep metric learning is illustrated with the help of block diagram in Figure 1. Let's denote an input speech as $X \in \mathbb{R}^d$, and the corresponding output at the first layer is given by,

$$\mathbf{y}_i^{(1)} = j(\mathbf{A}^{(1)} \mathbf{f}_i + \mathbf{b}^{(1)}) \in \mathbb{R}^{l^{(1)}} \quad (2)$$

where, $i \in \{m, p\}$, $\mathbf{A}^{(1)} \in \mathbb{R}^{l^{(1)} \times d}$ is a projection matrix,

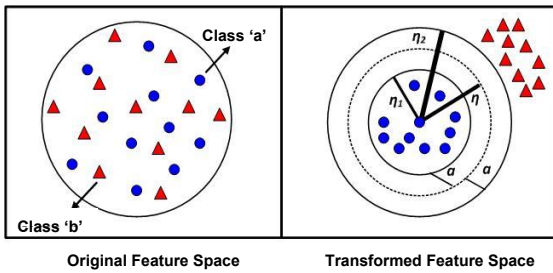


Figure 2: Visualization of the original and transformed feature space

$\mathbf{b}^{(1)} \in \mathbb{R}^{l^{(1)}}$ is a bias vector, and $j : \mathbb{R} \mapsto \mathbb{R}$ is an element-wise nonlinear activation function, where $l^{(1)}$ is the number of nodes in first hidden layer. Similarly the output of the N^{th} layer can be computed as:

$$\mathbf{F}(x) = \mathbf{y}_i^{(N)} = j(\mathbf{A}^{(N)} \mathbf{y}_i^{(N-1)} + \mathbf{b}^{(N)}) \in \mathbb{R}^{l^{(N)}} \quad (3)$$

where the mapping $\mathbf{F} : \mathbb{R}^d \mapsto \mathbb{R}^{l^{(n)}}$ is a parametric nonlinear function determined by the parameters $\mathbf{A}^{(n)}$ and $\mathbf{b}^{(n)}$. The last layer output, for a pair of input features \mathbf{f}_{i_j} and \mathbf{f}_{i_k} , is represented as $\mathbf{F}(\mathbf{f}_{i_j}) = \mathbf{y}_{i_j}^{(N)}$ and $\mathbf{F}(\mathbf{f}_{i_k}) = \mathbf{y}_{i_k}^{(N)}$. The distance between them is measured using two different criterion, namely Euclidean Distance shown in eq. (4) and Cosine Distance shown in eq. (5).

$$dist^2(\mathbf{f}_{i_j}, \mathbf{f}_{i_k}) = \|\mathbf{F}(\mathbf{f}_{i_j}) - \mathbf{F}(\mathbf{f}_{i_k})\|_2^2 \quad (4)$$

$$dist^2(\mathbf{f}_{i_j}, \mathbf{f}_{i_k}) = \left(1 - \frac{\langle \mathbf{F}(\mathbf{f}_{i_j}), \mathbf{F}(\mathbf{f}_{i_k}) \rangle}{\|\mathbf{F}(\mathbf{f}_{i_j})\| \|\mathbf{F}(\mathbf{f}_{i_k})\|}\right)^2 \quad (5)$$

The proposed network is trained to attain a non-linear mapping which ensures that the syllables belonging to same class ($p = 1$) are close to each other i.e. $dist^2(\mathbf{f}_{i_j}, \mathbf{f}_{i_k})$ is less than η_1 and the syllable belonging to different classes ($p = -1$) are far apart i.e. $dist^2(\mathbf{f}_{i_j}, \mathbf{f}_{i_k})$ is greater than η_2 . The pairwise label p denotes the similarity between an input pair \mathbf{f}_{i_j} and \mathbf{f}_{i_k} . η_1 and η_2 are connected with threshold η , such that $\eta_1 = \eta - \alpha$ and $\eta_2 = \eta + \alpha$. This condition can be enforced by using the constraint:

$$p(\eta - dist^2(\mathbf{f}_{i_j}, \mathbf{f}_{i_k})) > \alpha; \quad \eta > \alpha \quad (6)$$

Thus, the loss function is formulated as,

$$S = \sum_{j,k} h(\alpha - p(\eta - dist^2(\mathbf{f}_{i_j}, \mathbf{f}_{i_k}))) \quad (7)$$

where $h(z) = \frac{1}{\beta} \log(1 + \exp(\beta z))$ and β represents the sharpness parameter. This constraint is used to set a margin between each similar and dissimilar pairs of a syllable class in the transformed feature space, as shown in Figure 2.. By using learnt weights and biases after training, the transformed features can be obtained as,

$$\mathbf{f}_i^t = \mathbf{A}^{(n)} \mathbf{f}_i + \mathbf{b}^{(n)} \quad (8)$$

3. Attention Based LSTM Modeling for the Fusion of Spectral and Prosodic Information

This section proposes an strategy to handle fusion of two exclusive information, where each information has its own sequence of feature vectors. A basic fusion approach can be the simple concatenation of spectral and prosody features to form a single monolithic feature (Spectral+Prosody). But, as the spectral and prosodic information corresponding to a speech signal lie in different feature spaces, the simple concatenation of the two information is sub-optimal. So, we propose an attention based fusion mechanism as shown in Figure 3. This empowers the network to emphasize particular features for every frame depending on the context, thus providing the better accuracy. As the

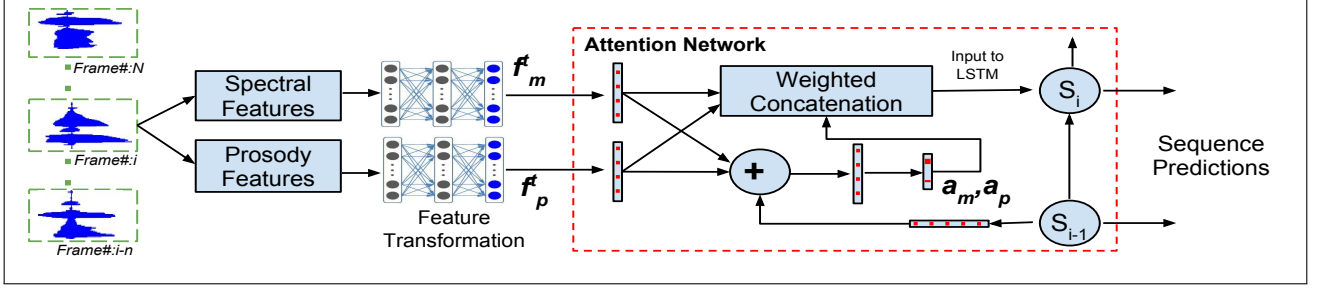


Figure 3: Training Architecture for Attention based LSTM Modeling

spectral and prosodic information corresponding to each frame of speech signal lie in different feature spaces, we pass them through separate fully connected layers,

$$\begin{aligned} \mathbf{x}_m &= \tanh(\mathbf{W}_m \mathbf{f}_m^t + \mathbf{b}_m) \\ \mathbf{x}_p &= \tanh(\mathbf{W}_p \mathbf{f}_p^t + \mathbf{b}_p) \end{aligned} \quad (9)$$

where $\mathbf{W}_m \in \mathbb{R}^{d_m \times n}$ and $\mathbf{W}_p \in \mathbb{R}^{d_p \times n}$ are the projection weights of the two features aligned representations. d_m, d_p are the feature dimensions respectively, n is the projected feature dimension, and $\mathbf{b}_m, \mathbf{b}_p$, are the corresponding biases. The information from all the past frames is encoded in \mathbf{x}_s as follows.

$$\mathbf{x}_s = \tanh(\mathbf{W}_s \mathbf{s} + \mathbf{b}_s), \quad (10)$$

where \mathbf{s} and $\mathbf{W}_s \in \mathbb{R}^{d_s \times n}$ denote the LSTM states, and the projection weights for the LSTM states, d_s being the dimension of the LSTM states. The aligned features and the history from past frames are then passed through a 2-layer neural network with a softmax in the end to generate attention weight distribution corresponding to spectral and prosody information.

$$\begin{aligned} \mathbf{x}'_i &= \tanh(\mathbf{W}_a \mathbf{x}_i + \mathbf{b}_a) \quad \forall i \in \{m, p, s\} \\ \mathbf{x}_a &= [\mathbf{x}'_m, \mathbf{x}'_p, \mathbf{x}'_s] \end{aligned} \quad (11)$$

$$\mathbf{a} = \text{softmax}(\mathbf{W}_{att} \mathbf{x}_a + \mathbf{b}_{att}) \quad (12)$$

where $\mathbf{W}_{att} \in \mathbb{R}^{3n \times 2}$ are the weights for the attention layer. Finally, \mathbf{a} contains the attention weights corresponding to the spectral and prosody information. Once the attention weights are obtained, each of the spectral and prosodic features are then multiplied with their corresponding weights.

$$\mathbf{f}_t = \mathbf{a}_{m,t} \mathbf{f}_{m,t}^t + \mathbf{a}_{p,t} \mathbf{f}_{p,t}^t \quad (13)$$

Where \mathbf{a}_m and \mathbf{a}_p are the attention weights corresponding to each frame t . The LSTM cell takes an input \mathbf{f}_t for every frame t , and updates the memory cell \mathbf{s}_t in consultation with its previous state \mathbf{s}_{t-1} . The LSTM is equipped with several gates which control the update process. A forget gate \mathbf{g}_t decides how much information from the past state \mathbf{s}_{t-1} is carried forward. An input gate \mathbf{i}_t supervises the information from the current input vector \mathbf{x}_t . An output gate \mathbf{o}_t puts a check on the information that need to fed to the output as a hidden state. The state update process followed by LSTM is as follows:

$$\begin{aligned} \mathbf{g}_t &= \sigma(\mathbf{W}_g \mathbf{x}_t + \mathbf{W}_g \mathbf{h}_{t-1} + \mathbf{b}_g) \\ \mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{W}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\ \mathbf{s}_t &= \mathbf{g}_t \mathbf{s}_{t-1} + \mathbf{i}_t \tanh(\mathbf{W}_s \mathbf{x}_t + \mathbf{W}_s \mathbf{h}_{t-1} + \mathbf{b}_s) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{W}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \end{aligned} \quad (14)$$

$$\mathbf{h}_t = \mathbf{o}_t \tanh(\mathbf{s}_t) \quad (15)$$

where, the \mathbf{W} s and \mathbf{b} s are the weight matrices and the bias vectors that are learned during the training phase.

Further, a Deep Denoising Autoencoders (DDA) [12] is used to correct the predicted sequence obtained from LSTM. DDA have been successful in the context of reconstructing a noisy signal. A syllable sequence is represented in form of a matrix, where each row is a one-hot encoded vector, pointing to the particular syllable out of all. Thus, the output of the LSTM is represented as a binary matrix. An input to autoencoder is converted to a fixed length sequence either by subdividing the sequence or by appending zero vectors depending on a length of the sequence. This fixed length matrixized sequence is fed-forwarded through a DDA to obtain the improved phoneme predictions. The DDA is trained with the matrixized output of LSTM as input and the matrixized original sequence as the labels. This helps in reconstructing the sequence and thus improving the syllable sequence prediction performance. Figure 4 shows the training architecture for learning DDA in context to syllable sequence prediction. In order to quantify the errors between denoised sequence and the ground truth, the loss function used is the cross-entropy loss of the matrixized sequences, which is given by,

$$\text{Loss}(x, z) = - \sum_{k=1}^d [x_k \log z_k + (1 - x_k) \log(1 - z_k)] \quad (16)$$

where x represents the matrixized predicted output from HMM and the z represents the ground truth sequence in vector form.

4. Performance Evaluation

The performance of the proposed method is validated on domain specific Hindi dataset (news and speech) collected from YouTube [19]. The dataset has 19-hours of audio data with 6840 audio clips of 10sec each. Each audio clip is transcribed manually at syllable level using IPA alphabets. Each audio contains average of 50 syllables. The dataset is divided into a split of 80 : 10 : 10% as *train* : *validation* : *test*.

Discriminability Analysis: A 2-layered network with {512, 256} nodes are used to learn the discriminative transformation. Figure 5 shows the visualization of distance between positive samples (similar phonemes) and negative samples (dissimilar phonemes) in native form and after proposed transformation. Each point in the plot represents a tuple which consists of the distances when Cosine distance (CD) and Euclidean distance (ED) are used respectively. Clearly, the distance between all

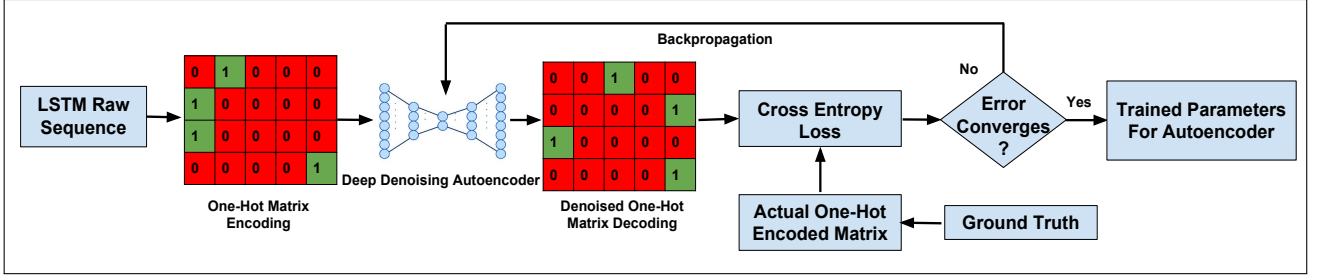


Figure 4: Training Architecture for Deep Denoising Autoencoder

positive samples is smaller than the predefined threshold i.e. $(\eta - \alpha) = 4(ED)$ and the distance between all negative samples is greater than the threshold i.e. $(\eta + \alpha) = 6(ED)$, using Equation (6).

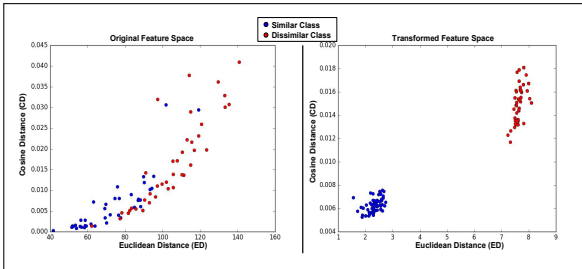


Figure 5: Visual representation of distances between similar and dissimilar samples obtained after metric learning

Syllable Sequence Predictions: After transformation, the spectral and prosody features within a volume of 20 frames are max pooled to learn 2 representational layers, one for each descriptor, which projects the respective descriptor to a 512 dimensional output space. The aligned descriptors, and the LSTM states from the previous time step are projected to 128 dimensions by another fully connected layer, and are stacked together (see Eq. (11)). Finally they are feedforwarded through the last layer, and a *softmax* is applied to obtain attention distribution over different informations (see Eq. (12)). We use the cross entropy loss as the loss function, and the RMSProp gradient descent method to learn the parameters of LSTM, and the attention network. Then, a 5-layer DDA with {256, 128, 64, 128, 256} nodes are used as an error correcting module. Clearly, the proposed framework performs reasonably better than other conventional method as shown in Table 1.

KWS using Sliding Syllable Protocol: KWS refers to searching a frequently occurring keyword within an audio archive. In order to accommodate a large numbers of syllables, for continuous speech, an extra class called 'gar' is used for all non-keyword syllables present in a continuous speech to facilitate the one-hot encoding of each syllable. A network is trained for the set of $\sum_i M_i$ classes, with total of 71 keywords and M_i being the number of syllables in the i^{th} keyword, and one class is trained as garbage class 'gar'. For KWS, an automatic syllabic transcription of the test audio is obtained along with the time stamp corresponding to each syllabic unit. Every syllable of a keyword is then matched with the recognized output string as illustrated in Figure 6. If at least two consecutive syllables in the overall string match, then the keyword is spotted [20].

Table 1: Performance of proposed methods for syllable recognition (SR) and keyword spotting (KWS) using recognition accuracy (Acc. (%)), TPR and FPR respectively

Proposed Methods	SR	KWS
	Acc.	Acc.(TPR,FPR)
Spectral	69.9	83.5 (0.90,0.35)
Spectral+Prosody	70.1	84.4 (0.91,0.34)
Spectral+Prosody+DDML	84.6	91.9 (0.95,0.26)
DDML+Attn-LSTM	86.6	93.01 (0.96,0.24)
DDML+Attn-LSTM+DDA	87.2	94.8 (0.98,0.21)
DNN-HMM [5] (SR:baseline)	81.3% (Acc.)	
DNN-HMM [5] (KWS:baseline)	88.5% (Acc.)	

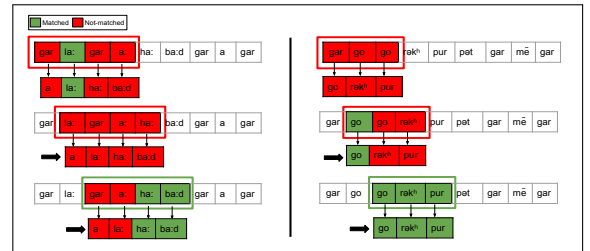


Figure 6: Sliding syllable protocol for keyword spotting. Green box shows the spotting of keywords (3rd step) in both examples, satisfying the criterion of atleast two consecutive matches

The effectiveness of the proposed framework are evaluated in terms of accuracy and True Positive Rate (TPR) and False Positive Rate (FPR) as shown in Table 1. It can be observed that the experiment (DDML+Attn-LSTM+DDA) performs the best as it has high TPR and low FPR.

5. Conclusions and Future Scope

In this work, an improved methodology for KWS in Hindi language, by fusing spectral and prosodic information using a deep network architecture, is proposed. Prior to fusion, a feature transformation method using deep neural nets is proposed. As a side result, a new error modeling technique for improving the predicted syllable sequence using denoising autoencoder is also presented. The performance of the proposed framework is evaluated on syllable recognition and keyword spotting which indicates 5.9 % and 6.3 % improvement over corresponding DNN-HMM baseline. Future work will focus on developing keyword search methods for reverberant and noisy speech.

6. References

- [1] Y. Benayed, D. Fohr, J. P. Haton, and G. Chollet, "Confidence measures for keyword spotting using support vector machines," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, 2003, pp. 588–591.
- [2] J. Junkawitsch, L. Neubauer, H. Hoge, and G. Ruske, "A new keyword spotting algorithm with pre-calculated optimal thresholds," in *Fourth International Conference on Spoken Language, ICSLP*, vol. 4, 1996, pp. 2067–2070.
- [3] J. Junkawitsch, G. Ruske, and H. Höge, "Efficient methods for detecting keywords in continuous speech," in *Fifth European Conference on Speech Communication and Technology (EUROSPEECH)*, vol. 97, 1997, pp. 259–262.
- [4] V. Mitra, J. Van Hout, H. Franco, D. Vergyri, Y. Lei, M. Graciarana, Y.-C. Tam, and J. Zheng, "Feature fusion for high-accuracy keyword spotting," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7143–7147.
- [5] V. Tyagi, "Hybrid context dependent CD-DNN-HMM keyword spotting (KWS) in speech conversations," in *26th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016, pp. 1–6.
- [6] I.-F. Chen and C.-H. Lee, "A hybrid HMM/DNN approach to keyword spotting of short words." in *INTERSPEECH*, 2013, pp. 1574–1578.
- [7] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 4087–4091.
- [8] K. Audhkhasi, A. Rosenberg, A. Sethy, B. Ramabhadran, and B. Kingsbury, "End-to-end asr-free keyword search from speech," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1351–1359, 2017.
- [9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [10] P. Goelet, V. F. Castellucci, S. Schacher, and E. R. Kandel, "The long and the short of long-term memory—a molecular framework," *Nature*, vol. 322, no. 6078, p. 419, 1986.
- [11] A. D. Baddeley and E. K. Warrington, "Amnesia and the distinction between long-and short-term memory," *Journal of verbal learning and verbal behavior*, vol. 9, no. 2, pp. 176–189, 1970.
- [12] X. Feng, Y. Zhang, and J. Glass, "Speech feature denoising and dereverberation via deep autoencoders for noisy reverberant speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1759–1763.
- [13] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder." in *INTERSPEECH*, 2013, pp. 436–440.
- [14] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371–3408, 2010.
- [15] W. Han, C.-F. Chan, C.-S. Choy, and K.-P. Pun, "An efficient MFCC extraction method in speech recognition," in *IEEE International Symposium on Circuits and Systems, 2006. (ISCAS)*, 2006, pp. 145–148.
- [16] M. J. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.
- [17] R. Hegde, B. Manoj, B. Rao, R. Rao *et al.*, "Emotion detection from speech signals and its applications in supporting enhanced qos in emergency response," in *Proceedings of the 3rd International ISCRAM Conference*, 2006.
- [18] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1875–1882.
- [19] L. Pandey, K. Nathwani, S. Kaur, I. Husain, R. Pathak, G. Singh, S. Tiwari, and R. M. Hegde, "Domain specific audio indexing using linguistic information," in *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2014, pp. 000 364–000 369.
- [20] L. Pandey, K. Chaudhary, and R. M. Hegde, "Fusion of spectral and prosodic information using combined error optimization for keyword spotting," in *Twenty-third IEEE National Conference on Communications (NCC)*, 2017, pp. 1–6.