AN INVESTIGATION OF LSTM-CTC BASED JOINT ACOUSTIC MODEL FOR INDIAN LANGUAGE IDENTIFICATION

by

Tirusha Mandava, Ravi Kumar Vuddagiri, Hari Vydana, Anil Kumar Vuppala

in

IEEE Automatic Speech Recognition and Understanding Workshop
(*ASRU-2019*)

Sentosa, Singapore

Report No: IIIT/TR/2019/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
December 2019

AN INVESTIGATION OF LSTM-CTC BASED JOINT ACOUSTIC MODEL FOR INDIAN LANGUAGE IDENTIFICATION

Tirusha Mandava, Ravi Kumar Vuddagiri, Hari Krishna Vydana, and Anil Kumar Vuppala

Speech Processing Laboratory
International Institute of Information Technology, Hyderabad, India.
{mandava.tirusha, ravikumar.v, hari.vydana}@research.iiit.ac.in, anil.vuppala@iiit.ac.in

ABSTRACT

In this paper, phonetic features derived from the joint acoustic model (JAM) of a multilingual end to end automatic speech recognition system are proposed for Indian language identification (LID). These features utilize contextual information learned by the JAM through long short-term memory-connectionist temporal classification (LSTM-CTC) framework. Hence, these features are referred to as CTC features. A multi-head self-attention network is trained using these features, which aggregates the frame-level features by selecting prominent frames through a parametrized attention layer. The proposed features have been tested on IIITH-ILSC database that consists of 22 official Indian languages and Indian English. Experimental results demonstrate that CTC features outperformed i-vector and phonetic temporal neural LID systems and produced an 8.70% equal error rate. The fusion of shifted delta cepstral and CTC feature-based LID systems at the model level and feature level further improved the performance.

Index Terms— Equal error rate, Joint acoustic model, Language identification system, Multi-head, Self-attention mechanism

1. INTRODUCTION

Language identification (LID) refers to the task of identifying the language being spoken by a speaker in a given utterance. LID is an active research area with notable applications in various fields such as multilingual automatic speech recognizer, multilingual dialogue system, and voice service systems [1]. The performance of the LID system is deteriorated due to the short duration of the speech, and significant noise originated from various sources like channels and background [2]. To address the problems of the short duration of speech and background noise, an efficient representation of language information and effective methods for language classification are needed.

The early-stage research on LID systems uses spectral, and prosody features in building statistical models like Gaussian mixture models, hidden Markov models, and support vector machines (SVM) [3, 4, 5, 6, 7]. From literature, shifted delta cepstral (SDC) feature is the most popular acoustic representation for LID systems [8, 9]. These features are derived by the augmentation of conventional Mel-frequency cepstral coefficients (MFCCs) to make use of long-term contextual information in the speech. The problem with the acoustic features for LID task is that the features extracted in a short time cannot characterize a relatively long duration sounds [10, 11]. This problem is addressed using the features which are embedded with contextual information learned from the network in a nonlinear discriminant fashion [12]. In this context, deep neural networks (DNN) which are trained to predict senones/phonemes are used as feature extractors to compute phone log-likelihood ratios, stacked bottleneck, and multilingual tandem bottleneck features for LID [13, 12, 14, 15, 16]. In [17], senone based long short-term memory-recurrent neural network (LSTM-RNN) framework is investigated. Bottleneck features followed by i-vector modeling have good performance [18]. Time delay neural network (TDNN) based acoustic model is trained to convert acoustic sequence to the corresponding sequence of phones or senones; LSTM networks are employed to model the sequential relations from the senone sequences to discriminate languages [19]. Dialect recognition using two-stage training has been done in [20].

Notably, in an Indian scenario, where almost every state has a language of its own and every language having hundreds of dialects, development of a LID system becomes crucial. In this paper, LID systems are studied on IIITH-Indian language speech corpus (ILSC) [21]. The standard acoustic features such as MFCC, SDC coefficients, and prosody (pitch, duration, and intonation) have been explored in the Indian context [5, 6, 7]. These features cannot adequately discriminate the languages since most of the Indian languages have an overlapped set of phonemes (sound units). This makes developing a LID framework for Indian languages challenging. However, due to phonotactic constraints, the characteristics of a particular sound unit may differ in different languages which motivates us to explore phonetic features. Our work mainly focuses on modeling phonetic information using a neural network to improve the performance of an Indian

LID system. In this context, we investigate the joint acoustic model (JAM) of a multi-lingual automatic speech recognizer (ASR) to extract phonetic features for LID. The motivation for this work is that JAM built using LSTM-connectionist temporal classification (LSTM-CTC) network [22] implicitly learns the language discriminant information. It is hypothesized that the phonetic features extracted from the JAM effectively represent the language information.

Frame level log-likelihood scores obtained from JAM are considered as phonetic features and are trained using multi-head self-attention networks which are recently explored for speaker recognition [23, 24]. Further, the concatenation of acoustic and phonetic features are analyzed to enhance the performance. We also investigated the fusion of scores obtained from the acoustic and phonetic feature-based LID systems. To the best of our knowledge, phonetic feature extraction using LSTM-CTC network and classification using multi-head self-attention networks have not been explored in the context of LID.

The remaining paper organized as follows: Section 2 describes the fundamental architectural aspects of LSTM-CTC and multi-head self-attention network. The experimental setup (including database, JAM, baseline, and proposed LID systems) is explained in Section 3. In Section 4 results and discussions are reported. The conclusion is presented in Section 5.

2. SYSTEM DESCRIPTION

2.1. LSTM-CTC

The sequence to sequence model is comprised of a stacked bidirectional LSTM (Bi-LSTM) network, which is aimed to predict a phone label sequence from a sequence of acoustic frames. The mapping is learned by maximizing the log-likelihood probability of a label sequence given an acoustic sequence ($P(Z/X)$). It is computed using intermediate CTC paths since the network is not provided with predefined alignments. CTC path contains a blank symbol corresponding to no emission and allows repetition of symbols, unlike the label sequence.

Let $X = [x_1, x_2, x_3, ..., x_L]$ be an acoustic vector, $Z = [z_1, z_2, z_3, ..., z_U]$ be the corresponding label sequence and $K = [k_1, k_2, k_3, ..., k_L]$ be an intermediate CTC path. Here L is length of input acoustic sequence and U is number of unique phonemes. $P(K/X)$ is computed assuming conditional independence as follows:

$$P(K/X) = \prod_{t=1}^{L} y_t^{k_t} \tag{1}$$

where $y_t^{k_t}$ is soft-max probability of label k at time t. Mapping is employed between K and Z by merging repeated symbols and by removing the blank symbol. This leads to a many to one mapping (different CTC paths mapped to same label sequence). The $P(Z/X)$ is calculated as a sum of $P(K/X)$ over all possible intermediate CTC paths $\Omega(Z)$ as given below:

$$\begin{aligned} P(Z/X) &= \sum_{K \in \Omega(Z)} P(K/X) \\ &= \sum_{k \in \Omega(Z)} \prod_{t=1}^{L} y_t^{k_t}. \end{aligned} \tag{2}$$

Likelihoods are computed using forward and backward algorithm and network parameters are updated using back-propagation.

2.2. Multi-head self-attention network

The architecture of self-attention network can be explained in three different blocks, i.e., (i) a frame-level feature extractor, (ii) self-attention layer, (iii) and an utterance-level feature extractor. A frame-level feature extractor is a feed-forward neural network which takes acoustic sequences as input and produces hidden activations for each frame. Further, these hidden activations are used by the self-attention layer to produce a scalar value for each frame. An utterance-level representation is obtained by concatenating the mean and standard deviation of attentively weighted hidden representations. This utterance-level representation is further passed through a feed-forward network whose output dimension is the number of language classes. The entire network is trained with a single objective function to maximize language identification accuracy. The block diagram of a self-attention network is described in Figure 1.

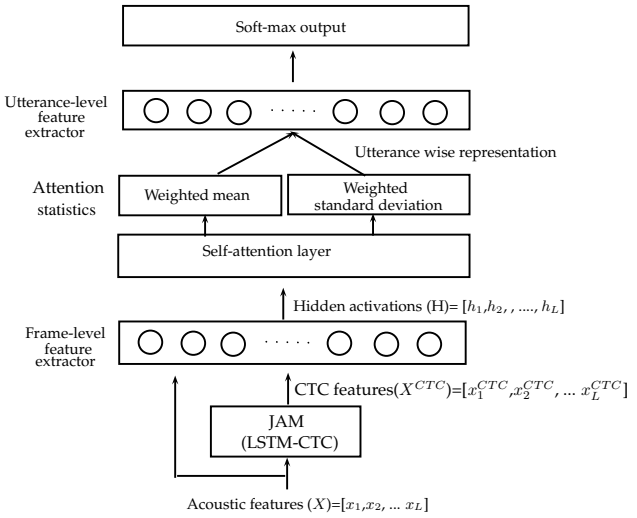

Fig. 1: Block diagram of a self-attention network. The input to the network is either acoustic or phonetic features (frame log-likelihood ratios).

Let $H = [h_1, h_2, h_3, ..., h_L]$ be the hidden activations corresponding to an acoustic vector X after the frame-level feature extractor block and are computed as follows:

$$H = f(W_{ih}X + b_{ih}) \tag{3}$$

where f is the nonlinear activation function. W_{ih} and b_{ih} are hidden layer parameters. A self-attention layer is a feed-forward layer which produces a scalar value e_t for every frame, and it is computed as follows:

$$e_t = tanh(W_a H^T). \tag{4}$$

The dimensions of e_t and W_a are $1 \times L$ and $1 \times d_h$ respectively. Here, d_h is the dimensionality of the hidden representations and W_a are attention layer parameters. The values of e_t are normalized using a soft-max activation and are given by

$$\alpha_t = \frac{\exp(e_t)}{\sum_{t=1}^{L} \exp(e_t)} \tag{5}$$

here α_t are the scalar values that specify the relative importance of each frame. The mean and standard deviation of attentively weighted hidden representations are computed as follows:

$$\mu = \sum_{t=1}^{L} \alpha_t h_t \tag{6}$$

$$\sigma = \sqrt{\sum_{t=1}^{L} \alpha_t h_t \odot h_t - \mu \odot \mu} \tag{7}$$

where $\odot$ represents the Hadamard product. The weighted mean and standard deviation vectors are concatenated and given as input to utterance-level feature extractor to predict language ID.

Multi-head self-attention network has more than one attention layer to compute various utterance-level representations for an acoustic sequence. These representations are concatenated and given as an input to the utterance-level feature extractors. To reduce the computation power in multi-head, concatenated representations are passed through the feed-forward layer before utterance-level feature extractor that brings down the dimension back to as it is just single-head. In the case of multi-head, the objective function (cross-entropy) is penalized by the factor (ϵ) [24] as defined below to ensure each head captures unique information.

$$\epsilon = ||W_a W_a^T - I||_F^2 \tag{8}$$

where $||.||_F$ represents the Frobenius norm of the matrix and T represents transpose of a matrix. Implementation details of the architectures has been explained in Section 3.

3. EXPERIMENTAL SETUP

3.1. Database

To evaluate the performance of the proposed method, we have considered IIITH-ILSC database [21]. It consists of 23 languages, Assamese, Bengali, Bodo, Dogri, Gujarati, Hindi, Kannada, Kashmiri, Konkani, Maithili, Malayalam, Manipuri, Marathi, Nepali, Odia, Punjabi, Sanskrit, Santali, Sindhi, Tamil, Telugu, Urdu, and Indian English. Each language consists of a minimum of 25 male and 25 female speakers. Amount of data for each language is 4.5 hours in which 3 hours is used for training, 0.5 hours is used for validation, and 1 hour of data is used for testing. Each utterance in the database is of 5-10 sec duration, sampled at 16kHz and a sample size of 16 bits. It contains data from both noisy and clean environments.

3.2. Baseline LID Systems

In this paper, we considered two types of baseline LID systems based on i-vector modeling and DNN models. Several experiments are carried out to know the configuration for baseline systems, and the results are presented for the best configuration.

3.2.1. i-vector

We followed the same procedure described in [25] to extract i-vectors. In i-vector modeling [26], universal background model which consists of 1024 Gaussian mixtures is trained with 56-dimensional SDC features (extracted from 13-dimensional MFCC using widely used configuration (7-1-3-7) [27]) and the dimension of extracted i-vector is 400. Linear discriminant analysis (LDA) is applied on these i-vectors to promote language-specific information. Back-end modeling techniques such as SVM and probabilistic linear discriminant analysis (PLDA) scoring are used to predict the language ID.

3.2.2. DNN models

In DNN models, TDNN [26], Bi-LSTM [28], and phonetic temporal neural (PTN) model [19] are considered. TDNN consists of three layers with temporal context [-1, 1], [-2, 2], and [-3, 3] respectively. Bi-LSTM network consists of two layers with 320 units, and each layer followed by a projection layer. Both the networks TDNN, Bi-LSTM are trained with raw acoustic features (SDC) while PTN network consists of TDNN acoustic model (trained with Microsoft [29] data) to get intermediate phonetic representations from acoustic sequence and these representations are modeled using LSTM network.

All DNN architectures are implemented in pytorch. Training of all networks used Adam as an optimizer. Learning rate

is halved upon observing an increase in validation cost. The training is halted upon encountering an increase in validation cost over three successive epochs. In TDNN a symmetric 4 frame window and in Bi-LSTM a symmetric 2 frame window is used to splice adjacent frames. In all networks, hidden units are followed by rectified linear unit (ReLU) activation function, and networks are trained with cross-entropy objective function.

3.3. Multilingual Joint Acoustic Model

JAM is built using Microsoft data which is released as a part of "Low Resource Speech Recognition Challenge for Indian languages-Interspeech 2018" [29]. The database consists of three languages, Telugu, Tamil, and Gujarati. Each language contains 40 hours of data for training, 5 hours of data for validation, and 5 hours for testing. Combined data is obtained by converting orthographic text from all languages to IT3 format using common phone-set representation [30, 31]. The common phone-set provides shared space representation across all Indian languages.

JAM is trained with the combined data (40-dimensional MFCC features extracted from the labeled data) by using the LSTM-CTC network. This network contains four hidden layers, in which a projection layer follows each layer, and these hidden layers are densely connected. Each layer in the network contains 320 cells except the output layer. The output of the last hidden layer is mapped to a linear function which in turn results into a 70 (the number of unique phones in all three languages plus a blank symbol) dimension vector. Adam optimizer with a learning rate of 0.001 is used. Early stopping criteria is employed with an increase in the validation cost for successive three epochs.

This model is not provided with any prior information of languages, but it is still able to produce good performance (word error rate in terms of %) for each language. Word error rate for Telugu, Tamil, and Gujarati are 19.90%, 19.33%, 14.71% respectively. To check the ability of the model in dealing with multiple languages, surface analysis at word level is carried out with the output text of JAM. As described in [32], test the word with ground-truth language, if it is not found, at that point check with other languages. If it is not found with any of the three languages, classify it as a mixed word. The confusion matrix for JAM is shown in Figure 2. It is observed that LSTM-CTC network is rarely confused between the three languages, and it demonstrates that network is implicitly learning the language discriminant information. We experimentally proved the hypothesis that features derived from JAM effectively represent the language information.

3.4. Proposed LID system

LID systems using phonetic features are proposed for Indian languages. To extract phonetic features, LSTM-CTC acoustic model is used, and these features are processed using the multi-head self-attention network. The LSTM-CTC and multi-head self-attention networks are described in section 2, and here configuration details are provided.

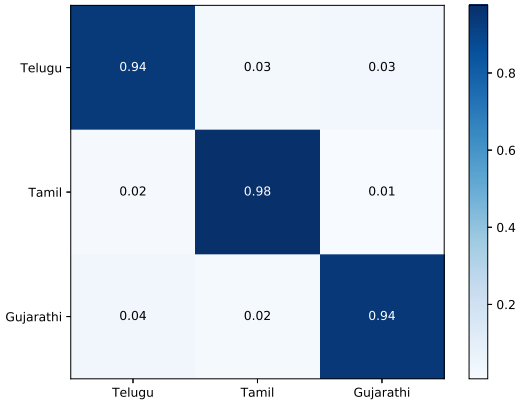

Fig. 2: Confusion matrix for JAM. It is rarely confused between languages.

3.4.1. CTC features

For every acoustic vector x_t (in this work x_t is 40-dimensional MFCC vector), phone posterior probabilities $y_t^{z_t} = P(z_t|x_t;\lambda)$ are computed from JAM which is trained on Microsoft data [29] using LSTM-CTC network. Here, λ denotes the parameters of JAM. Further, these posterior probabilities are converted to the scaled likelihoods by dividing them with phone prior probabilities. The scaled likelihoods ($y_t^{lz_t}$) are considered as CTC features in this work.

$$x_t^{CTC} = [y_t^{lz_1} y_t^{lz_2} y_t^{lz_3} ... y_t^{lz_{U+1}}]_{1\times 70} \tag{9}$$

where, x_t^{CTC} is a CTC feature corresponds to the acoustic vector x_t and having dimension 70.

3.4.2. Multi-head self-attention network

In multi-head self-attention network, frame-level feature extractor comprises of an input layer with six hidden layers and each hidden layer comprises of 1024 units. Attention and utterance level feature extractors are single feed-forward layers with 512, 23 (number of target languages) units respectively. The dimension of utterance wise representation is 512.

4. RESULTS AND DISCUSSION

Our work aims to improve the performance of LID systems in the Indian scenario. In this context, CTC features are proposed and are compared with the standard baseline LID systems. Importance of long-temporal information is investigated using different features and networks. The consistency

in the performance with CTC features is studied through stacking the neighboring features. By using stacked CTC features, multi-head self-attention architecture is explored. Further, the performance of CTC features with different duration utterances investigated. The performance of the LID system is presented in terms of the equal error rate (EER).

4.1. Multilingual CTC features

This section compares the TDNN acoustic model with LSTM-CTC acoustic model in the context of LID. TDNN acoustic model which is trained to predict tristate phones used as phonetic feature extractor and followed the same configuration as described in [19]. From these two networks, phonetic features are extracted and are trained with the self-attention network for LID. The corresponding results are listed in Table 1. The first column in Table 1 represents the language used to train the acoustic model and columns 2, 3 are the performance of the LID system. It is observed that in monolingual case, both the features have almost the same performance, but in the multilingual scenario, CTC features have better performance compared to the TDNN-phonetic features. The performance of multilingual TDNN-phonetic features is worse than monolingual. This may be due to senone based TDNN is confusing between the phone states of different languages. Results from Table 1 demonstrates the better contextual modeling capability of CTC features derived from LSTM-CTC based JAM (JAM refers to an acoustic model trained with three languages).

Table 1: Performance (in terms of EER (%)) of TDNN-phonetic and CTC features.

Language	TDNN-phonetic	CTC
Telugu	13.54	13.40
Tamil	**13.23**	13.99
Gujarati	13.67	13.25
Telugu-Tamil	14.24	12.83
Telugu-Gujarati	14.52	12.71
Tamil-Gujarati	14.89	12.65
Telugu-Tamil-Gujarati	14.87	**11.09**

Further, long temporal information in the CTC features is studied through comparing the performance with MFCC and SDC using different networks and results are tabulated in Table 2. It is noted that EER of CTC features is almost the same across all models. But with MFCC and SDC features, EER is decreasing from TDNN to self-attention. At the model level, the self-attention network has better performance compared to all models. Since it involves the utterance-level decision to predict language ID, while in other networks frame-level decisions are averaged for language prediction. From these observations, it is inferred that long-temporal information is playing a vital role in LID either at the feature (CTC) level or model (self-attention) level. In the next sub-section, experiments are carried out by stacking adjacent features to enhance the performance of the LID system.

Table 2: Performance (in terms of EER (%)) of LID systems using different features and networks.

Model	Features		
	MFCC	SDC	CTC
TDNN	21.42	17.79	**13.03**
Bi-LSTM	20.35	16.45	**12.12**
Self-attention	15.23	14.93	**11.09**

4.2. LID systems using Multi-head self-attention network

Results of the LID system with increasing temporal context through splicing frames are shown in Table 3. It can be noticed that stacking the successive features has improved the performance of LID systems. Stacking the features with a temporal context of $\pm$ 2 has produced the best EER and further increasing the temporal context does not produce a significant gain. Utterance wise representation containing both mean and standard deviation of hidden representations leading to significant improvement in overall temporal context. We speculate that this might be due to, standard deviation captures temporal variability over long context. Further multi-head attention network experiments are carried out using the temporal context of $\pm$ 2.

Table 3: Results (EER in %) of the self-attention network using stacked SDC and stacked CTC features. Here, system1, system2 are self-attention networks with utterance wise representation containing only mean and the combination of mean and standard deviation respectively.

Temporal context	CTC		SDC	
	system1	system2	system1	system2
0-1-0	11.09	9.76	14.93	13.32
1-1-1	10.25	9.47	12.79	11.49
2-1-2	**9.40**	**9.17**	**11.36**	**11.15**
3-1-3	9.84	9.23	12.82	11.84

This work explored the use of multi-head attention mechanism in a self-attention network, and the results are tabulated in Table 4. It can be seen that using multi-head attention has enhanced the performance of LID systems. The improvement in the performance can be attributed to its better utterance wise representation (each head captures distinct information) compared to the single-head attention. SDC captures long-term temporal information by spanning over multiple frames, whereas CTC features learn from multilingual JAM. Combination of these two features at the model level (fuse the scores obtained from the individual models) and at

feature level (train a model by concatenating SDC and CTC features) has a significant improvement compared to the individual models. These results are presented in columns 4 and 5 of Table 4.

Table 4: Results (EER in %) of LID systems trained using multi-head self-attention networks.

Number of heads	SDC	CTC	Fusion at model level	Fusion at feature level
1-head	11.15	9.17	7.49	8.99
2-head	9.90	8.83	7.20	7.81
3-head	**9.61**	**8.70**	**6.82**	**7.61**

To check the performance of CTC features with short duration speech, we created 5 test sets by dividing the test data into small utterances of different durations from 1 sec to 5 sec. The results on 5 test sets are shown in Figure 3. It depicts that, consistency in the performance with CTC features is observed at 3 sec, with i-vector (used as a feature to train multi-head self-attention network) at 4 sec, and with MFCC, SDC around 5 sec. It is also noted that for short duration utterances ($\leq$2 sec) the performance is poor even with CTC features. This might be due to the limitation of the multi-head self-attention network, i.e., the effectiveness of utterance wise representation depends on the length of a speech utterance.

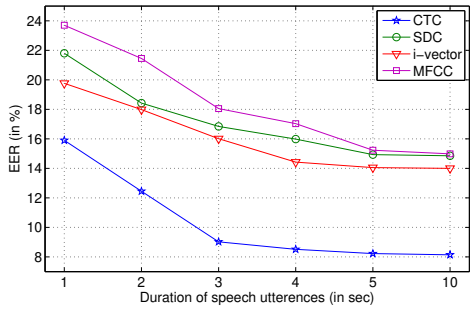

Fig. 3: Comparison of the effect of utterance duration with different features using a multi-head self-attention network.

4.3. Benchmark comparison of proposed LID system

The performance of the proposed LID system has been compared with state-of-the-art LID systems, are listed in Table 5. It is noted that the proposed system outperformed the other systems. Phonetic feature based LID systems (PTN, proposed) are performing better compared to the acoustic feature based LID systems (i-vector, TDNN, and Bi-LSTM). Acoustic features cannot characterize the relatively long duration sounds present in the speech signal while phonetic features capture the variations in sound units (contextual information). Indian LID systems take advantage with phonetic features since most of the languages have overlapped set of phonemes (basic sound units) but the characteristics of a particular sound unit different in different languages. CTC features followed by i-vector modeling results an EER of 14.49%. Comparing this (CTC- i-vector) result with the proposed system demonstrating the importance of attention mechanism in capturing the temporal information while the i-vector model ignores the temporal information by computing the statistical mean.

Table 5: Benchmark comparison (in terms of EER (%)).

Modeling method	LID system	EER
i-vector modeling	i-vector+SVM	17.45
	i-vector+LDA+SVM	17.08
	i-vector+PLDA	17.23
DNN modeling	TDNN	17.79
	Bi-LSTM	16.45
	PTN	13.35
	CTC-Multi head attention	**8.70**

5. CONCLUSION

This paper proposed connectionist temporal classification (CTC) features and multi-head self-attention network for Indian language identification. These features outperformed the other state-of-the-art LID systems.
The following are the observations from this paper.

- The joint acoustic model (JAM) builds using LSTM-CTC network implicitly learns language-specific information. So the features extracted from JAM effectively represent the language information.
- Multi-head self-attention network which captures the temporal information through attention mechanism has shown better performance than a time-delay neural network, and long short-term memory (LSTM) network. This network is also computationally less expensive than LSTM since it is a feed-forward network.
- CTC features have shown consistency in the performance even with the utterances of short duration.
- The fusion of shifted delta cepstra and CTC features using the multi-head self-attention network has also investigated, and the corresponding equal error rate is found to be 6.82%.

Performance of CTC features has to be studied in the presence of noise and other mismatched conditions.

6. REFERENCES

[1] E. Ambikairajah, H. Li, L. Wang, B. Yin, and V. Sethu, "Language identification: a tutorial," *IEEE Circuits and Systems Magazine*, vol. 11, no. 2, pp. 82–108, 2011.

[2] Y. Song, B. Jiang, Y. Bao, S. Wei, and L.-R. Dai, "i-vector representation based on bottleneck features for language identification," *Electronics Letters*, vol. 49, no. 24, pp. 1569–1570, 2013.

[3] P. A. Torres-Carrasquillo, D. A. Reynolds, and J. R. Deller, "Language identification using Gaussian mixture model tokenization," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2002, pp. 757–760.

[4] E. Wong and S. Sridharan, "Methods to improve Gaussian mixture model based language identification system," in *Proc. International Conference on Spoken Language Processing*, 2002, pp. 93–96.

[5] M. Kamsali Veera, R. K. Vuddagiri, S. V. Gangashetty, and A. K. Vuppala, "Combining evidences from excitation source and vocal tract system features for Indian language identification using deep neural networks," *International Journal of Speech Technology*, vol. 21, no. 3, pp. 501–508, 2018.

[6] D. Nandi, D. Pati, and K. S. Rao, "Sub-segmental, segmental and supra-segmental analysis of linear prediction residual signal for language identification," in *Proc. IEEE International Conference on Signal Processing and Communications*, 2014, pp. 1–6.

[7] V. R. Reddy, S. Maity, and K. S. Rao, "Identification of Indian languages using multi-level spectral and prosodic features," *International Journal of Speech Technology*, vol. 16, no. 4, pp. 489–511, 2013.

[8] M. A. Kohler and M. Kennedy, "Language identification using shifted delta cepstra," in *Proc. IEEE Midwest Symposium on Circuits and Systems*, 2002, pp. 69–72.

[9] F. Allen, E. Ambikairajah, and J. Epps, "Language identification using warping and the shifted delta cepstrum," in *Proc. IEEE Workshop on Multimedia Signal Processing*, 2005, pp. 1–4.

[10] B. Jiang, Y. Song, S. Wei, I. V. McLoughlin, and L.-R. Dai, "Task-aware deep bottleneck features for spoken language identification," in *Proc. INTERSPEECH*, 2014, pp. 3012–3016.

[11] T. Fu, Y. Qian, Y. Liu, and K. Yu, "Tandem deep features for text-dependent speaker verification," in *Proc. INTERSPEECH*, 2014, pp. 1327–1331.

[12] W. Geng, J. Li, S. Zhang, X. Cai, and B. Xu, "Multilingual tandem bottleneck feature for language identification," in *Proc. INTERSPEECH*, 2015, pp. 413–417.

[13] M. Diez, A. Varona, M. Penagarikano, L. J. Rodriguez-Fuentes, and G. Bordel, "On the use of phone log-likelihood ratios as features in spoken language recognition," in *Proc. IEEE Spoken Language Technology Workshop*, 2012, pp. 274–279.

[14] P. Matejka, L. Zhang, T. Ng, H. S. Mallidi, O. Glembek, J. Ma, and B. Zhang, "Neural network bottleneck features for language identification," in *Proc. Odyssey, 2014*, pp. 299–304.

[15] R. Fér, P. Matějka, F. Grézl, O. Plchot, and J. Černockỳ, "Multilingual bottleneck features for language recognition," in *Proc. INTERSPEECH*, 2015, pp. 389–393.

[16] M. A. Zissman, "Language identification using phoneme recognition and phonotactic language modeling," in *International Conference on Acoustics, Speech, and Signal Processing*, 1995, pp. 3503–3506.

[17] Y. Tian, L. He, Y. Liu, and J. Liu, "Investigation of senone-based long-short term memory RNNs for spoken language recognition," in *Proc. Odyssey*, 2016, pp. 89–93.

[18] L. Ferrer, Y. Lei, M. McLaren, and N. Scheffer, "Study of senone-based deep neural network approaches for spoken language recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 1, pp. 105–116, 2016.

[19] Z. Tang, D. Wang, Y. Chen, L. Li, and A. Abel, "Phonetic temporal neural model for language identification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 1, pp. 134–144, 2018.

[20] Z. Ren, G. Yang, and S. Xu, "Two-Stage Training for Chinese Dialect Recognition," in *Proc. INTERSPEECH*, 2019, pp. 4050–4054.

[21] R. K. Vuddagiri, K. Gurugubelli, P. Jain, H. K. Vydana, and A. K. Vuppala, "IIITH-ILSC speech database for Indian language identification," in *Proc. Workshop on Spoken Language Technologies for Under-Resourced Languages*, 2018, pp. 56–60.

[22] H. K. Vydana, K. Gurugubelli, V. Raju, and A. K. Vuppala, "An exploration towards joint acoustic modeling for Indian languages: IIIT-H submission for low resource speech recognition challenge for Indian languages," in *Proc. INTERSPEECH*, 2018, pp. 3192–3196.

[23] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Proc. INTERSPEECH*, 2018, pp. 2252–2256.

[24] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, "Self-attentive speaker embeddings for text-independent speaker verification," in *Proc. INTERSPEECH*, 2018, pp. 3573–3577.

[25] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Proc. INTERSPEECH*, 2011.

[26] Z. Tang, D. Wang, Y. Chen, and Q. Chen, "AP17-OLR challenge: Data, plan, and baseline," in *Proc. IEEE Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 2017, pp. 749–753.

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

[28] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, and P. J. Moreno, "Automatic language identification using long short-term memory recurrent neural networks," in *Proc. INTERSPEECH*, 2014.

[29] B. M. L. Srivastava, S. Sitaram, R. Kumar Mehta, K. Doss Mohan, P. Matani, S. Satpal, K. Bali, R. Srikanth, and N. Nayak, "Interspeech 2018 Low Resource Automatic Speech Recognition Challenge for Indian Languages," in *Proc. Workshop on Spoken Language Technologies for Under-Resourced Languages*, 2018, pp. 11–14.

[30] G. Madhavi, B. Mini, N. Balakrishnan, and R. Raj, "OM: one tool for many (Indian) languages," *Journal of Zhejiang University-SCIENCE A*, vol. 6, no. 11, pp. 1348–1353, 2005.

[31] A. Baby, N. N.L., A. L. Thomas, and H. A. Murthy, "A unified parser for developing Indian language text to speech synthesizers," in *Proc. International Conference on Text, Speech, and Dialogue*, 2016, pp. 514–521.

[32] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. J. Moreno, E. Weinstein, and K. Rao, "Multilingual speech recognition with a single end-to-end model," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 4904–4908.