SPOKEN LANGUAGE IDENTIFICATION USING BIDIRECTIONAL LSTM BASED LID SEQUENTIAL SENONES.

Muralikrishna H, Pulkit Sapra, Anuksha Jain, Dileep Aroor Dinesh

MANAS Lab, Indian Institute of Technology Mandi

ABSTRACT

The effectiveness of features used to represent speech utterances influences the performance of spoken language identification (LID) systems. Recent LID systems use bottleneck features (BNFs) obtained from deep neural networks (DNNs) to represent the utterances. These BNFs do not encode language-specific features. The recent advances in DNNs have led to the usage of effective language-sensitive features such as LID-senones, obtained using convolutional neural network (CNN) based architecture. In this work, we propose a novel approach to obtain LID-senones. The proposed approach combines BNF with bidirectional long short-term memory (BLSTM) networks to generate LID-senones. Since each LID-senones preserve sequence information, we term it as LID-sequential-senones (LID-seq-senones). The proposed LID-seq-senones are then used for LID in two ways. In the first approach, we propose to build an end-to-end structure with BLSTM as front end LID-seq-senones extractor followed by a fully connected classification layer. In the second approach, we consider each utterance as a sequence of LID-seq-senones and propose to use support vector machine (SVM) with sequence kernel (GMM-based segment level pyramid match kernel) to classify the utterance. The effectiveness of proposed representation is evaluated on Oregon graduate institute multi-language telephone speech corpus (OGI-TS) and IIT Madras Indian language corpus (IITM-IL).

Index Terms— Language Identification, LID-sequential-senones, sequence kernel, bidirectional LSTM

1. INTRODUCTION

The success of any language identification (LID) system heavily depends on the ability of the used features to capture the language-specific cues in the speech utterance. There have been several approaches for capturing this language-specific contents starting from low-level acoustic features to higher level syntactic features [1, 2, 3, 4].

State-of-the-art systems for LID use bottleneck features (BNFs) [5, 6, 7, 8, 9]. BNFs are the activations obtained from a bottleneck layer of a DNN. These BNFs encode the phonetic contents in the input speech in a compact form. Since the networks used to extract the BNFs are primarily trained for phone state classification, they do not specifically encode language-discriminative information, particularly for highly confusable languages that may have similar phoneme-level information. Also, this method leads to a variable length representation of speech, posing a challenge in building the back-end classifier.

Typically, these varying length sequences are handled by extracting a fixed length utterance-level representation before feeding them to a classifier. In [6, 7, 8], i-vector analysis is performed on the extracted BNFs to obtain a compact representation of the utterance. The i-vector jointly encodes information about the speaker, language, channel etc. and gives a fixed length representation [10]. However, i-vectors are extracted in an unsupervised fashion without using language labels. Hence the techniques such as linear discriminant analysis (LDA) and within-class covariance normalization (WCCN) are necessary before feeding them to a back-end classifier.

This motived researchers to use DNN based end-to-end systems that are trained directly with language labels. These type of networks have a front-end feature extractor stage to produce frame level features, followed by a back-end classifier. Because of the end-to-end nature of the network, the front-end feature extractor learns to produce language-discriminative features. Since the output of feature extractor will be varying length in nature, a summarization (or pooling) layer is used to extract a fixed length utterance level representation before feeding them to a back-end classifier. In [11], a sequence summarization layer is used which calculates the mean of all the frames to provide an utterance-level representation. In [12], a fixed dimensional representation called x-vector is used. The DNN used for extracting x-vector has a statistics pooling layer to convert variable length input sequence to a fixed length vector. This statistics layer calculates the mean and variance of all frame level features in that speech sample. A similar approach is reported in [13] to extract a fixed length representation called embeddings. In [14], a novel concept of LID-specific features called LID-senones is introduced. In [14], the BNFs extracted at the front end are further processed using a CNN stage to make them more language-discriminative in nature. These LID-senones are actually the activations obtained at the output of last convolutional layer. Then, a spatial pyramid pooling layer is used to

map these LID-senones to an utterance level representation, followed by a fully connected classification layer.

In our work, we borrow the idea of [14] and propose a novel approach for obtaining LID-senones to improve its effectiveness. In this approach, we propose to use the BNFs obtained from a multilingually trained network [15] as a front end representation and propose to use bidirectional long short-term memory (BLSTM) network to obtain LID-senones. Since these LID-senones are obtained using BLSTM, they preserve some sequence information. We propose to call these language-specific features as LID-sequential-senones (LID-seq-senones). We propose to build LID system using the LID-seq-senones in two ways. In the first approach, we propose to build an end-to-end structure with BLSTM as front end to extract LID-seq-senones followed by a fully connected network (FCN) classifier. In the second approach, each utterance is represented as a varying length sequence of LID-seq-senones, as the duration of each utterance is different. We then propose to use sequence kernel based support vector machines (SVMs) to classify these utterances into a language class. In this work, we consider the Gaussian mixture model based segment level pyramid match kernel (GSPMK) [16] as a sequence kernel that considers two varying length sequences as input to compute a kernel value between them. The ability of the proposed methods to differentiate closely related languages is assessed on Oregon graduate institute multi-language telephone speech corpus (OGI-TS) [17] and IIT Madras Indian language corpus (IITM-IL) [18].

The contribution of this paper are: (1) BLSTM-based LID network, a novel end-to-end structure to model LID-seq-senones, (2) GSPMK-based SVM framework for LID using sequence of LID-seq-senones, and (3) Extensive experimentation of proposed approaches for LID using LID-seq-senones and comparison with state-of-the-art approaches using LID-senones.

The remainder of this paper is as follows: In section 2, we describe our approaches for LID. In section 3, a description of the databases used is given. In section 4, details of various experiments and corresponding results are given followed by discussions and conclusions in section 5.

2. PROPOSED FRAMEWORK FOR LID USING LID-SEQ-SENONES

The framework for LID using LID-seq-senones is shown in Figure 1. The overall system consists of a BNF extractor at the front-end followed by a BLSTM-based network to extract LID-seq-senones. Then these LID-seq-senones are applied to either a FCN classifier or GSPMK-based SVM classifier.

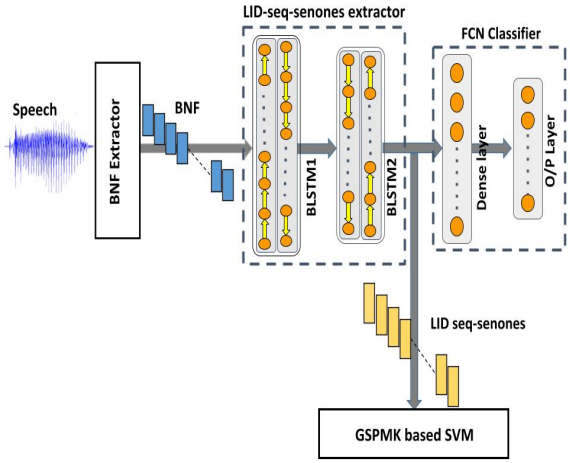

Fig. 1. Block diagram of the proposed framework for LID using LID-seq-senones.

2.1. BNF Extractor

For the front-end, we use a network pre-trained on 17 languages (provided by BUT/Phonexia bottleneck feature extractor [15]) to extract 80-dimensional BNFs. This network was originally trained with 3096 phone states as targets and consists of a cascade of two bottleneck networks. Each of the extracted BNF covers a total context of 31 frames (325 ms) of input speech. We have used an energy-based voice activity detector (VAD) to remove non-speech frames. Since a wide variety of languages (including Indian languages) are used in the training, we assume that these extracted features cover phonetics of most of the languages used in this work. More details about this package can be found at [15]. Since the BNF extractor is trained only to identify the phone states, these features do not contain language-discriminative information. Hence we use them to generate proposed LID-specific features using BLSTM-based LID-seq-senones extractor network.

2.2. LID using BLSTM-based end-to-end network

In our first approach, we propose a task-aware network that provides a compact representation of the speech utterance in terms of LID-seq-senones using the simple BNFs as input. Our proposed network contains two BLSTM layers followed by a FCN classifier which includes one dense hidden and an output layer. We train this network for identifying the languages using input BNFs. After training, the activations obtained from the second BLSTM layer are used as a new representation of the input utterance. We call these activations as LID-seq-senones. Being originally trained for LID, these BLSTM layers learn to discriminate between the languages

using the temporal variations in the input BNFs. Hence these new features are more LID-specific. A qualitative analysis of these LID-seq-senones will be done in Section 4.4.

Each of these LID-seq-senones is extracted by using an input sequence containing a fixed (N) number BNF vectors. Since each input BNF is already covering a context of 31 frames [15], each of these LID-seq-senones cover a total context of 31+N-1 frames of input speech. While it is not possible to know the exact number of frames over which the BLSTM units combine the information to produce output, we believe that the context (31+N-1) covered by the input sequence to the BLSTM network allows it to capture locally available language-specific cues in the input speech. Hence these LID-seq-senones carry sufficient language specific information to be directly mapped to a language label. Since each LID-seq-senone requires only a fixed number of input frames, multiple LID-seq-senones can be extracted from a single utterance.

In this approach, we consider these LID-seq-senones as independent units and the entire utterance is represented as a sequence of these feature vectors. During testing, the final decision about the class is taken by averaging the probabilities obtained for these individual LID-seq-senones. But, this approach does not preserve the sequence of LID-seq-senones. Hence we propose a different approach that preserves the temporal ordering of LID-seq-senones.

2.3. LID using SVM Classifier with GSPMK

In our second approach, we represent the utterance as a sequence of LID-seq-senones. Since we do not put any restriction on the length of the input speech sample, this representation results in varying length sequences. Approaches like i-vector analysis for obtaining a fixed length representation for the utterance fails to preserve the sequential information that might be present in these features. Hence we propose to use a GSPMK-based SVM classifier [16], which preserves the sequence information in a better way. This approach enables us to i) utilize the local information while comparing two speech utterances of unequal length and ii) preserve the temporal information in features to some extent. To compute GSPMK, each speech utterance represented as a sequence of LID-seq-senones is repeatedly divided to form a pyramid of increasingly finer segments. Then GSPMK between a pair of varying length sequence of LID-seq-senones is computed by matching the corresponding segments at every level of the pyramid. Here, each segment is represented as a bag-of-codewords, where the codewords are obtained by soft-clustering all the LID-seq-senones of all the utterances using class independent Gaussian mixture model (CIGMM). The GSPMK between a pair of sequence of LID-seq-senones is computed as a weighted sum of the number of new matches found at different levels of the pyramid of segments. More descriptions about the GSPMK is given in [16].

3. DATABASES USED IN THE STUDIES

Table 1. Details of languages used from OGI-TS

Language	Train		Test	
	Hours	Utterances	Hours	Utterances
English	1.26	465	0.53	195
Farsi	1.19	459	0.49	190
French	1.32	482	0.50	188
Germen	1.33	478	0.54	188
Japanese	1.26	474	0.51	195
Korean	1.12	446	0.408	166
Mandarin	1.13	462	0.47	182
Spanish	1.35	478	0.49	188
Tamil	1.17	458	0.43	177
Vietnamese	1.12	449	0.43	179

Table 2. Details of Indian languages used from IITM-IL

Language	Train		Test	
	Hours	Utterances	Hours	Utterances
Assamese	6.01	3260	21.1	14394
Bengali	6.20	3897	8.9	5543
Gujarati	6.12	1456	15.25	4228
Hindi	5.83	2509	4.01	2127
Kannada	5.93	2063	1.35	515
Malayalam	5.91	4058	10.52	7242
Manipuri	6.03	5702	13.72	12215
Odia	6.02	4674	2.74	2477
Rajastani	6.19	2505	13.90	4795
Tamil	6.12	1825	13.91	5137
Telugu	6.02	1243	19.95	5281

We have considered two data sets in our experiments. The first data set is the Oregon graduate institute multi-language telephone speech corpus (OGI-TS) [17] containing 10 languages. The number of utterances and total duration in hours for each of the languages used in this study is given in Table 1. Note that each of the language classes has only around 450 samples for training. This poses a challenge in training an end-to-end network with utterance level target labels.

The second data set contains closely related 13 Indian languages provided by the Indian Institute of Technology Madras [18]. In this work, we have considered only 11 languages by excluding Bodo and Marathi as these two languages have voice recordings from single gender only. This corpus was created by recording speech samples read by speakers in a controlled environment with a sampling rate of 48 KHz. In order to have a balanced training data set, we have used approximately 6 hours of speech from each language for training. The details about the number of utterances in each language class along with the total duration in hours for both

train and test sets are given in Table 2. For our experiment, each of these files has been downsampled to 8 KHz. Some of these languages have very similar words. Also, majority of the phonemes are common among these languages [19]. For example, south Indian languages like Kannada, Malayalam, Tamil and Telugu belongs to the Dravidian language family and have many similar phonemes. Similarly, Assamese, Bengali, Gujarati, Hindi and Rajasthani belongs to the Indo-Aryan language family. Since these languages are closely related, the correct identification of a language is very challenging.

4. EXPERIMENTS AND RESULTS

In this section, we study the effectiveness of the proposed LID-seq-senons for LID. All LID systems considered in this paper are evaluated using two different metrics. The first metric is the percentage of Error Rate (ER), which is defined as:

$$ER = \frac{N_{Miss}}{T} \star 100 \quad (1)$$

Where N_{Miss} is the total number of utterances misclassified and T is the total number of utterances used for testing. The second metric is C_{avg}, which is used in NIST Language Recognition Evaluation 2015 [20].

4.1. Baseline Systems

The first baseline system used for the comparison of our proposed method is obtained using a FCN with 6 layers, including input and output layers. The 80 dimensional BNFs along with the context of 21 frames (10 frames on either side) are used as input to this network. The configuration used in our FCN is: 21x80-2056-1024-512-256-N_l. Here N_l is the number of nodes in the output layer representing the number of language classes (11 for IITM-IL and 10 for OGI-TS). This configuration is the best one obtained empirically. We have used the sigmoidal activation function for all hidden layers and softmax activation for the final layer. All the networks in this paper are trained with cross-entropy loss and stochastic gradient descent (SGD) optimizer. We have used keras tool [21] for implementation. The final decision for an utterance is obtained by averaging of probabilities obtained for each context vector in an utterance and selecting the class with maximum probability. The LID performance of the FCN-21 (21 indicating the number of context frames) for OGI-TS and IITM-IL is given in the first row of Table 3.

We also implemented the LID-Net proposed in [14], using a slightly different architecture than the one used in the original work since it did not perform well on our datasets. We used the same 80 dimensional BNFs as in other experiments. We have used 512 filters of size 20x11 and 256 filters of size 1x5 respectively in the first and second convolution layers. The output of the second layer is then pooled using spatial pyramid pooling followed by a dense layer with 32 nodes. Final dense classification layer has N_l nodes. We have split all training samples in OGI-TS into non-overlapping segments of 40 BNF vectors to increase the number of training samples. This resulted in 108075 training segments for OGI-TS. Samples from IITM-IL were maintained without any change. Testing was done without splitting the sequences. The LID performance of the LID-net for OGI-TS and IITM-IL is given in the second row of Table 3. It can be seen that LID-net performs significantly better than FCN-21.

We have also implemented the LID system using embeddings as given in [13]. The results obtained in [13] indicates that the performance of embeddings based approach is close to that of the state-of-the-art i-vector based approach. It consists of two BLSTM layers followed by a dense layer, each of them with 256 nodes. Then it has a statistics pooling layer to compute average and variance of all frames in the sequence, followed by two dense layers with 256 and 150 nodes respectively. As suggested in the original work [13], we have used concatenated embeddings and used PCA for dimensionality reduction before applying them to a Gaussian linear classifier. Training sequences were split into sequences of 40 frames (instead of 300 frames as in [13]). During testing, we pass the entire sequence without splitting. The results obtained are presented in the third row of Table 3. This system performs much better than all other baseline systems.

Table 3. Comparision of the performance of baseline fully connected network (FCN-21), state-of-the-art LID-net & DNN-embed, proposed BLSTM network with different time steps and GSPMK-based SVM on BNF & proposed LID-seq-senones for LID. The best performance is marked in bold.

approach	OGI-TS		IITM-IL	
	ER	C_{avg}	*ER*	C_{avg}
FCN-21	31.31	17.43	10.68	5.87
LID-net	18.52	9.69	3.21	2.02
DNN-embed	**14.48**	**8.16**	**1.36**	**0.80**
BLSTM-21	19.60	10.44	3.95	1.72
BLSTM-31	15.72	8.40	2.90	1.15
BLSTM-35	14.60	8.24	1.37	0.80
BLSTM-41	14.75	8.24	1.37	0.83
BNF+GSPMK-SVM	26.94	15.01	9.89	5.47
LSS+GSPMK-SVM	**14.30**	**7.94**	**1.32**	**0.72**

4.2. BLSTM-based end-to-end network

In this section, we discuss the implementation and results for our proposed first approach. The overall architecture contains two BLSTM layers to extract LID-seq-senones, followed by a classifier stage with a dense hidden layer and a dense output layer with N_l nodes. The optimum number of nodes in the first two BLSTM layers and dense hidden layer are experimentally determined as 512, 64 and 32 respectively. The

LID-seq-senones extracted at the end of the second BLSTM layer are classified using the FCN classifier. In testing, the decision about the class for an utterance is made by taking an average of all individual LID-seq-senone level predictions. The extracted LID-seq-senones are of the length 128.

We have evaluated our network by varying the number of BNF frames (N) while forming the input sequence to the network by keeping other parameters same. In Table 3, rows 4-7 indicates the results obtained for input sequences with length 21, 31, 35 and 41 respectively. We see that the BLSTM network with 21 time steps (BLSTM-21) has outperformed the FCN-21. Certainly, the ability of BLSTM network to combine the sequential information present in the input BNFs has delivered this improvement. The best performance is obtained for 35 time steps. This covers an overall context of 65 frames (655 ms) at the input side. Clearly, this large context covered by the input sequence enables our network to capture language-specific cues in the utterance. This enables our network to distinguish even closely related languages in IITM-IL corpus. It can also be seen that the proposed BLSTM-35 network outperformed the state-of-the-art LID-net and performed almost equally to the embeddings based LID system (DNN-embed).

From the results, it is evident that BLSTM networks efficiently combines the input features over a fixed duration to provide a better representation. However, in all these cases we have taken the final decision by averaging the LID-seq-senone level probabilities. While this operation helps in us handling variable length utterances, it may destroy the information that might be present in the form of sequence of these LID-seq-senones. This motivates us to use the BLSTM network for extracting LID-seq-senones and use them with a classifier which preserves the sequence of these features. While each LID-seq-senone captures the locally available LID-specific contents, preserving their sequence might help the classifier to have a global picture about the utterance.

4.3. LID using GSPMK-based SVM

In this study, each utterance is represented as a sequence of 128-dimensional LID-seq-senones which are classified using GSPMK-based SVM classifier. For training the LID-seq-senones extractor, we used the same architecture and parameters as that of our best performing BLSTM network in the previous section. We used a CIGMM to obtain bag-of-codewords using which GSPMK is computed. The optimum number of components in CIGMM is experimentally determined as 512. LIBSVM [22] tool is used to build the SVM using one-against-the-rest approach. The value of trade-off parameter, *C*, in SVM is chosen empirically as 10^{-3}. In this study, we have restricted the number of levels in the pyramid to 3. If we consider more than 3 levels, each segments will have very less number of frames. This leads to a serious problem while computing the 512-dimentional bag-of-code-word representation, as most of the entries will be close to zero.

The performance of SVM-based classifier using GSPMK computed from sequence of LID-seq-senones is presented in the last row of Table 3. In order to compare the effectiveness of LID-seq-senones, we also developed a GSPMK from the utterances represented as sequence of 80-dimensional BNFs. The performance of SVM-based classifier using GSPMK computed from sequences of BNFs is also presented in the last-but-one row of Table 3. The system with LID seq-senones provides a significant improvement in the performance by effectively extracting a higher level representation compared to simple BNFs. Also, the GSPMK-based SVM has delivered a slight improvement in the performance compared to baseline DNN-embed and BLSTM-35. This indicates the presence of sequential information in these new features.

4.4. Qualitative analysis of LID-seq-senones

Figure 2 shows the bar plot of the first 64 coefficients of 128-dimensional LID-seq-senones for four Indian languages in which Kannada & Tamil are highly confusable and Assamese & Bengali are confusable. We have selected 20 utterances from each of these four languages which are stacked along the y-axis. For convenience, each utterance is represented by an average of all LID-seq-senones obtained for that file. The z-direction indicates the magnitude of features. Note that the magnitude can take negative values also. The plot shows a very good intra-class similarity and inter-class variability in these features. This shows the ability of LID-seq-senones to capture the language specific information from the input BNFs.

4.5. Effect of intersession variabilities

In this section, we study the effect of intersession variations like channel and speakers on the extracted LID-seq-senones. We did this by evaluating the performance of our system on Tamil language which is common to both databases. As mentioned in section 3, the recording conditions and speakers are different in these two databases. Firstly, we use the LID-seq-senones extractor trained on OGI-TS data to extract features of Tamil in IITM-IL corpus. These extracted LID-seq-senones are then classified using SVM which was trained using OGI-TS data. The results are in the first row of Table 4. The second row of Table 4 shows the results obtained by using IITM-IL database for training and Tamil from OGI-TS dataset used for testing.

The system trained on OGI-TS has performed very well on samples from IITM-IL. Since Tamil has very less similarity with other languages in this corpus, correctly identifying the testing sample with clean background was an easy task for the model. Though the second system was trained with more training data, the performance was very inferior to that

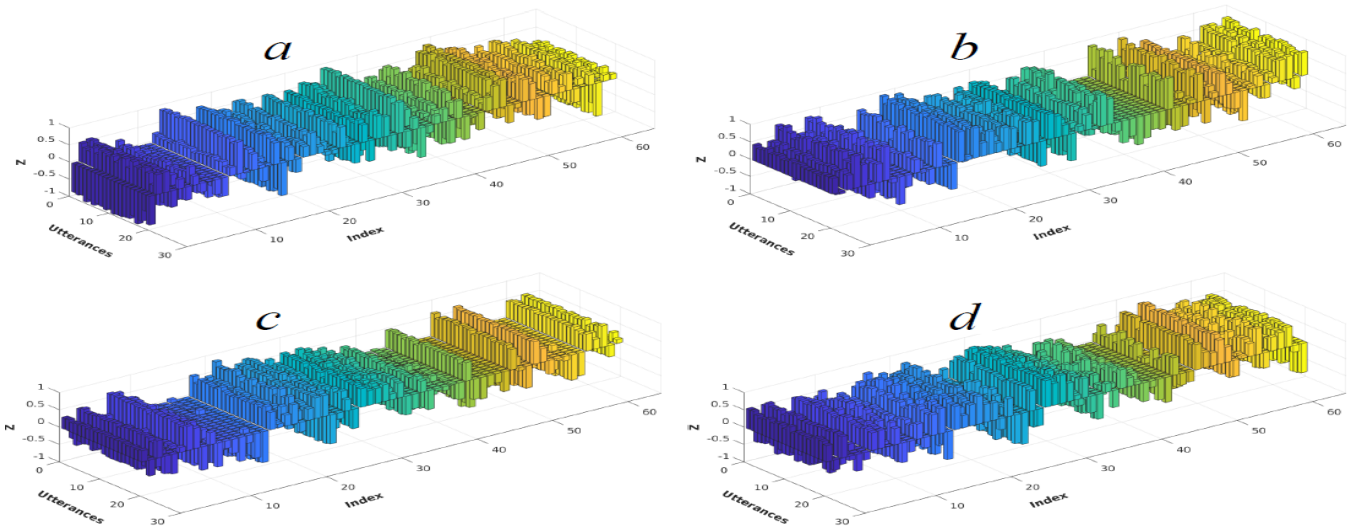

Fig. 2. LID-seq-senones for 20 utterances of (a) Assamese, (b) Bengali, (c) Kannada and (d) Tamil. Each row along y-axis is the average of all LID-seq-senones obtained for that utterance.

Table 4. Error Rates (ER) for Tamil language in cross-database experiments. LSS_OGI_IL indicates the system trained on OGI-TS used to test features of Tamil from IITM-IL. LSS_IL_OGI indicates the system trained on IITM-IL used to test features of Tamil from OGI-TS.

System	*ER*
LSS_OGI_IL	2.75
LSS_IL_OGI	50.60

trained with OGI-TS. The reason is that the system trained on clean recordings of IITM-IL could not overcome the noisy background of test samples from OGI-TS. Also, a high similarity between closely related classes in this model has lead to more confusion.

5. DISCUSSIONS AND CONCLUSIONS

In this paper, we proposed a method to extract LID-specific features called LID-seq-senones. These features are designed to capture the locally available language-specific information by analysing the BNFs over a fixed duration. Results obtained proves that these features carry more language discriminative information compared to simple BNFs. Results obtained using GSPMK-based SVM classifier suggests that preserving the sequence of these features leads to better performance. Cross-database experiments conducted indicates that these features are sensitive to intersession variabilities.

6. REFERENCES

[1] Yeshwant K Muthusamy, Etienne Barnard, and Ronald A Cole, "Reviewing automatic language identification," *IEEE Signal Processing Magazine*, vol. 11, no. 4, pp. 33–41, 1994.

[2] Haizhou Li, Bin Ma, and Kong Aik Lee, "Spoken language recognition: from fundamentals to practice," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1136–1159, 2013.

[3] Haizhou Li, Bin Ma, and Chin-Hui Lee, "A vector space modeling approach to spoken language identification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 1, pp. 271–284, 2007.

[4] Marc A Zissman, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Transactions on speech and audio processing*, vol. 4, no. 1, pp. 31, 1996.

[5] Yan Song, Bing Jiang, YeBo Bao, Si Wei, and Li-Rong Dai, "I-vector representation based on bottleneck features for language identification," *Electronics Letters*, vol. 49, no. 24, pp. 1569–1570, 2013.

[6] Pavel Matejka, Le Zhang, Tim Ng, Harish Sri Mallidi, Ondrej Glembek, Jeff Ma, and Bing Zhang, "Neural network bottleneck features for language identification," in *Proceedings of Odyssey*, 2014, vol. 2014, pp. 299–304.

[7] Bing Jiang, Yan Song, Si Wei, Jun-Hua Liu, Ian Vince McLoughlin, and Li-Rong Dai, "Deep bottleneck fea-

tures for spoken language identification," *PloS one*, vol. 9, no. 7, pp. e100795, 2014.

[8] Radek Fer, Pavel Matějka, František Grézl, Oldřich Plchot, Karel Veselỳ, and Jan Honza Černockỳ, "Multilingually trained bottleneck features in spoken language recognition," *Computer Speech & Language*, vol. 46, pp. 252–267, 2017.

[9] Mitchell McLaren, Luciana Ferrer, and Aaron Lawson, "Exploring the role of phonetic bottleneck features for speaker and language recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5575–5579.

[10] Najim Dehak, Pedro A Torres Carrasquillo, Douglas Reynolds, and Reda Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Twelfth annual conference of the international speech communication association*, 2011.

[11] Jan Pešán, Lukáš Burget, and Jan Černockỳ, "Sequence summarizing neural networks for spoken language recognition," *Interspeech 2016*, pp. 3285–3288, 2016.

[12] David Snyder, Daniel Garcia-Romero, Alan McCree, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "Spoken language recognition using x-vectors," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 105–111.

[13] Alicia Lozano-Diez, Oldřich Plchot, Pavel Matejka, and Joaquin Gonzalez-Rodriguez, "DNN based embeddings for language recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5184–5188.

[14] Ma Jin, Yan Song, Ian McLoughlin, and Li-Rong Dai, "LID-senones and their statistics for language identification," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 1, pp. 171–183, 2018.

[15] Anna Silnova, Pavel Matejka, Ondrej Glembek, Oldrich Plchot, Ondrej Novotny, Frantisek Grezl, Petr Schwarz, Lukas Burget, and Jan Cernocky, "BUT/phonexia bottleneck feature extractor," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 283–287.

[16] Shikha Gupta, Aroor Dinesh Dileep, and Veena Thenkanidiyoor, "Segment-level pyramid match kernels for the classification of varying length patterns of speech using SVMs," in *2016 24th European Signal Processing Conference (EUSIPCO)*. IEEE, 2016, pp. 2030–2034.

[17] Yeshwant K Muthusamy, Ronald A Cole, and Beatrice T Oshika, "The OGI multi-language telephone speech corpus," in *Second International Conference on Spoken Language Processing*, 1992.

[18] A. Baby, A. L. Thomas, N. L. Nishanthi, and T. Consortium, "Resources for indian languages," in *CBBLR Community Based Building of Language Resources*, Brno, Czech Republic:Tribun EU, 2016, pp. 37–43.

[19] KV Mounika, Sivanand Achanta, HR Lakshmi, Suryakanth V Gangashetty, and Anil Kumar Vuppala, "An investigation of deep neural network architectures for language recognition in indian languages.," in *INTERSPEECH*, 2016, pp. 2930–2933.

[20] "The 2015 NIST Language Recognition Evaluation Plan (LRE15)," .

[21] François Chollet et al., "Keras," `https://keras.io`, 2015.

[22] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, April 2011, Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.