DKU-Tencent Submission to Oriental Language Recognition AP18-OLR Challenge

Haiwei Wu*†, Weicheng Cai*†, Ming Li*, Ji Gao‡, Shanshan Zhang‡, Zhiqiang Lyu‡, Shen Huang‡
* Data Science Research Center, Duke Kunshan University, Kunshan, China
E-mail: ming.li369@dukekunshan.edu.cn
† School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China
‡ Tencent Research, Beijing, China

Abstract—**In this paper, we describe our submitted DKU-Tencent system for the oriental language recognition AP18-OLR Challenge. Our system pipeline consists of three main components, including data augmentation, frame-level feature extraction, and utterance-level modeling. First, we perform speed perturbation to increase the diversity and amount of training data. Second, we extract several kinds of frame-level features, including the hand-crafted acoustic features as well as the deep phonetic features. Third, we aggregate the frame-level features into fixed-dimensional utterance-level representation through i-vector and x-vector modelings. We also propose a deep residual network to obtain the utterance-level language posteriors in an end-to-end manner. Our submitted primary system achieves C_{avg} of 0.0499, 0.0146, and 0.0135 for the corresponding short-utterance, confusing language and open-set tasks on the evaluation set.**

I. Introduction

Language identification (LID) can be considered as a task of utterance-level speech attribute recognition whose goal is to identify the language category of a given variable-length speech. Different from the "sequence-to-sequence" tagging tasks like speech recognition, LID is a "sequence-to-one" summary task, which requires us to consider the entire audio content for decision.

To further boost the research and improve the techniques on LID, the center for speech and language technologies (CSLT) at Tsinghua University organizes the series Oriental Language Recognition (OLR) Challenge [1–3]. In this year, the organizers put forward three challenging tasks in LID, including short-utterance identification, confusing-language identification, and open-set recognition [3].

The performance of the LID system for short utterances usually degrades severely due to the insufficiency of language information in short audio segments. Short-utterance identification task focuses on this problem and provides the evaluation utterances as short as one second long. Confusing-language identification task requires us to develop systems identifying three confusing languages, containing Cantonese, Korean, and Mandarin. In real-world scenarios, sometimes we also need to reject speech from out of set languages in the open-set recognition task [3].

Generally, the focus of the LID task is to find out the discriminative and robust utterance-level representation for the variable-length audio sequence. The LID processing pipeline usually contains the following steps, including frame-level feature extraction, utterance modeling, and classification.

Given the raw waveform, considering the quasi-stationary property of speech, we typically convert it into a frame-level feature sequence. Several hand-crafted acoustic level features, such as log mel-filterbank energies (Fbank), mel-frequency cepstral coefficient (MFCC), perceptual linear prediction (PLP) [4], or shifted delta coefficients (SDC) features [5] are commonly adopted. We can also extract the phonetic features automatically from the phoneme decoder trained with deep neural network (DNN). The phonetics features include the bottleneck feature [6], the phoneme posterior probability (PPP) feature [7], and the tandem feature [8].

The feature sequence only describes the local frame-level pattern. The remaining question is how to aggregate the feature sequence into a global utterance-level representation. One of the most popular representative approaches is i-vector modeling. Variable-length speech utterances can be transformed into fixed-dimensional supervectors by accumulating the sufficient statistics over time, and then projected into a low-dimensional i-vector representation [9–11]. The process to extract the i-vector representation covers a series of separated models, and they are commonly trained in an unsupervised manner. In recent years, due to the excellent performance of deep learning approaches, many supervised methods managed to discriminate between language categories [12–15] directly. Among them, the x-vector [15] modeling based on the time-delay neural network (TDNN) [16] has become popular due to its superior performance. Recently, Cai *et al.* has built a deep residual network (ResNet) architecture for end-to-end LID, and yielded state-of-the-art performance [17–19].

After i-vectors and x-vectors are extracted, back-end classifiers such as logistic regression (LR) [20], support vector machine (SVM) [21] are employed to do the final decisions. For the deep ResNet methods, the final fully-connected layers can act as a classifier, and the utterance-level decision can be directly generated from the network output [19].

In AP18-OLR challenge, our primary system is built upon different types of frame-level features, including MFCC, Fbank, Bottleneck, PPP, and Tandem features. To extract utterance-level representations and make the final decisions, we adopt several kinds of utterance-level modeling schemes, including i-vector modeling, x-vector modeling, and end-to-

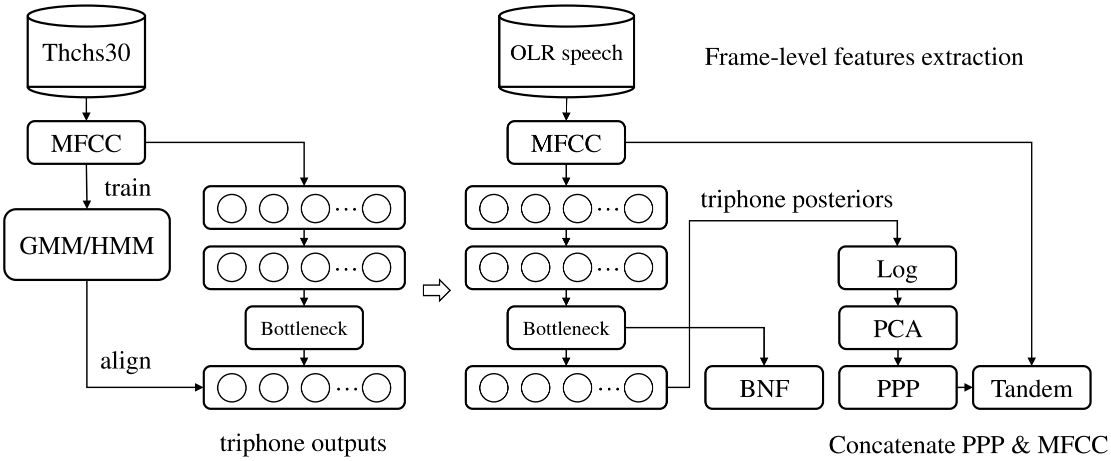

Fig. 1. The Process of extracting the PPP, BNF, Tandem features.

end ResNet modeling.

Our paper is organized as followed. Section 2 presents the data augmentation strategy. Section 3 introduces the frame-level features extraction computed in our system, and section 4 describes the modeling algorithms. Our experiments and results are presented in section 5, and the final section concludes the paper.

II. Data augmentation

Data augmentation is a common scheme to increase the quantity of training data and improve the robustness of the machine learning system. Recently, following the x-vector system [15], many works in speaker and language community prefer to augment the training data with additive noise and simulated room impulse responses (RIRs) [22]. However, it relies on additional noise source dataset, which is not allowed for the challenge.

To comply with the requirements of the fixed training condition of the evaluation, we adopt a simple speed perturbation strategy to augment the training set. Speed perturbation [23] is proven to be an effective augmentation method in speech recognition. It is easily implemented and does not need to rely on any external data. Following [23], in this challenge, we adopt speed perturbation with factor 0.9, 1.0, and 1.1 to augment the training data. We pool all of them together and obtain training data three times larger than the original one.

III. Frame-level feature extraction

This section describes our frame-level features in our system, including the hand-crafted acoustic features like MFCC, Fbank, and the phoneme discriminant features learned from DNN such as PPP, BNF, Tandem features. The extraction of deep phonetic features is illustrated in Fig. 1.

A. MFCC feature

A 25 ms window with 10 ms shifts is applied to compute the 20-dimensional MFCCs and their first and second derivatives. The filter banks are selected within the range of 20 to 7600 Hz. A simple energy-based voice activity detector (VAD), which classifies a frame as speech or non-speech based on the average log-energy with a given window centered at the current frame, is used. Before VAD, a short-time cepstral mean subtraction (CMS) is applied on the MFCC features over a 3-second sliding window.

B. Fbank feature

To get MFCCs, we need to perform the discrete cosine transform (DCT) operation on the Fbank features. Compared with the Fbank, MFCC is much more compressible and a bit more decorrelated, which is beneficial for linear models like Gaussian mixture models (GMMs).

However, for the end-to-end DNN modeling, especially for the convolutional neural network (CNN), the correlated information might be helpful. Therefore, our end-to-end ResNet system adopts Fbank feature as input. Each utterance is converted to 64-dimensional Fbank with a frame length of 25 ms. A short-time CMS is applied over a 3-second sliding window, and an energy-based VAD is used to drop the non-speech frames.

C. PPP feature

The PPP feature extractor utilizes a phoneme recognizer trained with the Chinese corpus, Thchs30 dataset [24]. First, we extract the 39-dimensional MFCC feature for each utterance to train GMM-HMM acoustic models. Then, the alignment of tri-phones is generated with the GMM-HMM model for each utterance. With the alignments, we train a DNN acoustic model with 3328 tied tri-phones states (or senones). The PPP feature is a low-dimensional representation of the frame-level phoneme posterior probability. After logarithm and Principal Component Analysis (PCA), we get the resulted 52-dimensional PPP features.

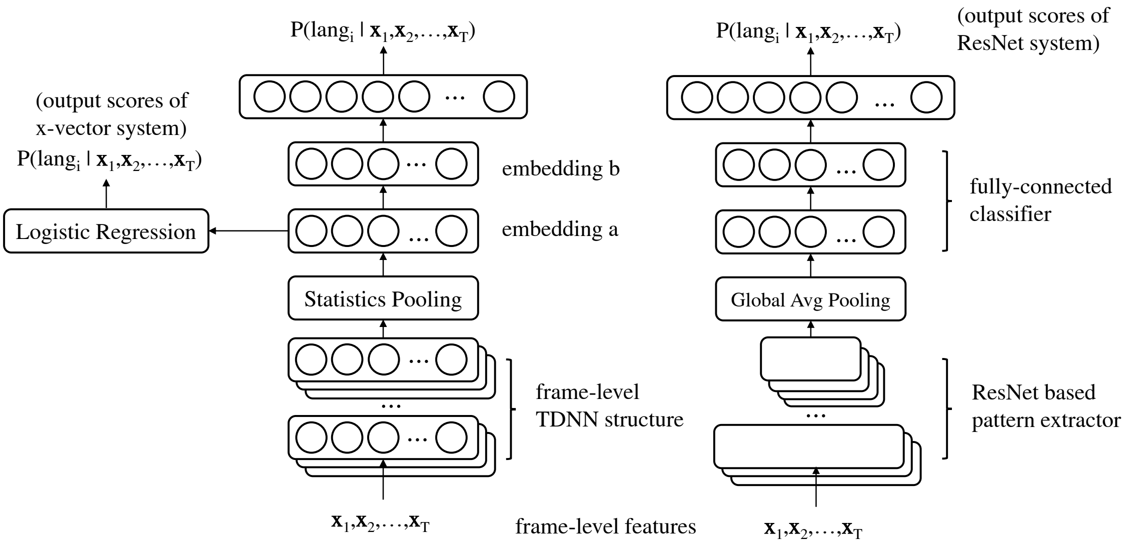
P(lang$_i$ | $\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T$)
(output scores of x-vector system)
P(lang$_i$ | $\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T$)
Logistic Regression
embedding b
embedding a
Statistics Pooling
frame-level TDNN structure
$\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T$
frame-level features
P(lang$_i$ | $\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T$)
(output scores of ResNet system)
fully-connected classifier
Global Avg Pooling
ResNet based pattern extractor
$\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T$

Fig. 2. Structures of the x-vector system and ResNet system.

D. Tandem feature

The resulted 52-dimensional PPP feature is fused with the 60-dimensional MFCC at the feature level to get the 112-dimensional tandem feature. Then a short-time CMS is computed followed by an energy-based VAD.

E. Bottleneck feature

The same as the process in the PPP feature extraction, a DNN acoustic model is trained on Thchs30 dataset. The BNF is directly extracted from the bottleneck layer of the DNN acoustic model rather than the output layer.

IV. Utterance-level modeling

We adopt three utterance-level modeling schemes for LID, including i-vector modeling, x-vector modeling, and end-to-end ResNet modeling. LR follows i-vector and x-vector extraction to produce final decisions, and the end-to-end ResNet directly generate the posteriors for each language from its output layer. The x-vector modeling and ResNet modelings are detailedly illustrated in Fig. 2.

A. i-vector + LR

I-vector system, which is the baseline system in OLR2018, has long dominated the speaker recognition task as well as the LID task for its excellent performance and high-efficiency [9, 10].

To train an i-vector extractor [8, 25], first, a 2048 components UBM model is trained on the MFCC, PPP, Tandem or BNF features with diagonal covariance matrices at the beginning. Then, with the initialization of the diagonal UBM, we train a full covariance UBM of 2048 components. The zero-order and first-order Baum-Welch statistics are computed on the UBM for each recording to obtain a supervector. Moreover, to reduce the dimensionality of supervectors, single factor analysis is employed to extract 600-dimensional i-vectors.

After the utterance-level i-vectors are extracted, a general logistic regression (LR) model is trained to get the utterance-level decisions.

B. x-vector + LR

The x-vector system [15] is developed using the data recipe available at Kaldi.

For the acoustic features, we extract 23-dimensional MFCC features (including c0) from 25 ms frames with a shift of 10 ms using a 23-channel mel-scale filter bank spanning the frequency range of 20 Hz to 7600 Hz. Besides the hand-crafted acoustic features, we also extract phoneme features including BNF and PPP feature for x-vector model training.

For x-vector extraction, a TDNN (Time-Delay Neural Network) is trained to discriminate among ten languages in the training set. The first five hidden layers, which are time delayed layers, operate at frame-level. Then a statistics pooling layer is employed to compute the mean and standard deviation over all frame-level outputs for each input segment. The resulted segment-level representations are then fed into two fully connected layers to classify the languages labels. After training, utterance-level x-vectors are extracted from the 512-dimensional affine component of the first fully-connected layer. Finally, an LR model is trained to obtain the final posteriors for each utterance.

C. End-to-end ResNet

Our end-to-end network structure is the same as [19] and is trained to identify ten target languages directly. The procedure

TABLE I
OUR END-TO-END RESNET MODEL ARCHITECTURE.

Layer	Output size	Structure	#Params
Conv1	$16\times64\times T$	3×3, stride 1	176
Res1	$16\times64\times T$	$\begin{bmatrix} 3\times3,\ 16 \\ 3\times3,\ 16 \end{bmatrix}\times3$, stride 1	14K
Res2	$32\times32\times\frac{T}{2}$	$\begin{bmatrix} 3\times3,\ 32 \\ 3\times3,\ 32 \end{bmatrix}\times4$, stride 2	70K
Res3	$64\times16\times\frac{T}{4}$	$\begin{bmatrix} 3\times3,\ 64 \\ 3\times3,\ 64 \end{bmatrix}\times6$, stride 2	427K
Res4	$128\times8\times\frac{T}{8}$	$\begin{bmatrix} 3\times3,\ 128 \\ 3\times3,\ 128 \end{bmatrix}\times3$, stride 2	821K
GAP	128	Global average pooling	0
FC	64	Fully-connected	8K
Output	10	Fully-connected	650

can be divided into three main components, local pattern extractor, global average pooling, and classifier.

1) Local pattern extractor: Deep Convolutional Neural Network (DCNN) structure can capture the high-level abstract patterns from local feature descriptors. Our DCNN structure is based on the well known ResNet-34 [26] architecture illustrated in Table I. It learns high-level features from frame-level Fbank, PPP, and Tandem features.

2) Global average pooling: The output of pattern extractor is still variable, and temporal pooling is necessary for generating utterance-level features. A global average pooling (GAP) layer [27] is then designated on top of our DCNN structure and transforms the local feature maps into a 256-dimensional utterance-level representation.

The GAP layer takes the means along with the time-frequency axis to accumulate the statistics. Given an output feature map $\mathbf{F}$ from the pattern extractor with the size of $C \times H \times W$, the accumulating process can be formulated as:

$$u_k = \frac{1}{H \times W} \times \sum_{H}^{i=1} \sum_{W}^{j=1} \mathbf{F}_{i,j,k}, \tag{1}$$

where $k \in [1, C]$. With this structure, we can obtain a fix-dimensional utterance-level feature $\mathbf{u} = [u_1, u_2, \ldots, u_C]$ of each audio sample for further classification.

3) Classifier: The utterance-level representation is then processed by two fully-connected layers and finally connected with an activation output layer. Each unit in the output layer is represented as a target language category. This structure acts as a back-end classifier and directly gives the final decisions.

V. EXPERIMENTS

A. Dataset and metric

For the training data, we have AP16-OL7, AP17-OL3, including AP16-OL7-train, AP16-OL7-dev, AP16-OL7-test, AP17-OL3-train, and AP17-OL3-dev. There are 72234 utterances and ten target languages [3].

We leave AP17-OLR-test as our development set, and there are 22051 utterances. The dataset contains audio files with different durations of 1 second and full length.

TABLE II
AP18-OLR DEVELOPMENT SET PERFORMANCE

Feature type	Modeling	$C_{avg}\times$ 100	
		Full-length	1 second
MFCC	i-vector + LR	3.58	14.23
PPP	i-vector + LR	2.23	14.54
Tandem	i-vector + LR	2.77	13.21
BNF	i-vector + LR	3.17	20.74
MFCC	x-vector + LR	3.45	11.85
PPP	x-vector + LR	1.78	11.47
BNF	x-vector + LR	1.97	15.48
Fbank	ResNet	4.63	8.98
PPP	ResNet	1.49	11.02
Tandem	ResNet	2.08	9.62
Fusion		0.85	5.76

TABLE III
AP18-OLR EVALUATION SET PERFORMANCE

Task	$C_{avg}\times 100$
Short-utterance identification task	4.99
Confusing-language identification task	1.42
Open-set recognition task	1.35

In our ResNet system, all the components in the pipeline are jointly learned in an end-to-end manner with a softmax classifier based cross-entropy loss. The model is trained with a mini-batch size of 256. The network is trained using standard stochastic gradient descent with momentum 0.9 and weight decay 1e-4. The learning rate is set to 0.1, 0.01, 0.001, and is switched when the training loss plateaus. The training is finished at 30 epochs. Since we have no separated validation set, the converged model after the last optimization step is used for evaluation. For each training step, an integer L within $[100, 700]$ interval is randomly generated, and each data in the mini-batch is cropped or extended to L frames. After model have been trained, the utterance-level posteriors can be directly computed from the output layer of the neural network for the given variable-length frame-level acoustic or phoneme features.

The primary evaluation metric in OLR2018 are C_{avg} [3] which is calculated by

$$\begin{aligned} C_{avg} = & \frac{1}{N} \sum_{L_t} \{ P_{Target} P_{Miss} \\ & + \sum_{L_n} P_{Non-Target} P_{FA}(L_t, L_n) \}. \end{aligned} \tag{2}$$

N denotes the number of language. P_{Target} is the prior probability of the target language and $P_{Non-Target}$ is computed by $P_{Non-Target} = (1 - P_{Target})/(N-1)$. L_t and L_n are the target and non-target language. P_{Miss} and P_{FA} refer to the missing rate and false alarm probability.

B. Performance and Analysis

The system performance on the development set is shown in Table II.

From the perspective of modeling methods, we find that the ResNet model with PPP features achieves the best performance. The x-vector system obtains lower C_{avg} than the i-vector system on MFCC, PPP and BNF respectively. For full-length audio data, the i-vector system reaches a relatively satisfying performance while in the scenario of short utterance, its performance degrades. It shows that the traditional i-vector method is more suitable for long duration tasks. The end-to-end ResNet and x-vector system show much better results in short utterance than the i-vector system, which means the neural network modeling is less sensitive to the length of the audio sequence and more robust.

Among different input features, the phoneme discriminant features automatically learned from DNN, including PPP and BNF, show excellent performance in the scenario of full-length utterances. The results reveal that phonetic information is effective in LID task. We could also find that PPP features perform better than BNF features. Although the traditional acoustic features like MFCC and Fbank receive higher C_{avg} in full-length, they are still competitive in short utterance scenario. Our end-to-end ResNet system with Fbank achieves the best performance in the 1-second development set.

We can observe from the results that, in the full-length task, phonetic information is dominant and brings superior performance. And in the 1-second task, traditional acoustic features are more informative than phoneme features. The tandem feature fused by MFCC and PPP feature is stable and robust in both full-length and 1-second scenario.

We evaluate our system performance and fusion parameters on the development set. Then with the fusion parameters, we pool all the training and development data together and re-train all the sub-system again. For evaluation, score-level fusion is further used to combine the utterance-level sub-system scores into our final submission. The fusion parameters are calibrated using the FoCal Multi-class toolkit [28]. The performance on the evaluation set of three tasks are shown in Table III. The results show that our system is robust and stable in short-utterance identification, confusing-language identification, and open-set recognition tasks.

VI. Conclusion

This paper presents our DKU-Tencent system for OLR2018. We extract MFCC, Fbank, PPP, BNF, and Tandem features as our input frame-level features. As for modeling, we employ i-vector/x-vector + LR as well as the end-to-end ResNet system. Results show that the x-vector and ResNet system obtain better performance than the baseline i-vector system. We also find that systems with phoneme features learned by DNN generate excellent results in full-length task and models trained with the traditional acoustic features achieve competitive performance in the short utterance scenario.

VII. Acknowledgment

This research was funded in part by the National Natural Science Foundation of China (61773413), Natural Science Foundation of Guangzhou City (201707010363), Six Talent Peaks project in Jiangsu Province (JY-074), Science and Technology Program of Guangzhou City (201903010040) and CCF-Tencent Open Fund.

References

[1] D. Wang, L. Li, D. Tang, and Q. Chen, “Ap16-ol7: A multilingual database for oriental languages and a language recognition baseline,” in *Proc. APSIPA*, pp. 1–5, 12 2016.

[2] Z. Tang, D. Wang, Y. Chen, and Q. Chen, “Ap17-olr challenge: Data, plan, and baseline,” in *Proc. APSIPA*, 06 2017.

[3] Z. Tang, D. Wang, and Q. Chen, “Ap18-olr challenge: Three tasks and their baselines,” in *Proc. APSIPA*, 2018.

[4] D. P. W. Ellis, “PLP and RASTA (and MFCC, and inversion) in Matlab,” 2005. online web resource.

[5] H. Li, B. Ma, and K. Lee, “Spoken language recognition: From fundamentals to practice,” *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1136–1159, 2013.

[6] 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 Workshop*, pp. 299–304, 2014.

[7] M. Li and W. Liu, “Speaker verification and spoken language identification using a generalized i-vector framework with phonetic tokenizations and tandem features,” in *Proc. INTERSPEECH*, pp. 1120–1124, 2014.

[8] M. Li, L. Liu, W. Cai, and W. Liu, “Generalized i-vector representation with phonetic tokenizations and tandem features for both text independent and text dependent speaker verification,” *Journal of Signal Processing Systems*, vol. 82, pp. 207–215, 07 2015.

[9] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, “Language recognition via i-vectors and dimensionality reduction,” in *Proc. INTERSPEECH*, pp. 857–860, 01 2011.

[10] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *Audio, Speech, and Language Processing, IEEE Transactions*, vol. 19, pp. 788 – 798, 06 2011.

[11] L. Li, H. Guo, F. Shang, Q. Hong, and K. Liu, “Evaluation of the i-vector system for text-dependent speaker verification,” in *Proc. ASID*, pp. 60–63, Oct 2017.

[12] 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, pp. 134–144, Jan 2018.

[13] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. Moreno, “Automatic language identification using deep neural networks,” in *Proc. ICASSP*, pp. 5337–5341, 2014.

[14] J. Ma, Y. Song, I. Mcloughlin, W. Guo, and L. Dai, “End-to-end language identification using high-order utterance representation with bilinear pooling,” in *Proc. INTERSPEECH*, pp. 2571–2575, 2017.

[15] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and

S. Khudanpur, “X-Vectors: Robust DNN Embeddings for Speaker Recognition,” in *Proc. ICASSP*, pp. 5329–5333, Apr. 2018.

[16] D. Snyder, D. Garcia-Romero, and D. Povey, “Time delay deep neural network-based universal background models for speaker recognition,” in *Proc. ASRU*, pp. 92–97, 2015.

[17] W. Cai, Z. Cai, W. Liu, X. Wang, and M. Li, “Insights into end-to-end learning scheme for language identification,” in *Proc. ICASSP*, pp. 5209–5213, 2018.

[18] W. Cai, Z. Cai, X. Zhang, and M. Li, “A novel learnable dictionary encoding layer for end-to-end language identification,” in *Proc. ICASSP*, pp. 5189–5193, 2018.

[19] W. Cai, J. Chen, and M. Li, “Exploring the encoding layer and loss function in end-to-end speaker and language recognition system,” in *Proc. Odyssey Workshop*, pp. 74–81, 2018.

[20] S. MENARD, “Applied logistic regression analysis,” *Sage University Paper Series on Quantitative Application in the Social Sciences*, vol. 7, p. 88, 1995.

[21] J. A. Suykens and J. Vandewalle, “Least squares support vector machine classifiers,” *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[22] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, “Spoken language recognition using x-vectors,” in *Proc. Odyssey Workshop*, pp. 105–111, 2018.

[23] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, “Audio augmentation for speech recognition,” in *Proc. INTERSPEECH*, pp. 3586–3589, 2015.

[24] D. Wang and X. Zhang, “Thchs-30: A free chinese speech corpus,” *arXiv preprint arXiv:1512.01882*, 2015.

[25] D. Garcia-Romero and C. Y. Espy-Wilson, “Analysis of I-Vector Length Normalization in Speaker Recognition Systems,” in *Proc INTERSPEECH*, pp. 249–252, 2011.

[26] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, pp. 770–778, 2016.

[27] M. Lin, Q. Chen, and S. Yan, “Network in network,” 2014.

[28] N. Brummer, “Focal: Tools for fusion and calibration of automatic speaker detection systems,” *URL: https://sites.google.com/site/nikobrummer/focal*, 01 2005.