Phone-Aware Multi-task Learning and Length Expanding for Short-Duration Language Recognition

Miao Zhao[1], Rongjin Li[2], Shijiang Yan[1], Zheng Li[2], Hao Lu[1], Shipeng Xia[1], Qingyang Hong[1], Lin Li[2]
[1]School of Information Science and Engineering, Xiamen University, China
[2]School of Electronic Science and Engineering, Xiamen University, China
E-mail:{qyhong,lilin}@xmu.edu.cn

***Abstract*—In the language recognition, the phonetic information has shown great potential for neural network to learn the high-level representations. In this paper, we explore two significant aspects to improve the system performance on oriental language recognition (OLR) challenge under the short-duration condition. Firstly, we propose to learn the language information and phonetic information jointly with multi-task learning. The classified networks can learn the extra phonetic representation from a frame-level phone-task and extract the language embedding at the segment level. Furthermore, we propose to introduce length expanding strategy to provide supplemental information of short-duration utterances by dithering the short duration evaluation utterances at different speeds. The evaluation results of the 3rd OLR Challenge showed that our proposed methods obtained the best results on the short-duration condition.**

Keywords: phonetic information, multi-task learning, length expanding, speed perturbation pooling, short-duration, language recognition

I. Introduction

Language recognition is to determine the category of language corresponding to a given spoken utterance. Recently, neural networks have been increasingly popular in the application of language recognition. One of the most remarkable use of neural networks is bottleneck feature (BNF). The BNF has been originally developed for speech recognition [1] and introduced successfully to speaker verification [2], [3], [4] and language recognition [5], [6], [7], [8], [9]. BNF is extracted from the output of one hidden layer of a trained automatic speech recognition (ASR) neural network, it can also be concatenated together with acoustic features to generate tandem features.

Besides the implement of BNF, recent researchers also improved the performance of language recognition systems using temporal modeling of neural networks. In [10], Tang et al. introduced long-short term memory (LSTM) deep neural network (DNN) to learn phonetic and language information in frame-level embedding framework, namely phonetic temporal neural (PTN) model. The LSTM based systems achieve promising performance because LSTM framework considers temporal information in language recognition. Later, bidirectional LSTM (BLSTM) [11] and time delay neural network (TDNN) [12], [13] are used to process the temporal information, which are further conducted at the segment-level after a statistics pooling. Furthermore, Cai et al. investigated various kinds of encoding layers to learn utterance-level representation based on convolutional neural network (CNN) and BLSTM [14].

However, the use of BNF is based on two independent neural networks, posing an uncooperative effect on language recognition since the relationship between the phone and language is weak. In this paper, a new framework of multi-task learning (MTL) is proposed in speaker recognition [15] to train the language classification with phonetic information jointly. Considering phonetic representation can not be modeled at the segment-level, we then train two tasks on different branches respectively and they share layers at the frame-level. The shared layers contain both language and phonetic information, and the remaining layers of phonetic branch can be cut to simplify the system after training. In order to enhance the temporal modeling, our systems are developed based on TDNN.

Furthermore, the very short-duration speech (about 1 second) is quite difficult to be recognized because of limited clues. In [16], the authors proposed to dither the same speech at different speeds and splice them at time-axis to provide supplement information, and the speed perturbation does not seriously affect the phonetic and language information. Motivated by this point, we investigate the length expanding strategy to enhance the short-duration speech and extract the subsequent x-vector [17], [18] based on TDNN. We firstly dither the same evaluation utterances at different speeds and splice the acoustic features at the time-axis to obtain the length expanded feature, which is fed into TDNN afterwards. However, such a length expanding strategy may not be suitable for temporal modeling neural network. Hence, a more effective strategy is proposed to expand the length, named speed perturbation pooling (SPP). Firstly, we directly extract several x-vectors of the dithered speech at different speeds, then an x-vector is obtained by calculating the average of these x-vectors, which could enrich the language information of short-duration speech on the segment level.

The remainder of this paper is as follows. Section II describes the multi-task learning structure based on TDNN for language recognition and phonetic information. Section III introduces the proposed speed perturbation pooling method to expand the utterance length for short duration language recognition. Then the experimental setup is presented in section IV, experimental results are analogized in section V. Finally, we conclude this paper in section VI.

II. X-vector Systems for Language Recognition

A. Standard X-vector

In this paper, our proposed systems are based on standard x-vector architecture. The first 5 layers of the neural network process the input at the frame-level, with a temporal context centered at the current frame t. Similar to standard TDNN configuration, the first layer splices the features at frames $t-2, t-1, t, t+1, t+2$. The inputs of next two hidden layers are the spliced output of its previous frames at $t-2, t, t+2$ and $t-3, t, t+3$, respectively, and then a statistic pooling layer aggregates the representation across the time-axis, hence the next layers operate on the segment-level. Besides the 5-th hidden layer, all layers are 512-dimensional.

B. Splicing with Phonetic Representation

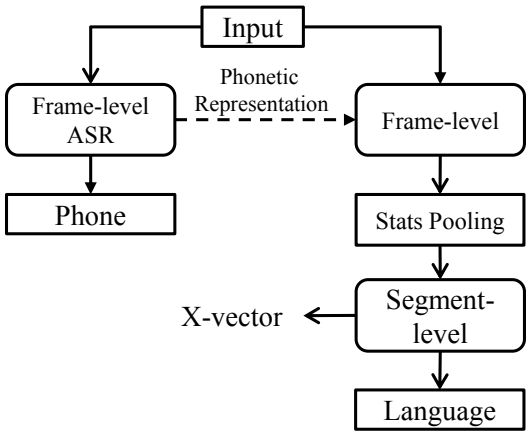

Fig. 1. The language classification network with the phonetic representation from ASR network.

For comparison, we also train a conventional joint training DNN with acoustic features and phonetic representation, named the phonetic network (PN). It is the same with the baseline framework in [10], [15]. In Figure 1, we firstly train an ASR network that is modified from TDNN without statistics pooling layer, and then we extract the phonetic representation from the last hidden layer and splice them to the 5-th frame-level layer of the language network. Since the phonetic representation can be seen as the auxiliary features, it can be trained based on a simple language-independent ASR alignment information, and the difference from the BNF is that the phonetic representation is spliced directly into the hidden layer of language classification network, rather than in tandem with acoustic features or being fed into the input layer.

C. Multi-Task Learning for Language and Phonetic Tasks

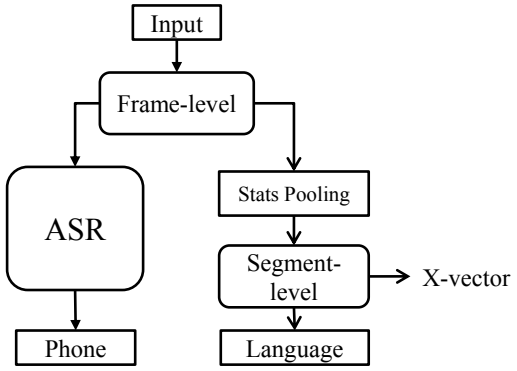

Fig. 2. The multi-task learning for the language classification and ASR.

Considering the relationship between language and phone tasks, we propose to utilize the multi-task learning to train the two tasks jointly. In Figure 2, the frame-level hidden layers are the shared part that learns the phonetic compensation information for the language task. From a view of feature space, the phonetic representation is invariant information that is not affected by differences in language and duration. The gradient descent of each task will affect the frame-level shared layers in training and the x-vector will be extracted from the penultimate segment-level layer in the language task branch.

III. Speed Perturbation Pooling

The language recognition performance is often greatly reduced when the length of evaluation utterance becomes too short, such as 1 second duration utterance. In order to compensate the short-duration condition, we propose to expand the length of evaluation utterances by dithering the speed. The difference in the speed not only enrich the language information but weaken the speaker factor. Furthermore, the speed perturbation used in the evaluation set does not require same processing for the training set, i.e. it does not increase the training cost, and if the speed of speech is modified too much, it will affect the recognition accuracy of the utterance [19], [20] and so we should control the disturbance factor carefully. Hence, the following procession is based on three kinds of speeds, including 0.9, 1.0 and 1.1 speed factors.

To integrate the different speeds of one utterance, the speed perturbation concatenating (SPC) is developed in the reference [16]. However, it's a very rough process. Before being fed into the DNN extractor, the acoustic features of different speeds are concatenated at the time-axis firstly. Then the new features will be mapped to the fixed-dimensional x-vector:

$$x_{spc} \leftarrow concat(x_{sp0.9}, x_{sp1.0}, x_{sp1.1}) \tag{1}$$

$$X_{spc} = F(x_{spc}) \tag{2}$$

where $F(x)$ denotes the extractor of neural network that maps the variable-length acoustic features x to the fixed-dimensional embedding.

But we have found that SPC does not work well on the x-vector framework and even make the performance become very worse. In this paper, we propose another strategy, named speed perturbation pooling (SPP). SPP averages the x-vectors of same utterance into a new x-vector and the formulas are as follows:

$$\begin{cases} X_{sp0.9} = F(x_{sp0.9}) \\ X_{sp1.0} = F(x_{sp1.0}) \\ X_{sp1.1} = F(x_{sp1.1}) \end{cases} \tag{3}$$

$$X_{spp} = \frac{n_{sp0.9} \cdot X_{sp0.9} + n_{sp1.0} \cdot X_{sp1.0} + n_{sp1.1} \cdot X_{sp1.1}}{n_{sp0.9} + n_{sp1.0} + n_{sp1.1}} \tag{4}$$

where n is the frames of corresponding utterance and (4) can be seen as the weighted integration of x-vectors X for the different speeds.

IV. Experimental Setup

A. Datasets

As described in Table 1, all the datasets are recorded by mobile phones with a sampling rate of 16 kHz and size of 16 bits. The Thchs30-train consists of 10 languages, including Mandarin, Cantonese, Indonesian, Japanese, Russian, Korean, Vietnamese, Kazakh, Tibetan and Uyghur [21]. The duration of most utterances in BaseTrain and Dev-all are among 1-30 seconds. Dev and Eval are segmented to the duration of 1 second from long-duration datasets, respectively. Specially, Dev-all is a full-length version of Dev.

The BaseTrain is used to train x-vector system with 10 languages. And the Thchs30-train is used to train phonetic DNN with phonetic labels. The phonetic labels of Thchs30-train are forced alignment based on a GMM-HMM model of Kaldi thchs30 recipe [22]. This GMM-HMM model is also trained by Thchs30-train.

We evaluate our methods on the short-duration datasets, Dev and Eval, and the long-duration Dev-all is used as a comparison.

B. Data Augmentation

Because most of the datasets described in Section 4.1 have little noise, we just consider two strategies, speed perturbation and volume perturbation, to increase the amount and diversity of the training data. For speed perturbation, we apply a speed factor of 0.9 or 1.1 to slow down or speed up the original recording, and then we get additional two copies of original recording and add them to the original dataset list directly. For volume perturbation, we apply a random volume factor to change the volume of every recording of a dataset.

C. Front-End

1) Acoustic features: The Perceptual Linear Prediction (PLP) coefficients are used in all experiments with a frame-length of 25ms and frame-shift 10ms. Firstly, we compute the acoustic features with 20 dimensions. Then the 3-dimensional pitch features are also computed to be pasted in the end and finally we attain 23-dimensional acoustic features.

Before the acoustic features being fed into DNN, nonspeech frames are filtered out using an energy-based voice activity detection (VAD) and then cepstral mean-normalized (CMN) is performed over a sliding window of 3 seconds.

2) Baseline x-vector system: The standard x-vector framework described in Section 2.1 is used as our baseline system. The BaseTrain dataset is used to train our baseline system. And 512-dimensional x-vectors are extracted at the segment layer which is closed to the statistics pooling layer.

3) Phonetic representation-spliced x-vector system: In our PN x-vector system, the phonetic architecture has 5 TDNN layers and all layers have 650 nodes except the last hidden layer with 128 nodes. The splicing information of these hidden layers are { -2, -1, 0, 1, 2} {-1, 0, 1} {-1, 0, 1} {-3, 0, 3} and {-6, -3, 0} by the first-to-last order. And the output layer is a log-softmax layer with 3,447 nodes corresponding to phonetic labels. During the training phase, the phonetic architecture is pre-trained by Thchs30-train and then attached to the standard x-vector architecture as described in Section 2.2. Finally, the BaseTrain is used to train the x-vector architecture and the phonetic architecture is still updated with a very low learning rate at the same time.

4) Multi-task learning x-vector system: In our multi-task learning x-vector system, there are four shared layers which are the first four hidden layers of the standard x-vector architecture. The last shared layer concatenates two branches. One branch consists of the remaining layers of standard x-vector architecture and another one contains three layers with 512 nodes and a log-softmax output layer with 3,447 nodes corresponding to phonetic labels. As the same as the phonetic x-vector system, the phonetic architecture of multi-task learning system is also trained by Thchs30-train and the x-vector architecture is trained by BaseTrain dataset.

D. Back-End

In the back-end, once the x-vectors are extracted, linear discriminant analysis (LDA), subtract-mean and length normalization are applied before the embeddings being fed into a logical regression (LR) classifier. Specially, the LDA transforms 512-dimensional x-vectors into 10-dimensional vectors. And a cross-validation strategy between Dev/Dev-all and Eval is used to avoid computing global mean from the evaluation datasets themselves in the subtract-mean step. Both LDA and LR are trained by BaseTrain. Finally, the scores of evaluation datasets are generated by the LR.

All experiments are conducted with Kaldi toolkit [22].

V. Results

In this section, we report results in terms of equal error-rate (EER). And we firstly compare SPC method with SPP and then compare the different x-vector frameworks based on the optimal back-end.

TABLE I

THE DATASETS PROFILES.

Name	Datasets	Total Utterances	Total Duration	Length
BaseTrain	AP16-OL7, AP17-OL3(Train,Dev)	72,234	106.6h	1-30s
Thchs30-train	Thchs30-Train	10,000	25.5h	4-16s
Dev-all	AP17-OLR-Test(test_all)	22,051	34.2h	1-30s
Dev	AP17-OLR-Test(test_1s)	22,051	6.1h	1s
Eval	AP18-OLR-Test(Task_1)	21,456	6.0h	1s

A. SPC vs. SPP

Before using SPC and SPP to expand the length of evaluation utterances, there are three basic questions:

1) Why SPC does not work well on the x-vector framework?
2) If SPP benefits to short-duration utterances, how about long-duration utterances?
3) As both SPP and data augmentation are related to speed perturbation, then what relations between SPP method and speed perturbation of training?

To discuss these three questions, we conduct a series of experiments based on two baseline systems. Specially, for question 1, the self-concatenating (SC) is added to analyze whether the concatenating process will result in bad performance. For question 3, we train the two baseline systems with and without speed perturbation and do not use volume perturbation. We also control this variable in the training of LDA and LR. And they are consistent with DNN training all the time.

The results are presented in Table II. Firstly, we find that both SC and SPC methods perform worse than the original configuration, especially in short-duration condition. And we also find that both CMN and extracting embedding are influenced extremely by the concatenating process. It may bring unnatural context and information redundancy. However, SPP always outperforms the Original by $3\%-5\%$ for short-duration datasets even using no SP in training. Therefore, SPP may not rely on SP augmentation used in the training of DNN, LDA and LR. Secondly, we see that SPP is not really suitable for long-duration utterances by comparing the results of Dev-all with Dev/Eval. But there is no negative influence if using data augmentation of speed perturbation (SP) in DNN training.

TABLE II

THE EER (%) RESULTS WITH SPC AND SPP.

Datasets	SP aug.	Original	SC	SPC	SPP
Dev	No	8.90	9.25	9.94	**8.58**
	Yes	8.18	8.32	8.31	**7.90**
Eval	No	7.95	8.23	8.55	**7.60**
	Yes	7.62	7.71	7.53	**7.24**
Dev-all	No	**2.35**	2.38	2.50	2.48
	Yes	2.00	2.09	2.13	**1.99**

B. Multi-Task Learning for Language Recognition

In this experiment, we compare different x-vector frameworks, which are referenced throughout Sections 4.3.2-4.3.4. Because the Thchs30-train is a small dataset with one language and the GMM-HMM model used to generate the alignments of Thchs30-train is also not good. To observe the original influence of phonetic information, data augmentation is not used in Thchs30-train. But we still use two data augmentation strategies in BaseTrain to train x-vector architectures. We also use speed perturbation in LDA and LR to achieve better performance. And SPP is used for all evaluation datasets.

As shown in Table III, training with phonetic information (PN/MTL) is about 6% better than Baseline on short-duration datasets. PN and MTL are 5% and 13% better than Baseline on long-duration dataset respectively. Although MTL outperforms PN by about 9% on long-duration dataset, the performances between PN and MTL are almost at the same level. This may be due to the limited context information of short-duration utterance.

TABLE III

THE EER (%) RESULTS IN DIFFERENT X-VECTOR FRAMEWORKS.

Datasets	Baseline	PN	MTL
Dev	7.68	**7.16**	7.29
Eval	6.92	6.62	**6.50**
Dev-all	1.72	1.64	**1.50**

VI. CONCLUSION

In this paper, we proposed SPP method to expand length of short-duration utterances. It enriched the information and benefited to short-duration utterances. Meanwhile, it had little influence on long-duration utterances when training language DNN with data augmentation of speed perturbation. We also adapted multi-task learning to combine phonetic information into the x-vector framework. We found that multi-task learning could achieve excellent performance as the conventional joint training and even better. The architecture of multi-task learning was also simpler, and it was very useful that the number of parameters can be reduced as same as the standard x-vector architecture in test time.

VII. ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China (Grant No.61876160).

REFERENCES

[1] F. Grézl, M. Karafiat, and L. Burget, "Investigation into Bottle-Neck Features for Meeting Speech Recognition," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[2] D. Garcia-Romero and A. McCree, "Insights into Deep Neural Networks for Speaker Recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[3] S. Yaman, J. Pelecanos, and R. Sarikaya, "Bottleneck Features for Speaker Recognition," in *Odyssey 2012-The Speaker and Language Recognition Workshop*, 2012.

[4] A. Lozano-Diez, A. Silnova, P. Matejka, O. Glembek, O. Plchot, J. Pešán, L. Burget, and J. Gonzalez-Rodriguez, "Analysis and Optimization of Bottleneck Features for Speaker Recognition," in *Proceedings of Odyssey*, vol. 2016. ISCA Bilbao, Spain, 2016, pp. 352–357.

[5] A. Lozano-Diez, R. Zazo, D. T. Toledano, and J. Gonzalez-Rodriguez, "An Analysis of the Influence of Deep Neural Network (DNN) Topology in Bottleneck Feature based Language Recognition," *PloS one*, vol. 12, no. 8, p. e0182580, 2017.

[6] R. Fér, P. Matějka, F. Grézl, O. Plchot, and J. Černockỳ, "Multilingual Bottleneck Features for Language Recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[7] P. Matejka, L. Zhang, T. Ng, H. S. Mallidi, O. Glembek, J. Ma, and B. Zhang, "Neural Network Bottleneck Features for Language Identification," in *Proceedings of Odyssey*, vol. 2014, 2014, pp. 299–304.

[8] F. Richardson, D. Reynolds, and N. Dehak, "Deep Neural Network Approaches to Speaker and Language Recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 2015.

[9] B. Jiang, Y. Song, S. Wei, J.-H. Liu, I. V. McLoughlin, and L.-R. Dai, "Deep Bottleneck Features for Spoken Language Identification," *PloS one*, vol. 9, no. 7, p. e100795, 2014.

[10] 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.

[11] A. Lozano-Diez, O. Plchot, P. Matejka, and J. 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.

[12] V. Peddinti, D. Povey, and S. Khudanpur, "A Time Delay Neural Network Architecture for Efficient Modeling of Long Temporal Contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[13] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken Language Recognition Using X-vectors," *submitted to Odyssey*, 2018.

[14] W. Cai, D. Cai, S. Huang, and M. Li, "Utterance-Level End-to-End Language Identification Using Attention-based CNN-BLSTM," *arXiv preprint arXiv:1902.07374*, 2019.

[15] Y. Liu, L. He, J. Liu, and M. T. Johnson, "Speaker Embedding Extraction with Phonetic Information," *arXiv preprint arXiv:1804.04862*, 2018.

[16] M. Xiaoxiao, Z. Jian, S. Hongbin, Z. Ruohua, and Y. Yonghong, "Expanding the length of short utterances for short-duration language recognition," *Journal of Tsinghua University (Science and Technology)*, vol. 58, no. 3, pp. 254–259, 2018.

[17] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification." in *Interspeech*, 2017, pp. 999–1003.

[18] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[19] J. Yuan, M. Liberman, and C. Cieri, "Towards an integrated understanding of speaking rate in conversation," in *Ninth International Conference on Spoken Language Processing*, 2006.

[20] C. J. van Heerden, E. Barnard, E. Van Heerden *et al.*, "Speech rate normalization used to improve speaker verification," in *Proceedings of the Symposium of the Pattern Recognition Association of South Africa*. Citeseer, 2007, pp. 2–7.

[21] Z. Tang, D. Wang, and Q. Chen, "Ap18-olr challenge: Three tasks and their baselines," *arXiv preprint arXiv:1806.00616*, 2018.

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, 2011.