Phonexia s.r.o. submission to OLR 2020

Michal Klčo, Ondřej Novotný, Ján Profant, Josef Slavíček

Phonexia s.r.o.

{michal.klco, ondrej.novotny, ...}@phonexia.com

Abstract

This is the description of the systems submitted to all threes task of the AP20-OLR Challenge.

The points where our systems differ from the baseline are described in detail: added spectral augmentation, training process of TDNN, and ResNet architecture and different data preprocessing techniques for each architecture.

1. Introduction

AP20-OLR Challenge is the fifth oriental language recognition (OLR) challenge with three tasks: (1) cross-channel language identification, (2) dialect identification, and (3) language identification in a noisy environment.

The following sections describe submitted systems for all the tasks.

2. System

2.1. Data processing

2.1.1. Data description

The AP16-OL7, AP17-OL3, and AP17-OLR-test datasets [1] in combination with the THCHS30 [2] dataset were used as the training data for all our systems. Together, the training set contains 165427 utterances of 10 training languages.

2.1.2. Augmentation

The speed and volume perturbations were used to augment the training data in the same way as in the baseline system [1]. Additionally, we used spectral augmentation to add random distortion to the signal.

In spectral augmentation, we used two ways to distort the signal. First, the signal was transformed into the spectrogram, followed with simultaneous usage of both masking approach: (i) time masking, where several randomly selected frames were replaced by spectrogram mean value; (ii) frequency masking, where several randomly selected frequency bins were replaced by spectral mean value across all frames. After the spectral masking procedure, the spectrogram was transformed back to signal, followed by feature extraction.

2.1.3. Features

As features, we used 64-dimensional filter banks for all the tasks. The filter banks were computed in Kaldi with 25 ms window length and 10 ms shift.

2.2. Architecture

2.2.1. TDNN – task 1, 3

TDNN [3] is a neural net architecture popularly used in speech processing for tasks like SID or LID. For OLR2020 tasks 1 and 3, we used the original TDNN architecture adapted to the input features dimensionality difference. The detailed network architecture is presented in Table 1

Table 1: *TDNN architecture.*

Layer	Layer context	Input × output
frame1	$[t-2, t+2]$	320×512
frame2	$\{t-2, t, t+2\}$	1536×512
frame3	$\{t-3, t, t+3\}$	1536×512
frame4	$\{t\}$	512×512
frame5	$\{t\}$	512×1500
stats pooling	$[0, T)$	1500×3000
segment6	0	3000×512
segment7	0	512×512
softmax	0	512×10

2.2.2. ResNet18 – task 2

ResNet [4] is a successful DNN architecture developed for image processing and used in a wide variety of tasks, including speech processing. For OLR2020 task 2, we used ResNet18 with a statistics pooling layer from [3]. The output of the first linear layer after statistics pooling is used as language embedding. Out Resnet18 architecture is depicted in Table 2

Table 2: *The structure of ResNet18 architecture. The first dimension of the input shows the number of filter-banks and the second dimension indicates the number of frames.*

Layer	Structure	Stride	Output
Input	-	-	$64 \times 100 \times 1$
Conv2D-1	$3 \times 3, 32$	1	$64 \times 100 \times 32$
ResNetBlock-1	$\begin{bmatrix} 3\times3,32 \\ 3\times3,32 \end{bmatrix} \times 2$	1	$64 \times 100 \times 32$
ResNetBlock-2	$\begin{bmatrix} 3\times3,64 \\ 3\times3,64 \end{bmatrix} \times 2$	2	$32 \times 50 \times 64$
ResNetBlock-3	$\begin{bmatrix} 3\times3,128 \\ 3\times3,128 \end{bmatrix} \times 2$	2	$16 \times 25 \times 128$
ResNetBlock-4	$\begin{bmatrix} 3\times3,256 \\ 3\times3,256 \end{bmatrix} \times 2$	2	$8 \times 13 \times 256$
Statistics Pooling	-	-	16×256
Flatten	-	-	4096
Linear1	-	-	256
Linear2	-	-	10

3. Experiments

3.1. Training

We trained our ResNet18 in PyTorch framework on 6 NVIDIA RTX 2080ti GPUs with SGD optimizer with Nesterov momentum 0.9 and weight decay 0.0001. The model was trained for 3 epochs with batch size 32 per GPU (i.e. 192 in total). The learn-

Table 3: *Cavg and EER results on the referenced development sets.*

Task	Cross-channel LID		Dialect Identification	
Enrollment Set	AP20-ref-dev-task1		AP20-OLR-dialect	
Test Set	AP19-OLR-channel		AP19-OLR-dev&eval-task3-test	
	Cavg	EER%	Cavg	EER%
Baseline Kaldi i-vector	0.2965	29.12	0.0703	9.33
Baseline Kaldi x-vector	0.3583	36.37	0.0807	14.67
Baseline Pytorch x-vector	0.2696	26.94	0.0849	12.40
TDNN	0.2889	29.09	–	–
ResNet18	–	–	0.0256	4.60

ing rate was exponentially decaying from 0.06 to 0.0003. The system was trained as a classifier with softmax head and cross-entropy loss. For the final submission, we used a snapshot of the neural net after 1.39 epoch of training.

Our TDNN system was trained on data processed with Kaldi energy based VAD. Input features were mean-variance normalized over a sliding window of length 100 frames. Our ResNet18 was trained on raw features (i.e. no VAD, no mean-variance normalization).

3.2. Scoring

The same back-end was used as in [1]. Linear Discriminant Analysis (LDA) is used to reduce the dimension of the embeddings from 256 to 100, the embeddings are then centered and the score is calculated for each trial using logistic regression (LR). LDA, centering mean and LR were trained on the enrollment set.

4. Results

Table 3 shows the results of our systems on the referenced development sets in comparison with the baseline systems.

5. Conclusion

In this paper, we presented the systems that we submitted to the AP20-OLR challenge. We described the data processing, used architectures, training process, and scoring process.

6. References

[1] Z. Li, M. Zhao, Q. Hong, L. Li, Z. Tang, D. Wang, L. Song, and C. Yang, "Ap20-olr challenge: Three tasks and their baselines," *arXiv preprint arXiv:2006.03473*, 2020.

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

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," *Submitted to ICASSP*, 2018.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.