TIGERIT SYSTEM DESCRIPTION FOR NIST 2021 SPEAKER RECOGNITION EVALUATION

Zabir Al Nazi, Md. Erfanul Haque, Nasir Uddin Ahmed

TigerIT Bangladesh Ltd., Dhaka, Bangladesh

ABSTRACT

In this work, we present the speaker recognition system submission of TigerIT Bangladesh Ltd. to the 2021 Speaker Recognition Evaluation (SRE21). Our system is based on two well-known neural network architectures ResNet and EfficientNet. We present our test results on 2016 NIST SRE Evaluation Set, and VoxCeleb 1 test set.

***Index Terms*—** NIST 2021 Speaker Recognition Evaluation, ResNet, EfficientNet, Angular Prototypical loss.

1. INTRODUCTION

Since 1996, the US National Institute of Standards and Technology (NIST) has undertaken a series of speaker recognition assessments, the most recent of which is the 2021 Speaker Recognition Evaluation (SRE21) [1].

Our system is based on deep neural network training, embedding extraction and cosine scoring. We use well-known deep neural network architectures ResNet and EfficientNet, the networks are trained with combined angular prototypical and softmax loss, cosine similarity is used to generate the scores.

2. DATASET DESCRIPTION

For our experiments, we use data corpora following fixed training conditions of NIST 2021 Speaker Recognition Evaluation Plan. We use LDC2021E08, Vox2, and Vox1 for training our models. For evaluation, we use the test set of voxceleb 1.

Data Augmentation: We apply both online and offline data augmentation while training. Before training, we generate an augmented dataset from the original data corpora. We apply a simple permutation trick to generate more instances from low-frequency speakers [1].

The following strategy is used for permutation based augmentation:

1. Generate speaker list S with number of utterances less than k.
2. Select p utterances for each speaker s in S.
3. For each p_i:
   (a) Apply energy based VAD to make a list of speech segments from the utterance. We denote each segment as "word".
   (b) Run a random permutation to get a different order of words than the original.
   (c) Store the new utterance.

We also apply online data augmentation using MUSAN[2] [2] and RIR[3] [3] dataset. Here, MUSAN is a corpus of music, speech, and noise recordings and RIR contains simulated and real room impulse responses, isotropic and point-source noises. We use additive noise augmentation and reverberation for online augmentation, reverberation is typically represented by convolution of the audio signals with a noise impulse.

3. DEEP NEURAL NETWORK BASED SYSTEM

We focus on deep learning based speaker identification system. The complete training pipeline has been described below:

1. We use energy-based VAD to process each utterance and generate more utterances using word-level permutation.
2. We extract mel spectogram with 64 dimensional mel-filter banks.
3. We augment the DNN with a 512 dimensional embedding extractor fully connected layer before the classification layer.
4. We use the embedding layer output for angular prototypical loss [4] and the final fully connected layer output for softmax loss. Both losses are combined to optimize the DNN weights.

3.1. ResNet

Residual Networks are used frequently for image recognition tasks [5]. It is also frequently used for speaker recognition. Here, we use a variation of resnet-34 as our base model. The detailed architecture is shown in table 2. We use attentive statistics pooling before the speaker embedding layer [6, 7].

[1] https://github.com/zabir-nabil/audioperm

[2] https://www.openslr.org/17/

[3] https://www.openslr.org/28/

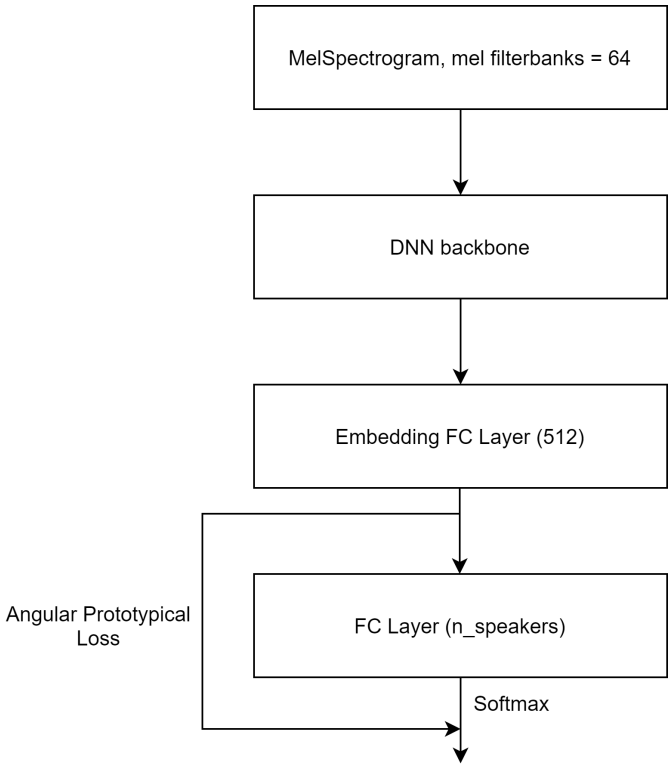

Fig. 1. DNN Training Pipeline

Table 1. Dataset Description

Dataset Identifier	Dataset	Usage
LDC2021E08	NIST SRE CTS Superset	Train
LDC2019S20	2016 NIST SRE Evaluation Set	Test
Vox2	Voxceleb 2	Train
Vox1	Voxceleb 1	Train
Vox1t	Voxceleb 1 test	Test
SRE21d	SRE21 dev set	Test

For trianing the model, we use angular prototypical loss from [4]. In angular prototypical loss, the centroids are made from exactly same number of utterances from the support set which is more aligned with the test setup. It also improves the robustness by adding scale invariance. We finally use a combined loss function which uses the vanilla softmax and the angular prototypical loss.

In our experiments, we trained ResNets with two different audio lengths (4.5 seconds and 2 seconds). Note that, ResNet models have 8028492 parameters in total. It takes 1.7 GB of GPU memory to load the model.

3.2. EfficientNet

EfficientNet is the current state of the art in image recognition. It has been explored in other domains such as speech [8, 9].

Table 2. ResNet architecture for speaker recognition

Layer	Kernel dim.	Filters	Stride	Feature shape (out)
input	-	-	-	L x 64 x 1
conv1	3x3	32	1x1	L x 64 x 32
residual 1	3x3	32	1x1	L x 64 x 32
residual 2	3x3	64	2x2	L/2 x 32 x 64
residual 3	3x3	128	2x2	L/4 x 16 x 128
residual 4	3x3	256	2x2	L/8 x 8 x 256
flatten				L/8 x 2048
AS pooling				4096
embedding (fc)				512
classification (fc)				n_speakers

We use the efficientnet-b0 model in a similar setup as resnet. We replace the resnet backbone with efficientnet-b0 and train with same loss function and pre-processing. We trained the EfficientNet with 4.5 seconds audio data.

3.3. System Detailes

For runnning our experiments, we use Intel(R) Xeon(R) Gold 5118 CPU with 187 GB physical ram and an Nvidia Tesla V100 SXM2 (32GB) GPU. It nearly takes 6 days of full training for the ResNet based models and 2 additional days for the EfficientNet based models to converge if only a single GPU is used.

ResNet models take 1.7 GB of the GPU space when loaded which have around 8 million parameters. We calculate the average number of audio samples processed per second on the voxceleb 1 test set. For the ResNet 4.5 s model, the model can extract embeddings from 52 samples per second, the second ResNet model can process 64 samples in the same time. For cosine matching, the calculated samples are 2200 and 2467 respectively. The numbers are rounded to nearest integer. The detailed result is shown in table 3.

4. FEATURE AND BACKEND

4.1. Voice Activity Detection (VAD)

For voice activity detection, we use a simple energy based silence removal scheme to remove the silence from the training audio. We use **pydub** [4] to remove the silence from the audio randomly with 0.5 probability. In 50% of the cases, we didn't remove the silence from the audio to further regularize the model.

4.2. Feature Extraction

For all of our sub-system, we use the same feature consistently. For our experiment, we extract melspectrogram with the following conditions. The detailed parameters are shown in table 4.

[4]https://github.com/jiaaro/pydub

Table 3. GPU memory and time comparison

Model	Memory	Parameters	Task	Samples / Sec
ResNet, 4.5 s	1.7 GB	8028492	Embedding Extraction	52 Hz
			Cosine Matching	2200 Hz
ResNet, 2 s	1.7 GB	8028492	Embedding Extraction	64 Hz
			Cosine Matching	2467 Hz

Table 4. MelSpectrogram feature parameters

Parameter	Value
Sample rate	16000
FFT	512
Window length	400
Hop length	160
Mel filterbanks	64
Window function	Hamming

4.3. Backend

We use cosine similarity without any score normalization or adaptation as our backend scoring method. From the enrollment audio or test utterances, 10 equally spaced time samples are sampled, which are then passed to the DNN after feature extraction, then we take 512-dimensional speaker embedding vectors, and run cosine similarity. We take the average cosine similarity as our score.

4.4. Facial verification

For facial verification in the audio-visual track, we use pretrained model from DeepFace [5]. For scoring we use cosine similarity.

4.5. Fusion

For fusing the scores of multiple models, we follow a very simple averaging strategy. First, we apply min-max normalization to all the model scores, and then we take the mean of those scores as the final score.

5. RESULTS

We have used NIST SRE CTS Superset and voxceleb corpus for training our models. For evaluation, we use 2016 NIST SRE Evaluation Set, SRE21 dev set and Voxceleb1 test set. As most of the speakers from our training corpus are from English speaking distribution, we have found that our model performs much better on a English test set than test set containing speakers speaking any other language. The results are reported in table 5.

[5]https://github.com/serengil/deepface

Table 5. Results

Data segment	Model	EER (%)	min DCF
Voxceleb 1 test	ResNet, 4.5 s	1.89	0.17
	ResNet, 2 s	1.96	0.23
	Fusion	1.85	0.21
	EfficientNet	2.82	0.32
SRE 16 eval	ResNet, 4.5 s	19.46	0.81
	ResNet, 2 s	21.53	0.78
SRE 21 dev	ResNet, 4.5 s	27.02	0.76
	ResNet, 2 s	29.31	0.83

6. CONCLUSION

In this work, we have worked on a deep learning based speaker recognition system. Deep Neural Network architectures ResNet and EfficientNet were used to build the backbone of the speaker recognition model. This shows a baseline result for our submission to NIST 2021 SRE. The system can be further improved by using LDA for dimensionality reduction, PLDA for scoring and score normalization, adaptation methods.

7. REFERENCES

[1] Omid Sadjadi, Craig Greenberg, Elliot Singer, Lisa Mason, Douglas Reynolds, et al., "Nist 2021 speaker recognition evaluation plan," 2021.

[2] David Snyder, Guoguo Chen, and Daniel Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.

[3] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L Seltzer, and Sanjeev Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[4] Joon Son Chung, Jaesung Huh, Seongkyu Mun, Minjae Lee, Hee Soo Heo, Soyeon Choe, Chiheon Ham, Sunghwan Jung, Bong-Jin Lee, and Icksang Han, "In defence of metric learning for speaker recognition," *arXiv preprint arXiv:2003.11982*, 2020.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[6] Koji Okabe, Takafumi Koshinaka, and Koichi Shinoda, "Attentive statistics pooling for deep speaker embedding," *arXiv preprint arXiv:1803.10963*, 2018.

[7] Hee Soo Heo, Bong-Jin Lee, Jaesung Huh, and Joon Son Chung, "Clova baseline system for the voxceleb speaker recognition challenge 2020," *arXiv preprint arXiv:2009.14153*, 2020.

[8] Qidong Lu, Yingying Li, Zhiliang Qin, Xiaowei Liu, and Yun Xie, "Speech recognition using efficientnet," in *Proceedings of the 2020 5th International Conference on Multimedia Systems and Signal Processing*, 2020, pp. 64–68.

[9] Mingxing Tan and Quoc Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.