INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Mixup Learning Strategies for Text-independent Speaker Verification

Yingke Zhu[1], *Tom Ko*[2*], *Brian Mak*[1]

[1]Department of Computer Science & Engineering
The Hong Kong University of Science & Technology
[2]Department of Computer Science and Engineering
South University of Science and Technology, Shenzhen, China
{yzhuav,mak}@cse.ust.hk, tomkocse@gmail.com

Abstract

Mixup is a learning strategy that constructs additional virtual training samples from existing training samples by linearly interpolating random pairs of them. It has been shown that mixup can help avoid data memorization and thus improve model generalization. This paper investigates the mixup learning strategy in training speaker-discriminative deep neural network (DNN) for better text-independent speaker verification.

In recent speaker verification systems, a DNN is usually trained to classify speakers in the training set. The DNN, at the same time, learns a low-dimensional embedding of speakers so that speaker embeddings can be generated for any speakers during evaluation. We adapted the mixup strategy to the speaker-discriminative DNN training procedure, and studied different mixup schemes, such as performing mixup on MFCC features or raw audio samples. The mixup learning strategy was evaluated on NIST SRE 2010, 2016 and SITW evaluation sets. Experimental results show consistent performance improvements both in terms of EER and DCF of up to 13% relative. We further find that mixup training also improves the DNN's speaker classification accuracy consistently without requiring any additional data sources.

Index Terms: speaker recognition, deep neural networks, mixup, x-vectors

1. Introduction

Speaker verification (SV) is the task of accepting or rejecting the identity claim of a speaker based on some given speech. Over the past decade, i-vector [1] based systems have been the dominant approach in speaker verification. A standard i-vector system consists of a universal background model (UBM), a large projection matrix T and a probabilistic linear discriminant analysis (PLDA) classifier [2]. I-vectors are extracted by projecting the statistics from a UBM to low-dimensional representations using the projection matrix. The PLDA classifier is used to compare pairs of i-vectors and make the decision.

In recent years, more powerful SV systems are built using deep learning. These SV systems usually consists of two components: a speaker-discriminative DNN [3, 4, 5, 6] to produce speaker embeddings, and a separately trained PLDA classifier to compare pairs of embeddings. The speaker-discriminative DNN is trained to classify speakers in the training set with a softmax output layer. After training, speaker embeddings of any speakers are extracted from an intermediate layer before the softmax layer. Since the speakers in the evaluation stage have never been seen during training, and these test speakers may utter under very different acoustic environments, how to enhance the generalization ability of the speaker-discriminative DNN is an important issue in order to improve the performance of an SV system.

Various techniques have been developed to improve the generalization ability of neural networks. One way is to introduce regularizers during the network training procedure. For example, batch normalization [7] normalizes outputs of hidden layers to address the internal covariate shift problem. Dropout [8] randomly drops units and their connections from a neural network during training to avoid over-fitting. Another choice is data augmentation. It constructs additional training samples using expert knowledge and extra data sources. Data augmentation can enrich the training data distribution and constantly improve the generalization ability [9] of the DNN. In [5], additive noises and reverberation [10] are employed to increase the diversity of training data, and lead to significant improvement of x-vector-based DNN systems.

Mixup [11] is a learning strategy that is recently proposed to improve the generalization of neural networks. It constructs additional virtual training samples by linearly interpolating random pairs of samples from the original training data and their labels. The interpolation weights follow some distribution, and beta distribution is commonly used. It is shown in [11] that mixup reduces the generalization error on a variety of machine learning tasks, such as image classification [12, 13], recognition of simple spoken commands [14], and tabular data classification [15]. In [16], mixup is also found to be effective in large-scale automatic speech recognition, especially when there is a mismatch between test data and training data.

In this paper, we investigate the mixup learning strategy in text-independent speaker verification tasks. We adapt the mixup strategy to the speaker-discriminative DNN training procedure of an x-vector system. Various mixup schemes and their effectiveness are studied.

2. Mixup training of speaker verification systems

2.1. Mixup training

Mixup [11] is a training strategy that aims at avoiding data memorization and improving the generalization ability of large neural networks without the need for additional training data. In supervised learning, we want to find a function f that maps input feature x to its output label y by minimizing the average of the loss function l over the joint data distribution $P(x, y)$:

$$R(f) = \int l(f(x), y) dP(x, y)\,. \tag{1}$$

* corresponding author

Copyright © 2019 ISCA

http://dx.doi.org/10.21437/Interspeech.2019-2250

This is known as expected risk. Since the data distribution P is usually unknown in practice, it is often empirically approximated from the training data set $D = \{(x_i, y_i)\}_{i=1}^{n}$ as follows:

$$P_\delta(x, y) = \frac{1}{n} \sum_{i=1}^{n} \delta(x = x_i, y = y_i) \,, \tag{2}$$

where $\delta(x = x_i, y = y_i)$ is a Dirac mass centered at (x_i, y_i). Another alternative is to use the following vicinal distribution:

$$P_v(\tilde{x}, \tilde{y}) = \frac{1}{n} \sum_{i=1}^{n} v(\tilde{x}, \tilde{y} | x_i, y_i) \,, \tag{3}$$

where v is a vicinity function that measures the probability of finding the virtual training sample $(\tilde{x}, \tilde{y})$ in the vicinity of the real training sample (x_i, y_i).

[11] proposes a generic vicinal distribution called mixup:

$$\mu(\tilde{x}, \tilde{y} | x_i, y_i) = \frac{1}{n} \sum_{j}^{n} \mathbb{E}_\lambda [\delta(\tilde{x} = \lambda x_i + (1-\lambda) x_j, \; \tilde{y} = \lambda y_i + (1-\lambda) y_j)] \,, \tag{4}$$

where $\lambda \sim Beta(\alpha, \alpha)$, for $\alpha > 0$.

In brief, sampling from the mixup distribution of Eq. (4) generates new samples (and labels) which are the result of interpolation between a pair of original training samples and their labels, respectively:

$$\tilde{x} = \lambda \, x_i + (1 - \lambda) \, x_j$$

$$\tilde{y} = \lambda \, y_i + (1 - \lambda) \, y_j$$

where x_i, x_j are input samples, and y_i, y_j are their one-hot label encodings.

On the one hand, mixup can be considered as a form of data augmentation that constructs virtual training samples through interpolation of randomly sampled data. On the other hand, mixup is also a regularization method that renders a neural network favoring simple linear interpolation behavior in-between training examples, and thus reduces data memorization.

2.2. The x-vector system with mixup

We employ the mixup training strategy on Kaldi's x-vector system [17]. The x-vector system consists of two modules. The frond-end is a deep neural network trained to classify speakers in the training set. The DNN training also produces an intermediate hidden layer from which speaker embeddings can be extracted. The back-end is a PLDA classifier [2] trained to compare pairs of speaker embeddings for SV.

We investigate two mixup schemes for training the speaker-discriminative neural network: raw audio sample mixup and feature mixup. In the raw audio sample mixup scheme, each virtual training sample is a linear combination of a pair of audio samples from the original training data. In the feature mixup scheme, the combination is conducted on MFCC features.

3. Experimental setup

3.1. Model configuration

The DNN used in the x-vector SV system is the baseline model defined in Kaldi's sre16/v2 recipe. The first five layers, l_1 to l_5, are constructed with a time-delay architecture that works at the frame level. Suppose t is the current time step; frames from $(t-2)$ to $(t+2)$ are spliced together in the input layer. The next two layers splice the output of the preceding layer at time steps $\{t-2, t, t+2\}$ and $\{t-3, t, t+3\}$, respectively. No temporal contexts are added to the fourth and the fifth layers. Thus, the total temporal context after the third layer is 15 frames. A statistics pooling layer aggregates over frame-level output vectors of the DNN, and computes their mean and standard deviation. The mean and standard deviation are concatenated together and forwarded to two additional hidden layers, l_6 and l_7, and finally to a softmax output layer. The DNN is trained to classify speakers in the training set.

After training, the softmax output layer and the last hidden layer are discarded, and speaker embeddings are extracted from the affine component of layer l_6.

The PLDA back-end is used for comparing pairs of speaker embeddings. The speaker embeddings are centered and then projected using LDA, which reduces their dimension from 512 to 150. After dimensionality reduction, the representations are length-normalized and modeled by PLDA. The PLDA scores are further normalized using adaptive s-norm [18].

The input acoustic features are 23-dimensional MFCCs extracted every 10ms from a window of 25ms, and are mean-normalized over a sliding window of up to 3 seconds. An energy-based VAD is employed to filter out non-speech frames from the utterances.

3.2. Training data

There are two sets of training data. The set of 8kHz training data from NIST evaluation consists of primarily English telephone speech (with a smaller amount of non-English and microphone speech) taken from Switchboard data sets and past NIST speaker recognition evaluations (SREs). The Switchboard portion consists of Switchboard 2 Phase 1/2/3 and Switchboard Cellular, and it contains about 28k recordings from 2.6k speakers. The SRE portion consists of NIST SRE data from 2004 to 2008 for a total of about 35k recordings from 3.8k speakers. The set of 16kHz training data for the Speaker In The Wild (SITW) [19] evaluation set consists of Voxceleb1 (excluding speakers that overlap with SITW data set) [20] and Voxceleb2 [21] with a total of 1,236,567 segments from 7,185 speakers.

The clean data, together with the virtual data generated by the mixup learning strategy are used to train the DNN system and speaker embeddings, but the PLDA classifier is trained only on the clean data.

3.3. Evaluation

System performance is assessed on NIST 2010, 2016 speaker recognition evaluations [22, 23] and SITW, which will be denoted as SRE10, SRE16 and SITW, respectively in the rest of this paper. SRE10 consists of English telephone speech and our evaluation is based on the extended core condition 5. SRE16 consists of Cantonese and Tagalog telephone speech. The length of enrollment segments is about 60 seconds, and the length of test segments varies from 10 to 60 seconds. SITW consists of unconstrained audio of English speakers from videos with naturally occurring noises, reverberation, as well as device and codec variations.

The performance is reported in terms of equal error rate (EER) in percentage as well as the official evaluation metric. For SRE10, the metric is the minimum of the normalized detection cost function (DCF) with $P_{Target} = 0.001$ [22]. For SRE16 [23], it is computed from a normalized DCF averaged over two operation points with $P_{Target} = 0.01$ and

$P_{Target} = 0.005$, respectively. For SITW, it is the minimum of the normalized DCF at $P_{Target} = 0.01$ and $P_{Target} = 0.001$.

3.4. Mixup details

The whole SV system is built using Kaldi. In its neural network training framework using multiple machines, stochastic gradient decent is separately applied with different randomized subsets of the training data. Neural network parameters are averaged across all the jobs periodically, and the averaged parameters are redistributed to the machines for further training. The data subset used in a single job is referred to as a training archive, which contains a certain number of samples for all the speakers in the training set.

3.4.1. MFCC feature mixup

In the MFCC feature mixup scheme, the samples are MFCCs with their corresponding speaker labels, and the mixup is conducted within each training archive. We randomly choose two training samples with different speaker labels to generate a new virtual training sample.

3.4.2. Raw audio sample mixup

In the raw audio sample mixup scheme, we randomly choose a pair of audio samples from different speakers to generate a new audio. To deal with the audio pairs with different lengths, we repeat the speech of the shorter one to match the length of the longer one. Mixup is then performed on the two length-aligned audios to generate the additional virtual audio for training.

4. Results

Table 1: *Results of MFCC feature mixup on SRE10*

System	EER	DCF10
Baseline	2.50	0.465
Uniform ($\alpha = 1$)	2.34	0.460
Beta ($\alpha = 0.5$)	2.27	0.491
Beta ($\alpha = 0.2$)	2.33	0.455
Beta ($\alpha = 0.1$)	2.44	0.454

Table 2: *Results of MFCC feature mixup on SRE16*

	Cantonese		Tagalog	
	EER	DCF16	EER	DCF16
Baseline	6.06	0.518	15.30	0.785
Uniform ($\alpha = 1$)	5.69	0.498	14.32	0.784
Beta ($\alpha = 0.5$)	5.94	0.519	14.25	0.782
Beta ($\alpha = 0.2$)	5.92	0.515	14.71	0.778
Beta ($\alpha = 0.1$)	6.12	0.508	14.75	0.785

4.1. Mixup on MFCC feature

We investigate mixup training with MFCC features on SRE10 and SRE16. Experimental results are summarized in Table 1 and Table 2. The baseline system is trained with data described in Section 3.2 without any data augmentation techniques. We try beta distribution with different α values. When $\alpha = 1$, we have the special case that the beta distribution becomes the uniform distribution.

On SRE10, when α increases from 0.1 to 0.5, EER drops while DCF10 increases. When the mixup weights are sampled from the uniform distribution, EER improves by 6%. For SRE16, the system performance is similar for beta distribution with different α values, except that EER on Tagalog improves 4-7% compared to the baseline. With uniform distribution, the system is 6% better in EER and 3% better in DCF16 on both Cantonese and Tagalog. In general, mixup training helps little when it is performed on MFCC features.

Table 3: *Results of raw audio mixup on SRE10*

System	EER	DCF10
Baseline	2.50	0.465
Uniform ($\alpha = 1$)	2.18	0.418

Table 4: *Results of raw audio mixup on SRE16*

	Cantonese		Tagalog	
	EER	DCF16	EER	DCF16
Baseline	6.06	0.518	15.30	0.785
Uniform ($\alpha = 1$)	5.45	0.491	14.30	0.764

Table 5: *Results of raw audio mixup on SITW*

	Dev			Eval		
	EER	DCF2	DCF3	EER	DCF2	DCF3
Baseline	3.93	0.372	0.551	4.265	0.397	0.604
Uniform ($\alpha = 1$)	3.73	0.358	0.546	4.073	0.383	0.591

4.2. Mixup on raw audio data

We investigate mixup training with raw audio data on SRE10, SRE16 and SITW data sets. The results are summarized in Table 3, 4 and 5. Only results from using uniform distribution for sampling the interpolation weights are shown as we find from preliminary results on SRE10 that it usually gives better performance than using other beta distribution. Moreover, the use of uniform distribution also has the advantage that no hyper-parameters tuning is required, and the system performance seems to be more stable.

In general, mixup training with raw audio data consistently outperforms the baseline on all test sets. On SRE10, compared with the baseline, it is 13% better in EER and 10% better in DCF10. For SRE16, it outperforms the baseline by 10% in EER and 5% in DCF16 on Cantonese, and 7% better in EER and 3% better in DCF16 on Tagalog. The improvement on SITW is mainly on EER and DCF2. On both the development and evaluation sets, audio mixup training achieves 5% improvement in EER anf 4% improvement in DCF2.

Figure 1 and 2 report the DET curves for systems evaluated on SRE10 and SRE16, respectively.

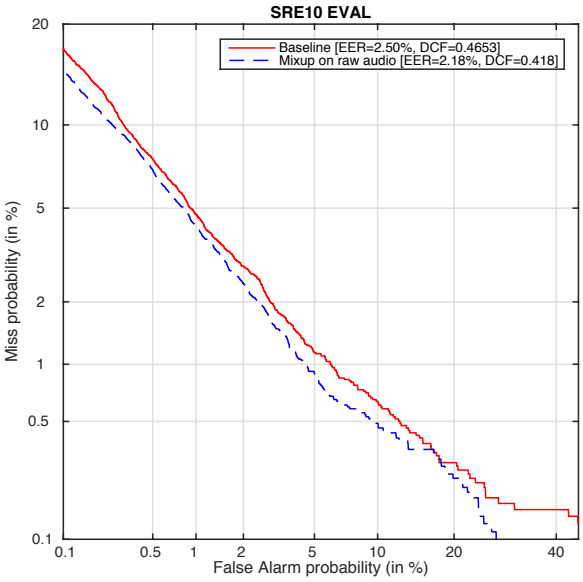

Figure 1: *DET curve for SRE10 results*

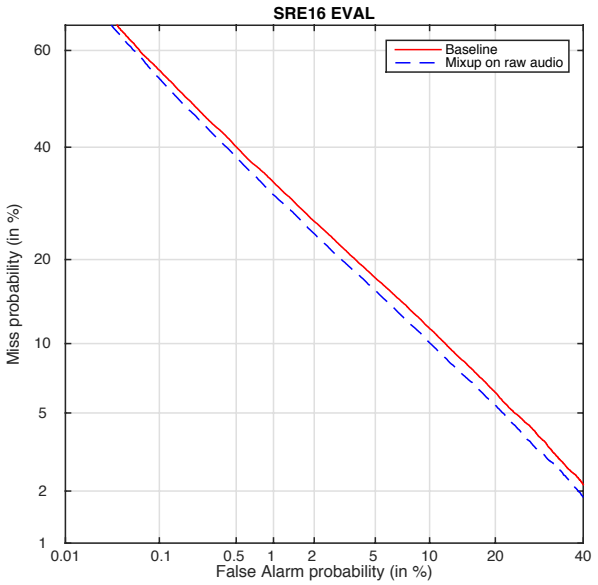

Figure 2: *DET curve for SRE16 when results are pooled across Cantonese and Tagalog.*

Table 6: *Speaker DNN classification accuracy on training and validation sets*

Data	System	Training Set	Validation Set
8kHz	Baseline	89.8%	78.5%
	Uniform ($\alpha = 1$)	94.2%	85.9%
16kHz	Baseline	94.9%	96.8%
	Uniform ($\alpha = 1$)	96.4%	97.8%

4.3. Classification accuracy of the speaker DNN

Mixup training is employed in training the speaker-discriminative DNN, and it is supposed to improve the generalization of the DNN as well. To verify the effectiveness of mixup training, we further check the DNN performance on classifying the speakers in the training and validation sets. The results are shown in Table 6, where audio sample mixup training is employed.

We can see that mixup training with weights sampled from the uniform distribution can consistently improve the classification accuracy on both training and validation speakers. For the DNN trained on 8kHz data, mixup training improves the accuracy by 5% absolute on the training set and 7% absolute on the validation set. In other words, it reduces the error rate by 43% and 34% relative, respectively. Moreover, it reduces the performance gap between the training and validation sets from 11.3% to 8.3% absolute. For the DNN trained on 16kHz data, we can see that the baseline already achieves high accuracy on both data sets. Nevertheless, mixup training can still reduce the classification errors by about 30% relative on both training and validation data sets. The results are consistent with speaker verification results presented in Section 4.2 when mixup training is performed on raw audio data.

5. Conclusion

We investigate the mixup learning strategy in training a speaker-discriminative DNN for text-independent speaker verification. Mixup training is employed in training two systems with data sampled at 8kHz and 16kHz, respectively. The 8kHz speaker verification system is evaluated on SRE10 and SRE16, and the performance of the 16kHz system is reported on the SITW evaluation set.

We find that mixup training on MFCC features does not bring consistent and significant improvement, especially on DCF, while mixup training with raw audio data can greatly improve the SV performance on both EER and DCF. Besides speaker verification, we also investigate the DNN performance in terms of speaker classification accuracy. The classification results are consistent with the findings in verification tasks: mixup training with raw audio data can help improve the classification accuracy and reduce the performance gap between the training set and validation set. We also find the relative error reductions are comparable in both 8kHz system and 16kHz system, though the reduction in the latter is a bit smaller. That is reasonable since the 16kHz DNN already has very high speaker classification accuracy as the data (sampled at a higher sampling rate) contain richer speaker information.

The computational cost of creating additional virtual mixup training samples from raw audio training data is low when compared to the computational cost of the whole system. Yet it brings significant improvement without using any extra data sources. It can also be used together with other existing data augmentation techniques to further improve the robustness and generalization ability of the large DNN. In the future, we plan to further explore other mixup schemes as well as other interpolation weight distributions.

6. Acknowledgements

The work described in this paper was partially supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. HKUST16200118 and HKUST16215816).

7. References

[1] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[2] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*. Springer, 2006, pp. 531–542.

[3] E. Variani, X. Lei, E. McDermott, I. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2014, pp. 4052–4056.

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

[5] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2018.

[6] Y. Zhu, T. Ko, D. Snyder, B. Mak, and D. Povey, "Self-attentive speaker embeddings for text-independent speaker verification," in *Proceedings of Interspeech*, 2018, pp. 3573–3577.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[9] P. Y. Simard, Y. A. LeCun, J. S. Denker, and B. Victorri, "Transformation invariance in pattern recognitiontangent distance and tangent propagation," in *Neural networks: tricks of the trade*. Springer, 1998, pp. 239–274.

[10] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. 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.

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," *ICLR*, 2017.

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[13] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[14] P. Warden, "Launching the speech commands dataset," https://ai.googleblog.com/2017/08/launching-speech-commands-dataset.html, 2017.

[15] "UCI machine learning repository," https://archive.ics.uci.edu/ml/index.php.

[16] I. Medennikov, Y. Khokhlov, A. Romanenko, D. Popov, N. Tomashenko, I. Sorokin, and A. Zatvornitskiy, "An investigation of mixup training strategies for acoustic models in ASR," in *Proceedings of Interspeech*, 2018.

[17] 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 *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*, 2011.

[18] D. E. Sturim and D. A. Reynolds, "Speaker adaptive cohort selection for Tnorm in text-independent speaker verification," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, 2005, pp. I–741.

[19] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The 2016 speakers in the wild speaker recognition evaluation." in *Proceedings of Interspeech*, 2016, pp. 823–827.

[20] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Proceedings of Interspeech*, 2017.

[21] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Proceedings of Interspeech*, 2018.

[22] "The NIST year 2010 speaker recognition evaluation plan," https://www.nist.gov/itl/iad/mig/speaker-recognition-evaluation-2010, 2010.

[23] "NIST speaker recognition evaluation 2016," https://www.nist.gov/itl/iad/mig/speaker-recognition-evaluation-2016, 2016.