Odyssey 2020 The Speaker and Language Recognition Workshop
1-5 November 2020, Tokyo, Japan

Orthogonality Regularizations for End-to-End Speaker Verification

Yingke Zhu, Brian Mak

Department of Computer Science & Engineering
The Hong Kong University of Science & Technology
{yzhuav,mak}@cse.ust.hk

Abstract

This paper seeks to explore orthogonal training in end-to-end speaker verification (SV) tasks. In various end-to-end speaker verification systems, cosine similarity has been used as the distance measurement for speaker embeddings. However, the effectiveness of cosine similarity is based on the assumption that the dimensions of the speaker embeddings are orthogonal. In our previous orthogonal training work, we have shown that in SV systems with cosine similarity backend, introducing orthogonality on the weights in speaker-discriminative deep neural networks can significantly improve the system performance. In this paper, we introduce two orthogonality regularizers to end-to-end speaker verification systems. The first one is based on the Frobenius norm, and the second one utilizes restricted isometry property. Both regularization methods can be handily incorporated into end-to-end training. We build systems based on the state-of-the-art end-to-end models. Two network architectures, LSTM and TDNN, are used in order to investigate the effects of orthogonality regularization on different types of models. Systems are assessed on the Voxceleb corpus and significant gains are obtained with our new regularized orthogonal training.

1. Introduction

Speaker verification (SV) is the process of verifying whether an utterance belongs to the claimed speaker, based on some enrolled reference utterances. The typical speaker verification process has three stages: training, enrollment and evaluation. The training stage aims to learn a low-dimensional embedding rich in speaker information, and a scoring function for computing similarity between embeddings. The enrollment stage is to estimate a speaker model for every new speaker using a limited amount of his/her utterances. Finally the evaluation stage is to score unknown utterances against estimated speaker models. The unknown utterance is considered to be produced by the claimed speaker if the evaluation score is above a pre-defined threshold, but is rejected otherwise. Speaker verification can be categorized in to text-dependent and text-independent. The text-dependent SV system requires the same set of phrases for enrollment and test, while the text-independent system has no constraints on the spoken content for enrollment and test.

Hybrid systems have been the dominant solutions to SV tasks over the years. They usually consist of multiple modules. The speaker embeddings can be learned from i-vectors [1], hybrids of i-vector and deep neural networks (DNNs) [2, 3, 4, 5], or purely speaker discriminative neural networks [6, 7, 8]. The scoring function is commonly built with a probabilistic linear discriminant analysis (PLDA) classifier [9].

In these hybrid systems, each module is optimized with respect to its own target function, which is usually not the same as the final system metrics. Moreover, speaker verification is an open-set classification problem. It needs to handle unknown speakers that are not contained in the training dataset, so the generalization ability of the system is important. Recently more and more SV systems are trained in an end-to-end fashion. Compared to the hybrid system, end-to-end system simplified the training pipeline and is optimized in a more consistent manner. Besides, it enables learning a task-specific metric during the training stage. In SV tasks, the metric is usually aimed to minimize intra-speaker variations and maximize inter-speaker differences, which can help the system generalize to unseen speakers in the verification stage.

End-to-end training has been widely used in text-dependent SV tasks. [10] presented an integrated approach that maps a test utterance and a few enrollment utterances directly to a single score. It used long short-term memory recurrent neural networks (LSTMs) or DNNs to learn the speaker embeddings, cosine similarity to measure the distant between embeddings and a simple logistic regression layer to produce the final output. In both training and verification stages, one test utterance together with N enrollment utterances were fed to the network as an input sample. The network first produced speaker embeddings for all the utterances, and estimated a target speaker model by averaging over the enrollment embeddings. Then cosine similarity was computed between the test utterance and the target speaker model. Finally the logistic regression layer took this similarity score as input and output a score indicating the probability the the test utterance is spoken by the target speaker. [11] improved the work in [10] by adapting the attention mechanism in the target speaker model estimation step. [12] used a similar framework, but convolutional neural networks (CNNs) were used to learn speaker embeddings. Besides, it incorporated phonetic information as additional input features. [13, 14, 15] extended the end-to-end training to text-independent tasks. In [13, 14], deep CNNs and LSTMs were used to learn speaker embeddings, and triplet loss [16, 17] was adopted to optimize the whole system. The training of triplet loss based system requires non-trivial triplet sampling schemes. [15] proposed a generalized end-to-end loss based on [10]. It made the training more efficient and did not require any special training examples selection.

In the above mentioned end-to-end systems, all the loss functions try to minimize the distance between embeddings from the same speaker and maximize the distance between embeddings from different speakers. Most of them use the simple cosine distance [10, 11, 12, 15]: the cosine of the angle between two embedding vectors measures the distance between two embeddings. The major underlying assumption for the effectiveness of the cosine similarity measure is that the embeddings space is orthogonal. In our previous work, we have shown that introducing orthogonality of weights in speaker discriminative neural networks can significantly improve the effectiveness of cosine similarity measurements in hybrid SV systems.

10.21437/Odyssey.2020-3

In this work, orthogonal training is further explored in end-to-end SV systems. In addition, we propose two orthogonality regularizers: the first one is based on the Frobenius norm, and it requires the Gram matrix to be close to the identity. The second one utilizes restricted isometry property and aims to minimize the largest singular value of the Gram matrix.

The proposed regularizers are evaluated on different types of end-to-end SV systems: LSTM-based and time-delay neural network (TDNN)-based, building from frameworks in [15]. System performance is assessed on Voxceleb corpus. Experimental results show that orthogonality regularizations can significantly improve the performance. Besides, they can be directly incorporated into end-to-end training loss without much extra work.

The rest of this paper is organized as follows. Section 2 introduces the end-to-end loss and two neural network structures used in this work. Section 3 introduces orthogonal training and two orthogonality regularizers. We discuss the experiments in Section 4 and analyze the results in Section 5. Section 6 concludes the paper.

2. End-to-end System

Generalized end-to-end (GE2E) loss for speaker verification is proposed in [15]. It greatly improves system performance, and also makes the training of end-to-end SV systems more efficient. In this work we built two systems trained with GE2E loss, one used LSTM network for speaker representation leaning, and the other used TDNN instead.

2.1. GE2E loss

In our system, one mini-batch consists of $N \times M$ utterance, meaning N different speakers and M utterances from each speaker. Let $\mathbf{x}_{ij}$ ($1 \leq i \leq N$, $1 \leq j \leq M$) represents the acoustic feature vector computed from the whole utterance j of speaker i. Each input $\mathbf{x}_{ij}$ is fed to the neural network, and the network produces a corresponding embedding vector $\mathbf{e}_{ij}$. The centroid of the embedding vectors $\mathbf{c}_i$ from the i-th speaker [$\mathbf{e}_{i1}$, $\cdots$, $\mathbf{e}_{iM}$] is defined as

$$\mathbf{c}_i = \frac{1}{M}\sum_{j=1}^{M}\mathbf{e}_{ij}\ . \tag{1}$$

The similarity matrix $\mathbf{S}_{ij,k}$ is defined as the scaled cosine distances between each embedding vector $\mathbf{e}_{ij}$ to all centroids $\mathbf{c}_k$ ($1 \leq i,k \leq N$, $1 \leq j \leq M$):

$$\mathbf{S}_{ij,k} = w \cdot \cos(\mathbf{e}_{ij}, \mathbf{c}_k) + b, \tag{2}$$

where w and b are trainable parameters. The weight w is constrained to be positive so that the similarity will be larger when cosine distance is larger.

During the training, we want each utterance embedding $\mathbf{e}_{ij}$ to be close to its own speaker's centroid $\mathbf{c}_i$, while far away from other speakers' centroids $\mathbf{c}_k, k \neq i$. Here we put a softmax on $\mathbf{S}_{ij,k}$ for $k = 1, \cdots, N$, we want the output equals to 1 only when $i = k$, and 0 otherwise. Thus the loss on each $\mathbf{e}_{ij}$ is defined as:

$$L(\mathbf{e}_{ij}) = -\mathbf{S}_{ij,i} + \log\sum_{k=1}^{N}\exp(\mathbf{S}_{ij,k}). \tag{3}$$

The final GE2E loss L_G is the summation of losses over all embedding vectors:

$$L_G = \sum_{i,j} L(\mathbf{e}_{ij}). \tag{4}$$

In practice, it is found that when computing the similarity $\mathbf{S}_{ij,i}$ between embedding $\mathbf{e}_{ij}$ and its own speaker centroid $\mathbf{c}_i$, it's better to remove $\mathbf{e}_{ij}$ when computing the centroid $\mathbf{c}_i$, so Eq. 2 is actually implemented as:

$$\mathbf{S}_{ij,k} = \begin{cases} w \cdot \cos(\mathbf{e}_{ij}, \mathbf{c}_i^{(-j)}) + b, & \text{if } k = i \\ w \cdot \cos(\mathbf{e}_{ij}, \mathbf{c}_k) + b, & \text{otherwise} \end{cases} \tag{5}$$

$$\mathbf{c}_i^{(-j)} = \frac{1}{M-1}\sum_{m\neq j}^{M}\mathbf{e}_{im}, \tag{6}$$

2.2. Neural network structure

We explored two types of neural networks in this work. The LSTM-based system uses a 3-layer LSTM with projection [18]. Each LSTM layer has 768 hidden nodes and the projection size is 256. After processing the input utterance, only the last frame output from LSTM is kept as the summary of the whole utterance.

The TDNN-based system uses the same TDNN structure as in Kaldi's x-vector model [7]. The configuration is depicted in Table 1, where T represents the length of the input speech segments.

Table 1: The architecture of TDNN.

Layer	Layer Context	Total Context	Input × Ouput
frame1	[t-2, t+2]	5	200×512
frame2	{t-2, t, t+2}	9	1536×512
frame3	{t-3, t, t+3}	15	1536×512
frame4	{t}	15	512×512
frame5	{t}	15	512×1500
stats pooling	[0,T)	T	1500T×3000
segment6	{0}	T	3000×512
segment7	{0}	T	512×256

The embedding vector is computed as the L2 normalization of the network output.

3. Orthogonality Regularizations

Imposing orthogonality on linear transformations has been widely explored in deep convolutional neural networks (CNNs) and deep recurrent neural networks (RNNs). An important property of orthogonal matrices is that they preserve gradient norm during back-propagation. Therefore encouraging orthogonality on linear transformations can help solve the gradient vanishing or exploding problem when training very deep neural networks or networks with long term dependencies. [19, 20] also show that orthogonal weights can stabilize the distribution of layer activations in CNNs, and make optimization more efficient. Various work have explored gradient stability, convergence and training speed of orthogonal training/initialization techniques [21, 22, 23, 24].

Orthogonality regularizations have been applied in many classification problems [22, 24, 25, 26] and obtained encouraging improvements. In our previous work, we investigated or-

thogonal training in the hybrid SV system with a cosine similarity backend. We found that enforcing orthogonality on the weight matrices in the speaker-discriminative neural network greatly improve the system performance. The reason is that it makes the speaker embedding space orthogonal and thus improves the effectiveness of the cosine similarity backend.

In this work, we explore the orthogonality regularization in end-to-end SV systems. We introduce two regularizers in this section, which can be applied to any fully-connected layers.

3.1. Soft orthogonality regularization

Suppose a fully connected layer has weight matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$. One straightforward way to impose orthogonality on matrices can be done via a soft constraint of the form:

$$\lambda \parallel \mathbf{W}^T\mathbf{W} - \mathbf{I} \parallel^2, \tag{7}$$

where λ is the regularization coefficient and $\parallel \cdot \parallel$ represents standard Frobenius norm. Soft orthogonality regularization requires the Gram matrix of $\mathbf{W}$ to be close to identity. The gradient of soft orthogonal regularization term w.r.t the weight $\mathbf{W}$ can be computed in a stable form. So this regularization term can be directly added to the original loss function and optimized together.

3.2. Spectral restricted isometry property regularization

The restricted isometry property (RIP) [27] characterizes matrices which are nearly orthonormal and is widely used in the field of compressed sensing. [25] takes an extreme case of RIP and makes a new criterion that enforces matrix to be close to orthogonal. The new criterion can be formulated as:

$$\left| \frac{\parallel \mathbf{Wz} \parallel^2}{\parallel \mathbf{z} \parallel^2} - 1 \right| < \delta_{\mathbf{W}}, \forall \mathbf{z} \in \mathbb{R}^n, \tag{8}$$

where $\delta_{\mathbf{W}} \in (0, 1)$ is a small constant, and $\mathbf{z} \in \mathbb{R}^n$ is an n-dimensional vector.

In order to enforce the orthogonality of $\mathbf{W}$, we wish to minimize the constant $\delta_{\mathbf{W}}$ in Eq.8, which equals to minimize the supremum of $\left| \frac{\|\mathbf{Wz}\|^2}{\|\mathbf{z}\|^2} - 1 \right|$. Since the spectral norm of $\mathbf{W}$ is defined as:

$$\sigma(\mathbf{W}) = \sup_{\mathbf{z} \in \mathbb{R}^n, \mathbf{z} \neq \mathbf{0}} \frac{\parallel \mathbf{Wz} \parallel^2}{\parallel \mathbf{z} \parallel^2}, \tag{9}$$

where $\sigma(\cdot)$ represent the spectral norm and $\sup(\cdot)$ means the supremum of a set, the spectral norm of $\mathbf{W}^T\mathbf{W} - \mathbf{I}$ can then be written as:

$$\sigma(\mathbf{W}^T\mathbf{W} - I) = \sup_{\mathbf{z} \in \mathbb{R}^n, \mathbf{z} \neq \mathbf{0}} \left| \frac{\parallel \mathbf{Wz} \parallel^2}{\parallel \mathbf{z} \parallel^2} - 1 \right|. \tag{10}$$

Therefore, the criterion finally equals to minimize the spectral norm of $\mathbf{W}^T\mathbf{W} - \mathbf{I}$.

The spectral restricted isometry property regularization is formulated as:

$$\lambda \cdot \sigma(\mathbf{W}^T\mathbf{W} - \mathbf{I}), \tag{11}$$

where λ is the regularization coefficient. It requires all singular values of $\mathbf{W}$ to be close to 1.

Since the spectral norm of a matrix equals to its largest singular value, computing regularization term Eq.11 requires eigenvalue decomposition, and resulting in non-stable gradients. Power iteration is thus used to approximate the spectral norm computation [28, 25]:

$$\begin{aligned} \mathbf{u} &\leftarrow (\mathbf{W}^T\mathbf{W} - \mathbf{I})\mathbf{v} \\ \mathbf{v} &\leftarrow (\mathbf{W}^T\mathbf{W} - \mathbf{I})\mathbf{u} \\ \sigma(\mathbf{W}^T\mathbf{W} - \mathbf{I}) &\leftarrow \frac{\parallel \mathbf{v} \parallel}{\parallel \mathbf{u} \parallel}. \end{aligned} \tag{12}$$

In our experiments, we randomly initialize the vector $\mathbf{v} \in \mathbb{R}^n$ and the above iterative procedure is performed two times.

3.3. Regularization coefficient schedule

The choice of regularization coefficient plays an important role in the training process as well as final system performance. We investigate two schedules:

- Constant: keep a constant coefficient ($\lambda = 0.1$) throughout the training stage.
- Decreasing: start with $\lambda = 0.2$, and gradually reduce λ to 0.

For the LSTM-based systems, the maximum number of training epochs is set to 100. The λ is reduced to 1e-2, 1e-4, 1e-6 after 20, 40, 60 and 80 epochs, respectively, and it is set to 0 after 80 epochs. For the TDNN-based systems, the maximum number of training epochs is set to 50. The λ is reduced to 1e-2, 1e-4, 1e-6 after 10, 20, 30 and 40 epochs, respectively, and it is set to 0 after 40 epochs.

4. Experiments

4.1. Training data

The training data consists of 133,778 utterance from 1,211 celebrities from the training set of Voxceleb1[29], which are extracted from YouTube videos. Each person in the corpus has 18 videos on average. Each of these videos has been split into approximately 123 short speech utterances of an average duration of 8.2 seconds. The four data augmentation techniques described in [7], namely, babble, music, noise, and reverb are applied to increase the amount of training data and to improve the robustness of the system. The clean data, together with the augmented data are used as the training set, and 10% of them are held out for validation during training.

The input acoustic features are 40-dimensional filterbank coefficients with a frame length of 25ms that 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.

4.2. Evaluation

System performance is assessed on the Voxceleb1 evaluation set. The test set consists of 4715 utterance from 40 speakers. System performance is reported in terms of equal error rate (EER) as well as the normalized detection cost function (DCF) at $P_{Target} = 0.01$ (DCF2) and $P_{Target} = 0.001$ (DCF3).

4.3. Training scheme

In all the experiments we use $N = 64$ speakers and $M = 8$ segments per speaker in each batch. We also tried with $M = 10$ segments per speaker, and the performance is similar to the system trained with $M = 8$. We finally choose $M = 8$ mainly due to the concern of our GPU memory capacity. To generate one batch, we first randomly sample 64 speakers, then sample

8 segments for each speaker from all his/her utterances. The segment length are randomly sampled from [140, 180] for every batch, following the settings in [15].

The network is trained with SGD with initial learning rate of 0.01, and the rate is decreased by half if the loss on validation set increases for 2 epochs. The training will be terminated if the validation loss does not decrease for 6 epochs. The weight decay is set to 1e-5 and the momentum is set to 0.8. The L2-norm of a gradient is clipped at 10 [30]. Dropout is also used and its probability [31] is set to 0.8 when training LSTM networks.

In this work, we apply the orthogonality constraints on the weight matrix of speaker embedding producing layer; i.e., the last projection layer in LSTM-based systems, and layer *segment7* in TDNN-based systems. Since the bias will disrupt the orthogonality, we use a linear layer without bias for speaker embedding producing layer.

5. Results

The overall results on two types of end-to-end systems are summarized in Table 2 and Table 3. Regularizer '*None*' means no orthogonality regularization is employed during training, i.e., a baseline system. *SO* represents the use of soft orthogonality regularization and *SRIP* represents the use of spectral restricted isometry property regularization introduced in Section 3.1 and Section 3.2, respectively. Two regularization coefficient schedules, *Constant* and *Decreasing*, are discussed in Section 3.3.

Table 2: *Results of orthogonality regularization with LSTM-based systems on Voxceleb.*

Regularizer	λ schedule	EER(%)	DCF2	DCF3
None	None	2.35	0.311	0.428
SO	Constant	1.99	0.289	0.482
SO	Decreasing	1.98	0.279	0.427
SRIP	Constant	2.17	0.275	0.434
SRIP	Decreasing	1.89	0.249	0.346

Table 3: *Results of orthogonality regularization with TDNN-based systems on Voxceleb.*

Regularizer	λ schedule	EER(%)	DCF2	DCF3
None	None	2.12	0.245	0.408
SO	Constant	2.09	0.279	0.425
SO	Decreasing	1.83	0.221	0.342
SRIP	Constant	1.95	0.230	0.390
SRIP	Decreasing	1.85	0.229	0.333

5.1. Results on LSTM-based systems

Table 2 compares the performance of various LSTM-based systems. To summarize, SRIP regularization outperforms SO as well as the baseline system without any regularization with remarkable performance gains; i.e., 20% improvements in EER, DCF2 and DCF3 compared to the baseline. SO regularization improves the system performance in terms of EER and DCF2 under both λ schedules, but no gain is obtained in DCF3. The best system with SO regularization obtained 16% improvement in EER, 10% improvement in DCF2 and similar performance in DCF3 compared to the baseline system. Besides, the decreasing schedule of λ performs better than constant schedule for both regularizers. The systems trained with constant λ schedule have worse performance in terms of DCF3, and the decreasing schedule is effective under all the 3 evaluation metrics.

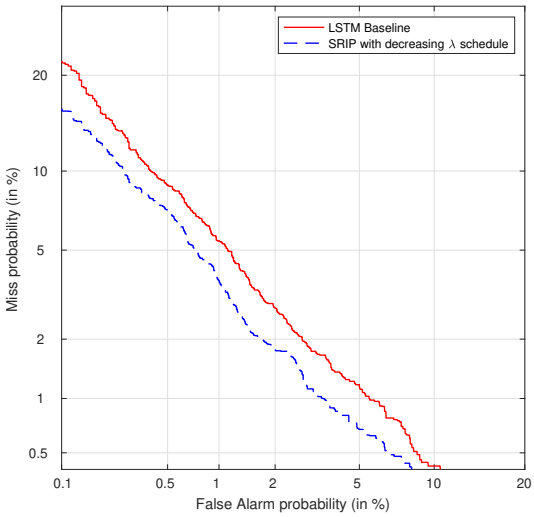

Figure 1: DET curves for LSTM-based systems.

We plot DET curves for the baseline and the best LSTM-based system trained with SRIP regularization and decreasing λ schedule in Figure 1.

5.2. Results on TDNN-based systems

Table 3 compares the performance of various TDNN-based systems. The SRIP regularization gives consistent improvement when trained with different λ schedules. Even when λ was kept constant throughout the training stage, it still achieves 8%, 6% and 4% improvement in EER, DCF2 and DCF3, respectively. The best system is obtained with SRIP regularization and decreasing λ schedule, and it is 12% better in EER and 18% better in DCF3 when compared to the baseline.

Unlike its performance in LSTM-based systems, SO regularization can be beneficial to TDNN-based systems. When trained with decreasing λ schedule, the system is 14% better in EER, 9% better in DCF2 and 16% better in DCF3 than the baseline.

We plot the DET curves for the baseline and two of the best TDNN-based systems trained with 2 regularizers and decreasing λ schedule in Figure 2.

The performance of SRIP regularizer trained under different schedules on TDNN-based systems is consistent with that on LSTM-based systems. In most cases, SRIP regularizer provides improvements and it is a better choice than SO. For the regularization coefficient training schedules, gradually reducing λ in the training stage leads to better system performance in all cases.

5.3. System performance comparison

Performance of our systems and other recent studies using the VoxCeleb dataset are compared in Table 4. Systems in the first part of the table use only development set of Voxceleb for training. [33] adopts attentive pooling into the X-vector framework and acheives the best EER of 3.85%. [35] proposes an end-to-end speaker verification system that starts directly from raw waveforms. The system comprises two DNNs, one for speaker embedding extraction and the other for back-end classification,

Table 4: *System performance comparison on VoxCeleb.*

	Training Data	Architecture	EER(%)
Nagrani *et al.*[29]	Voxceleb	VGG-M	7.8
Hajibabaei *et al.*[32]	Voxceleb	ResNet-20 + Additive margin loss	4.3
Okabe *et al.*[33]	Voxceleb	TDNN + Attentive pooling	3.85
Shon *et al.*[34]	Voxceleb	1D-CNN	5.9
Jung *et al.*[35]	Voxceleb	RawNet + Center loss + Between-class variation loss	4.0
Heo *et al.*[36]	Voxceleb	ResNet-34 + Between-class variation loss + End-to-end loss	5.5
Chung *et al.*[37]	Voxceleb2	ResNet-50	3.95
Heo *et al.*[36]	Voxceleb2	ResNet-34 + Between-class variation loss + End-to-end loss	2.66
Zeinali *et al.*[38]	Voxceleb2	ResNet-160 + Additive angular margin loss	1.31
Kaldi recipe[39]	Voxceleb + Voxceleb2	X-vector	3.13
Ours	Voxceleb	TDNN + GE2E	2.12
Ours	Voxceleb	TDNN + GE2E + Orthogonality regularizer	1.83

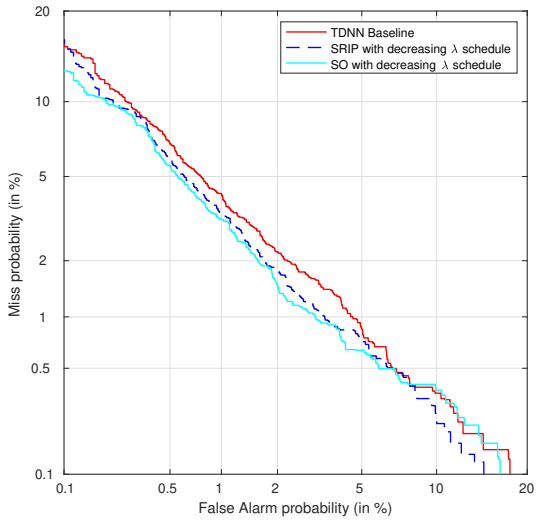

Figure 2: DET curves for TDNN-based systems.

it achieves an EER of 4.0%. [36] proposes two end-to-end loss functions, one is to conduct hard negative mining, and the other is designed to increase the inter-speaker variation. The inter-speaker variation loss is the sum of non-diagonal elements of the Gram matrix of last hidden layer weight matrix. It is similar to the soft orthogonality regularizer in this paper. The main difference is that, the inter-speaker variation loss only requires the weight vectors to be orthogonal to each other, while the soft orthogonality regularizer has an additional constraint that the Frobenius norm of each weight vector should be close to 1. The additional constraint provides normalization on weight vectors, it can avoid the norm of certain weight vectors become too large when the training data has some level of class imbalance, and thus improve the effectiveness of cosine similarity.

Systems in the second part of the table are trained on larger training datasets, Voxceleb2 or its combination of Voxceleb. The standard X-vector framework in Kaldi acheives an EER of 3.13%. [38] proposes various advanced architectures and loss functions, and the EER of its best single system is 1.31%.

Our baseline systems with GE2E loss have surpassed all the previous systems trained on Voxceleb dataset, with an EER of 2.12%. By introducing orthogonality regularization terms, the EER is further reduced to 1.83%.

5.4. Effect of orthogonality regularizations

To explore the effect of orthogonality regularization during training, we plot the validation loss curves in Figure 3 and Figure 4. The validation loss here is the GE2E loss, indicating the discriminability of the system.

Figure 3 shows the validation loss curves for all the LSTM-based systems. Notice that the actual number of training epochs is different for different systems. As described in Section 4.3, it is because we set the maximum number of training epochs to 100, and stop training if the validation loss does not decrease for 6 consecutive epochs. From the validation loss curves we can see that all the regularizers accelerate the training process in the early training stage, and maintain at lower loss throughout training compared to the baseline. In general, SRIP regularization achieves lower validation loss than SO regularization, and this finding is consistent with their system performance. Besides, for both regularizers, training with a constant λ leads to more training epochs and lower final loss. However, according to the results reported in Table 2 and 3, training with a decreasing λ schedule always results in better performance. One possible reason is that at the final training stage, changes of model parameters are very small and it's more like a fine tuning phase. Keeping the same regularization strength throughout training might be overly strict at this stage. By decreasing the coefficient λ, we loosen the orthogonality constraint and allow the model parameters having more flexibility in the final stage, thus leading to better system performance.

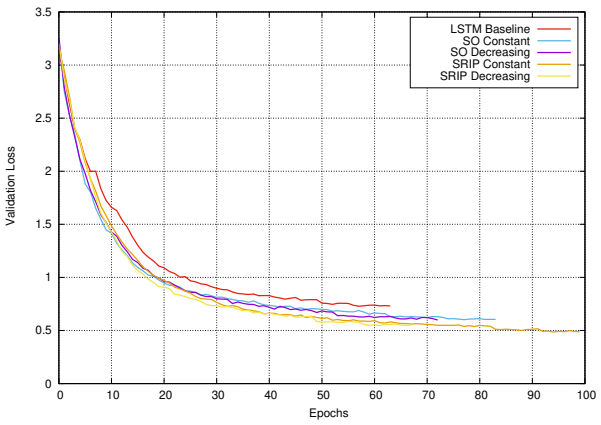

Figure 3: Validation loss curves during training of LSTM-based systems.

Figure 3 shows the validation loss curves for all the TDNN-based systems. Compared to the LSTM-based models, TDNN-based systems converge much faster and can achieve lower loss. Systems trained with orthogonality regularization have lower validation loss in the final training stage, but the gap is much smaller than that in LSTM-based systems. In general, from the validation loss curves, we do not see clear differences or characteristics between regularizers and their training schemes. We conjecture that we should train the TDNN-based system with more batches in each epoch. In the current training scheme, we randomly sample 2500 batches in every epoch and use them to train the systems. Since one batch has 64 speakers and 8 segments from each speaker, we have around 1 million segments trained in one epoch, and the segment length ranges from 1.4 to 1.8 seconds. Compared to the standard x-vector training scheme, which takes 2-5 million segments with lengths ranging from 2-4 seconds to train one epoch, the amount of data used in training our TDNN-based systems is much smaller.

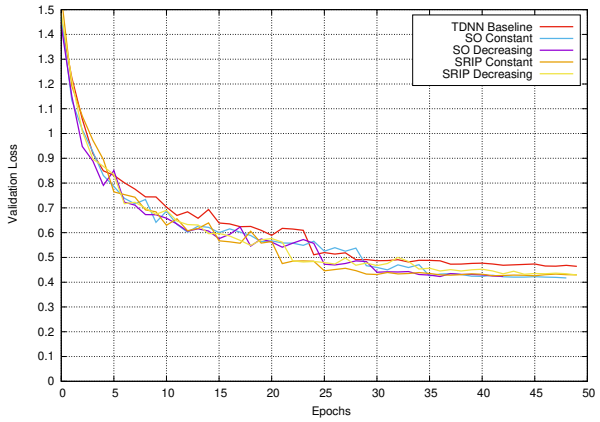

Figure 4: Validation loss curves during training of TDNN-based systems.

6. Conclusion

We introduced two orthogonality regularizers in training end-to-end text-independent speaker verification systems. The first one is based on the Frobenius norm. It requires the Gram matrix to be close to the identity. The second regularizer aims to minimize the largest singular value of the Gram matrix based on the restricted isometry property. We tried different regularization coefficient training schedules and investigated their effect on the training stage as well as the evaluation performance. SV systems in this work are all built with the generalized end-to-end loss. Two different neural network architectures, LSTM and TDNN, were investigated in order to explore the effectiveness of orthogonality regularization under different model structures. All the systems were evaluated on the Voxceleb corpus. We find that spectral restricted isometry property regularization performs best in all the cases, and achieves, in the best case, 20% improvement under all the criteria. Moreover, both regularizers can be easily combined into the original training loss and optimized together with little computation overhead.

In our current experiments, we only incorporated orthogonality regularization on the weight matrix of the speaker embedding producing layer. It is worth trying with more layers. In the future, we will investigate other orthogonality regularization methods and other model architectures.

7. 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).

8. References

[1] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[2] Patrick Kenny, Themos Stafylakis, Pierre Ouellet, Vishwa Gupta, and Md Jahangir Alam, "Deep neural networks for extracting Baum-Welch statistics for speaker recognition.," in *Odyssey*, 2014, vol. 2014, pp. 293–298.

[3] Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1695–1699.

[4] Daniel Garcia-Romero, Xiaohui Zhang, Alan McCree, and Daniel Povey, "Improving speaker recognition performance in the domain adaptation challenge using deep neural networks," in *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2014, pp. 378–383.

[5] David Snyder, Daniel Garcia-Romero, and Daniel Povey, "Time delay deep neural network-based universal background models for speaker recognition," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 92–97.

[6] E. Variani, X. Lei, E. McDermott, I. Moreno, and Javier 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.

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

[8] Yingke Zhu, Tom Ko, David Snyder, Brian Mak, and Daniel Povey, "Self-attentive speaker embeddings for text-independent speaker verification," in *Proceedings of Interspeech*, 2018, pp. 3573–3577.

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

[10] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer, "End-to-end text-dependent speaker verification," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2016, pp. 5115–5119.

[11] FA Rezaur rahman Chowdhury, Quan Wang, Ignacio Lopez Moreno, and Li Wan, "Attention-based models for text-dependent speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5359–5363.

[12] Shi-Xiong Zhang, Zhuo Chen, Yong Zhao, Jinyu Li, and Yifan Gong, "End-to-end attention based text-dependent speaker verification," in *Proceedings of the IEEE Workshop on Spoken Language Technology*, 2016, pp. 171–178.

[13] Hervé Bredin, "TristouNet: triplet loss for speaker turn embedding," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 5430–5434.

[14] Chunlei Zhang and Kazuhito Koishida, "End-to-end text-independent speaker verification with triplet loss on short utterances," in *Interspeech*, 2017, pp. 1487–1491.

[15] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno, "Generalized end-to-end loss for speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4879–4883.

[16] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[17] Elad Hoffer and Nir Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015, pp. 84–92.

[18] Hasim Sak, Andrew W Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," 2014.

[19] Pau Rodríguez, Jordi Gonzalez, Guillem Cucurull, Josep M Gonfaus, and Xavier Roca, "Regularizing CNNs with locally constrained decorrelations," *arXiv preprint arXiv:1611.01967*, 2016.

[20] Guillaume Desjardins, Karen Simonyan, Razvan Pascanu, et al., "Natural neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2071–2079.

[21] Eugene Vorontsov, Chiheb Trabelsi, Samuel Kadoury, and Chris Pal, "On orthogonality and learning recurrent networks with long term dependencies," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3570–3578.

[22] Mehrtash Harandi and Basura Fernando, "Generalized backpropagation, Étude de cas: Orthogonality," *arXiv preprint arXiv:1611.05927*, 2016.

[23] Kui Jia, Dacheng Tao, Shenghua Gao, and Xiangmin Xu, "Improving training of deep neural networks via singular value bounding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4344–4352.

[24] Lei Huang, Xianglong Liu, Bo Lang, Adams Wei Yu, Yongliang Wang, and Bo Li, "Orthogonal weight normalization: Solution to optimization over multiple dependent Stiefel manifolds in deep neural networks," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[25] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang, "Can we gain more from orthogonality regularizations in training deep CNNs," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 4266–4276.

[26] Di Xie, Jiang Xiong, and Shiliang Pu, "All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6176–6185.

[27] Emmanuel J Candes and Terence Tao, "Decoding by linear programming," *IEEE transactions on information theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[28] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[29] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[30] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio, "Understanding the exploding gradient problem," *CoRR, abs/1211.5063*, vol. 2, pp. 417, 2012.

[31] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan 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.

[32] Mahdi Hajibabaei and Dengxin Dai, "Unified hypersphere embedding for speaker recognition," *arXiv preprint arXiv:1807.08312*, 2018.

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

[34] Suwon Shon, Hao Tang, and James Glass, "Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 1007–1013.

[35] Jee-weon Jung, Hee-Soo Heo, Ju-ho Kim, Hye-jin Shim, and Ha-Jin Yu, "RawNet: Advanced end-to-end deep neural network using raw waveforms for text-independent speaker verification," *arXiv preprint arXiv:1904.08104*, 2019.

[36] Hee-Soo Heo, Jee-weon Jung, IL-Ho Yang, Sung-Hyun Yoon, Hye-jin Shim, and Ha-Jin Yu, "End-to-end losses based on speaker basis vectors and all-speaker hard negative mining for speaker verification," *arXiv preprint arXiv:1902.02455*, 2019.

[37] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[38] Hossein Zeinali, Shuai Wang, Anna Silnova, Pavel Matvejka, and Oldvrich Plchot, "BUT system description to Voxceleb speaker recognition challenge 2019," *arXiv preprint arXiv:1910.12592*, 2019.

[39] "Kaldi Voxceleb recipe," `https://github.com/kaldi-asr/kaldi/tree/master/egs/voxceleb/v2`.