Speaker embeddings by modeling channel-wise correlations

Themos Stafylakis\textsuperscript{1}, Johan Rohdin\textsuperscript{1,2}, Lukáš Burget\textsuperscript{2}

\textsuperscript{1}Omilia - Conversational Intelligence, Athens, Greece
\textsuperscript{2}Brno University of Technology, Faculty of Information Technology, Speech@FIT, Czechia

{tstafylakis}@omilia.com, {rohdin, burget}@fit.vutbr.cz

Abstract

Speaker embeddings extracted with deep 2D convolutional neural networks are typically modeled as projections of first and second order statistics of channel-frequency pairs onto a linear layer, using either average or attentive pooling along the time axis. In this paper we examine an alternative pooling method, where pairwise correlations between channels for given frequencies are used as statistics. The method is inspired by style-transfer methods in computer vision, where the style of an image, modeled by the matrix of channel-wise correlations, is transferred to another image, in order to produce a new image having the style of the first and the content of the second. By drawing analogies between image style and speaker characteristics, and between image content and phonetic sequence, we explore the use of such channel-wise correlations features to train a ResNet architecture in an end-to-end fashion. Our experiments on VoxCeleb demonstrate the effectiveness of the proposed pooling method in speaker recognition.

Index Terms: speaker recognition, style-transfer, deep learning

1. Introduction

Extracting low-dimensional representations from speech utterances that characterize the speaker is a challenging task with numerous application. Apart from speaker recognition, speaker representations are used in automatic speech recognition (e.g. for speaker adaptation), in speaker diarization and separation, and in speech synthesis, for mimicking a target speaker using text-to-speech or voice conversion.

Until recently, i-vectors were the most commonly used means for representing speaker characteristics [1, 2]. With the advent of deep learning, neural architectures such as time-delay neural networks (TDNNs) [3], Long Short-Term Memory networks (LSTMs), 2D convolutional neural networks (such as ResNets and its extensions [4]) and Transformers have emerged, surpassing the performance of i-vectors in most applications and speaker recognition settings.

In order to obtain fixed-size representation vectors from utterances of variable duration and arbitrary word sequences (i.e. text-independent), a pooling method is required. The most common pooling method is a simple aggregation over the time axis, where the mean vector is concatenated with standard deviation (known as statistics pooling layer). The speaker embedding (a vector of few hundred dimensions) is obtained by projecting this pooled representation (a vector of few thousand dimensions) onto a linear layer or a shallow feed-forward neural network. Other pooling approaches have been proposed, such as single and multi-head attention, as well as the use of higher-order statistics [5, 6, 7].

In this paper we present an alternative way of modeling speaker information, that is largely inspired by style-transfer [8]. This seminal algorithm in computer vision is capable of producing new images that combine the content of an image with the style of other images or artworks. The style of an image is modeled by the correlations between the channels of the convolutional neural network. By considering styles as broadly analogous to speakers, we experiment with modeling speakers by channel-wise correlation. This is in contrast to averaging along the time dimension of each channel frequency pair of the output tensor that is used in statistics pooling. The other sources of variability that may also be considered as style of the recording, such as noise, channel, and emotion, can be suppressed by training the model end-to-end using supervised learning with speakers as targets. The resulting embeddings are highly speaker-discriminative, yielding state-of-the-art results on VoxCeleb using plain cosine similarity scoring.

The rest of the paper is organized as follows. In Sect. 2 we briefly review some of the related work, such as pooling methods and style-transfer. In Sect. 3 we introduce the proposed method, and provide all the implementation and architectural details. In Sect. 4, we present our experiments on a standard speaker recognition benchmark (VoxCeleb). Finally, in Sect. 5 we provide conclusions and future work directions.

2. Related work

2.1. Style-transfer in computer vision and voice conversion

A few years ago, the seminal work of L.A. Gatys et al. [8] in neural style-transfer gained the attention of the computer vision community and the industry, and received wide media coverage. Since then, several works extended their method, some of which can be found in [9]. A theoretical interpretation of the way style and image texture is encoded in the channel-wise correlations (which can be considered as a Gram-Matrix) is given in [10]. The authors treat style-transfer as a domain adaptation problem, and prove the equivalence between trying to match the Gram-Matrices of feature maps and minimizing the Maximum Mean Discrepancy (MMD) with the second-order polynomial kernel.

One of the few attempts to apply neural-style transfer in speech synthesis is described in [11]. The authors employ a pretrained CNN and apply style-transfer to train a speech generation and voice conversion system. Moreover, they demonstrate the capacity of the intermediate tensors to encode speaker characteristics using the VCTK dataset. However, the scope of their work is different to ours, the network is not trained end-to-end while the representations are high-dimensional. Furthermore, their method is not tested on a speaker recognition benchmark containing noise and high intrinsic and extrinsic within-speaker variability (such as VoxCeleb or NIST-SRE). Hence, the ability of their utterance-level representation to suppress session variability has not been examined.
The speaker embeddings are extracted by projecting the (vectorized) pooling layer onto a lower dimensional space (e.g. \(d_{e} = 256\)) using a single linear layer.

3.3. Style modeling in images

The style of an image in computer vision for a given convolutional layer is modeled by the symmetric positive-semidefinite matrix \(S\) with elements defined as

\[
S_{f,c,c'} = \frac{1}{TF} \sum_{t} Y_{t,f,c} Y_{t,f,c'}
\]

Note though that images differ from spectral representations of acoustic signals. Image characteristics are considered invariant along both spatial axes, while spectral representations are invariant only along the time axis. In fact, this property explains the use of frequency-channel pairs in statistics pooling instead of channels alone, as shown in (1)-(2). Due to this property of spectral representations, pooling along both time and frequency axes as in (3) is expected to yield less discriminative speaker representations.

3.4. Pooling via frequency-dependent channel-wise correlations

A natural modification of (3) that addresses the above issue is to pool only along the time axis, i.e.

\[
S_{f,c,c'} = \frac{1}{F} \sum_{t} Y_{t,f,c} Y_{t,f,c'}
\]

The 3D tensor \(S\) can be viewed as a list of \(F\) symmetric and positive semidefinite matrices \(S_{f,c}\) and has therefore \(FC(C+1)/2\) free variables. As its size can be too large we propose to reduce it in a frequency-dependent manner as shown later.

A further proposed modification is the use of mean and variance normalization along the time axis of \(Y\) for each frequency-channel pair, i.e. \(Z_{f,c} = \frac{Y_{f,c}}{\sqrt{\sum_{t} Y_{t,f,c}^2}}\). When variance normalization is applied, the variables in the diagonal of \(S_{f,c}\) (i.e. the variance of each channel-frequency combination) become equal to 1, and therefore they can be removed from the pooling layer resulting in \(C(C+1)/2\) free variables for each frequency \(f\) and \(FC(C+1)/2\) overall. Moreover, \(S_{f,c,c'} \in [-1,1]\) for all \(f,c,c'\). The pooling layer is followed by a linear layer which reduces the size of the representation to \(d_{e} = 256\) (the speaker embedding space), as in the standard statistics pooling.

Note that when mean and variance normalization are applied, the pooled representation encodes complementary information to that of the standard pooling; the means of each channel-frequency are being subtracted, while their variance is also set equal to one.

3.5. Additional extensions and modifications

In the pooling method described above, we add the following operations in order to decrease the size of the pooling layer and regularize the network. Recall that regularization is needed in order to extract embeddings that generalize well to speakers unseen during training.

3.5.1. Channel dropout

Dropout is a standard choice for regularizing a network. We employ it in a channel-wise fashion, that is, we sample a 1D mask...
of size $C$ (with dropout probability $p_{d,t} = 0.25$) and we broadcast it along the time and frequency axes. Channel-wise dropout is chosen because of its Bayesian interpretation as model averaging between models with variable number of convolutional filters [19].

### 3.5.2. Frequency ranges

As neighboring frequency bins in $Y$ have similar statistics, we consider merging them into $F_r$ frequency ranges. After experimentation, we concluded that merging $f_r = 2$ consecutive frequency bins in a non-overlapping manner yields more stable performance. Merging is implemented by reshaping $Y$, so that $F_r \leftarrow F_r f_r^{-1}$ and $T_r \leftarrow T f_r$. For example, when experimenting with $f_r = 2$, we obtain $F_r = 5$ and $T_r = 100$ given that $F = 10$ and $T = 50$. Note that by setting $f_r = F$ we recover the original style-transfer used in computer vision, where pooling is performed along both spatial axes.

### 3.5.3. Frequency-dependent channel reduction

The number of channels in $Y$ is $C = 256$ which would result in $C(C - 1)/2 \approx 33K$ variables per frequency bin. So we transform $Y$ as follows

$$Y_{t,f,c} \leftarrow \sum_c L_{f,c,c'} Y_{t,f,c'}$$

where $L$ is a learnable tensor of shape $(F_r, C, C')$, acting on the channel axis and reducing its size from $C$ to $C'$ (we choose $C' = 64$). The rationale for using a 3D kernel instead of a 2D is that different linear combinations of channels should be better suited for obtaining speaker discriminative information from each frequency range.

### 3.6. Summary and order of operations

We summarise here the set of operations and their order we use in order to extract speaker embeddings. Recall that the output tensor of the last ResNet block is $Y$ with shape $(T, F, C)$.

1. **Dropout** We apply channel-wise dropout with $p_{d,t}$ by sampling a binary mask of shape $C$ and broadcasting it along the other two axes.

2. **Frequency ranges** We reshape the tensor $Y$ with new shape $(T_r, F_r, C')$, by merging $f_r$ consecutive frequency bins into $F_r$ frequency ranges.

3. **Channel Reduction** We reduce the channel size by applying $L$ as shown in (5), i.e. $Y \leftarrow \sum_c L_{f,c,c'} Y_{t,f,c'}$.

4. **Normalization** We apply mean and variance normalization along the time axis, i.e.

$$Y \leftarrow \underbrace{Y}_{norm}$$

5. **Pooling** We apply pooling as shown in (4), but using $Z$ instead of $Y$, by which we obtain $S$, a 3D tensor with shape $(F_r, C', C')$.

6. **Flattening** We flatten the pooled representation to a vector of size $d_s = F_r C'(C_r - 1)/2$, which corresponds to the number of free variables in $S$.

7. **Embedding** We reduce the size of the pooled representation vector to $d_e$ using a linear layer. The resulting vector is the speaker embedding.

### 4. Experiments

#### 4.1. VoxCeleb 1 & 2 datasets

We evaluate the systems on the three VoxCeleb test sets, namely the original VoxCeleb 1 test set (composed of only 40 speakers) [20] and the (much larger and representative) Extended and Hard sets [14]. As training set, we use a subset of the VoxCeleb 2 development set which we augmented with babble, noise, reverberation, and music using Kaldi [21, 3]. The training set contains 7570 speakers and approximately 4.9M utterances including the augmented versions. As a held-out set we used 455 utterances (plus augmentations) from 90 training speakers.

#### 4.2. Architecture, training, and loss function

The backbone of the network is a 34-layer ResNet. All convolutional kernels are $3 \times 3$, the strides are $(1, 2, 2, 2)$ for the 4 blocks in both time and frequency axes, while the number of channels is $(64, 128, 256, 256)$ (the first convolutional layer also outputs 64 channels). The number of convolutional layers per block is $(3, 4, 6, 3)$. Finally, Squeeze and Excitation layers (with reduction ratio $r = 4$) are added only to the first 2 blocks of the ResNet [22, 23].

The network is trained using multi-speaker classification. As optimization criterion we employ the Additive Angular Margin (AAM) loss, which has shown notable improvements over the plain Cross Entropy loss [24]. We set the scale coefficient of the loss equal to 30, while for the margin coefficient we adopt a curriculum learning approach and progressively increase it as (0.1, 0.2, 0.3) during training [25].

As optimizer we use stochastic gradient descent with momentum equal to 0.9. The minibatch size is 256, however to train the model with a single GPU we split the minibatch into 16 “microbatches” of 16 examples each and use gradient accumulation. The initial learning rate (LR) is equal to 0.2 which we divide by 2 when the loss does not improve for more than 3000 model updates in the held-out set (the final LR is 0.2/64). The experiments are conducted using TensorFlow [26].

We emphasize that the model architecture, the training method, and its hyperparameters are optimized with the baseline, where mean and std features are used for pooling. Moreover, we did not observe any differences in time required per epoch between the baseline and the proposed architectures.

#### 4.3. Experimental Results

To evaluate our method we experiment we examine several configurations and ablations. The results we obtained are given in Table 1, and are obtained using cosine-similarity, without score or any other normalization. For the two baseline experiments we train and evaluate a ResNet-34 with the standard temporal pooling. The first includes mean and std features while the second only std (suggested in [18]). We emphasize that for the baseline experiments the tensor operations are not included, i.e. mean and std pooling is performed directly on the output of the last ResNet block.

The first experiment using a variant of the proposed pooling (P1) is essentially a direct implementation of the pooling as suggested in image style-transfer (recall that pooling with $F_r = 1$ is equivalent to (3), i.e. to pooling along both time and frequency axes.). Two such networks are examined, one with mean and a second one with mean and variance normalization, denoted by P1 and P5, respectively. Note that when mean-only normalization is applied, the number of free-parameters is $F_r C_r (C_r + 1)/2$, i.e. the size of the pooled representa-
ranges) may attain this goal more effectively and naturally. via modeling correlations between different representations to move such undesired sources of variability. Removing them as features to the diverse factors of within-speaker variability. statistical quantities, should be more resilient cally normalized relevant information from utterances. Correlations, as aggregation being the natural means of extracting speaker-sentations. This finding challenges the common view of frame relation matrices results in more speaker-discriminative repre-

5. Conclusions and research directions

In this paper, we introduced a new way of pooling information extracted from a CNN for modeling speaker information. The approach is inspired by neural style-transfer in computer vision, where style is modeled by a Gram-Matrix of channel-wise dot-products. We proposed a way to adapt this method to speech signals, e.g. by calculating several such Gram-Matrices, one per frequency range. We further demonstrated that transforming them to correlation matrices increases the speaker-discriminability of the extracted representations. Our experiments on VoxCeleb indicate that the proposed method yields notable improvements over the standard statistics pooling.

There are several ways by which our approach can be extended. For estimating the correlations between channels, the dot-product can be replaced by kernel functions, some of which have been examined in image style-transfer [9]. The property of Gram-Matrix being a symmetric positive semi-definite matrix can be explored in several ways, using e.g. statistical divergences (e.g. Kullback-Leibler, Jensen-Shannon, a.o.). Alternative training/finetuning schemes (e.g. prototypical, triplet, contrastive and self-supervised losses [25, 27, 28, 29]) that do not require a classification head are probably better suited for such kind of experimentation. Finally, the whole chain of transforms we apply on the tensor $\mathbf{Y}$ is not a result of exhaustive experimentation, leaving room for further improvements.

6. Acknowledgements

The work was supported by Czech National Science Foundation (GACR) project “NEUREM” No. 19-26934X, and European Union’s Horizon 2020 project No. 833635 ROXANNE. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 101007666 / ESPERANTO / H2020-MSCA-RISE-2020. The authors do not see any significant ethical or privacy concerns that would prevent the processing of the data used in the study. The datasets do contain personal data, and these are processed in compliance with the GDPR and national law.
7. References


