SINGING STYLE INVESTIGATION BY RESIDUAL SIAMESE CONVOLUTIONAL NEURAL NETWORKS

Cheng-i Wang

University of California, San Diego
Department of Music
La Jolla, CA, USA

George Tzanetakis

University of Victoria
Electrical and Computer Engineering
Victoria, BC, Canada

ABSTRACT

Investigating singing style is a difficult problem as individual styles are intertwined with melodies from different songs. In this paper, a methodology to investigate singing style is proposed. The proposed approach utilizes convolutional neural networks in a siamese architecture. In addition, we investigate variants of the networks to improve the audio feature extraction process. The potential of the proposed method for analyzing singing style is demonstrated using experiments on pop music singing recordings. The results indicate that the use of the proposed method is indeed effective in learning audio features that are relevant for characterizing singing style.

Index Terms— music information retrieval, convolutional neural network, singing voice, siamese neural network

1. INTRODUCTION

Singing is the most natural activity that allows human to express themselves musically. There are general terms people use to describe the styles or characteristics of a person's singing, such as adenoidal, breathy, operatic, soft, rough, etc. In this paper, a trainable algorithm capable of capturing singing style or singing characteristics is proposed. The proposed algorithm is a deep neural network that uses ResNeXt convolutional blocks [1] to process spectral inputs, followed by a feed-forward attention layer [2] handling temporal dependencies and fully-connected dense layers to learn the non-linear embeddings. Since there is not enough labeled data describing the singing styles of singing recordings, a proxy problem is defined to tackle this problem. The proxy problem is to learn an embedding space that recordings sang by the same singers are close to each other while the those sang by different singers are far away from each other. The assumption behind the proxy problem is that the singing style is relatively more consistent across performances by one singer than those by different singers. After such embedding space is learned, the embedded singing recordings that are closer together should ideally possess common singing characteristics. Also the embedding task is more general than singer identification in that there are too many singers in the real world for classification to be practical. For training the proposed neural networks and to evaluate its results, a newly published dataset extending the DAMP [3] dataset named DAMP-balanced[1] is used. The DAMP-balanced dataset has 24874 unaccompanied solo singing recordings and provides "unbiased" splits of the dataset such that singers included in subsets from the splits all sang the same collection of songs. Details of the DAMP-balanced dataset will be described later.

Previous works related to singer style investigation are singer identification researches. Traditional approaches focus on melody enhancement and background music reduction so that the analysis could focus on the sang melodies [4, 5, 6, 7]. Features used for speech such as LPC coefficients or MFCCs [4, 5, 6, 7] are used as the main features for training the classifiers. Gaussian mixture models are the favorite classifiers used to identify singer's identity [4, 5, 6, 7]. There are two major differences between the pervious works and the work done in this paper. The first one is that all of them had to reduce the impact from background music which introduced error whether it is during the singing voice detection stage or singer classification stage. Since the recordings in the DAMP-balanced dataset have only singing voices, and in most cases only negligible background leaks, the work done in this paper focuses on the singing voice itself. The second difference is that all of the mentioned previous work did not address the problem of "song" or "song collection" effect except for [5] where $4 \sim 6$ performances from 13 singers are collected by asking them to sing the same collection of melodies. On the contrary, the DAMP-balanced dataset provides train/validation/test splits that all singers in each split sang the same collection of songs.

2. SINGING PERFORMANCE EMBEDDING

To learn an embedding space that places performances by the same singer close to each other while pushes those by different singers away from each other, a siamese network architecture is used to learn such embedding [9, 2]. The inner networks of the siamese networks start with convolutional

[1] https://ccrma.stanford.edu/damp/

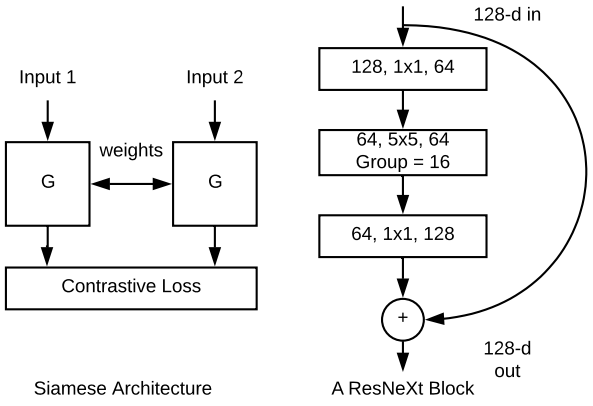

Fig. 1: (Left) A visualization of the siamese neural network architecture. (Right) A visualization of the ResNeXt convolution block. The numbers in boxes represent (# of input channels, kernel size, # of output channels.)

layers, followed by a feed-forward attention layer, then fully connected layers with a linear output layer at the end.

2.1. Siamese Architecture

The siamese network architecture ties two identical neural networks, with shared weights, by a loss function. A depiction of the siamese architecture can be seen in figure 1. A siamese network learns the embedding by minimizing the contrastive loss [9, 2]. Contrastive loss is defined over the distance between the outputs from the two identical inner neural networks. In this paper, squared euclidean distance between a pair of outputs from the siamese networks G given a pair of inputs x_1, $x_2 \in \mathbb{R}^D$ is used. Denote the squared euclidean distance as $D(x_1, x_2) = \|G(x_1) - G(x_2)\|_2^2$ and y a binary label that equals 1 when x_1, x_2 have the same identity and equals 0 when their identities differ, the contrastive loss takes the form

$$\mathcal{L}(y, x_1, x_2) = \frac{1}{2} yD + \frac{1}{2}(1-y)\max\{0, m - D\}.$$

By examining $\mathcal{L}$, it could be concluded that reducing $\mathcal{L}$ will have the effect of encouraging samples with the same identity be close to each other while the ones with different identities be pushed away from each other. m is the target margin between theembedded vectors having different identities and $m = 1$ for all the experiments.

2.2. ResNeXt Convolutional Block

The specific convolutional operation used to process input samples is the ResNeXt configuration [1]. A visualization of the ResNeXt convolutional block can be seen in figure 1. The ResNeXt configuration differs from a vanilla convolutional operation in two aspects: 1) a skip connection skipping 2 vanilla convolutional layers, or skipping 3 layers with the bottleneck design. 2) grouped convolution in the middle convolutional layers. The skipped connection allows a smoother gradient back-propagation from the loss function to the 1st layer in the networks. The grouped convolution generalizes the partitioned convolution proposed in [10] and allows the number of parameters to be reduced while keeping the learning effective.

2.3. Feed-Forward Attention

The feed-forward attention was proposed in [2] to aggregate over time axis in neural networks without the training burden brought by recurrent neural networks, and shown to improving the siamese neural network training [2] on spectral inputs. The feed-forward attention is defined as follows: Given the input matrix $X \in \mathbb{R}^{N \times D}$ representing N frames of D dimensional feature vectors, a weight vector $\sigma \in \mathbb{R}^N$ over the time frames is calculated via

$$\sigma = \mathrm{softmax}(f(Xw + b))$$

where $\mathrm{softmax}(x_m) = \frac{e^{x_m}}{\sum_{n=1}^{N} e^{x_n}}$ and f is a non-linear function ($tanh$ for the experiments done in this paper). $w \in \mathbb{R}^D$ and $b \in \mathbb{R}$ are the learnable parameters, which can be learned by back-propagation. The output $\hat{X}$ of the feed-forward attention layer is calculated by

$$\hat{X} = \sum_{n=1}^{N} \sigma_n X_n$$

and $\hat{X}$ can be considered as a weighted average of X by weights σ, determined by the learnable parameters w and b.

3. EXPERIMENTS

3.1. Datasets

The DAMP-balanced dataset contains 24874 solo singing recordings from 5429 singers singing a collection of 14 songs. For this paper, a subset from the dataset splitting the 14 songs into $6/4/4$ train/validation/test sets having $276/88/224$ performances sang by $46/22/56$ singers are used. Each singer in the train, validation and test set, sang each of the $6/4/4$ songs once respectively, thus making the collections of performances "balanced" with respect to song collections. Details of the split and list of songs could be found in `https://ccrma.stanford.edu/damp/`.

3.2. Feature Extraction

Two spectral features are used and compared in this experiment, constant-Q transformed spectrogram (CQT) and Mel-scaled spectrogram (Mel-spectrogram). The frequency axis of a CQT has equal number of bins per octave, so that the

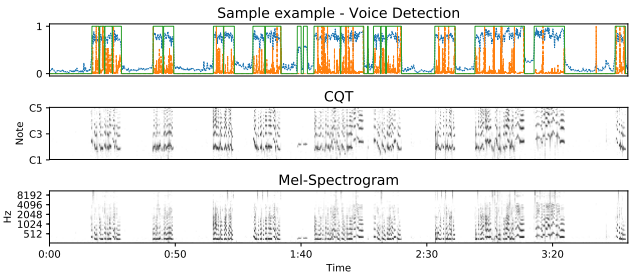

Fig. 2: (Top) Voice detection results, 1 being voiced and 0 otherwise. Green line is the voice detection, blue line is RMSE, and orange line is the voice confidence score. (Middle) CQT. (Bottom) Mel-spectrogram.

distances between different harmonics are the same between different fundamental frequencies. Mel-spectrogram scales its frequency axis according to human perception of pitch height. For both CQT and Mel-spectrogram, raw audio signals are re-sampled to 22050Hz. Spectrograms are obtained by STFT with a 2048 window length, 512 hop size, and a Hanning window. Both CQT and Mel-spectrogram has 96 bins. For CQT, each octave has 24 bins to capture sharp/flat pitches. The extracted spectrograms are squared to obtain the power spectrogram which are then scaled into decibels (dB). The values below -60dB are clipped to be zero and the whole power spectrogram is offset to be between 0 and 60. Plots of sampled CQT and Mel-spectrogram could be seen in the second and third row in figure 2.

3.3. Voice Detection

A voice detection is done on the singing recordings to extract regions that contain singing activities to ensure that the neural networks learn singing styles instead of voice detection. A K-means clustering with $K = 2$ is used as the voicing detector. The inputs are the root mean squared energy (RMSE) from the waveform and voicing confidence values. The voicing confidence value is the maximum value of the normalized autocorrelation at each analysis frame. The frames assigned to the class having higher RMSE and voicing confidence values are considered voiced. A sampled voice detection is shown in figure 2 on the top row with sampled RMSE and voicing confidence values as well. The CQT and Mel-spectrogram of each recording are chopped into overlapping matrices each of which has a duration of 6 seconds (256 time steps) and 20% hop size. The chopped samples having more than half the frames clustered as voiced from the voice detection are used in the experiment.

3.4. Neural Networks Training

The gradient descent of the contrastive loss is optimized by ADAM [11] with a learning rate 0.0001 and a batch size of 32. A drop out of 10% is applied at the last fully connected dense layers. L_2 weight regularizations with a weight $1e-6$ are applied on all the learnable weights in the neural network. The above hyper parameters are chosen by the Bayesian optimization package SPEARMINT [12]. An early stopping test on the validation set is applied every 50 epochs. The patience for early stopping is 1000 epochs with at least 99.5% improvement.

Feature	Feed-Forward Attention	# of Parameters	Test Loss
CQT	No	200000k	0.3916
CQT	Yes	5300k	0.3932
Mel-spectrogram	No	200000k	0.3351
Mel-spectrogram	Yes	5300k	0.3315

Table 1: Singing Performance Embedding Test Loss

The parameters for constructing the neural networks are as follows: the inputs are pairs of chopped CQT or Mel-spectrogram in the shape of $(256, 96)$. To train the siamese networks, pairs of chopped samples from the same singer or different ones are randomly sampled in a $1:1$ ratio and fed into the siamese networks. The first layer is a vanilla convolutional layer with 128 channels, kernel size of $(10, 10)$ and strides $(1, 1)$, followed by a max pooling layer with kernel size $(2, 2)$ and strides $(2, 2)$. After the max pooling layer is a ResNeXt convolutional block (as shown in figure 1). The first convolutional layer inside the block has 64 channels with $(1, 1)$ kernel. The middle layer is the grouped convolution layer that slices the input on the channel axis according to the cardinality parameters. In this experiment, the cardinality is 4, which means the incoming 64 channels are sliced into 4 partitioned "tunnels" with each "tunnel" contains 16 channels from the incoming layer. For each "tunnel", a convolutional layer is applied with 16 channels of kernel size of $(5, 5)$ and strides $(1, 1)$. The "tunnels" are concatenated back along the channel axis after the grouped convolution. The last convolutional layer inside the ResNeXt block takes the concatenated layer (64 channels) and applies a 128 channel convolutional layer with kernel size $(1, 1)$. The skipped connection from the input to the ResNeXt block is added to the output of the last layer inside the block. Another max pooling layer having kernel size $(2, 2)$ and strides $(2, 2)$ is applied after the ResNeXt block. The feature maps from the last max pooling layer having shapes (# of channels, height, width) are then reshaped to having shapes (height, # of channels $\times$ width) and are fed into the feed-forward attention layer, followed by fully connected layers. The fully connected layers have 3 layers with each layer having 1024 hidden units. Finally, a 16-dimension linear layer accepts the output from the fully connected layer is the output layer of the neural networks. The non-linear activation function used in all convolutional and fully connected layers is the ReLU function. Batch normalization is applied before each activation functions in the convolutional blocks.

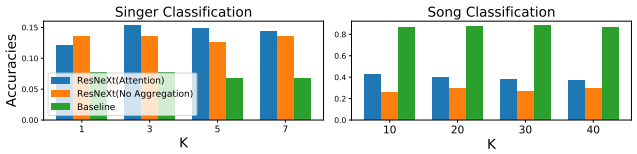

Fig. 3: Bar plots of k-nearest neighbor classification accuracies, on the embeddings learned from the embedding experiment. ResNeXt configurations with/without feed-forward attention and the handcrafted features (baseline) are used and four k values are experimented. Left bar plot is for singer classification, and the right plot is song classification.

4. SINGING STYLE INVESTIGATION BY EMBEDDING

To evaluate the networks after the training is done. The test set from the DAMP-balanced are fed into the trained neural networks to obtain their 16-dimensional embedded vectors. In Table 1, results of either training with CQT or Mel-spectrogram and of either using feed-forward attention or not are shown. From Table 1, Mel-spectrogram has noticeably better performance than CQT. To visualize how the embedding captured the singing characteristic of individual singers, the embedded vectors from the test set are projected onto a 2-D space using t-SNE [13]. The visualization is shown in figure 4. In order to better demonstrate how the learned model is able to successfully capture singing styles rather than melody or song characteristics, bag-of-feature vectors having mean and standard deviation of chroma, MFCC, spectral centroid, spectral roll-off, and spectral flux [14] extracted from each $\sim$ 6 second clip are used as a baseline to compare to our proposed embeddings. From figure 4, it is obvious that song characteristics dominate the baseline audio features, not singer characteristics. On the contrary, the embeddings from the proposed models show the capability of clustering the performances sang by the same singer closer to each other, while making the embeddings invariant to song effects. To have a quantitative assessment of the embeddings, leave-one-out k-nearest neighbor classifications using the embedded 16-dimensional vectors are used as training points. The "performance" vectors are obtained by averaging over the $\sim$ 6 second clips for each performance. Every sample is used as test sample once and the classification accuracies are obtained by averaging over the outcomes of every test sample for all k and network configurations. The classification results with multiple ks among the ResNeXt configurations with/without feed-forward attention and the baseline features are shown in figure 3. In addition, k-nearest neighbor classifications on performed songs are also conducted to demonstrate the "song effect". From the k-nearest neighbor classification results on singers and songs, it is evidential that the "song" effect exists and the singing performance embedding learning is able to dilute the "song" effect while extracting features that are more relevant to singer characterization. Also the feed-forward attention layer helped the enhancement of "singer style" while reducing "song effect" slightly by looking at the k-nearest neighbor classification accuracies. It is worth mentioning that the k-nearest neighbor classification on performed songs is only possible due to the "balanced" nature of the dataset.

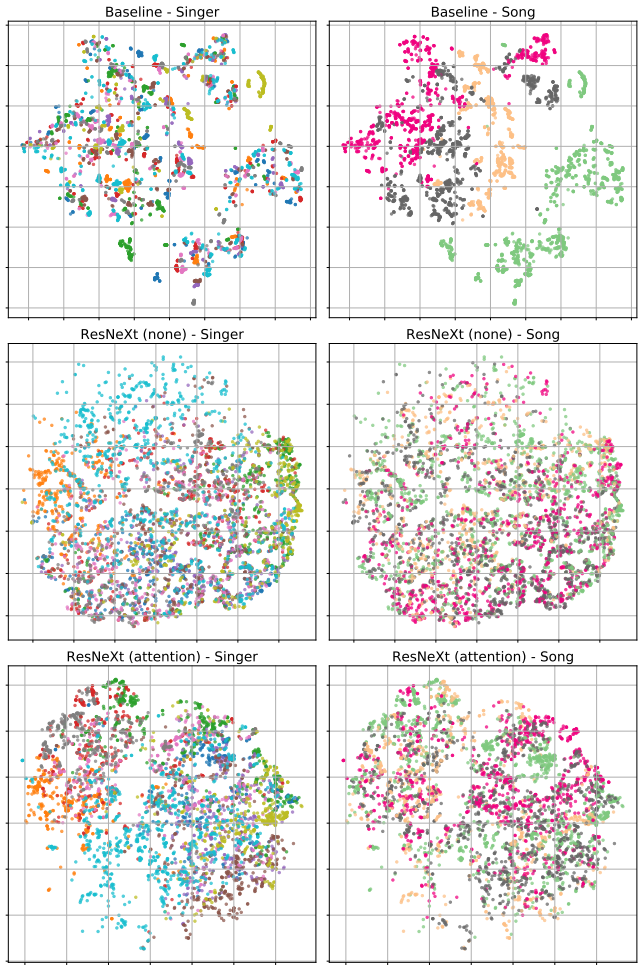

Fig. 4: t-SNE projections of the embedded clips compared to baseline handcrafted audio features. The left column is colored by singer identities while the right column is colored by song identities. Only performances from 10 singers from the test set are shown here for clearer visualization purposes.

5. ACKNOWLEDGEMENT

The research work done in this paper was supported by both the Center for Research in Entertainment and Learning (CREL) at UCSD and the internship program at Smule, Inc, with great help from Prof. Perry Cook, Mr. John Shimmin and Mr. Stefan Sullivan and the audio/video team at Smule, Inc.

6. REFERENCES

[1] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He, "Aggregated residual transformations for deep neural networks," *arXiv preprint arXiv:1611.05431*, 2016.

[2] Colin Raffel and Daniel PW Ellis, "Pruning subsequence search with attention-based embedding," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 554–558.

[3] Jeffrey C Smith, *Correlation analyses of encoded music performance*, Ph.D. thesis, Stanford University, 2013.

[4] Youngmoo E Kim and Brian Whitman, "Singer identification in popular music recordings using voice coding features," in *3rd International Society for Music Information Retrieval Conference (ISMIR)*, 2002, vol. 13, p. 17.

[5] Annamaria Mesaros, Tuomas Virtanen, and Anssi Klapuri, "Singer identification in polyphonic music using vocal separation and pattern recognition methods.," in *8th International Society for Music Information Retrieval Conference (ISMIR)*, 2007, pp. 375–378.

[6] Hiromasa Fujihara, Masataka Goto, Tetsuro Kitahara, and Hiroshi G Okuno, "A modeling of singing voice robust to accompaniment sounds and its application to singer identification and vocal-timbre-similarity-based music information retrieval," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 638–648, 2010.

[7] Mathieu Lagrange, Alexey Ozerov, and Emmanuel Vincent, "Robust singer identification in polyphonic music using melody enhancement and uncertainty-based learning," in *13th International Society for Music Information Retrieval Conference (ISMIR)*, 2012.

[8] Maria Panteli, Rachel Bittner, Juan Pablo Bello, and Simon Dixon, "Towards the characterization of singing styles in world music," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 636–640.

[9] Raia Hadsell, Sumit Chopra, and Yann LeCun, "Dimensionality reduction by learning an invariant mapping," in *Computer vision and pattern recognition, 2006 IEEE computer society conference on*. IEEE, 2006, vol. 2, pp. 1735–1742.

[10] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning.," in *AAAI*, 2017, pp. 4278–4284.

[11] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[13] Laurens van der Maaten and Geoffrey Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[14] Michael A Casey, Remco Veltkamp, Masataka Goto, Marc Leman, Christophe Rhodes, and Malcolm Slaney, "Content-based music information retrieval: Current directions and future challenges," *Proceedings of the IEEE*, vol. 96, no. 4, pp. 668–696, 2008.