INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Siamese Autoencoders for Speech Style Extraction and Switching Applied to Voice Identification and Conversion

Seyed Hamidreza Mohammadi and Alexander Kain

Center for Spoken Language Understanding, Oregon Health & Science University
Portland, OR, USA

`mohammah@ohsu.edu, kaina@ohsu.edu`

Abstract

We propose an architecture called siamese autoencoders for extracting and switching pre-determined styles of speech signals while retaining the content. We apply this architecture to a voice conversion task in which we define the content to be the linguistic message and the style to be the speaker's voice. We assume two or more data streams with the same content but unique styles. The architecture is composed of two or more separate but shared-weight autoencoders that are joined by loss functions at the hidden layers. A hidden vector is composed of style and content sub-vectors and the loss functions constrain the encodings to decompose style and content. We can select an intended target speaker either by supplying the associated style vector, or by extracting a new style vector from a new utterance, using a proposed style extraction algorithm. We focus on in-training speakers but perform some initial experiments for out-of-training speakers as well. We propose and study several types of loss functions. The experiment results show that the proposed many-to-many model is able to convert voices successfully; however, its performance does not surpass that of the state-of-the-art one-to-one model's.

Index Terms: siamese autoencoders, style extraction, style switching, voice conversion

1. Introduction

Some categories of data can be modeled as a composition of *style* and *content* information that are independent of each other. In certain applications, we are interested in changing the style without affecting the content. For example, style conversion can be used to apply a desired artistic style, such as "chalk drawing", to an existing photographic image [1]. Another example is Voice Conversion (VC) [2], which changes the style of a speech utterance, in this case the speaker characteristics, while keeping the content, the linguistic message, unchanged [3]; thus an utterance produced by a *source* speaker will be perceived as if it had been spoken by a *target* speaker. In this study, we create a *many-to-many* VC system [4, 5, 6, 7] that uses a special autoencoder architecture which first decomposes speech utterances into speaker identity and content representations, and that then re-composes new speech utterances from *modified* speaker identity and *unchanged* content representations. The VC system is many-to-many because a single architecture is used to convert *any* source speaker in the training set to sound like *any* desired target in the training set (the utterances can be unseen). Moreover, the system also works for unseen source or target speakers, albeit at potentially reduced performance.

An autoencoder (AE) is a type of artificial neural network typically used for unsupervised learning of efficient encodings, or representations [8]. The representations are learned by requiring the network to reconstruct the data.in the presence of non-linear functions in the architecture; in this manner the network learns to extract useful patterns from the data in order to be able to encode and reconstruct them [9]. When modeling speech data, these representations typically contain a combination of both content and style information. In this study, we want to create an architecture that decouples style and content representation. To this effect, we model the hidden representation vector as the *concatenation* of a style vector and a content vector. This key concept leads to an architecture called siamese autoencoder, which enables learning style and/or content in a supervised or unsupervised manner.

Similar architectures called Siamese deep networks have been proposed previously [10, 11]; these architectures were two siamese feed-forward networks, and the content representation was unconstrained. However, in this study we propose using multiple siamese representation learning networks and constrain content encodings by means of using a parallel training speech corpus which contains multiple speakers speaking the same text material, thus providing training data with identical content but unique style.

We also propose a style extraction algorithm inspired by the style extraction algorithm for photographs [1]. The style of a single data point can be computed by presenting the data point to the autoencoder and examining the style representation. For a sequence, the style could be computed by averaging the style representations of all the samples of that sequence; however, this is likely to be suboptimal. Instead, we search for the style representation that reconstructs the sequence with minimum reconstruction error, implemented as a stochastic gradient descent on the style vector, while keeping all the parameters of the architecture fixed. This study provides a way of capturing and transforming speaker dependent information from an utterance, enabling a new approach for the unification of speaker verification/identification and conversion frameworks [12].

2. Siamese autoencoders

We will use the following notation: Let $\mathbf{X}^k_{N\times D} = [\mathbf{x}^k_1, ..., \mathbf{x}^k_N]^\top$, where $\mathbf{x} = [x_1, \ldots, x_D]^\top$, represent the k^{th} parallel data streams of N observations of D-dimensional training feature vectors. For any given n, $\{\mathbf{x}^k_n\}$ are vectors with *distinct* style k (out of a total of K styles) and identical content. For the VC task, style represents speaker identity, and content represents the linguistic message.

A basic AE is composed of an *encoder* $f(\cdot)$ and a *decoder* $g(\cdot)$. The network encodes the input to the hidden representation $\mathbf{h} = f(\mathbf{x})$, and then reconstructs the input by decoding the hidden representation, $\hat{\mathbf{x}} = g(\mathbf{h})$. Minimizing the reconstruction loss, $L(\mathbf{x}, \hat{\mathbf{x}})$, forces the network to learn useful representation patterns during training. To force the network to learn more generalized representations, the input is corrupted by adding noise; the resulting training method is called *denoising* [9].

http://dx.doi.org/10.21437/Interspeech.2017-1434

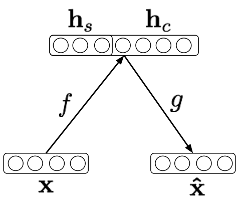

Figure 1: *Autoencoder for decomposing style and content.*

We extend the basic AE to force the network to learn reconstruction, but, in addition, learn the decomposition of style and content. The latter goal is achieved by splitting the hidden representation vector $\mathbf{h}$ into two sub-vectors $\mathbf{h}_s$ and $\mathbf{h}_c$, which represent style and content, respectively; i. e. $\mathbf{h} = [\mathbf{h}_s, \mathbf{h}_c]$, as shown in Figure 1. For training this type of AE, we propose a novel siamese autoencoder (SiAE) architecture [10], in which K AEs (one for each available style in the training data) have *shared* parameters, but operate on inputs that are parallel in content, but distinct in style. Specifically, the SiAE is composed of multiple copies of the encoder, mapping the k^{th} data stream to style and content representations $[\mathbf{h}_s^k, \mathbf{h}_c^k] = f(\mathbf{x}^k)$, and multiple copies of the decoder, reconstructing the data from style and content representations $\hat{\mathbf{x}}^k = g([\mathbf{h}_s^k, \mathbf{h}_c^k])$. This training approach, shown in Figure 2, learns to reconstruct the training data using a reconstruction loss function. However, importantly, the SiAE also allows constraints on either style or content (or both) by means of *additional* loss functions that are applied to the hidden representations $\mathbf{h}_s$ and $\mathbf{h}_c$, resulting in the total loss function (for a single observation) $L = L_r + L_c + L_s$, where $L_r = \sum_{k=1}^{K} L(\mathbf{x}^k, \hat{\mathbf{x}}^k)$, L_c, and L_s are the reconstruction, content, and style loss functions, respectively. We will discuss possible choices for L_c and L_s in Sections 2.1 and 2.2. Note that the SiAE architecture is used during training only, resulting in a standard AE architecture during testing, referred to as decomposing AE or DCAE.

2.1. Content loss function L_c

We propose two approaches to constrain the content representation. First, we consider learning the content in a supervised manner by presenting a content target vector $\mathbf{c}$ such that:

$$L_c = \sum_{k=1}^{K} L(\mathbf{h}_c^k, \mathbf{c}). \tag{1}$$

A content target could be derived from a context-dependent phoneme description, or it could be set to one of the streams $\mathbf{x}^\tau$, $1 \le \tau \le K$, effectively treating the data stream with style τ as a content template; this idea can be compared to using a particular font as a representation of the symbols of the alphabet. Second, we consider learning content in an unsupervised manner by using a *similarity* measure

$$L_c = \sum_{i=1}^{K} \sum_{j=i+1}^{K} L(\mathbf{h}_c^i, \mathbf{h}_c^j) \tag{2}$$

since all data streams have the same content. The proposed approaches to content loss functions are depicted in Figure 3.

2.2. Style loss function L_s

We propose two approaches to constrain the style representation. First, we consider learning the style in a supervised manner by presenting K distinct one-hot style target vector $\mathbf{s}^k$ such that:

$$L_s = \sum_{k=1}^{K} L(\mathbf{h}_s^k, \mathbf{s}^k). \tag{3}$$

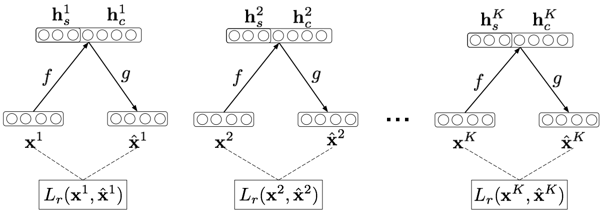
$\mathbf{h}_s^1$ $\mathbf{h}_c^1$
f
g
$\mathbf{x}^1$
$\hat{\mathbf{x}}^1$
$L_r(\mathbf{x}^1, \hat{\mathbf{x}}^1)$
$\mathbf{h}_s^2$ $\mathbf{h}_c^2$
f
g
$\mathbf{x}^2$
$\hat{\mathbf{x}}^2$
$L_r(\mathbf{x}^2, \hat{\mathbf{x}}^2)$
...
$\mathbf{h}_s^K$ $\mathbf{h}_c^K$
f
g
$\mathbf{x}^K$
$\hat{\mathbf{x}}^K$
$L_r(\mathbf{x}^K, \hat{\mathbf{x}}^K)$

Figure 2: *Basic siamese autoencoder (SiAE) architecture. The hidden representation layers are not yet constrained by means of additional loss functions.*

Second, we consider learning distinct styles in an unsupervised manner by using a *dissimilarity* measure. We posit that the average distance between all styles should be maximized while learning the speaker style space:

$$L_s = -\sum_{i=1}^{K} \sum_{j=i+1}^{K} L(\bar{\mathbf{h}}_s^i, \bar{\mathbf{h}}_s^j) \tag{4}$$

where the accent bar signifies the average value over all available vectors. The proposed approaches to style loss functions are depicted in Figure 4.

3. Style extraction algorithm

The style of any arbitrary datapoint $\mathbf{x}$ can be extracted by inspecting the style representation vector $\mathbf{h}_s$ after AE encoding: $[\mathbf{h}_s, \mathbf{h}_c] = f(\mathbf{x})$, or $\mathbf{h}_s = f(\mathbf{x})_s$. The style vector is not required to be one-hot; for example, when encoding an out-of-training speaker the style vector may reflect the mixture of styles that best represents the presented style. One approach to determining the style of a *sequence* $\mathbf{X}$ could be to average the associated style vectors. However, if the goal is finding the style vector that best reconstructs the sequence, averaging is probably sub-optimal, because it is likely that not all data contain reliable style information. For example, when using speech data, pauses are completely independent of the speaker, and certain classes of speech sounds (e. g. voiceless fricatives) are highly speaker-independent. To address this, we propose to determine the optimal style vector given an input sequence of length M:

$$\mathbf{s}^* = \arg_{\mathbf{s}} \min \frac{1}{M} \sum_{m=1}^{M} L\left(\mathbf{x}_m, g\left([\mathbf{s}, f(\mathbf{x}_m)_c]\right)\right). \tag{5}$$

During the optimization, the model parameters are kept fixed. In this study, we use stochastic gradient descent as the optimization algorithm. For computing the gradients with regards to an input, we build a new network in which the decoding weight associated with the style is provided as input and the style is considered a part of the computational graph that produces the output. This allows us to compute the gradient of style vector that is going to be learned using the Stochastic Gradient Descent (SGD) algorithm, with the style vector being the only parameters in the new network.

4. Experiments

4.1. Training

We use the VC Challenge (VCC) 2016 corpus which consists of 10 speakers each with 162 parallel sentences [13]. We split the sentences into training (100), validation (30), and objective testing (32) sentences for the objective experiments. We utilize the subjective testing set as proposed in the corpus for subjective experiments. As speech features, we used 39th-order

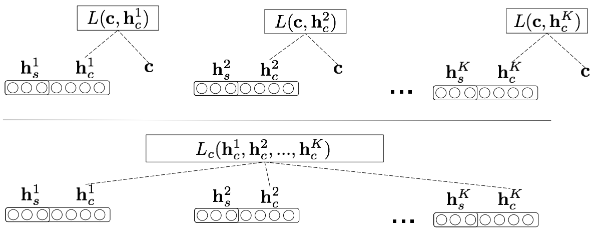
$L(\mathbf{c}, \mathbf{h}_c^1)$
$L(\mathbf{c}, \mathbf{h}_c^2)$
$L(\mathbf{c}, \mathbf{h}_c^K)$
$\mathbf{h}_s^1$ $\mathbf{h}_c^1$ $\mathbf{c}$
$\mathbf{h}_s^2$ $\mathbf{h}_c^2$ $\mathbf{c}$
...
$\mathbf{h}_s^K$ $\mathbf{h}_c^K$ $\mathbf{c}$
$L_c(\mathbf{h}_c^1, \mathbf{h}_c^2, ..., \mathbf{h}_c^K)$
$\mathbf{h}_s^1$ $\mathbf{h}_c^1$
$\mathbf{h}_s^2$ $\mathbf{h}_c^2$
...
$\mathbf{h}_s^K$ $\mathbf{h}_c^K$

Figure 3: *Two approaches to content loss functions: supervised using targets (top) and unsupervised using a similarity constraint (bottom).*

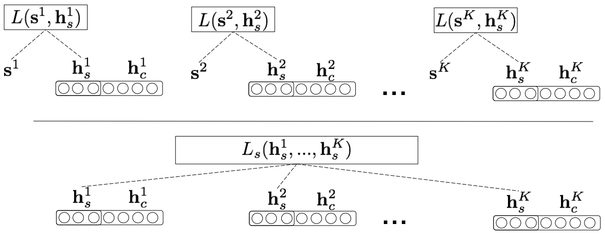
$L(\mathbf{s}^1, \mathbf{h}_s^1)$
$L(\mathbf{s}^2, \mathbf{h}_s^2)$
$L(\mathbf{s}^K, \mathbf{h}_s^K)$
$\mathbf{s}^1$ $\mathbf{h}_s^1$ $\mathbf{h}_c^1$
$\mathbf{s}^2$ $\mathbf{h}_s^2$ $\mathbf{h}_c^2$
...
$\mathbf{s}^K$ $\mathbf{h}_s^K$ $\mathbf{h}_c^K$
$L_s(\mathbf{h}_s^1, ..., \mathbf{h}_s^K)$
$\mathbf{h}_s^1$ $\mathbf{h}_c^1$
$\mathbf{h}_s^2$ $\mathbf{h}_c^2$
...
$\mathbf{h}_s^K$ $\mathbf{h}_c^K$

Figure 4: *Two approaches to style loss functions: supervised using targets (top) and unsupervised using a dissimilarity constraint (bottom).*

mel-cepstral (MCEP) features (excluding the zero$^{\text{th}}$ coefficient representing energy), extracted using the Ahocoder toolkit [14] with a 5 ms frame shift. We consider all 10 speakers for this experiment. We parallelize the speakers against a base speaker, SF1, in this study. We build three DCAE models all with supervised style constraint but three different content constraints: Triphones represented as three appended one-hot vectors (with total dimension of 126), gold speaker's min-max normalized MCEPs (SF1, with dimension of 39), and the similarity constraint (with dimension 39 for comparability). We use the autoencoder architecture of [39, 512, X], with X depending on the style and content constraints. The AE is tied-weight with sigmoidal transfer functions, except the output layer which is linear. We use SGD with batch size of 10 and learning rate of 0.01 decayed to 0.001 for training. The loss function between any two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ is defined as $L(\mathbf{v}_1, \mathbf{v}_2) = \mathbb{E}\left[|\mathbf{v}_1 - \mathbf{v}_2|^2\right]$ for all loss functions.

4.2. Visualization

For validating the architecture, we study style and content encodings for the DCAE trained with the similarity content constraint and supervised style constraint. We perform the principal component analysis (PCA) over all of the speaker content encodings, depicted in Figure 5. The root mean square error (RMSE) of mean and variance normalized two-dimensional PCA values of content encodings and similarly normalized Mel-Cepstrum (MCEP) features averaged over all speakers in a pairwise manner are 0.043 and 0.558, respectively. This shows that the content encoding values of the speakers are an order of magnitude closer to each other compared to MCEP features, showing that the similarity constraint is successfully imposing the content encodings to learn similar encodings. As shown in the figure, the distribution of these encodings for different speakers are similar. The three emphasized time-synchronous data points in Figure 5 fall in the same regions for different speakers, further validating this visually.

The style encoding for a sequence of samples can be computed using either the average of the style encodings of all of the samples, or the proposed style extraction algorithm proposed in Section 3. We plot the 2D principal components of the per-utterance computed style encodings in Figure 6, using both style averaging and extraction approaches. The figure shows speaker clusters are further apart when using style extraction compared to the averaging approach. Furthermore, the style vectors calculated using the proposed extraction algorithm have more contrastive values (i. e. they are more similar to one-hot vectors) as evident from Figure 6-right. These confirm our hypothesis that frames with less style information affect the computed style of the sequence when we use simple averaging.

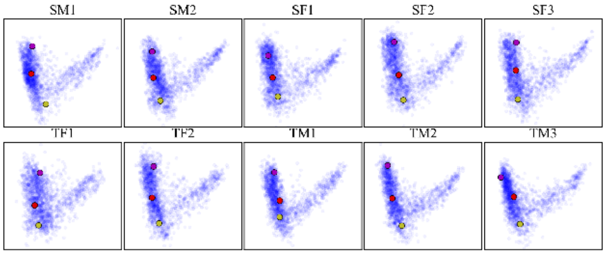

Figure 5: *2D PCA plot of computed content encodings of the test frames shown for all of the 10 training speakers. Three time-synchronous data points are emphasized.*

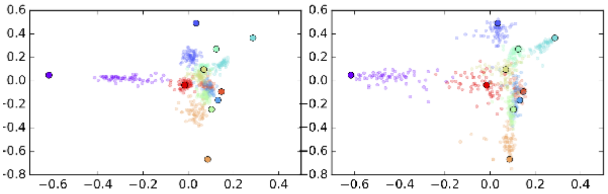

Figure 6: *2D PCA plot of computed styles from the utterances of the 10 training speakers using averaging (left) and the proposed style extraction (right) techniques. PCA-transformed one-hot vectors are added as reference (dots with black circles).*

4.3. Voice identification

We perform a voice identification experiment on in-training speakers using the proposed architectures. In this experiment, we compute the style from 2 second regions of test utterances using both style averaging and extraction methods. We then fit a Gaussian distribution with diagonal covariance matrix over the distribution of the computed style vectors for each speaker. At test time, the style is extracted from the input using either of the methods and used to find the closest Gaussian component, hence identifying the speaker. For rejecting the speakers, we decide based on a threshold on the posterior likelihood of each of the speaker's style distributions, for example 0.2. We did not perform any rejection in this study. The results are shown in Table 1. A feed forward neural network (FFNN) with a softmax top layer achieves 100% accuracy in this task by averaging and finding the speaker index with maximum value. Applying the averaging to the DCAE style encodings however achieves less than the FFNN accuracy which is likely due to the presence of other loss functions during training. Furthermore, we observe that the style extraction algorithm does not perform as well as the averaging method regarding identification. One reason is that the optimization algorithm gets stuck in local minima, mistakenly identifying another similar speaker.

Finally, we perform a similar experiment for out-of-training speakers; four speakers from the CMU-Arctic database [15]. We use the same models trained on the VCC corpus for this experiment. We perform the style averaging and extraction algorithms on the 100 utterances from each of the speakers to build the Gaussian distributions. We then perform the experiment similar to in-training speakers. The results are shown in

Table 1: *Voice identification and reconstruction scores using DCAE using different style computation methods.*

	In-training Speakers						Out-of-training Speakers					
	Identification Accuracy		Reconstruction Error (dB)				Identification Accuracy		Reconstruction Error (dB)			
Content (↓) & Style (→)	Averaging	Proposed	Frame	Averaging	Proposed	Gold	Averaging	Proposed	Frame	Averaging	Proposed	Gold
Supervised-Triphones	98.9%	95.2%	4.98	5.27	5.15	5.17	91.1%	86.9%	5.47	5.87	5.60	5.55
Supervised-Speaker	99.1%	95.3%	4.23	4.45	4.33	4.37	91.4%	87.3%	4.86	5.12	4.99	5.03
Unsupervised-Similarity	98.8%	94.9%	4.36	4.56	4.45	4.51	90.9%	87.5%	4.95	5.26	5.04	5.11

Table 2: *Conversion objective scores in mel-CD (dB)*

Model	Style	Content	K=2	K=10
1-DCAE	Supervised	Triphones	7.83	8.01
2-DCAE	Dissimilarity	Triphones	8.06	8.92
3-DCAE	Supervised	Speaker	7.19	7.45
4-DCAE	Dissimilarity	Speaker	7.89	8.71
5-DCAE	Supervised	Similarity	7.32	7.57
6-DCAE	Dissimilarity	Similarity	7.96	8.76
7-FFNN	-	-	6.74	-

Table 1. The identification accuracies are lower compared to in-training speakers, as expected. Similar to the previous case, the style extraction approach is prone to converging to local minima, resulting in lower accuracy.

4.4. Voice Reconstruction

We study the capability of the trained architecture to reconstruct the individual speakers. We perform this experiment to validate that the network is able to reconstruct individual speakers before performing voice conversion experiments. We reconstruct each speakers' utterance test frames using different style vectors including: extracted style from all of the training frames of each speaker (gold-standard), encoded from frames (encode, then decode the frame), averaging, and the proposed extraction algorithm (encode, then compute and impose style, then decode). We report mel-cepstral distortion (mel-CD in dB) as the objective measure [16]. We also perform the reconstruction experiment for out-of-training speakers as well. The reconstruction errors are higher than the ones for in-training speakers, as expected. The results show that the style extraction algorithm finds a unique style vector for a given utterance that reconstructs the features better than other style computation approaches, except the completely unconstrained frame-wise condition, which serves merely as a baseline comparison. The style extraction algorithm is prone to getting stuck in local minima since it usually finds a more varied range of styles, however, it is able to find a style vector that has lower reconstruction error compared to averaging. This property might be more desirable for use-cases which require generating speech features, such as VC.

4.5. Voice Conversion

A VC system is a speech generation system which converts speech produced by a *source* speaker to sound similar to that of a *target* speaker's. Typically, the problem is defined as a regression problem, in which the source spectral feature is mapped to target spectral features. In this study, we use the proposed DCAE to decompose style and content. VC is achieved by replacing the style vector by the intended target's style vector. We use the one-hot target selection. We use similar configurations as described in subsection 4.1 for $K = 10$ and a smaller configuration for $K = 2$ (to mimic the traditional one-to-one approach).

We explore several DCAE architectures and a one-to-one FFNN as a baseline [17, 18, 19], listed in Table 2. The average mel-CD between source and target features is 10.42. The results show that the one-to-one optimization of FFNN results in the lowest error. This is expected since the training criterion is directly related to the objective measure. For a direct one-to-one comparison, we build DCAE with the two speakers. The results show that the objective scores are close to FFNN. The generated features using the one-to-one DCAE have slightly less over-averaging compared to FFNN from our observation, which compensates for the lower mel-CD score. Using the supervised style and supervised content constraint resulted in the lowest errors. The dissimilarity style performance was not very satisfactory for K=10, which might be due to a convergence issue, since the performance for K=2 is comparatively better, showing that the loss function are behaving as expected. Nevertheless, it still lowers the error from 10.42 dB to under 9 dB which is in the right direction.

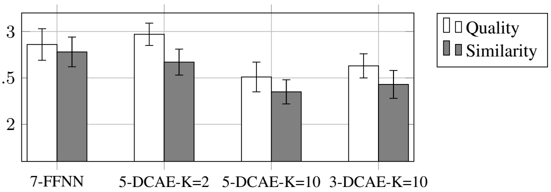

Figure 7: *Similarity/Quality plot*

We perform speech quality and speaker similarity experiment to assess the performance of the VC systems for the architectures: number 7, 5-K=2, 5-K=10, and 3-K=10. We use 100 listeners for each experiment. For the speech quality (a.k.a naturalness) test, we play a stimuli for the listeners and ask them to rate the quality from 1 (very bad) to 5 (very good) and report the Mean Opinion Score (MOS). We use four conversion pairs: SM1→TF1, SF2→TM1, SF1→TF2, and SM2→TM2. The quality results in Figure 7 show that the one-to-one DCAE performs marginally better compared to FFNNs. Both one-to-one approaches perform significantly better than the many-to-many models. We perform speaker similarity experiment by playing two stimuli for the listener, one target and the other converted. The listeners are asked to score from 1 (the utterances are definitely from different speakers) to 5 (the utterances are definitely from the same speaker). The results show that both one-to-one approaches perform significantly better than the many-to-many models, with FFNN performing marginally better than the one-to-one DCAE.

5. Conclusion

We proposed a training architecture called siamese autoencoders for constructing a decomposing autoencoder which decomposes the style and content of speech signals. For voice identification and conversion tasks, we define the content to be the linguistic message and the style to be the speaker's voice. We trained this architecture on a 10 speaker speech corpus and showed that it is able to perform voice identification and conversion for in-training speakers. The VC experiment results show a similar performance of the one-to-one model compared to baseline FFNNs. The many-to-many models performance was worse than one-to-one models. We also performed some preliminary identification and reconstruction experiments on out-of-training speakers. A next step is to train the model on a larger number of speakers to build a more general-purpose model that is able to better handle style and content decomposition for unseen speakers, resulting in robust style and content representation vectors that could be used for various speech processing tasks, including VC without any training.

6. References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the CVPR*, 2016.

[2] S. H. Mohammadi and A. Kain, “An overview of voice conversion systems,” *Speech Communication*, 2017.

[3] V. Popa, J. Nurminen, and M. Gabbouj, “A novel technique for voice conversion based on style and content decomposition with bilinear models.” in *Proceedings of the INTERSPEECH*, 2009.

[4] T. Toda, Y. Ohtani, and K. Shikano, “One-to-many and many-to-one voice conversion based on eigenvoices,” in *Proceedings of the ICASSP*, 2007.

[5] Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, “Many-to-many eigenvoice conversion with reference voice,” in *Proceedings of the INTERSPEECH*, 2009.

[6] D. Saito, K. Yamamoto, N. Minematsu, and K. Hirose, “One-to-many voice conversion based on tensor representation of speaker space.” in *Proceedings of the INTERSPEECH*, 2011.

[7] Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, “Non-parallel training for many-to-many eigenvoice conversion,” in *Proceedings of the ICASSP*, 2010.

[8] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[9] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[10] K. Chen and A. Salman, “Extracting speaker-specific information with a regularized siamese deep network.” in *Advances in Neural Information Processing Systems 24*, 2011, pp. 298–306.

[11] N. Zeghidour, G. Synnaeve, N. Usunier, and E. Dupoux, “Joint learning of speaker and phonetic similarities with siamese networks,” in *Proceedings of the INTERSPEECH*, 2016.

[12] T. Kinnunen, L. Juvela, P. Alku, and J. Yamagishi, “Non-parallel voice conversion using i-vector plda: Towards unifying speaker verification and transformation,” in *Proceedings of the ICASSP*, 2017.

[13] T. Toda, D. Saito, F. Villavicencio, J. Yamagishi, M. Wester, Z. Wu, L.-H. Chen *et al.*, “The voice conversion challenge 2016,” in *Proceedings of the INTERSPEECH*, 2016.

[14] D. Erro, I. Sainz, E. Navas, and I. Hernáez, “Improved HNM-based vocoder for statistical synthesizers.” in *Proceedings of the INTERSPEECH*, 2011.

[15] J. Kominek and A. W. Black, “The CMU arctic speech databases,” in *Proceedings of the SSW*, 2004.

[16] T. Toda, A. W. Black, and K. Tokuda, “Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2222–2235, 2007.

[17] S. Desai, A. W. Black, B. Yegnanarayana, and K. Prahallad, “Spectral mapping using artificial neural networks for voice conversion,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 5, pp. 954–964, 2010.

[18] L.-H. Chen, Z.-H. Ling, L.-J. Liu, and L.-R. Dai, “Voice conversion using deep neural networks with layer-wise generative training,” *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 22, no. 12, pp. 1859–1872, 2014.

[19] S. H. Mohammadi and A. Kain, “Voice conversion using deep neural networks with speaker-independent pre-training,” in *Proceedings of the SLT*, 2014.