ON INVESTIGATION OF UNSUPERVISED SPEECH FACTORIZATION BASED ON NORMALIZATION FLOW

Haoran Sun, Yunqi Cai, Lantian Li, Dong Wang

CSLT, Tsinghua University, China

ABSTRACT

Speech signals are complex composites of various information, including phonetic content, speaker traits, channel effect, etc. Decomposing this complicated mixture into independent factors, i.e., speech factorization, is fundamentally important and plays the central role in many important algorithms of modern speech processing tasks. In this paper, we present a preliminary investigation on unsupervised speech factorization based on the normalization flow model. This model constructs a complex invertible transform, by which we can project speech segments into a latent code space where the distribution is a simple diagonal Gaussian. Our preliminary investigation on the TIMIT database shows that this code space exhibits favorable properties such as denseness and pseudo linearity, and perceptually important factors such as phonetic content and speaker trait can be represented as particular directions within the code space.

Index Terms— speech factorization, normalization flow, deep learning

1. INTRODUCTION

Speech signals are complex composite that involves enormous information, such as phonetic content, speaker traits, emotion, channel and ambient noise. These different types of information are intermingled in an unknown manner, which leads to the fundamental difficulty in all speech processing tasks. For example, speaker variation is among the most challenging problems that speech recognition researchers have been annoyed for several decades, and the variation on emotion status and speaking styles causes notorious trouble for speaker recognition [1].

A natural idea to deal with the information blending is to factorize the speech signal into separate informative factors. This idea has been partly demonstrated by some well-known factorization-based models in speaker recognition, such as J-FA [2] and the i-vector model [3], where speech signals are assumed to be factorized into phonetic content, speaker traits and channel effect. A clear shortage of these methods is that they assume the informative factors are composed by a linear Gaussian model, which might be oversimple and is incapable of describing the complex generation process of speech signals.

Recently, we proposed a deep cascade factorization (D-CF) approach to factorize speech signals at the frame level [4]. The DCF approach follows the layer-wised generation process proposed by Fujisaki [5] and factorize speech signals into information factors one by one, and each new factor depends on the factors that have been inferred already, by using a task-oriented deep neural network (DNN) trained using task-specific data. DCF is the first successful speech factorization model based on deep learning, however it suffers from two shortcomings: (1) it is based on supervised learning and requires labelled data for all interesting information factors; (2) it is frame-based and does not consider the temporal dependency within speech signals.

In this paper, we present an unsupervised speech factorization approach based on deep generative models. The basic hypothesis is that if we can find a way to generate the data, then we probably can gain a better understanding of the underlying informative factors. The i-vector model is such a generative model, but the linear Gaussian assumption is too strong to suit the generation process of speech. In this study, we make use of the powerful generation capability of DNNs to deal with this problem. More specifically, we build a latent code space where the distribution is as simple as a diagonal Gaussian, and train a complex DNN to generate the speech signals from these latent codes. We found that perceptually important speech factors could be represented as particular directions within the code space.

There are three popular deep generative models: generative adversarial network (GAN) [6], variational auto-encoder (VAE) [7] and normalization flow [8, 9, 10, 11]. Among these models, GAN is capable of generation but lack of inference. VAE is capable of both generation and inference, but the model is trained with a variational bound of the true likelihood, hence not accurate. Normalization flow is trained to maximize the true likelihood, and the generation and inference are simple. This model has been successfully applied to image generation [12] and speech synthesis [13, 14, 15]. In this work, we extensively use the normalization flow model to study the speech generation process and investigate the property of the latent code space.

2. RELATED WORK

The idea of discovering and manipulating speech factors plays the central role in many important algorithms in speech processing. The most important example is the famous source-filter model and the associated linear prediction coding (LPC) algorithm, which decomposes speech signals into vocal fold excitation and vocal tract modulation [16, 17]. This decomposition places the foundation of modern speech processing theory, however, it is mostly psychologically inspired and the factors it derives (excitation and modulation) are not directly related to speech processing tasks. For example, neither excitation nor modulation directly represents speaker traits. By contrast, the Fujisaki model [5] treats speech generation as a process of convolution of different layers of informative factors, and each factor is related to a specific speech processing task. However, the inference with Fujisaki model is difficult. The DCF algorithm [4] provides the inference approach, however, the model training requires a large amount of labelled data. This paper presents an unsupervised approach that can train a factorization model with unlabelled data and infer informative factors in an easy way.

This work is related to the flow-based speech synthesis [14, 15], but our goal is to analyze rather than generate speech signals. The unsupervised factorization idea was also seen in recent work on multi-speaker and multi-style speech synthesis [18, 19]. Finally, this work is mostly inspired by the flow-based image generation [12].

3. NORMALIZATION FLOW FOR SPEECH FACTORIZATION

3.1. Review on normalization flow

The basic idea of normalization flow is to design a chain of invertible transforms that map a simple distribution to a complex distribution [8], as shown in Fig. 1. In this figure, each single-step transform is invertible, so the whole transform is invertible. By this invertible transforming, a variable z that follows a simple distribution can be mapped to a variable x whose distribution is very complex. Conversely, a variable x whose distribution is complex can be mapped back to a variable z whose distribution is simple. This transform chain is called a *normalization flow*. In our study, we will treat x as a speech signal, and our goal is to transform the signal to a code z, which encodes the latent factor underlying x.

According to the principle of distribution transforming [20], the probabilities of the observation x and the corresponding code z possess the following relation:

$$\ln p(x) = \ln p(z(x)) + \ln |\det(\frac{\mathrm{d}z}{\mathrm{d}x})|,$$

where $z(x)$ is the inverse function of $x(z)$, and $\det(\frac{\mathrm{d}z}{\mathrm{d}x})$ is the determinant of the Jacobian matrix $\frac{\mathrm{d}z}{\mathrm{d}x}$.

The flow can be trained following the maximum likelihood (ML) criterion, for which the objective function can be written as follows:

$$L_\theta = \sum_i \ln p(x_i) = \sum_i \ln p(z_\theta(x_i)) + \ln \frac{\mathrm{d}z_\theta(x_i)}{\mathrm{d}x_i}.$$

where θ denotes the parameters of the flow. Maximizing this objective leads to a deep generative model where the generation process simply casts to sampling z from $p(z)$ and transforming it to the observation space by $x(z)$. Conversely, the (inverse) flow can be used to transform an observation x to its code z, offering a tool to describe data in the code space.
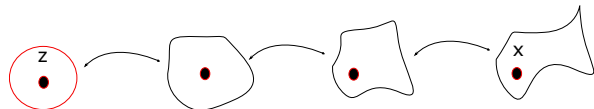

Fig. 1. Normalization flow transforms a simple distribution of z to a complex distribution of x, by a series of invertible transforming functions.

Note that normalization flow is a general framework, where both the transforming functions and the latent distribution can be selected freely. Regarding the transforming function, popular choices are linear transform [10], inverse autoregressive transform [11], and 1×1 convolutional transform [12]. All these transformations are invertible and the associated Jacobians hold simple forms. Regarding the latent distribution, the most popular choice is the diagonal Gaussian.

3.2. Speech factorization by flow

With the normalization flow model, it is possible to transform a speech signal x to a latent code z whose distribution is as simple as a diagonal Gaussian. Since the distribution of z is much simpler than x, it becomes easier for us to analyze speech signals in the code space, paving the way of discovering important informative factors there.

A key concern of speech factorization is the dependency over time. Although frame-based factorization worked in our previous work [4], we suppose taking temporal dependency into account would help. Therefore, the data sample x we choose in this study is a speech segment rather than a speech frame, the length of which is fixed. We first compute these spectrograms of the speech segments, and then treat these spectrograms as observations in the flow model. Considering that the spectrograms are 2-dimensional images, we choose the Glow structure [12] to implement the normalization flow, as it has worked well in image generation tasks.

As a preliminary study, we choose to analyze English vowels. The goal of the analysis is to study the distributional properties of different vowels and speakers in the code space,

and investigate the possibility to discover important factors that are perceptually salient for human ears.

4. EXPERIMENTS

4.1. Data preparation

We use the TIMIT database to conduct the experiments, and choose five English vowels (aa, ae, iy, ow, uh) in the investigation. Firstly, the speech segments of the five target vowels are extracted from the TIMIT database according to the meta information of the speech utterances. Secondly, these speech segments are converted to spectrograms, by setting the window length, window hop and FFT length to be 25ms, 1ms, 257, respectively. Thirdly, spectrograms longer than 288 frames are discarded, and all the rest spectrograms are lengthened to 288 frames by appending zeros. This leads to spectrograms in size of 288×288 pixels, which are used as the observations of the Glow model. The first 288 denotes the number of frames in the time domain, and the second 288 denotes the number of frequency bands in the frequency domain with a dimension of FFT length appending 31 zeros.

We use the code from an available version of the Glow model to conduct training and inference[1]. All the audio examples reported in the following experiments can be downloaded from http://project.cslt.org.

4.2. Distribution of observations and codes

Fig. 2 shows the distribution of the observations and codes, where we randomly select two dimensions for each piece of data. It can be seen clearly that the distribution of the codes is much more Gaussian compared to the distribution of the observations. This verified that the flow model has been well trained and it indeed normalized the distribution.
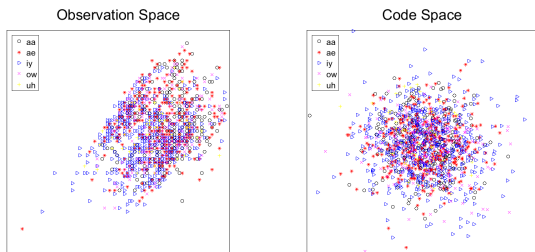

Fig. 2. Distribution in the observation space (left) and the code space (right).

4.3. Sampling

In this experiment, we test the flow by sampling some speech segments in the code space. This can be achieved by sampling a code z following a diagonal Gaussian, and then trans-

---

[1]https://github.com/chaiyujin/glow-pytorch

forming it to an observation x (spectrogram) through the flow $x(z)$. Fig. 3 shows some spectrograms of sampled examples. It can be seen that the sampled spectrograms exhibit similar formant structures to those of true speech. By converting them to waveforms using phase of a true speech, we found these samples are meaningful speech. This is not surprising as the distribution of meaningful speech segments is Gaussian in the code space, so samples obtained following the Gaussian will have a high probability to be meaningful. An observation is that most of the samples we obtained are silence. This could be attributed to the large proportion of silence in the training data, caused by the silence padding.
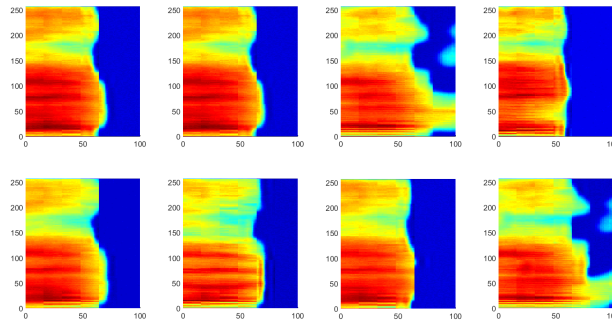

Fig. 3. Spectrograms obtained by sampling in the code space and transforming to the observation space.

4.4. Interpolation

In this experiment, we investigate the (pseudo) linear property of the code space. Considering two meaningful speech segments, both should be located in a *dense* area as they are in the code space. This is because they are meaningful and so should be granted high probabilities by the model. According to the property of the diagonal Gaussian, in the code space, the probability at the location of any interpolation of the two segments will be between the dense locations of these two segments. This means that any interpolation will result in a meaningful speech segment. Ideally, the speech properties will change gradually from one segment to the other by the interpolation.

To test this hypothesis, we choose a segment of aa and a segment of ae spoken by the same person, and interpolate them in the code space. Results are shown in Fig. 4. It is interesting that by this interpolation, a segment of aa gradually changed to a segment of ae, without much change on other properties, e.g., speaker traits. The audio clips reconstructed from the spectrograms (by using the phase of aa) can be downloaded online; they sound rather reasonable. We also test interpolation between genders and speakers, both work well.

This result is highly interesting, as it suggests that the code space is likely pseudo linear for factors that are salient for human ears. In other words, in the code space, it is possi-

ble to find a direction following which only one perceptually important factor changes. An implication of this property is that a speech factor can be represented by a particular direction in the code space. This suggests a possible speech factorization strategy that starts from a neutral speech, and change its properties by moving in the code space following the directions that correspond to the desired properties sequentially.
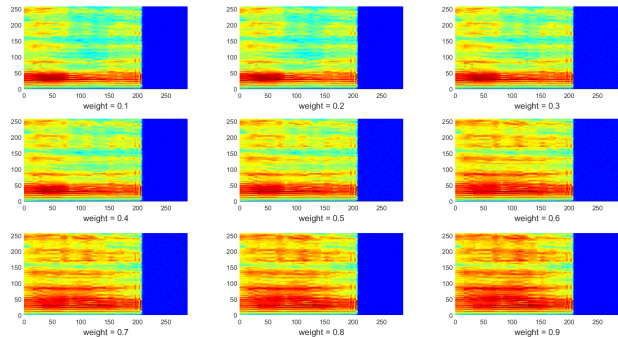

Fig. 4. Interpolation between vowel aa and ae in the code space. From top-left corner the bottom-right corner, the interpolation weight for ae changes from 0.1 to 0.9.

4.5. Denoising

The pseudo linear property of the code space can be used to remove noise. Firstly we add white noise to the training data randomly, and then compute the codes for these noise-contaminated segments. The averaged codes for clean and noisy speech are computed respectively, and the displacement between them, denoted by $\xi_{c \to n}$, is used to recover the clean version for noise-corrupted segments, by $z_c = z_n - \beta \xi_{c \to n}$, where β is a denoising scale. The effect of the factorization-based denoising is shown in Fig. 5. It can be seen that noise is removed gradually when stepping towards the opposite direction of $\xi_{c \to n}$.
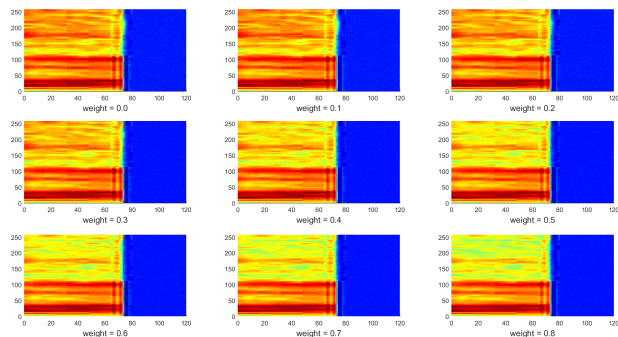

Fig. 5. Denoising using the pseudo linear property of the code space. From the top-left corner to bottom-right corner, the denoising scale β changes from 0 to 0.8.

4.6. Discrimination

The last experiment examines if the discriminative information of the observations is preserved in the code space. To gain this purpose, we train an LDA to select the two most discriminative directions (for a particular classification task) and plot the samples in both the observation space and code space. Three classification tasks are investigated: (1) vowel aa vs ae; (2) male vs female; (3) two different speakers. The results are shown in Fig. 6. It can be seen that the class information is largely lost when transforming to the code space. This unwanted property for most speech processing tasks, however, seems not surprising, as the flow model tries to compress all the codes into a Gaussian ball which is compact and dense, without any class information taken into account.
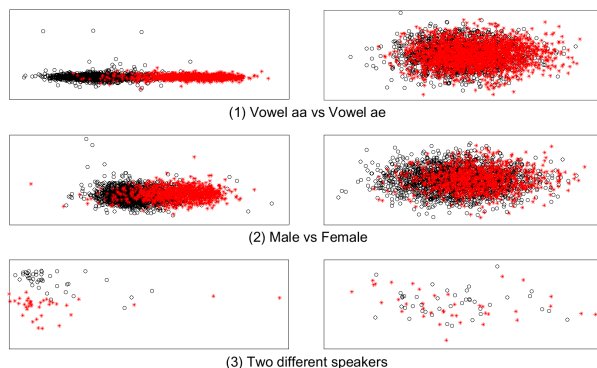

(1) Vowel aa vs Vowel ae

(2) Male vs Female

(3) Two different speakers

Fig. 6. Distribution with class labels in the observation space (left column) and the code space (right column). The two colors/shapes represent different vowels in the first row, different genders in the second row, and different speakers in the third row.

5. CONCLUSIONS

We presented a preliminary study on the properties of the latent space derived by a normalization flow model for speech segments. The experimental results showed that this code space possesses a favorable pseudo linear property, which means that perceptually important factors such as phonetic content and speaker traits can be changed gradually by moving in the code space following a particular direction. This provides an interesting way of unsupervised speech factorization, where each salient factor corresponds to a particular direction in the code space. Potential applications of this factorization include voice conversion and noise cancellation. Future work will conduct more thorough studies on large databases and continuous speech. Another work will investigate discriminative flow models which take class information into consideration.

6. REFERENCES

[1] Jacob Benesty, M Mohan Sondhi, and Yiteng Huang, *Springer handbook of speech processing*, Springer Science & Business Media, 2007.

[2] Patrick Kenny, Gilles Boulianne, Pierre Ouellet, and Pierre Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[3] 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, 2011.

[4] Lantian Li, Dong Wang, Yixiang Chen, Ying Shing, and Zhiyuan Tang, "Deep factorization for speech signal," in *ICASSP*, 2018.

[5] Hiroya Fujisaki, "Prosody, models, and spontaneous speech," in *Computing prosody*, pp. 27–42. Springer, 1996.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[7] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[8] Laurent Dinh, David Krueger, and Yoshua Bengio, "Nice: Non-linear independent components estimation," *arXiv preprint arXiv:1410.8516*, 2014.

[9] Laurent Dinh, Jascha Sohldickstein, and Samy Bengio, "Density estimation using real nvp," *arXiv preprint arXiv:1605.08803*, 2016.

[10] Danilo Jimenez Rezende and Shakir Mohamed, "Variational Inference with Normalizing Flows," *arXiv preprint arXiv:1505.05770*, 2015.

[11] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling, "Improving variational inference with inverse autoregressive flow," *arXiv pre-print arXiv:1606.04934*, 2016.

[12] Durk P Kingma and Prafulla Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems 31*, 2018, pp. 10215–10224.

[13] Aaron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis, "Parallel WaveNet: Fast high-fidelity speech synthesis," in *Proceedings of the 35th International Conference on Machine Learning*. 2018, vol. 80 of *Proceedings of Machine Learning Research*, pp. 3918–3926, PMLR.

[14] Sungwon Kim, Sang Gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon, "Flowavenet : A generative flow for raw audio," *arXiv pre-print arXiv:1811.02155*, 2018.

[15] Ryan Prenger, Rafael Valle, and Bryan Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP 2019*, 2019.

[16] Gunnar Fant, *Acoustic Theory of Speech Production*, Mouton,The Hague, 1960.

[17] B. S. Atal, "The history of linear prediction," *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 154–161, 2006.

[18] Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ Skerry-Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Fei Ren, Ye Jia, and Rif A. Saurous, "Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis," *arXiv preprint arXiv:1803.09017*, 2018.

[19] Wei-Ning Hsu, Yu Zhang, Ron Weiss, Yu-An Chung, Yuxuan Wang, Yonghui Wu, and James Glass, "Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization," in *ICASSP 2019*, 2019.

[20] Joseph C. Watkins, "Transformations of random variables," 2009.