Fine-tuning Style Transfer in TTS with Tacotron 2 and WaveGlow

Fuyang Zhang
Simon Fraser University
fuyangz@sfu.ca

Brian White
Simon Fraser University
bjw10@sfu.ca

Abstract

Generating natural speech from text has been a challenging conversational AI task for decades. Although immense progress has been made in recent years, existing methods largely do not provide users a way to control the generated speech style. In this paper we propose a modular architecture that can accomplish this task by employing two existing state-of-the-art models: Tacotron 2 [10], a recurrent sequence-to-sequence network for predicting mel-spectrograms from text, and WaveGlow [9], a flow-based generative network to synthesize natural sounding speech conditioned on mel-spectrograms. In particular, we explore how to learn better representative latent style vectors and accomplish more fine-grained control when applying stylistic emphasis on generated speech by additionally training WaveGlow using a triplet loss function, as opposed only using the data likelihood loss function as done in [9].

1 Introduction

Generating natural speech from text (text-to-speech, TTS) has seen great progress in recent years. In fact, current state-of-the-art (SOTA) works [12, 10, 8] can already produce audio quality that rivals that of human speech. However, these methods largely lack support for control over style transfer in the synthesized speech, and instead only synthesize speech with neutral style [11].

Most popular SOTA TTS methods are autoregressive, and employ an encoder-decoder architecture. In this framework, the encoder converts a sequence of text to a hidden representation, while the decoder then generates outputs frame by frame [5]. The decoder typically outputs audio encoded feature representations, e.g. spectrograms, which capture information such as word pronunciation, volume, and prosody, (variations in intonation, tone, stress, and rhythm).

Recently, non-autoregressive models have also shown great potential and significant efforts have been made in the development of such models [7]. However, non-autoregressive models for TTS often are highly compute-intensive, making training time prohibitively long in practice without access to modern parallel hardware.

Meanwhile, flow-based generative models show high performance on image generation [5, 3], especially when sophisticated distribution approximation methods are used. Compared to other popular generative models, (VAEs, GANs), flow-based models typically have the advantage of producing meaningful latent spaces. This is because flow-based models apply exact latent variable inference and log-likelihood evaluation without approximation. On the contrary, VAEs are only able to infer approximately of the latent variable, and GANs are constrained by latent vector representations. Meaningful latent spaces can provide great benefit for downstream tasks, such as interpolations between data points, i.e. style transfer.

In this paper we focus on controlling style transfer in TTS systems, and thus propose an architecture that utilises a flow-based vocoder model to perform this task. Our proposed system combines two

Preprint. Under review.

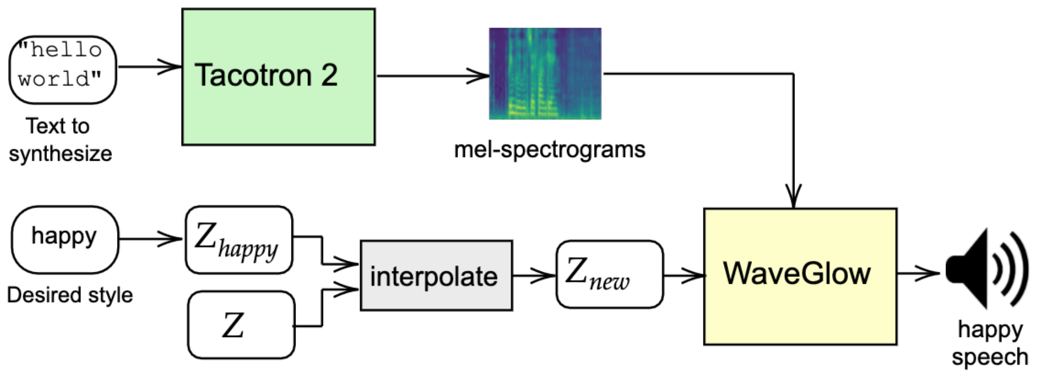

Figure 1: High-level model architecture.

existing models that have individually demonstrated SOTA performance in generating natural speech: Tacotron 2 [10], an end-to-end neural network architecture for achieving TTS directly from character sequences, and WaveGlow [9], a flow-based generative network for synthesizing speech and providing better control in style transfer. In particular, we explore potential qualitative improvements with respect to controlling style transfer when fine-tuning WaveGlow on an additional triplet loss function to learn more representative latent feature spaces associated with different speech styles. Our code is available at `https://github.com/zhangfuyang/TTS-Style-Transfer`.

2 Related work

Recent TTS systems are typically divided in two parts: a synthesis model, and a neural vocoder. First, the synthesis model transforms a sequence of text into time-aligned latent features to encode audio information, such as mel-spectrograms, and the neural vocoder is responsible for generating audio samples from those latent feature representations [13].

Models [12, 10] follow this two stage architecture, and the synthesis model employed in these models are among the most popular choices of synthesis models in TTS systems. At the backbone of these particular synthesis models is a recurrent sequence-to-sequence model with attention, which takes a character sequence as input and outputs the corresponding mel-spectrogram. The neural vocoder component of these models are different variants of [8], an autoregressive model that predicts output frames serially, that is, in an autoregressive way. In contrast to the original Tacotron, Tacotron 2 uses simpler components, such as standard LSTM and convolutional layers in the encoder and decoder instead of the rather complex CBHG stacks [12] and GRU recurrent layers in [12]. Further, location-sensitive attention is used instead of additive attention [10].

The second component in this typical two stage architecture, the neural vocoder, is computationally challenging and can affect quality as well. The most famous vocoder is [8], however, due to inherent attribution of autoregressive flow, [8] has difficulty synthesising audio in real time without sacrificing quality. Another approach is use a non-autoregressive vocoder such as WaveGlow [9], which generates high quality speech from mel-spectrograms via a flow-based model. The model is trained only with data likelihood loss function and can surprisingly generate high quality speech in real time.

NICE [2] is the first popular generative model in what is referred to as normalizing flow. The transformation in NICE is an additive coupling layer with a scale term. RealNVP [3] is a more recent model, which generalizes the coupling layer and successfully introduces convolutional layers into normalizing flow models. Glow [5] extends the previous models by replacing the reverse permutation operation on channel order with an invertible 1x1 convolutional layer. These methods have been shown to successfully discover efficient feature sampling that can be used for manipulating attributes of the human face when applied to the field of computer vision.

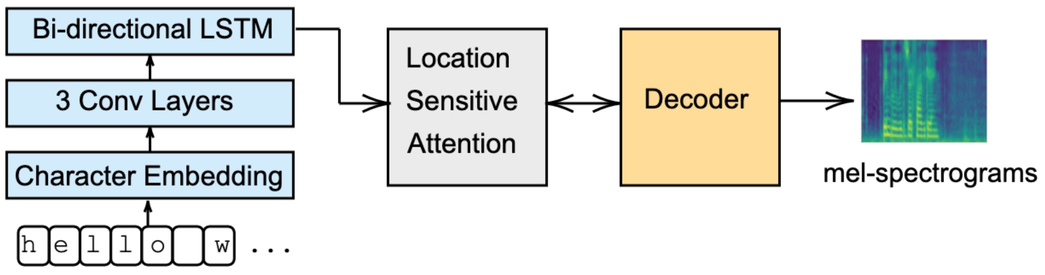

Figure 2: Tacotron 2 feature prediction network (simplified).

3 Approach

To provide more control over stylised speech synthesis to end users of the TTS system, we propose an architecture that combines the synthesis model component of Tacotron 2 with a flow-based vocoder, WaveGlow, in place of the original vocoder [8] used in [10]. Further, we train WaveGlow using a triplet loss function in addition to the data likelihood loss function used in [9]. By training WaveGlow on this alternative loss function, we show that we can learn more representative latent style vectors, and argue this allows for more fine-grained control over speech style.

3.1 Tacotron 2

The feature prediction network in [10], i.e. the synthesis model, is sequence-to-sequence model with attention, as shown in Figure 2.

In Tacotron 2, the encoder maps an input sequence, $X = \{X_1, X_2, ..., X_{T_x}\}$, to a series of annotations, (encoder hidden states), $H = \{H_1, H_2, ..., H_{T_x}\}$, both of the same length. Each annotation H_i contains information about token X_i in the input sequence X. Tacotron 2 operates on a character sequence, so each token in the input sequence X corresponds to a single character.

The encoder is composed of three convolutional layers followed by a bidirectional LSTM layer. The convolutional layers are useful for detecting local correlations between characters, allowing the model to handle more difficult words, such as words with silent letters, e.g., "know", or words with unusual pronunciation, e.g., "cafe".

These annotations, (the encoder outputs), are then passed to an attention network which generates a context vector. This allows the decoder to better align outputs with corresponding inputs, and in effect allows the decoder to focus on the most relevant tokens in the character sequence. From here, the decoder then produces mel-spectrograms one frame at a time.

3.2 WaveGlow

The neural vocoder component, WaveGlow, is a flow-based generative model composed of a sequence of invertible transform layers. WaveGlow learns an invertible mapping from mel-spectrograms to a latent feature space that can be exploited to manipulate many aspects of speech synthesis, which we refer to simply as "speech style". (Note that this speech style can be discussed more specifically in terms of accent, tone, and prosody information, but we will limit our discussion of speech style to a high level. E.g., in this paper we consider "happy speech" and "sad speech" to be two instances of distinct speech styles.)

The model explicitly learns the training data distribution $p(x) = p(x|z)p(z)dz$ with maximum log-likelihood. With a good estimation of $p(x)$, the model can sample unobserved data by sampling from the learned latent vector z. An important mathematical property of the transform layers is they are invertible, and their corresponding Jacobian determinants are relatively easy to compute. We ask the readers to look into their original paper [9] for details.

3.3 WaveGlow fine-tuning

Besides training WaveGlow on the data likelihood loss as originally done in [9], we additionally train on a triplet loss function to force the network to preserve similarities between same-style examples and differences between different-style examples existing in high-dimensional feature space when transforming into lower-dimensional latent feature space. I.e., this forces the network to keep same-style training instances close together and different-style training instances far apart when discovering their latent feature representations. In this way, the model can have more separated estimations of the distributions for different speech styles, and potentially improve transferring style when applying average style vectors to obtain new styles. Thus, our loss function is the combination of maximum log-likelihood and minimum triplet loss.

This overall loss function is formulated as shown here,

$$\begin{aligned}\log p_\theta(\mathbf{x}) &= \log p_\theta(\mathbf{z}) + \sum_{i=1}^{k} \log|\det(\mathbf{J}(f_i^{-1}(\mathbf{x}))| \\ &= -\frac{\mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{x})}{2\sigma^2} + \sum_{j=0}^{\#\text{coupling}} \log \mathbf{s}_j(\mathbf{x}) + \sum_{k=0}^{\#\text{conv}} \log\det|\mathbf{W}_k|\end{aligned} \tag{1}$$

$$l(\mathbf{x}_a, \mathbf{x}_p, \mathbf{x}_n) = \max(|\mathbf{z}(\mathbf{x}_a) - \mathbf{z}(\mathbf{x}_p)|^2 - |\mathbf{z}(\mathbf{x}_a) - \mathbf{z}(\mathbf{x}_n)|^2 + \alpha, 0) \tag{2}$$

$$\mathcal{L} = -\lambda \log p_\theta(\mathbf{x}) + l(\mathbf{x}_a, \mathbf{x}_p, \mathbf{x}_n), \tag{3}$$

where the first term in (3) comes from the log-likelihood and Jacobian determinants, (which is the same as in [9]), and second term comes from the triplet loss. We set margin value α as 1 and λ as 5.

3.4 Inference

Figure 1 shows an overview of the working pipeline during inference time. The user provides the system with a raw text input, optionally accompanied with a style emphasis category label, e.g. ("hello world", happy). Tacotron 2 then processes this input text as a character sequence, and outputs a corresponding sequence of mel-spectrograms. These mel-spectrograms are then fed into WaveGlow, which is responsible for converting these into audio wave forms. WaveGlow achieves this by drawing samples from a simple zero mean spherical Gaussian distribution, and then passing these samples through a series of layers, reshaping the samples to align with the desired distribution of audio, (learned during training), conditioned on the provided mel-spectrograms from Tacotron 2.

4 Experiments

This section describes training datasets, implementation details and provides qualitative results. We provide images to illustrate our results, but the best way is to simply listen to them. We ask the readers go to our website[1] to listen to some audio samples.

Our system is fairly modified from WaveGlow and Tacotron2 and our implementation is mostly based on repository `https://github.com/NVIDIA/waveglow`. Our code and pre-trained model can be found at `https://github.com/zhangfuyang/TTS-Style-Transfer`.

4.1 Datasets

Both Tacotron2 and WaveGlow are trained on LJ Speech dataset [4]. This dataset consists of 13,100 short audio clips of a single speaker reading passages from 7 non-fiction books, coupled with a transcription for each clip.

As the LJ Speech dataset does not include style annotations, WaveGlow is also fine-tuned on CREMA-D [1] and RAVDESS [6] datasets for discovering the latent vectors of different style categories.

[1]audio samples can be found at `https://zhangfuyang.github.io/TTS`

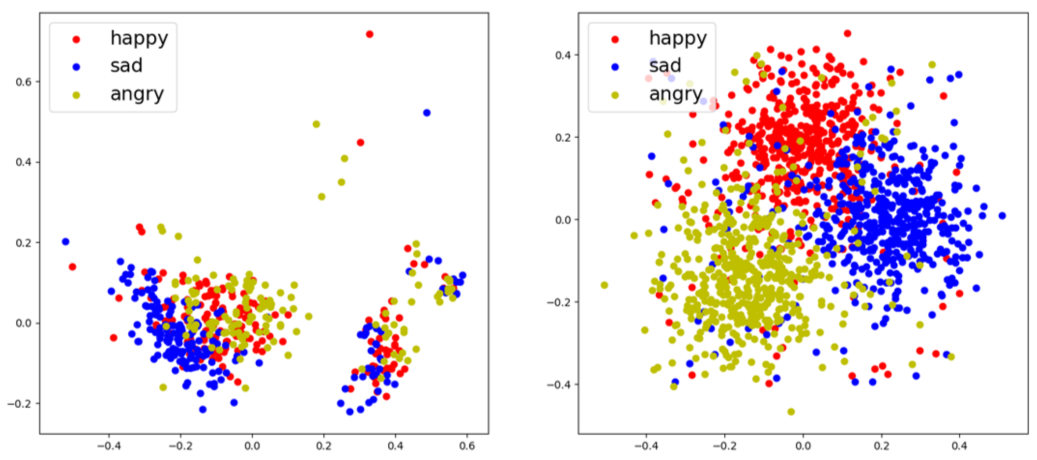

Figure 3: Visualization of feature embedding. Left is the result w/o fine-tuning. Right is the result w/ fine-tuning

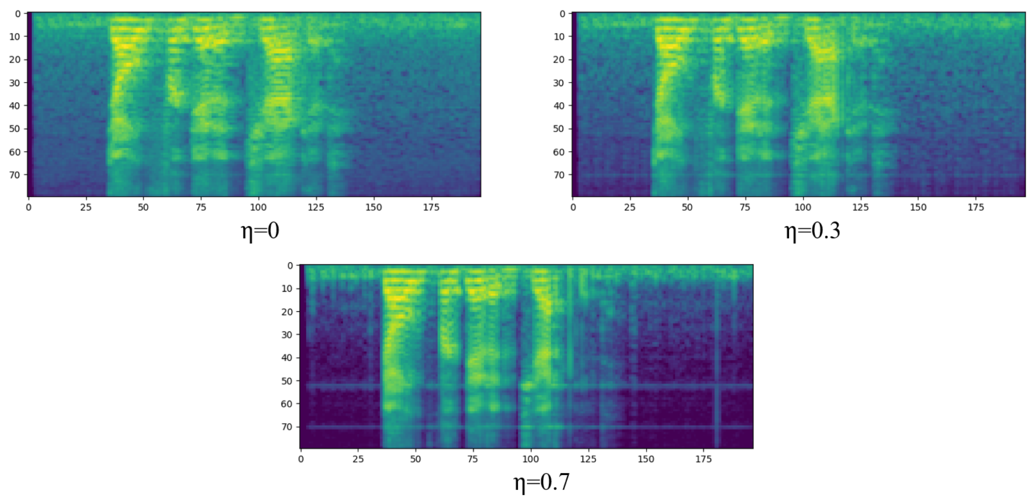

Figure 4: Interpolation latent vector of a sad style audio with average happy style vector. $\hat{\mathbf{z}} = (1-\eta)\cdot\mathbf{z}_{origin} + \eta\cdot\mathbf{z}_{happy}$. Figure shows mel-spectrorams of generated audio.

Though [1, 6] include style-annotated training data across six common style categories, in our experiment we fine-tune on only three of these six categories, namely, happy, sad and angry styles.

4.2 Implementation details

We initialize with a pre-trained WaveGlow model on LJ Speech Dataset. Then we further fine-tune the network on an NVIDIA TitanX GPU for 10 epochs with roughly 1500 samples per category (happy, sad, angry). For more details about our fine-tuning procedure, refer to Approach section 3.3.

We use a sampling rate of 22050 Hz, mel-spectrograms with 80 bins and librosa mel filter as defaults– which is the same setting used in [9]. In order to ensure all the audio data use the same sampling rate, we also use Python resampy package to re-sample the audio.

For training, we use Adam optimizer with 1×10^{-4} learning rate and 1×10^{-6} weight decay.

4.3 Results

Figure 3 shows a visualization of vector embeddings (latent space) for different emotions. We use principal component analysis to reduce latent vectors to 2-dimensional vectors. The result shows that our fine-tuning procedure can successfully make different emotions data more separable.

Figure 4 shows the interpolation in z-space to the corresponding generated mel-spectrograms. For this experiment we expect to see audio linearly transferring from initial style to reference style. We recommend our reader go to our website to hear the real audio samples.

However, even though our model can clearly embed different stylistic audio into different region of latent space, mostly we fail when transferring styles. There are a couple of reasons for the failure cases. First, the latent space size is too small, which may not be easy to capture high level information, such as emotions. Second, emotions can be mostly affected by the distribution of spectrograms of the sounds, though our model cannot directly control spectrogram instead using it as a condition. However, we argue that our model still shows the potential of doing style transfer. We believe if we gain more data and use a much larger latent space, it is possible to improve performance significantly.

5 Conclusion

In this paper we propose a modular architecture using two existing SOTA models which together can offer more control over style transfer in TTS systems. These models are Tacotron 2 [10], a recurrent sequence-to-sequence network for predicting mel-spectrograms from text, and WaveGlow [9], a flow-based generative network to synthesize natural sounding speech conditioned on mel-spectrograms. We explore how by training WaveGlow with a triplet loss function, together with the data likelihood loss function used in [9], we can obtain well-defined latent style vectors for each distinct speech style we support. With more separation between these latent style vectors, we argue that we can achieve more precise style emphases when synthesizing stylistic speech from text.

Broader impact

Although our system does not introduce much novelty to the existing SOTA models employed in this paper, (with the exception of fine tuning the vocoder with an alternative loss function), there are still some ethical considerations to keep in mind as TTS systems continue to advance at large. This is especially true for TTS systems that can control stylistic characteristics in generated speech. With the ability to control stylistic audio features, e.g. accent, tone, prosody, when generating speech, there are some potentially malicious applications that could be developed, such as reproducing an existing person's voice without consent.

Contributions

Both authors contributed equally to the work presented in this paper.

References

[1] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, and R. Verma. Crema-d: Crowd-sourced emotional multimodal actors dataset. *IEEE transactions on affective computing*, 5(4):377–390, 2014.

[2] L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[3] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[4] K. Ito and L. Johnson. The lj speech dataset. `https://keithito.com/LJ-Speech-Dataset/`, 2017.

[5] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in neural information processing systems*, pages 10215–10224, 2018.

[6] S. R. Livingstone and F. A. Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391, 2018.

[7] C. Miao, S. Liang, Z. Liu, M. Chen, J. Ma, S. Wang, and J. Xiao. Efficienttts: An efficient and high-quality text-to-speech architecture. *arXiv preprint arXiv:2012.03500*, 2020.

[8] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[9] R. Prenger, R. Valle, and B. Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621. IEEE, 2019.

[10] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783. IEEE, 2018.

[11] R. Valle, K. Shih, R. Prenger, and B. Catanzaro. Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. *arXiv preprint arXiv:2005.05957*, 2020.

[12] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*, 2017.

[13] R. J. Weiss, R. Skerry-Ryan, E. Battenberg, S. Mariooryad, and D. P. Kingma. Wave-tacotron: Spectrogram-free end-to-end text-to-speech synthesis. *arXiv preprint arXiv:2011.03568*, 2020.