SNAC: Speaker-Normalized Affine Coupling Layer in Flow-Based Architecture for Zero-Shot Multi-Speaker Text-to-Speech

Byoung Jin Choi, Student Member, IEEE, Myeonghun Jeong, Student Member, IEEE, Joun Yeop Lee, and Nam Soo Kim, Senior Member, IEEE

Abstract—Zero-shot multi-speaker text-to-speech (ZSM-TTS) models aim to generate a speech sample with the voice characteristic of an unseen speaker. The main challenge of ZSM-TTS is to increase the overall speaker similarity for unseen speakers. One of the most successful speaker conditioning methods for flow-based multi-speaker text-to-speech (TTS) models is to utilize the functions which predict the scale and bias parameters of the affine coupling layers according to the given speaker embedding vector. In this letter, we improve on the previous speaker conditioning method by introducing a speaker-normalized affine coupling (SNAC) layer which allows for unseen speaker speech synthesis in a zero-shot manner leveraging a normalization-based conditioning technique. The newly designed coupling layer explicitly normalizes the input by the parameters predicted from a speaker embedding vector while training, enabling an inverse process of denormalizing for a new speaker embedding at inference. The proposed conditioning scheme yields the state-of-the-art performance in terms of the speech quality and speaker similarity in a ZSM-TTS setting.

Index Terms—Conditional normalizing flow, speech synthesis, zero-shot multi-speaker text-to-speech.
scenario. The proposed method explicitly normalizes the input by the speaker-dependent parameters in order to preserve speaker-independent information at training, while target speaker’s information is inversely injected at inference through denormalization. We compare the proposed conditioning method to the existing method in several different experimental settings using VITS [7] as our base model and demonstrate that the proposed method outperforms the conventional technique in both subjective and objective measures. The audio samples are available on the demo page.¹

II. AFFINE COUPLING-BASED GENERATIVE FLOW

Normalizing flow models, [31], [32], [33], learn an invertible mapping between a prior distribution, \( p_\theta(z) \), and a more complex data distribution \( p_\theta(x) \) using a sequence of bijective functions. The log-likelihood computation is tractable via the rule of change-of-variables. Let \( f_\theta : \mathbb{R}^D \rightarrow \mathbb{R}^D \) be a bijective function which maps the observed data \( x \) to the latent variable \( z \) from a simple prior distribution \( p_\theta(z) \) where \( x, z \in \mathbb{R}^D \). Then the log-likelihood is obtained by

\[
\log p_\theta(x) = \log p_\theta(z) + \log \left| \det \frac{\partial f_\theta(x)}{\partial x} \right|. \tag{1}
\]

Computing the log determinant of the Jacobian matrix in (1) is computationally expensive in general. In addition, \( f_\theta \) is strictly restricted to be a bijective function in which only certain types of transformations can be easily inverted. An affine coupling layer, first introduced in [31], allows for an efficient computation of the log determinant with invertible transformations by generating an output \( y \in \mathbb{R}^D \) given an input \( x \in \mathbb{R}^D \) and \( d < D \) via

\[
y_{1:d} = x_{1:d}
\]

\[
y_{d+1:D} = x_{d+1:D} \odot \exp(s_\theta(x_{1:d})) + b_\theta(x_{1:d}) \tag{2}
\]

where \( s_\theta \) and \( b_\theta \) are parameterized scale and bias functions mapping \( \mathbb{R}^d \rightarrow \mathbb{R}^{D-d} \) and \( \odot \) is an element-wise product. With this coupling architecture, the Jacobian becomes a lower triangular matrix as given by

\[
\frac{\partial y}{\partial x} = \begin{bmatrix}
I_d & 0 \\
\frac{\partial s_\theta(x_{1:d})}{\partial x_{1:d}} & \text{diag}(\exp(s_\theta(x_{1:d})))
\end{bmatrix} \tag{3}
\]

where \( I_d \) represents a \( d \times d \) identity matrix. Computing the determinant of Jacobian matrix of the affine coupling does not depend on the Jacobians of \( s_\theta \) and \( b_\theta \). Therefore, they can be any type of complex functions modeled by highly expressive neural networks, such as non-causal Wavenet [34]. The inverse transformation of the coupling layer can be easily derived as

\[
x_{1:d} = y_{1:d}
\]

\[
x_{d+1:D} = \frac{y_{d+1:D} - b_\theta(y_{1:d})}{\exp(s_\theta(y_{1:d}))}, \tag{4}
\]

hence sampling is also efficient. Each coupling layer is then followed by a layer which permutes the ordering of the channels along the feature dimension.

¹[Online]. Available: https://byoungjinchoi.github.io/snac/

III. SPEAKER-NORMALIZED AFFINE COUPLING LAYER FOR ZSM-TTS

A conditional generative flow [35], [36] models a conditional probability distribution \( p_\theta(x|z) \) where \( z \) represents a conditioning term. Conventionally, a conditional flow extends the forward and inverse transformations of an affine coupling layer given in (2) and (4) by modifying \( s_\theta \) and \( b_\theta \) such that they take \( g \) as an additional input.

For ZSM-TTS, the condition \( g \) usually represents a specific speaker embedding vector. Our strategy for ZSM-TTS is to convert the speaker-dependent data distribution to a latent prior distribution which becomes speaker-independent. Then, when synthesizing speech, the speaker-independent latent prior distribution is mapped back to a speaker-specific data distribution depending on the given speaker embedding. In order to achieve this, we design each affine coupling layer to remove the information related to \( g \) at the forward transformation. Contrarily, \( g \) is injected to the input embedding sequence at the inverse transformation. To obtain such bijective transformation with explicit \( g \) conditioning, we propose a speaker-normalized affine coupling (SNAC) layer, which normalizes and denormalizes the input embedding sequence by the mean and standard deviation parameters predicted by \( g \). Speaker normalization (SN) and speaker denormalization (SDN) in SNAC are performed as follows:

\[
SN(x; g) = \frac{x - m_\theta(g)}{\exp(v_\theta(g))}
\]

\[
SDN(x; g) = x \odot \exp(v_\theta(g)) + m_\theta(g) \tag{5}
\]

where \( m_\theta \) and \( v_\theta \) are simple linear projections to obtain the mean and standard deviation parameters from \( g \). \( SN \) and \( SDN \) are applied across the temporal axis, thus normalizing and denormalizing each frame of the input \( x \) with the same mean and standard deviation parameters.

The forward transformation of the SNAC layer is now given by

\[
y_{1:d} = x_{1:d}
\]

\[
y_{d+1:D} = SN(x_{d+1:D};g) \odot \exp(s_\theta(SN(x_{1:d};g))) + b_\theta(SN(x_{1:d};g)) \tag{6}
\]

The inverse transformation can be derived straightforwardly as follows:

\[
x_{1:d} = y_{1:d}
\]

\[
x_{d+1:D} = SDN\left(\frac{y_{d+1:D} - b_\theta(SN(y_{1:d};g))}{\exp(s_\theta(SN(y_{1:d};g)))}, g\right) \tag{7}
\]

At each SNAC layer, \( SN \) is applied to the input of \( s_\theta \) and \( b_\theta \) such that the affine parameters contain the information unrelated to the speaker. Since \( x_{d+1:D} \) is also speaker-normalized in the forward transformation, this results in the extensive removal of speaker information at training. When inferring \( x_{d+1:D} \) through the inverse transformation, \( SDN \) is enforced after the affine transformation is applied to \( y_{d+1:D} \) to appropriately inject information related to the target speaker.
For the partitioning scheme of affine coupling layer represents \( q \frac{\partial f}{\partial g} \) and \( g \). In this manner. The objective is to maximize the variational lower bound of a conditional log-likelihood.

The log-determinant of the conditional flow with SNAC layers can be obtained by

\[
\log \left| \det \frac{\partial f_\theta(x)}{\partial x} \right| = \log \sum_j \frac{\exp(s_\theta(SN(x_1:l; g)))_j}{\exp(v_\theta(g)_j)}. \tag{8}
\]

The complete architecture of the SNAC layer is presented in Fig. 1.

IV. EXPERIMENTS

A. Model

We performed experiments on the proposed method by replacing the affine coupling layer of the flow module in VITS with the SNAC layer.

1) VITS Overview: VITS leverages the variational autoencoder (VAE) [37] formulation and the adversarial training strategy to successfully combine the joint training of acoustic feature generation, vocoding, and duration prediction in an end-to-end manner. The objective is to maximize the variational lower bound of a conditional log-likelihood

\[
\log p_\theta(o|l) \geq E_{q_\phi(z|o)} \left[ \log p_\theta(o|z) - \log q_\phi(z|o) \right] \tag{9}
\]

where \( p_\theta(o|z) \), \( q_\phi(z|o) \) and \( p_\theta(z|l) \) respectively denote the likelihood, the approximate posterior and the conditional prior distributions. A target speech and the corresponding phoneme sequence are denoted as \( o \) and \( l = [l_{ext}, A] \), and \( z \) is a frame-level latent sequence representing the intermediate acoustic features. The alignment \( A \) is estimated using the monotonic alignment search (MAS) algorithm proposed in [10]. The generator part of VITS architecture consists of a posterior encoder, prior encoder, decoder, and duration predictor and is trained with a discriminator in an adversarial manner. The prior encoder is composed of two parts: a text encoder and a flow module. The flow module plays an essential role in transforming a simple text-conditional distribution to a more complex distribution.

2) Multi-Speaker VITS: In a multi-speaker setting, the likelihood \( p_\theta(o|l) \) is substituted with \( p_\theta(o|l; g) \) where \( g \) represents a speaker embedding. For training, the original work uses a speaker label as an additional input which is transformed to a fixed-dimensional vector \( g \) via a learnable embedding table. \( g \) is then conditioned to every module of the generator.

B. Datasets

All tested models were trained on VCTK [38] dataset. VCTK is a multi-speaker audio dataset which contains approximately 44 hours of speech recorded by 109 speakers. We selected 11 speakers as an in-domain test set following [24]. To evaluate the performance on the out-of-domain dataset, we randomly selected 20 speakers from LibriTTS [39] test-clean dataset, which consists of 8.56 hours of audio from 39 speakers. Each utterance was downsampling to 22050 Hz for training.

C. Implementation Details

Our proposed method modifies the official implementation of VITS.² For the partitioning scheme of affine coupling layer at flow module, we chose channel-wise masking pattern [32]. To ensure that all input entries are processed, we reverse the ordering of the feature dimension at each layer of flow module. We employ a reference encoder to extract the speaker embedding vector. The reference encoder is composed of a stack of 2-D convolutional layers and a gated recurrent unit (GRU) [40], following global style token (GST) [41]. The reference encoder takes a sequence of linear spectrograms of the reference audio as an input and outputs a 256-dimensional embedding vector. Two separate linear projection layers, \( m_\theta \) and \( v_\theta \), are employed to predict the mean and standard deviation parameters of the speaker in (5) from the reference embedding vector.

D. Experimental Setup

We evaluated our method in several different settings. We built our first baseline, Baseline+Ref+ALL, by attaching a reference encoder to the vanilla multi-speaker VITS model, which applied the speaker conditioning at every module of the generator. The second baseline, Baseline+Ref+Flow, conditioned the speaker embedding only to the duration predictor and the flow module to focus on the effect of the proposed method. The last baseline, Baseline+Pre-Trained+Flow, substituted the reference encoder with a pre-trained speaker encoder which was trained from a speaker verification for a different speaker embedding scenario. We used a H/ASP model [42] which can be obtained from an open-source project.³ In this baseline, the pre-trained speaker encoder weights were fixed to consistently draw speaker embedding vectors from a learned speaker embedding space. For the above three baselines, the speaker embedding vector was used as a conditional input to produce \( s_\theta \) and \( b_\theta \) at affine coupling layers of the flow module.

²https://github.com/jaywalnut310/vits
³https://github.com/clovaai/voxceleb_trainer
To demonstrate the effect of the proposed method for each setting, we replaced the conventional affine coupling layers with the SNAC layers for the above three baselines. We name the three proposed models corresponding to each baseline as follows: Proposed+REF+ALL, Proposed+REF+FLOW, Proposed+PRE-TRAINED+FLOW.

Furthermore, we compared our models with two other baseline models: Meta-StyleSpeech [25] and YourTTS [24]. Meta-StyleSpeech is trained with a meta-learning scheme with a modified structure of FastSpeech2 [9]. YourTTS is built on VITS architecture with an external speaker encoder and an additional speaker consistency loss. We used an open-source implementation\(^4\) for Meta-StyleSpeech and followed the paper to implement YourTTS on the official VITS code.

## Evaluation Method

We first conducted subjective tests to measure the overall speech quality using mean opinion score (MOS). To assess the effectiveness of the proposed speaker conditioning method, we also measured similarity mean opinion score (SMOS). SMOS is employed to evaluate how similar the synthesized samples are to the reference speech samples in terms of speaker characteristic. Both MOS and SMOS are 5-scale scores ranging from 1 to 5 and reported with the 95% confidence interval. For in-domain evaluations, we drew 3 pairs of text and reference audio randomly from each of the 11 unseen speakers of VCTK test dataset. To evaluate on the out-of-domain case, 2 pairs of text and reference audio from 20 randomly selected speakers were drawn from the LibriTTS test-clean dataset. 15 judges participated in the subjective tests with both VCTK and LibriTTS unseen speakers.

In addition, we also evaluated the objective score for speaker similarity between the generated samples and the ground truth samples using speaker embedding cosine similarity (SECS). The SECS ranges from -1 to 1, where 1 indicates both samples are from the same speaker. We computed SECS using a pre-trained speaker encoder model provided by SpeechBrain toolkit\(^5\) [43]. The results of MOS, SMOS, and SECS are presented in Table I.

### Table I

| Model                        | VCTK MOS(\(\dagger\)) | VCTK SMOS(\(\dagger\)) | VCTK SECS(\(\dagger\)) | LibriTTS MOS(\(\dagger\)) | LibriTTS SMOS(\(\dagger\)) | LibriTTS SECS(\(\dagger\)) |
|------------------------------|------------------------|--------------------------|--------------------------|-----------------------------|-----------------------------|-----------------------------|
| Ground Truth                | 4.76±0.02              | 4.19±0.04                | 0.748                    | 4.80±0.02                   | 4.51±0.03                   | 0.646                       |
| Meta-StyleSpeech            | 2.06±0.04              | 2.62±0.05                | 0.212                    | 2.00±0.03                   | 2.50±0.04                   | 0.131                       |
| YourTTS                     | 4.42±0.03              | 3.86±0.04                | \(\mathbf{0.447}\)       | 4.23±0.03                   | 3.35±0.04                   | \(\mathbf{0.317}\)          |
| Baseline+REF+ALL            | 4.22±0.04              | 4.11±0.04                | 0.350                    | 4.30±0.03                   | 3.67±0.04                   | 0.143                       |
| Baseline+REF+FLOW           | 4.08±0.04              | 4.01±0.04                | 0.339                    | 3.98±0.04                   | 3.64±0.04                   | 0.135                       |
| Baseline+PRE-TRAINED+FLOW   | 4.38±0.03              | 3.52±0.04                | 0.321                    | 4.17±0.03                   | 2.91±0.05                   | 0.135                       |
| Proposed+REF+ALL            | 4.30±0.03              | 4.07±0.04                | 0.320                    | 4.11±0.03                   | 3.56±0.04                   | 0.145                       |
| Proposed+REF+FLOW           | \(\mathbf{4.48±0.03}\) | \(\mathbf{4.19±0.04}\)   | 0.352                    | \(\mathbf{4.41±0.03}\)     | \(\mathbf{3.70±0.04}\)     | 0.151                       |
| Proposed+PRE-TRAINED+FLOW   | 4.46±0.03              | 3.61±0.04                | 0.270                    | 4.40±0.03                   | 3.18±0.04                   | 0.116                       |

\(^4\)https://github.com/keonlee9420/StyleSpeech
\(^5\)https://github.com/speechbrain/speechbrain

**F. Results**

The MOS and SMOS results shown in Table I indicate that Proposed+REF+FLOW consistently shows superior performance over the baseline models in terms of sample quality and speaker similarity.

In Proposed+REF+ALL, the SNAC-based flow module enforces the explicit removal of speaker information at the forward transformation while the speaker information is conversely injected to the generator modules. Training this model results in neutralizing the effect of the SNAC layer, and this accounts for the MOS and SMOS drop from Baseline+PRE-TRAINED+FLOW to Proposed+PRE-TRAINED+FLOW in both VCTK and LibriTTS datasets. On the contrary, the best performance for subjective tests is consistently achieved by Proposed+REF+FLOW in both VCTK and LibriTTS datasets. Nonetheless, YourTTS shows the highest SECS scores among all models since YourTTS is trained to minimize the speaker embedding cosine similarity.

From the synthesized samples, we have noticed that the models using pre-trained speaker encoders occasionally produce the voice of different speakers. This phenomenon is reflected in the lower SMOS of Baseline+PRE-TRAINED+FLOW and Proposed+PRE-TRAINED+FLOW. This shows that the joint training of a reference encoder may be more suitable for ZSM-TTS task than using a pre-trained speaker encoder in terms of speaker stability. However, this does not affect the MOS as much since the generated samples maintain the consistent quality.

Although Proposed+REF+ALL outperforms the baseline models in both in-domain and out-of-domain datasets, the overall performance drop between the two settings still exists in terms of speaker similarity. Since LibriTTS dataset inherently includes various types of channel conditions which may interfere with the accurate inference of speaker embedding while VCTK contains only clean speech data, we conjecture such domain mismatch accounts for the performance drop.

## Conclusion

We have proposed a novel speaker conditioning method for flow-based multi-speaker TTS. The experimental results show that the proposed method outperforms the conventional conditioning technique in a ZSM-TTS setting and achieves the best performance in subjective tests compared to the other baseline models. For a future work, we intend to incorporate locally-varying features related to prosody and accents.
REFERENCES

[1] Y. Wang et al., “Tacotron: Towards end-to-end speech synthesis,” in Proc. Interspeech, 2017, pp. 4006–4010.

[2] J. Shen et al., “Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., 2018, pp. 4779–4783.

[3] H. Tachibana, K. Uenoyma, and S. Aihara, “Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., 2018, pp. 4784–4788.

[4] S. Ö. Arık et al., “Deep voice: Real-time neural text-to-speech,” in Proc. Int. Conf. Mach. Learn., 2017, pp. 195–204.

[5] W. Ping et al., “Deep voice 3: Scaling text-to-speech with convolutional sequence learning,” in Proc. Int. Conf. Learn. Representations, 2018.

[6] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, “Neural speech synthesis with transformer network,” in Proc. AAAI Conf. Artif. Intell., 2019, vol. 33, pp. 6706–6713.

[7] J. Kim, J. Kong, and J. Son, “Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 5530–5540.

[8] Y. Ren et al., “FastSpeech: Fast, robust and controllable text to speech,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019, pp. 3171–3180.

[9] Y. Ren et al., “FastSpeech 2: Fast and high-quality end-to-end text-to-speech,” 2020, arXiv:2006.04558.

[10] J. Kim, S. Kim, J. Kong, and S. Yoon, “ Glow-TTS: A generative flow for text-to-speech via monotonic alignment search,” in Proc. Adv. Neural Inf. Process. Syst., 2020, vol. 33, pp. 8067–8077.

[11] J. Donahue, S. Dieleman, M. Bärkówski, E. Olsen, and K. Simonyan, “End-to-end adversarial text-to-speech,” in Proc. Int. Conf. Learn. Representations, 2021.

[12] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, “Adaptation of pitch for HMM-based speech synthesis system using MLLR,” in Proc. 5th Eur. Conf. Speech Commun. Technol., 1999, pp. 2374–2380.

[13] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi, “Speaker adaptation for HMM-based speech synthesis using MLLR,” in Proc. 3rd ESCA/COCOSDA Workshop (ETRW) Speech Synth., 1998.

[14] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi, “Adaptation of pitch and spectrum for HMM-based speech synthesis using MLLR,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process., 2001, vol. 2, pp. 805–808.

[15] O. Siohan, T. A. Myrvoll, and C.-H. Lee, “Structural maximum a posteriori sequence learning,” in Proc. Int. Conf. Mach. Learn., 2014.

[16] L. Dinh, J. Sohl-Dickstein, and S. Bengio, “Density estimation using real NVP,” in Proc. Int. Conf. Learn. Representations, 2017.

[17] J. Hirsh, B. Lee, J. Y. Yang, and S. J. Hwang, “Semi-conditional normalizing flows for semi-supervised learning,” in Proc. Int. Conf. Learn. Representations, 2018.

[18] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” in Proc. Neural Inf. Process. Syst. Workshop Deep Learn., 2014.

[19] Y. Wang et al., “Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 5180–5189.

[20] H. S. Heo, B.-J. Lee, J. Huh, and J. S. Chung, “Clova baseline system for the VoxCeleb speaker recognition challenge 2020,” 2020, arXiv:2009.14153.

[21] M. Ravaneli et al., “SpeechBrain: A general-purpose speech toolkit,” 2021, arXiv:2106.04624.