INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

A New GAN-based End-to-End TTS Training Algorithm

Haohan Guo[*], *Frank K. Soong*[†], *Lei He*[†], *Lei Xie*[*]

[*]School of Computer Science, Northwestern Polytechnical University, Xian, China
[†]Microsoft China

{hhguo,lxie}@nwpu-aslp.org, {frankkps, helei}@microsoft.com

Abstract

End-to-end, autoregressive model-based TTS has shown significant performance improvements over the conventional ones. However, the autoregressive module training is affected by the exposure bias, or the mismatch between different distributions of real and predicted data. While real data is provided in training, in testing, predicted data is available only. By introducing both real and generated data sequences in training, we can alleviate the effects of the exposure bias. We propose to use Generative Adversarial Network (GAN) along with the idea of "Professor Forcing" in training. A discriminator in GAN is jointly trained to equalize the difference between real and the predicted data. In AB subjective listening test, the results show that the new approach is preferred over the standard transfer learning with a CMOS improvement of 0.1. Sentence level intelligibility tests also show significant improvement in a pathological test set. The GAN-trained new model is shown more stable than the baseline to produce better alignments for the Tacotron output.

Index Terms: speech synthesis, end-to-end TTS synthesis, auto-regressive model, generative adversarial model, adversarial training

1. Introduction

Statistical parametric text-to-speech (TTS) is a sequence generator, which generates a sequence of speech samples according to the input text or phoneme sequence. To achieve better intelligibility, naturalness and expressiveness, enhancing the model's prediction capability is very important. From HMM [1] and DNN [2] to LSTM [3] and BLSTM [4], effective sequence modelling plays an important role in TTS. In recent years, autoregressive (AR) model has been widely used in sequence-to-sequence model to further improve the performance of the sequential model. It has shown significant improvement in speech synthesis, e.g. autoregressive acoustic model [5, 6], WaveNet-based [7] or WaveRNN-based [8] neural vocoder, and end-to-end TTS system [9, 10, 11].

Autoregressive model specifies that the output sample $\hat{y}_t$ depends on its own previous samples $\hat{y}_{1:t-1}$, as:

$$p(\hat{y}_{1:T}|X,\Theta) = \prod_{t=1}^{T} p(\hat{y}_t|\hat{y}_{1:t-1},X,\Theta) \quad (1)$$

Here, X and Θ denote the inputs and the network weights. Although AR model has improved an end-to-end TTS model, the conventional training algorithm (also known as *teacher forcing* [12]) has a intrinsic problem in training, namely *exposure bias* [13]. As shown in Fig.1, in training, the model is only exposed to the real data, which predicts output $\hat{y}_t$ given the real data of previous samples as input.

$$p(\hat{y}_{1:T}|X,\Theta) = \prod_{t=1}^{T} p(\hat{y}_t|y_{1:t-1},X,\Theta) \quad (2)$$

However, in testing the model can only predict the next step using its own predicted samples. The model distribution, however, can not be the same as the real one, so the discrepancy between these two distributions can quickly accumulate errors in decoding. In the end-to-end TTS system (e.g. *Tacotron*), we often adopt a *data dropout* strategy to alleviate the above problems, which randomly discards part of feedback information in both training and testing to reduce the autoregressive effect in prediction such the generation can rely more on the linguistic information which is available in both training and testing. But it is still not enough to avoid the exposure bias, especially in decoding a long sequence.

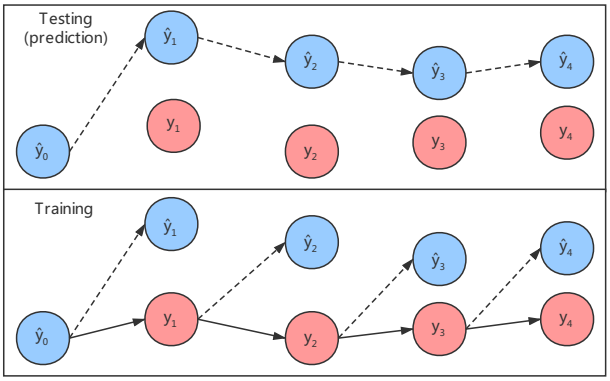

Figure 1: *The difference between training (teacher forcing) and testing (prediction) of AR model (ŷ: predicted, y: real)*

Exposure bias is caused by the mismatch between autoregressive predictions and real data used in training. We can avoid it by using predicted samples in training the autoregressive model. The widely used algorithm is to feedback the generated data in training with a sampling strategy, e.g. *data as demonstrator (DAD)* [14] or *scheduled sampling (SS)* [15]. In training with scheduled sampling, we decide to feedback real data with a given probability in each time step, or we feedback the generated data. The probability decreases based upon an annealing schedule. Since this training algorithm ignores the temporal dependency of the sequence [16], it may result in misalignment between the target and the predicted sequence. Thus it forces the model trained with MSE to predict an incorrect sequence (see 3.1.2 in [13]). How to introduce the complete generated sequence properly in training is necessary for avoiding the exposure bias in the autoregressive model.

Work performed as an intern at Microsoft. Lei Xie is the corresponding author. The research work is supported by the National Key Research and Development Program of China (No.2017YFB1002102).

http://dx.doi.org/10.21437/Interspeech.2019-2176

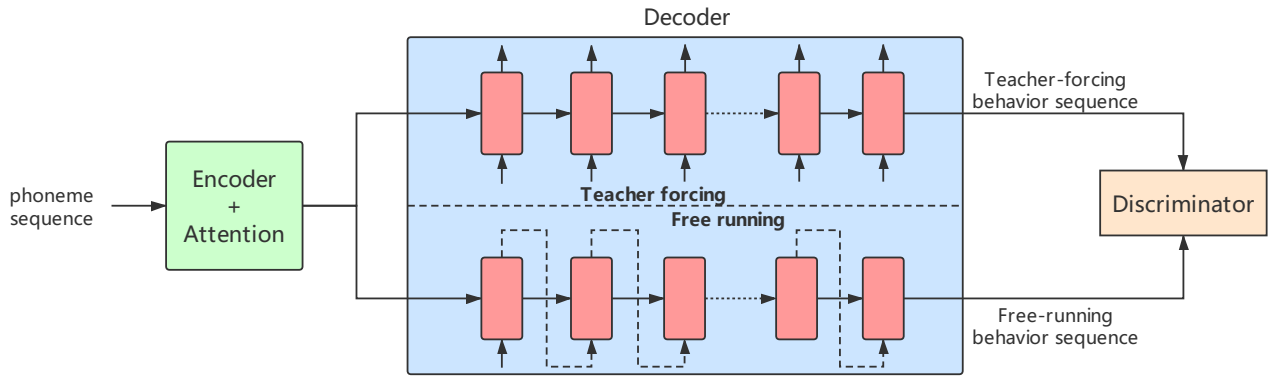

Figure 2: *The framework of GAN-based end-to-end TTS training algorithm*

Recently adversarial training has been used in many sequential training tasks, e.g. text classification[17], machine translation[18], speech recognition[19], etc. In domain adaptation [20], it has been successfully applied to help learning a domain-invariant representation to improve predictions and model generalization in the target domain. *Professor Forcing* [21] describes a GAN-based adversarial training for generative autoregressive model, which can make predictions with features that cannot be discriminated between real and model distributions. Inspired by it, we propose a new GAN-based, end-to-end TTS training algorithm to introduce generated sequence in training to avoid exposure bias in the autoregressive decoder.

In this paper, we will introduce *Professor Forcing* first, then present our proposed training framework and its training algorithm. Finally, we compare the performance of the training algorithms in different aspects with two subjective evaluation methods. The experimental results show that GAN-based training algorithm can significantly improve the model, including the naturalness of the generated speech and model generalization. We compared it with scheduled sampling to show it is more effective and proper for end-to-end TTS training.

2. Methods

2.1. Professor Forcing

There are two modules in *Professor Forcing*, a generative RNN (generator) and a discriminator. To introduce the complete predicted sequence in training, the generator will generate sequences in two different modes, teacher forcing (TF) and free running (FR, iteratively generate the sequential predictions). Discriminator is trained as a probabilistic classifier to determine in which mode the behavior sequence b (chosen hidden states and output values) is generated.

The training process of *Professor Forcing* is different from teacher forcing. There are two training objectives for the generator. The first one is to maximize the likelihood of data (depending on the task) using the output sequence generated in the teacher forcing mode. The second one is to equalize the discriminator so as to force the distributions of hidden states to be close to each other. This adversarial process reduces the discrepancy between real and model distributions of the model.

When the end-to-end TTS tries to synthesize a long sentence, it becomes vulnerable to the error accumulation in the AR process. Inspired by *Professor Forcing*, we propose a GAN-based end-to-end TTS training algorithm to train a better autoregressive decoder by avoiding exposure bias.

2.2. GAN-based End-to-end TTS Training Algorithm

There have been some studies on GAN in TTS in the past two years, such as GAN-based post filter [22], and GAN-based multi-task for TTS [23, 24]. These algorithms focus on the generated output sequence of acoustic model, trying to make the outputs be closer to the real data. Our proposed algorithm is different. Specifically, our algorithm focuses on the hidden states of the autoregressive decoder in the end-to-end TTS model, trying to make the behavior sequences generated in different modes be similar to each other. The proposed algorithm is introduced in two parts, in the training framework and model structure, and the training algorithm.

2.2.1. Training Framework & Model Structure

As shown in Fig.2, we have two models, which are the original end-to-end TTS model as generator, and a discriminator. In this paper, we adopt *Tacotron2* [10] as the generator, which has shown good performance in generating high-quality speech. The model structure of discriminator in *Professor forcing* is too simple, which can easily lead to training failure or no convergence. To meet our requirements for stable and effective training, we propose a new model structure based on *Self-Attention GAN (SAGAN)* [25] in the discriminator.

Fig.3 shows the model structure of the discriminator, which has two main components: a linear module and masked (unidirectional) self attention. Linear module is composed of a fully connected layer, spectrum normalization and an activation layer (leaky ReLU). Spectrum normalization [26] can help control Lipschitz constant to stabilize the training of the discriminator and back propagate more effective gradients to the generator, in case of mode collapse and a non-converging generator. Self-attention has shown excellent performance on modelling long-range dependency in many tasks, including: image generation [25], machine translation [27] and TTS [28]. Because autoregressive decoder is a uni-directional model, the cumulative errors are caused by its history. So we adopt masked self-attention in this work to make the discriminator focus on the impact of the history.

2.2.2. Training Algorithm

The discriminator is trained to distinguish in which mode the behavior sequence is generated, either teacher forcing or free

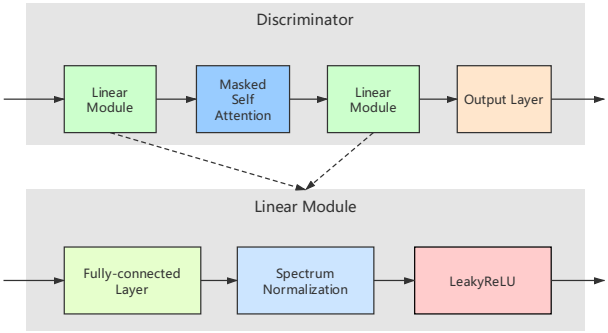

Figure 3: *Model architecture of the discriminator*

running. But the training criterion is different. We train it by minimizing the hinge version of the adversarial loss, which has shown good performance in GAN-based image generation.

$$\begin{aligned} L_D = &- \mathbb{E}_{(x,y)\sim data}[min(0, -1 + D(B_t(x,y)))] \\ &- \mathbb{E}_{(x,y)\sim data}[min(0, -1 - D(B_f(x)))] \end{aligned} \tag{3}$$

$x, D(*)$ refer to the input sequence and the classification results of the discriminator. We adopt the sequence which is composed of the hidden states of the attention RNN layer and the decoder RNN layer in Tacotron2 as the behavior sequence b to the discriminator. $B_t(*)$ and $B_f(*)$ refer to the behavior sequences generated in teacher forcing mode and free running mode, respectively.

The generator (TTS model) has two tasks in the framework. The first one is to minimize the loss L_T in Tacotron2 between the sequence generated in teacher forcing mode and the target sequence. The second task is to fool the discriminator by making the teacher-forcing and the free-running behavior sequences to be similar, distributions wise. We add a weighting coefficient to balance the two losses for more stabilised training. The training criterion of the generator is defined as

$$L_G = L_T - \alpha * (D(B_f(x)) - D(B_t(x,y))) \tag{4}$$

Eq.4 can be considered as a regularizer to restrain the model from over-fitting the distribution of teacher forcing in the training stage.

The training process has two phases: 1. we pre-train an end-to-end TTS model in the teacher forcing mode; 2. we train the TTS model and discriminator in turn. When the discriminator performance is below a lower bound, we will not back propagate the "bad" gradients from discriminator to update the generator parameters. Also, we will clamp the discriminator performance to a preset upper bound so as to prevent the discriminator from being too good to continue the training process. So we often test the accuracy every hundreds of steps. The detailed training algorithm is shown below.

GAN-based end-to-end TTS training algorithm

Input:
Training set: $D = \{x_k, y_k\}_{k=1}^{K}$
x_k: phoneme sequence, y_k: acoustic feature sequence
Steps for pre-training and GAN-based training: N_p, N_g
The required range of the accuracy: $[R_L, R_U]$
The period of testing discriminator accuracy: N_s
Output:
θ_g: TTS model
Initialize TTS model θ_g, discriminator θ_d
Initialize states $s_g = False$, $s_d = True$
Pre-train θ_g in teacher forcing mode for N_p steps.
for $i = 0; i < N_g; i = i + 1$ **do**
  Read a batch from D, and decode it in two modes
  Update θ_g
    if $s_g == False$
      Back propagate the gradient of L_T, update θ_g
    else
      Back propagate the gradient of L_G, update θ_g
  Update θ_d
    if $s_d == True$
      Back propagate the gradient of L_D, update θ_d
  if i mod N_s == 0, update s_g, s_d
    Get $accuracy$ of the discriminator on the training set
    if $accuracy > R_L$, $s_g = True$; **else**, $s_g = False$
    if $accuracy < R_U$, $s_d = True$; **else**, $s_d = False$
end for
return θ_g

3. Experiments

3.1. Training Setup

We use Tacotron2 [10] as TTS model, include WaveNet as vocoder for all experiments. We use one-hot feature as input, which contains phonemes, punctuation and the blank between two adjacent words. The model output is an 80-channel Mel spectrum (12.5 ms frame shift, 50 ms frame length), one frame at a time. The model structure of the discriminator has been shown in Fig.3, which has 1536-dim input (1024-dim 2nd RNN layer's output in decoder and 512-dim attention context), 512-dim hidden size, and 1-dim output.

When we calculate L_T in GAN-based algorithm, teacher forcing can also be replaced with scheduled sampling to generate sequence. We train four TTS models with 4 different training algorithms: teacher forcing (TF), scheduled sampling (SS), GAN-based algorithm with teacher forcing (TF-GAN) and GAN-based algorithm with scheduled sampling (SS-GAN, replace TF in TF-GAN with SS). These experiments are performed based on an American English speech data set, which has 14 hours of speech, recorded by a single female speaker.

All models are trained with a batch size of 128 sequences. We train these models using the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. The learning rate is exponentially decayed from 10^{-3} to 10^{-5} after 50,000 iterations. The TF model trained with 100,000 steps is set to be the baseline model. In SS, TF-GAN and SS-GAN training, we adopt the TF model trained with 50,000 steps as the pre-trained model, and train it for another 50,000 steps with these algorithms. The scheduled sampling strategy is to use real data with a linear decay, from probability 1 to 0.5, in the first 50,000 steps. We set the initial learning rate $lr_g = 10^{-3}, lr_d = 10^{-3}$, adversarial weight $\alpha = 10^{-3}$ for GAN-based algorithms. The range of the required discriminator accuracy is set to $75\% \sim 97\%$.

3.2. Subjective Evaluation

We design two TTS test sets to compare these algorithms in two aspects, speech quality and model generalization (stability). We use the common test set, which contains 50 typical sentences used in news and general conversation, to compare the performance of these models in speech quality and naturalness by a CMOS test. Each pair of samples is rated by 10 native

English speakers on a scale from -3 to 3 with 1 point discrete increments. Another test set containing 225 sentences is used to evaluate generalization of these models with an intelligibility test. These sentences have richer text content, such as long sentence, URL, the sequence of numbers or characters, abbreviation, etc. The test sentences and their contextual information are not well covered in the training set, so that the audios synthesized by these sentences tend to have lower intelligibility. We use it to evaluate the generalization capability of these models by the corresponding diagnostic sentence level intelligibility tests. The listeners need to mark a sentence unintelligible when any part of it is unintelligible in listening.[1]

Table 1: *The results of the CMOS tests*

System B	CMOS	Preference (%)			p-value
		TF	Neutral	System B	
SS	-0.04 $\pm$ 0.11	42.80	16.60	40.60	0.22
TF-GAN	0.10 $\pm$ 0.07	22.73	49.21	28.06	0.02
SS-GAN	0.01 $\pm$ 0.12	28.60	39.00	32.40	0.40

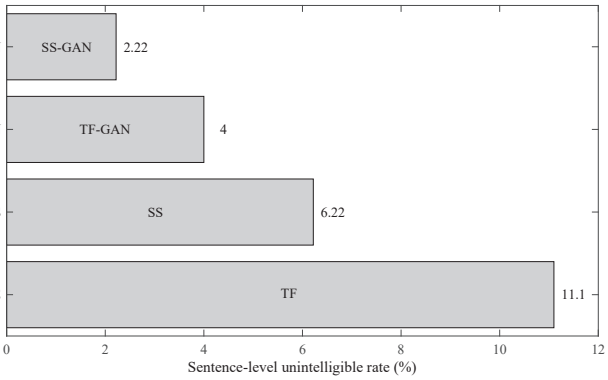

Figure 4: *The results of the intelligibility tests*

Table.1 and Fig.4 show the results of these two subjective evaluations. Compared with TF, both of CMOS score (-0.04) and preference (-2.2%) show that the performance of SS is worse, but the intelligibility is improved on the pathological test set. In the comparison between TF-GAN and TF, the votes on TF-GAN is 5.33% more than TF when 50% of the votes are neutral. TF-GAN shows significantly better performance than TF with a higher CMOS (0.1) and preference (5.33%). It also achieves a lower unintelligible rate (4%) than TF (11.1%) and SS (6.22%). So compared with SS, GAN-based training algorithm is more effective. It can improve both naturalness and generalization for end-to-end TTS. As the combination of SS and GAN-based training algorithm, SS-GAN can further improve the intelligibility rate (2.22%). SS-GAN does not achieve improvement in speech quality and naturalness due to SS, but has better performance in model generalization.

3.3. Analysis

We also try other decay strategies for scheduled sampling, but these experiments show that when we lower the sampling probability, more deterioration of the speech quality. Fig.5 shows the Mel spectrum synthesized by the models trained with TF and SS. When we linearly decay the sampling probability from 1 to 0 within 50,000 iterations, the sound quality and clarity deteriorate significantly (shown in the middle part). It shows that calculating the frame-level loss for non-aligned data will lead to loss of model output quality, although predicted data is helpful for improving the generalization capability of TTS model. So we finally set the sampling probability to 0.5 to alleviate the misalignment between output and the target.

[1]Samples are available at `https://hhguo.github.io/demo/publications/GANTTS/index.html`

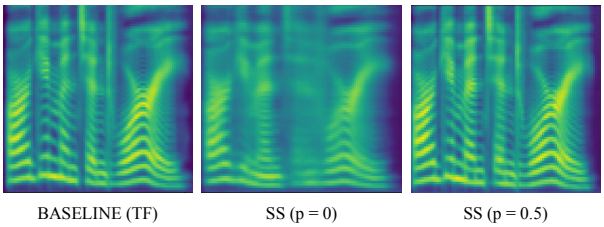

Figure 5: *Mel spectrum synthesized by different models*

To compare the performance before and after improving model generalization, we investigate the bad cases in the intelligibility test. We find that some long sentences can easily lead to garbled pronunciations, that is, the model suddenly starts to generate unintelligible and repeated garbled speech in decoding. Fig.6 shows the alignment and Mel spectrum of such a case. We hypothesize this problem is due to the fact that a long and unseen context in the sentence can lead to higher cumulative errors in decoding. These errors, in turn, can distract the attention to the correct context. After improving the generalization with the proposed algorithm, the decoder is more robust in decoding noisy sequence. In our test set, over 50% of these bad cases are fixed. The remaining bad cases with longer sentence and more complex contexts, may need better encoder to fix the problem.

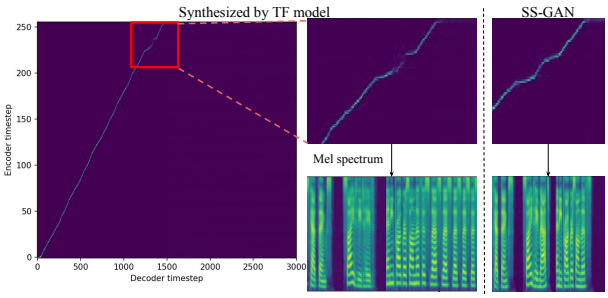

Figure 6: *The alignment and Mel spectrum of a long sentence*

4. Conclusions

This paper proposes a new GAN-based, end-to-end TTS training algorithm, which introduces the generated sequence to GAN training to avoid exposure bias in autoregressive decoder. Experimental results show that schedule sampling is harmful to synthesized speech quality, but can improve the model generalization capability of TTS model. Compared with scheduled sampling, our proposed algorithm improves both output quality and generalization of the model. By combining SS and GAN, we can further improve the generalization of the model by maintaining the speech quality and naturalness at the same level with a slight preference advantage of 3.8%.

5. References

[1] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *ICASSP*, 2000, pp. 1315–1318.

[2] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *ICASSP*, 2013, pp. 7962–7966.

[3] H. Zen, "Acoustic modeling in statistical parametric speech synthesis-from HMM to LSTM-RNN," in *MLSLP*, 2015.

[4] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *INTERSPEECH*, 2014.

[5] X. Wang, S. Takaki, and J. Yamagishi, "Autoregressive neural F0 model for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pp. 1406–1419, 2018.

[6] X. Wang, S. Takaki, and J. Yamagishi, "An autoregressive recurrent mixture density network for parametric speech synthesis," in *ICASSP*, 2017, pp. 4895–4899.

[7] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio." in *SSW*, 2016, p. 125.

[8] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. v. d. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *ICML*, 2018.

[9] 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," in *INTERSPEECH*, 2017.

[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 *ICASSP*, 2018, pp. 4779–4783.

[11] W. Ping, K. Peng, and J. Chen, "Clarinet: Parallel wave generation in end-to-end text-to-speech," *arXiv preprint arXiv:1807.07281*, 2018.

[12] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural computation*, pp. 270–280, 1989.

[13] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *ICLR*, 2015.

[14] A. Venkatraman, M. Hebert, and J. A. Bagnell, "Improving multi-step prediction of learned time series models," in *AAAI*, 2015.

[15] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.

[16] F. Huszár, "How (not) to train your generative model: Scheduled sampling, likelihood, adversary?" *arXiv preprint arXiv:1511.05101*, 2015.

[17] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," in *ICLR*, 2017.

[18] L. Wu, Y. Xia, L. Zhao, F. Tian, T. Qin, J. Lai, and T.-Y. Liu, "Adversarial neural machine translation," *ACML*, 2017.

[19] S. Sun, C.-F. Yeh, M.-Y. Hwang, M. Ostendorf, and L. Xie, "Domain adversarial training for accented speech recognition," in *ICASSP*, 2018, pp. 4854–4858.

[20] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, pp. 2096–2030, 2016.

[21] A. Goyal, A. Lamb, Y. Zhang, S. Zhang, A. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," in *Advances In Neural Information Processing Systems*, 2016, pp. 4601–4609.

[22] T. Kaneko, S. Takaki, H. Kameoka, and J. Yamagishi, "Generative adversarial network-based postfilter for STFT Spectrograms," in *INTERSPEECH*, 2017.

[23] S. Yang, L. Xie, X. Chen, X. Lou, X. Zhu, D. Huang, and H. Li, "Statistical parametric speech synthesis using generative adversarial networks under a multi-task learning framework," in *ASRU*, 2017, pp. 685–691.

[24] Y. Saito, S. Takamichi, and H. Saruwatari, "Statistical parametric speech synthesis incorporating generative adversarial networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pp. 84–96, 2018.

[25] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[26] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *ICLR*, 2018.

[27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances In Neural Information Processing Systems*, 2017, pp. 5998–6008.

[28] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, and M. Zhou, "Close to human quality TTS with transformer," in *AAAI*, 2019.