HIGH FIDELITY SPEECH SYNTHESIS WITH ADVERSARIAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative adversarial networks have seen rapid development in recent years and have led to remarkable improvements in generative modelling of images. However, their application in the audio domain has received limited attention, and autoregressive models, such as WaveNet, remain the state of the art in generative modelling of audio signals such as human speech. To address this paucity, we introduce GAN-TTS, a Generative Adversarial Network for Text-to-Speech. Our architecture is composed of a conditional feed-forward generator producing raw speech audio, and an ensemble of discriminators which operate on random windows of different sizes. The discriminators analyse the audio both in terms of general realism, as well as how well the audio corresponds to the utterance that should be pronounced. To measure the performance of GAN-TTS, we employ both subjective human evaluation (MOS – Mean Opinion Score), as well as novel quantitative metrics (Fréchet DeepSpeech Distance and Kernel DeepSpeech Distance), which we find to be well correlated with MOS. We show that GAN-TTS is capable of generating high-fidelity speech with naturalness comparable to the state-of-the-art models, and unlike autoregressive models, it is highly parallelisable thanks to an efficient feed-forward generator. Listen to GAN-TTS reading this abstract at `http://tiny.cc/gantts`.

1 INTRODUCTION

The Text-to-Speech (TTS) task consists in the conversion of text into speech audio. In recent years, the TTS field has seen remarkable progress, sparked by the development of neural autoregressive models for raw audio waveforms such as WaveNet (van den Oord et al., 2016), SampleRNN (Mehri et al., 2017) and WaveRNN (Kalchbrenner et al., 2018). A notable limitation of these models is that they are difficult to parallelise over time: they predict each time step of an audio signal in sequence, which is computationally expensive and often impractical. A lot of recent research on neural models for TTS has focused on improving parallelism by predicting multiple time steps in parallel, e.g. using flow-based models (van den Oord et al., 2018; Ping et al., 2019; Prenger et al., 2019; Kim et al., 2019). Such highly parallelisable models are more suitable to run efficiently on modern hardware.

An alternative approach for parallel waveform generation would be to use Generative Adversarial Networks (GANs, Goodfellow et al., 2014). GANs currently constitute one of the dominant paradigms for generative modelling of images, and they are able to produce high-fidelity samples that are almost indistinguishable from real data. However, their application to audio generation tasks has seen relatively limited success so far. In this paper, we explore raw waveform generation with GANs, and demonstrate that adversarially trained feed-forward generators are indeed able to synthesise high-fidelity speech audio. Our contributions are as follows:

- We introduce GAN-TTS, a Generative Adversarial Network for text-conditional high-fidelity speech synthesis. Its feed-forward generator is a convolutional neural network, coupled with an ensemble of multiple discriminators which evaluate the generated (and real) audio based on multi-frequency random windows. Notably, some discriminators take the linguistic conditioning into account (so they can measure how well the generated audio corresponds to the input utterance), while others ignore the conditioning, and can only assess the general realism of the audio.

- We propose a family of quantitative metrics for speech generation based on *Fréchet Inception Distance* (FID, Heusel et al., 2017) and *Kernel Inception Distance* (KID, Bińkowski et al., 2018), where we replace the Inception image recognition network with the DeepSpeech audio recognition network.
- We present quantitative and subjective evaluation of TTS-GAN and its ablations, demonstrating the importance of our architectural choices. Our best-performing model achieves a MOS of 4.2, which is comparable to the state-of-the-art WaveNet MOS of 4.4, and establishes GANs as a viable option for efficient TTS.

2 Related Work

2.1 Audio generation

Most neural models for audio generation are likelihood-based: they represent an explicit probability distribution and the likelihood of the observed data is maximised under this distribution. Autoregressive models achieve this by factorising the joint distribution into a product of conditional distributions (van den Oord et al., 2016; Mehri et al., 2017; Kalchbrenner et al., 2018; Arik et al., 2017). Another strategy is to use an invertible feed-forward neural network to model the joint density directly (Prenger et al., 2019; Kim et al., 2019). Alternatively, an invertible feed-forward model can be trained by distilling an autoregressive model using probability density distillation (van den Oord et al., 2018; Ping et al., 2019), which enables it to focus on particular modes. Such mode-seeking behaviour is often desirable in conditional generation settings: we want the generated speech signals to sound realistic and correspond to the given text, but we are not interested in modelling every possible variation that occurs in the data. This reduces model capacity requirements, because parts of the data distribution may be ignored. Note that adversarial models exhibit similar behaviour, but without the distillation and invertibility requirements.

Many audio generation models, including all of those discussed so far, operate in the waveform domain: they directly model the amplitude of the waveform as it evolves over time. This is in stark contrast to most audio models designed for discriminative tasks (e.g. audio classification): such models tend to operate on time-frequency representations of audio (*spectrograms*), which encode certain inductive biases with respect to the human perception of sound, and usually discard all phase information in the signal. While phase information is often inconsequential for discriminative tasks, generated audio signals must have a realistic phase component, because fidelity as judged by humans is severely affected otherwise. Because no special treatment for the phase component of the signal is required when generating directly in the waveform domain, this is usually more practical.

Tacotron (Wang et al., 2017) and MelNet (Vasquez & Lewis, 2019) constitute notable exceptions, and they use the Griffin-Lim algorithm (Griffin & Lim, 1984) to reconstruct missing phase information, which the models themselves do not generate. Models like Deep Voice 2 & 3 (Gibiansky et al., 2017; Ping et al., 2018) and Tacotron 2 (Shen et al., 2018) achieve a compromise by first generating a spectral representation, and then using a separate autoregressive model to turn it into a waveform and fill in any missing spectral information. Because the generated spectrograms are imperfect, the waveform model has the additional task of correcting any mistakes. Char2wav (Sotelo et al., 2017) uses intermediate vocoder features in a similar fashion.

2.2 Generative adversarial networks

Generative Adversarial Networks (GANs, Goodfellow et al., 2014) form a subclass of implicit generative models that relies on adversarial training of two networks: the *generator*, which attempts to produce samples that mimic the reference distribution, and the *discriminator*, which tries to differentiate between real and generated samples and, in doing so, provides a useful gradient signal to the generator. Following rapid development, GANs have achieved state-of-the-art results in image (Zhang et al., 2019; Brock et al., 2019; Karras et al., 2019) and video generation (Clark et al., 2019), and have been successfully applied for unsupervised feature learning (Donahue et al., 2017; Dumoulin et al., 2017a; Donahue & Simonyan, 2019), among many other applications.

Despite achieving impressive results in these domains, limited work has so far shown good performance of GANs in audio generation. Two notable exceptions include WaveGAN (Donahue et al.,

2019) and GANSynth (Engel et al., 2019), which both successfully applied GANs to simple datasets of audio data. The former is the most similar to this work in the sense that it uses GANs to generate raw audio; results were obtained for a dataset of spoken commands of digits from zero to nine. The latter provides state-of-the-art results for a dataset of single note recordings from various musical instruments (NSynth, Engel et al., 2017) by training GANs to generate invertible spectrograms of the notes. Neekhara et al. (2019) propose an adversarial vocoder model that is able to synthesise magnitude spectrograms from mel-spectrograms generated by Tacotron 2, and combine this with phase estimation using the Local Weighted Sums technique (Le Roux et al., 2010).

To the best of our knowledge, GANs have not yet been applied at large scale to non-visual domains. Two seconds of audio at 24kHz[1] has a dimensionality of 48000, which is comparable to RGB images at 128×128 resolution. Until recently, high-quality GAN-generated images at such or higher resolution were uncommon (Zhang et al., 2019; Karras et al., 2019), and it was not clear that training GANs at scale would lead to extensive improvements (Brock et al., 2019).

Multiple discriminators have been used in GANs for different purposes. For images, Denton et al. (2015); Zhang et al. (2017); Karras et al. (2018) proposed to use separate discriminators for different resolutions. Similar approaches have also been used in image-to-image transfer (Huang et al., 2018) and video synthesis (Saito & Saito, 2018). Clark et al. (2019), on the other hand, combine a 3D-discriminator that scores the video at lower resolution and a 2D-frame discriminator which looks at individual frames. In adversarial feature learning, Donahue & Simonyan (2019) combine outputs from three discriminators to differentiate between joint distributions of images and latents. Discriminators operating on windows of the input have been used in adversarial texture synthesis (Li & Wand, 2016) and image translation (Isola et al., 2017; Zhu et al., 2017).

3 GAN-TTS

3.1 Dataset

Our text-to-speech models are trained on a dataset which contains high-fidelity audio of human speech with the corresponding linguistic features and pitch information. The linguistic features encode phonetic and duration information, while the pitch is represented by the logarithmic fundamental frequency $\log F_0$. In total, there are 567 features. We do not use ground-truth duration and pitch for subjective evaluation; we instead use duration and pitch predicted by separate models.

The dataset is formed of variable-length audio clips containing single sequences, spoken by a professional voice actor in North American English. For training, we sample 2 second windows (filtering out shorter examples) together with corresponding linguistic features. The total length of the filtered dataset is 44 hours. The sampling frequency of the audio is 24kHz, while the linguistic features and pitch are computed for 5ms windows (at 200Hz). This means that the generator network needs to learn how to convert the linguistic features and pitch into raw audio, while upsampling the signal by a factor of 120. We apply a μ-law transform to account for the logarithmic perception of volume (see Appendix C).

3.2 Generator

A summary of generator G's architecture is presented in Table 2 in Appendix A.2. The input to G is a sequence of linguistic and pitch features at 200Hz, and its output is the raw waveform at 24kHz. The generator is composed of seven blocks (GBlocks, Figure 1a), each of which is a stack of two residual blocks (He et al., 2016). As the generator is producing raw audio (e.g. a 2s training clip corresponds to a sequence of 48000 samples), we use dilated convolutions (Yu & Koltun, 2016) to ensure that the receptive field of G is large enough to capture long-term dependencies. The four kernel size-3 convolutions in each GBlock have increasing dilation factors: $1, 2, 4, 8$. Convolutions are preceded by Conditional Batch Normalisation (Dumoulin et al., 2017b), conditioned on the linear embeddings of the noise term $z \sim \mathcal{N}(0, \boldsymbol{I}_{128})$ in the single-speaker case, or the concatenation of z and a one-hot representation of the speaker ID in the multi-speaker case. The embeddings are different for each BatchNorm instance. A GBlock contains two skip connections, the first of which performs

[1] 24kHz is a commonly used frequency for speech, because the absence of frequencies above 12kHz does not meaningfully affect fidelity.

upsampling if the output frequency is higher than the input, and it also contains a size-1 convolution if the number of output channels is different from the input. GBlocks 3–7 gradually upsample the temporal dimension of hidden representations by factors of $2, 2, 2, 3, 5$, while the number of channels is reduced by GBlocks 3, 6 and 7 (by a factor of 2 each). The final convolutional layer with *Tanh* activation produces a single-channel audio waveform.

3.3 Ensemble of Random Window Discriminators

Instead of a single discriminator, we use an ensemble of Random Window Discriminators (RWDs) which operate on randomly sub-sampled fragments of the real or generated samples. The ensemble allows for the evaluation of audio in different complementary ways, and is obtained by taking a Cartesian product of two parameter spaces: (i) the size of the random windows fed into the discriminator; (ii) whether a discriminator is conditioned on linguistic and pitch features. For example, in our best-performing model, we consider five window sizes ($240, 480, 960, 1920, 3600$ samples), which yields 10 discriminators in total. Notably, the number of discriminators only affects the training computation requirements, as at inference time only the generator network is used, while the discriminators are discarded. However, thanks to the use of relatively short random windows, the proposed ensemble leads to faster training than conventional discriminators.

Using random windows of different size, rather than the full generated sample, has a data augmentation effect and also reduces the computational complexity of RWDs, as explained next. In the first layer of each discriminator, we reshape (downsample) the input raw waveform to a constant temporal dimension $\omega = 240$ by moving consecutive blocks of samples into the channel dimension, i.e. from $[\omega k, 1]$ to $[\omega, k]$, where k is the downsampling factor (e.g. $k = 8$ for input window size 1920). This way, all the RWDs have the same architecture and similar computational complexity despite different window sizes. We confirm these design choices experimentally in Section 5.

The conditional discriminators have access to linguistic and pitch features, and can measure whether the generated audio matches the input conditioning. This means that random windows in conditional discriminators need to be aligned with the conditioning frequency to preserve the correspondence between the waveform and linguistic features within the sampled window. This limits the valid sampling to that of the frequency of the conditioning signal (200Hz, or every 5ms). The unconditional discriminators, on the contrary, only evaluate whether the generated audio sounds realistic regardless of the conditioning. The random windows for these discriminators are sampled without constraints at full 24kHz frequency, which further increases the amount of training data. More formally, we define conditional and unconditional RWDs as stochastic functions:

$$\mathsf{cRWD}_{k,\omega}(x, c; \theta) = \mathsf{D}^{\mathsf{C}}_{k}(x_{\mathsf{j}:\mathsf{j}+\omega k}, c_{\mathsf{j}/\lambda:(\mathsf{j}+\omega k)/\lambda}; \theta), \qquad \mathsf{j} \sim \mathcal{U}\left(\{0, \lambda, 2\lambda, \ldots, N - \omega k\}\right) \tag{1}$$

$$\mathsf{uRWD}_{k,\omega}(x; \theta) = \mathsf{D}^{\mathsf{U}}_{k}(x_{\mathsf{j}:\mathsf{j}+\omega k}; \theta), \qquad \mathsf{j} \sim \mathcal{U}\left(\{0, 1, \ldots, N - \omega k\}\right), \tag{2}$$

where x and c are respectively the waveform and linguistic features, θ is set of network parameters, and $\lambda = 120$ is a frequency ratio between x and c.

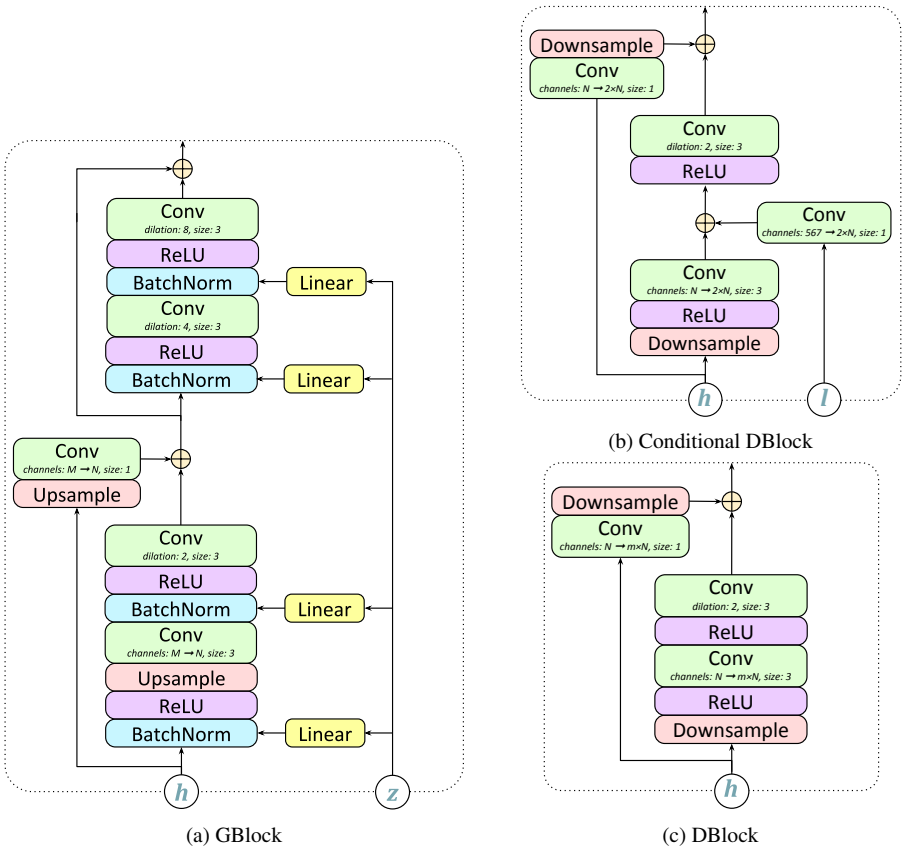
The final ensemble discriminator combines 10 different RWD's:

$$\mathsf{RWD}^{*}_{\omega}(x, c; \theta^{*}) = \sum_{k \in \{1,2,4,8,15\}} \mathsf{cRWD}_{k,\omega}(x, c; \theta_k) + \mathsf{uRWD}_{k,\omega}(x; \theta'_k), \qquad \theta^{*} = \bigcup_{k} (\theta_k \cup \theta'_k). \tag{3}$$

In Section 5 we describe other combinations of RWDs as well as a full, deterministic discriminator which we used in our ablation study.

3.4 Discriminator Architecture

The full discriminator architecture is shown in Figure 2. The discriminators consists of blocks (DBlocks) that are similar to the GBlocks used in the generator, but without batch normalisation. The architectures of standard and conditional DBlocks are shown in Figures 1b and 1c respectively. The only difference between the two DBlocks is that in the conditional DBlock, the embedding of the linguistic features is added after the first convolution. The first and the last two DBlocks do not downsample (i.e. keep the temporal dimension fixed). Apart from that, we add at least two downsampling blocks in the middle, with downsample factors depending on k, so as to match the



(a) GBlock (b) Conditional DBlock (c) DBlock

Figure 1: Residual blocks used in the model. Convolutional layers have the same number of input and output channels and no dilation unless stated otherwise. h - hidden layer representation, l - linguistic features, z - noise vector, $m \in \{1, 2\}$ - channel multiplier, M- G's input channels, $M = 2N$ in blocks 3, 6, 7, and $M = N$ otherwise; *size* refers to kernel size.

frequency of the linguistic features (see Appendix A.2 for details). Unconditional RWDs are composed entirely of DBlocks. In conditional RWDs, the input waveform is gradually downsampled by DBlocks, until the temporal dimension of the activation is equal to that of the conditioning, at which point a conditional DBlock is used. This joint information is then passed to the remaining DBlocks, whose final output is average-pooled to obtain a scalar. The dilation factors in the DBlocks' convolutions follow the pattern $1, 2, 1, 2$ – unlike the generator, the discriminator operates on a relatively small window, and we did not observe any benefit from using larger dilation factors.

4 Evaluation

We provide subjective human evaluation of our model using Mean Opinion Scores (*MOS*), as well as quantitative metrics.

4.1 MOS

We evaluate our model on a set of 1000 sentences, using human evaluators. Each evaluator was asked to mark the subjective *naturalness* of a sentence on a 1-5 Likert scale, comparing to the scores reported by van den Oord et al. (2018) for WaveNet and Parallel WaveNet.

Although our model was trained to generate 2 second audio clips with the starting point not necessarily aligned with the beginning of a sentence, we are able to generate samples of arbitrary length. This is feasible due to the fully convolutional nature of the generator and carried out using a *convolutional masking* trick, detailed in Appendix A.1. Human evaluators scored full sentences with a length of up to 15 seconds.

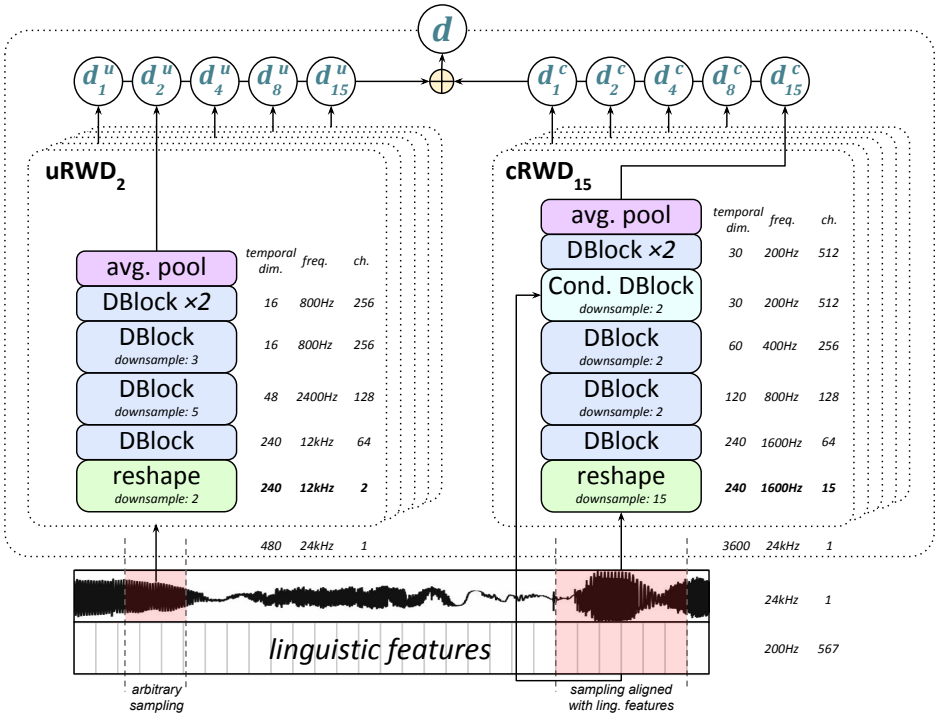


Figure 2: *Multiple Random Window Discriminator* architecture. The discriminator combines outputs from 5 unconditional (uRWDs, left) and 5 conditional (cRWDs, right) discriminators; one of each group is detailed in the diagram. The number of downsampling blocks is fixed for uRWDs and depends on the input window size ωk for cRWDs, see Table 3. $ch.$ - number of output channels.

4.2 Speech Distances

We introduce a family of quantitative metrics for generative models of speech, which include the unconditional and conditional *Fréchet DeepSpeech Distance (FDSD, cFDSD)* and *Kernel DeepSpeech Distance (KDSD, cKDSD)*. These metrics follow common metrics used in evaluation of GANs for images, *Fréchet Inception Distance* (FID, Heusel et al., 2017) and *Kernel Inception Distance* (KID, Bińkowski et al., 2018).

FID and KID compute the *Fréchet distance* and the *Maximum Mean Discrepancy* (MMD, Gretton et al., 2012) respectively between representations of reference and generated distributions extracted from a pre-trained *Inception* network (Szegedy et al., 2016). To obtain analogous metrics for speech, we extract the features from an open-source implementation of an accurate speech recognition model, *DeepSpeech2* (Amodei et al., 2016). Specifically, we use the implementation available in the *NVIDIA OpenSeq2Seq* library (Kuchaiev et al., 2018) and extract features from the last layer, whose output is used in the CTC loss during training. We use representations in the resulting feature space to compute the Fréchet distance and MMD (See Appendix B.1 for details).

We note that Kilgour et al. (2019) proposed a similar metric, *Fréchet Audio Distance*. This metric, however, has been designed for music datasets and uses a music classifier as a feature extractor; therefore it is not well-suited to evaluate text-to-speech models.

As conditioning plays a crucial role in our task, we compute two variants of these metrics, conditional (cFDSD, cKDSD) and unconditional (FDSD, KDSD). Both Fréchet and Kernel distance provide scores with respect to a *reference real* sample and require both the real sample and the generated one to be independent and identically distributed. Assume that variables x^{real} and x^{G} are drawn from the real and and generated distributions, while c is drawn from the distribution of linguistic features. In the conditional case, cFDSD and cKDSD compute distances between conditional distributions $p(\mathrm{x}^{G}|\mathrm{c})$ and $p(\mathrm{x}^{real}|\mathrm{c})$. In the unconditional case, FDSD and KDSD compare $p(\mathrm{x}^{G})$ and $p(\mathrm{x}^{real})$.

Both metrics are estimated using 10,000 generated and reference samples, drawn independently with the same (in the conditional case), or independent (in the unconditional case) linguistic features. This procedure is detailed in Appendix B.3.

The main reason for using both Fréchet and Kernel distances is the popularity of FID in the image domain, despite the issue of its biased estimator, as shown by Bińkowski et al. (2018). Thanks to the availability of an unbiased estimator of MMD, this issue does not apply to kernel-based distances. For instance, they yield zero values for real data, which allows comparison in the conditional case. We give more details on these distances in Appendix B.2.

5 EXPERIMENTS

In this section we discuss the experiments, comparing GAN-TTS with WaveNet and carrying out ablations that validate our architectural choices.

As mentioned in Section 3, the main architectural choices made in our model include the use of multiple RWDs, conditional and unconditional, with a number of different downsampling factors. We thus consider the following ablations of our best-performing model:

1. full-input discriminator $\mathsf{FullD} = \mathsf{D}_1^{\mathsf{C}}$,
2. single conditional RWD: cRWD_1,
3. multiple conditional RWDs: $\mathsf{cRWD}_{\{1,2,4,8,15\}} = \sum_{k \in \{1,2,4,8,15\}} \mathsf{cRWD}_k$,
4. single conditional and single unconditional RWD: $\mathsf{cRWD}_1 + \mathsf{uRWD}_1$,
5. five independent cRWDs and uRWDs:
   $(\mathsf{cRWD}_1 + \mathsf{uRWD}_1)^{\times 5}(x, c) := \sum_{i=1}^{5} \mathsf{cRWD}_1(x, c; \theta_i) + \mathsf{uRWD}_1(x; \theta'_i)$,
6. 10 RWDs without downsampling but with different window sizes:
   $\mathsf{RWD}_{1,240 \times \{1,2,4,8,15\}} = \sum_{k \in \{1,2,4,8,15\}} (\mathsf{cRWD}_{1,240k} + \mathsf{uRWD}_{1,240k})$
7. 10 RWDs with longer window: RWD^*_{480}.

All other parameters of these models were the same as in the proposed one. In Appendix D we present details of the hyperparameters used during training.

5.1 RESULTS

model	MOS	FDSD	cFDSD	KDSD $\times 10^5$	cKDSD $\times 10^5$
natural speech	4.55 ± 0.075	0.161	N/A	0	0
WaveNet, van den Oord et al. (2016)	4.41 ± 0.069				
Parallel WaveNet, van den Oord et al. (2018)	4.41 ± 0.078				
FullD	1.889 ± 0.057	4.51	4.46	785	782
cRWD_1	3.394 ± 0.058	0.362	0.247	35.2	30.9
$\mathsf{cRWD}_{\{1,2,4,8,15\}}$	3.498 ± 0.059	0.398	0.284	42.1	37.9
$\mathsf{cRWD}_1 + \mathsf{uRWD}_1$	3.502 ± 0.057	0.259	0.144	16.6	12.3
$(\mathsf{cRWD}_1 + \mathsf{uRWD}_1)^{\times 5}$	3.526 ± 0.054	0.194	0.073	5.59	1.34
$\mathsf{RWD}_{1,240 \times \{1,2,4,8,15\}}$	4.154 ± 0.050	0.184	0.061	3.73	0.54
RWD^*_{480}	4.195 ± 0.045	0.193	0.069	5.28	0.98
GAN-TTS (RWD^*)	4.213 ± 0.046	0.184	0.060	3.84	0.37

Table 1: Results from prior work, the ablation study and the proposed model. Mean opinion scores for natural speech, WaveNet and Parallel WaveNet are taken from van den Oord et al. (2018) and are not directly comparable due to dataset differences. For natural speech we present estimated FDSD – non-zero due to the bias of the estimator – and theoretical values of KDSD and cKDSD. cFDSD is unavailable; see Appendix B.2.

Table 1 presents quantitative evaluations of the proposed model, together with benchmarks and other variants of GAN-TTS that we considered in this work.

Our best model achieves worse yet comparable scores to the strong baselines, WaveNet and Parallel WaveNet. This performance, however, has not yet been achieved using adversarial techniques and is still very good, especially when compared to parametric text-to-speech models. These results are however not directly comparable due to dataset differences; for instance WaveNet and Parallel WaveNet were trained on 65 hours of data, a bit more than GAN-TTS.

Our ablation study confirms the importance of combining multiple RWDs. The deterministic full discriminator achieved the worst scores. All multiple-RWD models achieved better results than a single cRWD_1; all models that used unconditional RWDs were better than those that did not. Comparing 10-discriminator models, it is clear that combinations of different window sizes were beneficial, as a simple ensemble of 10 fixed-size windows was significantly worse. All three 10-RWD models with varying discriminator sizes achieved similar mean opinion scores, with the downsampling model with base window size 240 performing best.

We also observe a noticeable correlation between human evaluation scores (MOS) and the proposed metrics, which demonstrates that these metrics are well-suited for the evaluation of neural audio synthesis models.

5.2 Discussion

Random window discriminators. Although it is difficult to say why RWDs work much better than the full discriminator, we conjecture that this is because of the relative simplicity of the distributions that the former must discriminate between, and the number of different samples we can draw from these distributions. For example, the largest window discriminators used in our best model discriminate between distributions supported on $\mathbb{R}^{3600}$, and there are respectively 371 and 44,401 different windows that can be sub-sampled from a 2s clip (real or generated) by conditional and unconditional RWDs of effective window size 3600. The full discriminator, on the other hand, always sees full real or generated examples sampled from a distribution supported on $\mathbb{R}^{48000}$.

Computational efficiency. Our Generator has a larger receptive field (590ms, i.e. 118 steps at the frequency of the linguistic features) and three times fewer FLOPs (0.638 MFLOP/sample) than Parallel WaveNet (receptive field size: 320ms, 1.966 MFLOP/sample). However, the discriminators used in our ensemble compare windows of shorter sizes, from 10ms to 150ms. Since these windows are much shorter than the entire generated clips, training with ensembles of such RWDs is faster than with FullD. In terms of depth, our generator has 30 layers, which is a half of Parallel WaveNet's, while the depths of the discriminators vary between 11 and 17 layers, as discussed in Appendix A.2.

Stability. The proposed model enjoyed very stable training, with gradual improvement of subjective sample quality and decreasing values of the proposed metrics. Despite training for as many as 1 million steps, we have not experienced model collapses often reported in GAN literature and studied in detail by Brock et al. (2019).

6 Conclusion

We have introduced GAN-TTS, a GAN for raw audio text-to-speech generation. Unlike state-of-the-art text-to-speech models, GAN-TTS is adversarially trained and the resulting generator is a feed-forward convolutional network. This allows for very efficient audio generation, which is important in practical applications. Our architectural exploration lead to the development of a model with an ensemble of unconditional and conditional *Random Window Discriminators* operating at different window sizes, which respectively assess the realism of the generated speech and its correspondence with the input text. We showed in an ablation study that each of these components is instrumental to achieving good performance. We have also proposed a family of quantitative metrics for generative models of speech: *(conditional) Fréchet DeepSpeech Distance* and *(conditional) Kernel DeepSpeech Distance*, and demonstrated experimentally that these metrics rank models in line with Mean Opinion Scores obtained through human evaluation. As they are based on the publicly available DeepSpeech recognition model, they will be made available for the machine learning community. Our quantitative results as well as subjective evaluation of the generated samples showcase the feasibility of text-to-speech generation with GANs.

REFERENCES

Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse Engel, Linxi Fan, Christopher Fougner, Tony Han, Awni Hannun, Billy Jun, Patrick LeGresley, Libby Lin, Sharan Narang, Andrew Ng, Sherjil Ozair, Ryan Prenger, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Yi Wang, Zhiqian Wang, Chong Wang, Bo Xiao, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. Deep Speech 2: End-to-end speech recognition in English and Mandarin. In *ICML*, 2016.

Sercan Ö Arik, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, et al. Deep Voice: Real-time neural text-to-speech. In *ICML*, 2017.

Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *ICLR*, 2018.

Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. In *ICLR*, 2016.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

Aidan Clark, Jeff Donahue, and Karen Simonyan. Efficient video generation on complex datasets. *arXiv:1907.06571*, 2019.

Emily L Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a Laplacian pyramid of adversarial networks. In *NeurIPS*, 2015.

Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *ICLR*, 2019.

Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. *arXiv:1907.02544*, 2019.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *ICLR*, 2017.

Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. In *ICLR*, 2017a.

Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017b.

Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with WaveNet autoencoders. In *ICML*, 2017.

Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. GANSynth: Adversarial neural audio synthesis. In *ICLR*, 2019.

Andrew Gibiansky, Sercan Arik, Gregory Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep Voice 2: Multi-speaker neural text-to-speech. In *NeurIPS*, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *JMLR*, 2012.

Daniel Griffin and Jae Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 1984.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NeurIPS*, 2017.

Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aäron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *ICML*, 2018.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet audio distance: A metric for evaluating music enhancement algorithms. In *Interspeech*, 2019.

Sungwon Kim, Sang-Gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. FloWaveNet: A generative flow for raw audio. In *ICML*, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Oleksii Kuchaiev, Boris Ginsburg, Igor Gitman, Vitaly Lavrukhin, Carl Case, and Paulius Micikevicius. OpenSeq2Seq: Extensible toolkit for distributed and mixed precision training of sequence-to-sequence models. In *NLP-OSS*, 2018.

Jonathan Le Roux, Hirokazu Kameoka, Nobutaka Ono, and Shigeki Sagayama. Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency. In *DAFx*, 2010.

Chuan Li and Michael Wand. Precomputed real-time texture synthesis with Markovian generative adversarial networks. In *ECCV*, 2016.

Jae Hyun Lim and Jong Chul Ye. Geometric GAN. *arXiv:1705.02894*, 2017.

Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. SampleRNN: An unconditional end-to-end neural audio generation model. In *ICLR*, 2017.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

Paarth Neekhara, Chris Donahue, Miller Puckette, Shlomo Dubnov, and Julian McAuley. Expediting TTS synthesis with adversarial vocoding. In *Interspeech*, 2019.

Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep Voice 3: 2000-speaker neural text-to-speech. In *ICLR*, 2018.

Wei Ping, Kainan Peng, and Jitong Chen. ClariNet: Parallel wave generation in end-to-end text-to-speech. In *ICLR*, 2019.

Ryan Prenger, Rafael Valle, and Bryan Catanzaro. WaveGlow: A flow-based generative network for speech synthesis. In *ICASSP*, 2019.

Masaki Saito and Shunta Saito. TGANv2: Efficient training of large models for video generation with multiple subsampling layers. *arXiv:1811.09245*, 2018.

Andrew Saxe, James McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR*, 2014.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural TTS synthesis by conditioning WaveNet on Mel spectrogram predictions. In *ICASSP*, 2018.

Jose Sotelo, Soroush Mehri, Kundan Kumar, Joao Felipe Santos, Kyle Kastner, Aaron Courville, and Yoshua Bengio. Char2Wav: End-to-end speech synthesis. In *ICLR*, 2017.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception architecture for computer vision. In *CVPR*, 2016.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv:1609.03499*, 2016.

Aäron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George 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 *ICML*, 2018.

Sean Vasquez and Mike Lewis. MelNet: A generative model for audio in the frequency domain. *arXiv:1906.01083*, 2019.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. In *Interspeech*, 2017.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *ICML*, 2019.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

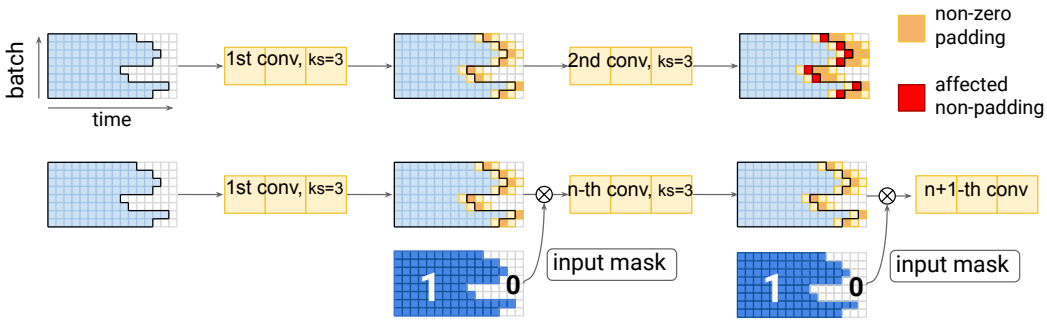


Figure 3: Masking scheme for sampling different-length samples. Top: processing a batch of samples of different lengths padded with zeros leads to interference between padding and non-padding after the second convolution, not seen during training. Bottom: masking after each convolution ensures that the meaningful input seen by each layer is padded with zeros only.

A ARCHITECTURE DETAILS

A.1 MASKING CONVOLUTIONS TO GENERATE LONGER SAMPLES

Since our generator is a fully-convolutional network, in theory it is capable of generating samples of arbitrary length. However, since deep learning frameworks usually require processing fixed-size samples in batches for efficiency reasons, our inputs of different lengths need to be zero-padded to fit in a fixed-size tensor. Convolutional layers, including the ones used in our model, often pad their inputs to create outputs of the desired dimensionality, hence we only need to ensure that the padded part of the input tensors to all layers is always zero. As shown in Figure 3, this would not normally be the case after the second convolutional layer, since convolutions (with kernel sizes greater than one) would propagate non-zero values outside the border between meaningful input and padding. A simple way to address this issue is masking, i.e. multiplying the input by a zero-one mask tensor, directly before each convolutional layer. This enables batched sampling of utterances of different length, which is efficient on many hardware platforms, optimised for batching.

A.2 ARCHITECTURE DETAILS

layer/input	t	*freq.*	ch
linguistic features, z	400	200Hz	567
conv, *kernel size 3*	400	200Hz	768
GBlock	400	200Hz	768
GBlock	400	200Hz	768
GBlock, *upsample* $\times 2$	800	400Hz	384
GBlock, *upsample* $\times 2$	1600	800Hz	384
GBlock, *upsample* $\times 2$	3200	1600Hz	384
GBlock, *upsample* $\times 3$	9600	4800Hz	192
GBlock, *upsample* $\times 5$	48000	24kHz	96
conv, *kernel size 3*	48000	24kHz	1
Tanh			

Table 2: Architecture of GAN-TTS's Generator. t denotes the temporal dimension, while ch denotes the number of channels. The rightmost three columns describe dimensions of the *output* of the corresponding layer.

In Table 2 we present the details of Generator architecture. Overall, the generator has 30 layers, most of which are parts of dilated residual blocks.

Table 3 shows the numbers of residual DBlocks and downsample factors in these blocks for different initial downsample factors of RWDs.

k	$\mathsf{D}_k^{\mathsf{C}}$			$\mathsf{D}_k^{\mathsf{U}}$		
	factors	num. blocks	depth	*factors*	num. blocks	depth
1	$5, 3, 2, 2, 2$	8	17	$5, 3$	5	11
2	$5, 3, 2, 2$	7	15	$5, 3$	5	11
4	$5, 3, 2$	6	13	$5, 3$	5	11
8	$5, 3$	5	11	$5, 3$	5	11
15	$2, 2, 2$	6	13	$2, 2$	5	11

Table 3: Downsample factors in discriminators for different initial stride values k. The number of blocks includes non-downsampling DBlocks.

All conditional discriminators eventually add the representations of the waveform and the linguistic features. This happens once the temporal dimension of the main residual stack is downsampled to the dimension of the linguistic features, i.e. by a factor of 120. Downsampling is carried out via an initial reshape operation (by a factor k varying per RWD) and then in residual blocks, whose downsample factors are prime divisors of $120/k$, in decreasing order. For unconditional discriminators, we use only the first two largest prime divisors of $120/k$.

B DEEPSPEECH DISTANCES - DETAILS

B.1 DEEPSPEECH2

Our evaluation metrics extract high-level features from raw audio using the pre-trained DeepSpeech2 model from the *NVIDIA OpenSeq2Seq* library (Kuchaiev et al., 2018). Let $w = 480$ be a 20ms window of raw audio at 24kHz, and let $f : \mathbb{R}^w \longrightarrow \mathbb{R}^{1600}$ be a function that maps such a window through the DeepSpeech2 network up to the 1600-dimensional output of the layer labeled `ForwardPass/ds2_encoder/Reshape_2:0`. We use default values for all settings of the DeepSpeech2 model; f also includes the model's preprocessing layers.

For a 2s audio clip $\boldsymbol{a} \in \mathbb{R}^{100w}$, we define

$$\mathsf{DS}(\boldsymbol{a}) = \frac{1}{199} \sum_{i=0}^{198} f(\boldsymbol{a}_{iw/2:iw/2+w}) \in \mathbb{R}^{1600}, \tag{4}$$

where $\boldsymbol{a}_{i:j} = (\boldsymbol{a}_i, \boldsymbol{a}_{i+1}, \ldots, \boldsymbol{a}_{j-1})'$ is a vector slice.

The function DS therefore computes 1600 features for each 20ms window, sampled evenly with 10ms overlap, and then takes the average of the features along the temporal dimension.

B.2 METRICS IN DISTRIBUTION SPACE

Given samples $\boldsymbol{X} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times d}$, where d is the representation dimension, the Fréchet distance and MMD can be computed using the following estimators:

$$\widehat{\text{Fréchet}^2}(\boldsymbol{X}, \boldsymbol{Y}) = \|\mu_{\boldsymbol{X}} - \mu_{\boldsymbol{Y}}\|_2^2 + \operatorname{Tr}\left(\Sigma_{\boldsymbol{X}} + \Sigma_{\boldsymbol{Y}} - 2(\Sigma_{\boldsymbol{X}}\Sigma_{\boldsymbol{Y}})^{1/2}\right) \tag{5}$$

$$\widehat{\text{MMD}^2}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{m(m-1)} \sum_{\substack{1 \le i,j \le m \\ i \ne j}} k(\boldsymbol{X}_i, \boldsymbol{X}_j) + \frac{1}{n(n-1)} \sum_{\substack{1 \le i,j \le n \\ i \ne j}} k(\boldsymbol{Y}_i, \boldsymbol{Y}_j) + \sum_{i=1}^{m} \sum_{j=1}^{n} k(\boldsymbol{X}_i, \boldsymbol{Y}_j), \tag{6}$$

where $\mu_{\boldsymbol{X}}, \mu_{\boldsymbol{Y}}$ and $\Sigma_{\boldsymbol{X}}, \Sigma_{\boldsymbol{Y}}$ are the means and covariance matrices of $\boldsymbol{X}$ and $\boldsymbol{Y}$ respectively, while $k : \mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}$ is a positive definite kernel function. Following Bińkowski et al. (2018) we use the polynomial kernel

$$k(x, y) = \left(\tfrac{1}{d} x^T y + 1\right)^3. \tag{7}$$

Estimator (5) has been found to be biased (Bińkowski et al., 2018), even for large sample sizes. For this reason, FID estimates for real data (i.e. when $\boldsymbol{X}$ and $\boldsymbol{Y}$ are both drawn independently from the same distribution) are positive, even though the theoretical value of such a metric is zero. KID, however, does not suffer from this issue thanks to the use of the unbiased estimator (6). These properties also apply to the proposed DeepSpeech metrics.

The lack of bias in an estimator is particularly important for establishing scores on real data for conditional distances. In our conditional text-to-speech setting, we cannot sample two independent real samples with the same conditioning, and for this reason we cannot estimate the value of cFDSD for real data, which would be positive due to bias of estimator (5). For cKDSD, however, we know that such an estimator would have given values very close to zero, if we had been able to evaluate it on two real i.i.d. samples with the same conditioning.

B.3 Distance estimation

Let G and DS represent the generator function and a function that maps audio to DeepSpeech2 features as defined in Eq. 4. Let

$$\boldsymbol{X}^{\mathsf{G}} = \{\mathsf{DS}\left(\mathsf{G}(c_i, z_i)\right)\}_{i=1}^{N}, \qquad \boldsymbol{X}_{:N}^{real} = \{\mathsf{DS}(x_i)\}_{i=1}^{N}, \qquad \boldsymbol{X}_{N:}^{real} = \{\mathsf{DS}(x_i)\}_{i=N+1}^{2N}, \tag{8}$$

where $(x_i, c_i) \overset{iid}{\sim} p(\mathrm{x}^{real}, \mathrm{c}), i = 1, \ldots, 2N$ are jointly sampled real examples and linguistic features, and $z_i \overset{iid}{\sim} \mathcal{N}(0, 1)$. In the conditional case, we use the same conditioning in the reference and generated samples, comparing conditional distributions $p(\mathrm{x}^G|\mathrm{c})$ and $p(\mathrm{x}^{real}|\mathrm{c})$:

$$\widehat{\mathsf{cFDSD}}\left(p(\mathrm{x}^{\mathsf{G}}|\mathrm{c}), p(\mathrm{x}^{real}|\mathrm{c})\right) = \widehat{\text{Fréchet}}\left(\boldsymbol{X}^{\mathsf{G}}, \boldsymbol{X}_{:N}^{real}\right), \tag{9}$$

$$\widehat{\mathsf{cKDSD}}\left(p(\mathrm{x}^{G}|\mathrm{c}), p(\mathrm{x}^{real}|\mathrm{c})\right) = \widehat{\text{MMD}}\left(\boldsymbol{X}^{\mathsf{G}}, \boldsymbol{X}_{:N}^{real}\right), \tag{10}$$

where $\widehat{\text{Fréchet}}$ and $\widehat{\text{MMD}}$ are estimators of the Fréchet distance and MMD defined in Eq. 5 and 6, respectively.

In the unconditional case, we compare $p(\mathrm{x}^G)$ and $p(\mathrm{x}^{real})$:

$$\widehat{\mathsf{FDSD}}\left(p(\mathrm{x}^{\mathsf{G}}), p(\mathrm{x}^{real})\right) = \widehat{\text{Fréchet}}\left(\boldsymbol{X}^{\mathsf{G}}, \boldsymbol{X}_{N:}^{real}\right), \tag{11}$$

$$\widehat{\mathsf{KDSD}}\left(p(\mathrm{x}^{\mathsf{G}}), p(\mathrm{x}^{real})\right) = \widehat{\text{MMD}}\left(\boldsymbol{X}^{\mathsf{G}}, \boldsymbol{X}_{N:}^{real}\right). \tag{12}$$

C μ-law preprocessing

Many generative models of audio use the μ-law transform to account for the logarithmic perception of volume. Although μ-law is typically used in the context of non-uniform quantisation, we use the transform without the quantisation step as our model operates in the continuous domain:

$$F(x) = \text{sgn}(x)\frac{\ln(1 + \mu|x|)}{\ln(1 + \mu)}, \tag{13}$$

where $x \in [-1, 1]$ and $\mu = 2^8 - 1 = 255$ for 8-bit encoding or $\mu = 2^{16} - 1 = 65,535$ for 16-bit encoding.

Our early experiments showed better performance of models generating μ-law transformed audio than non-transformed waveforms. We used the 16-bit transformation.

D Training details

We train all models with a single discriminator step per generator step, but with doubled learning rate: 10^{-4} for the former, compared to 5×10^{-5} for the latter. We use the *hinge loss* (Lim & Ye, 2017), a batch size of 1024 and the Adam optimizer (Kingma & Ba, 2015) with hyperparameters $\beta_1 = 0, \beta_2 = 0.999$.

Following Brock et al. (2019), we use spectral normalisation (Miyato et al., 2018) and orthogonal initialisation (Saxe et al., 2014) in both the generator and discriminator(s), and apply off-diagonal

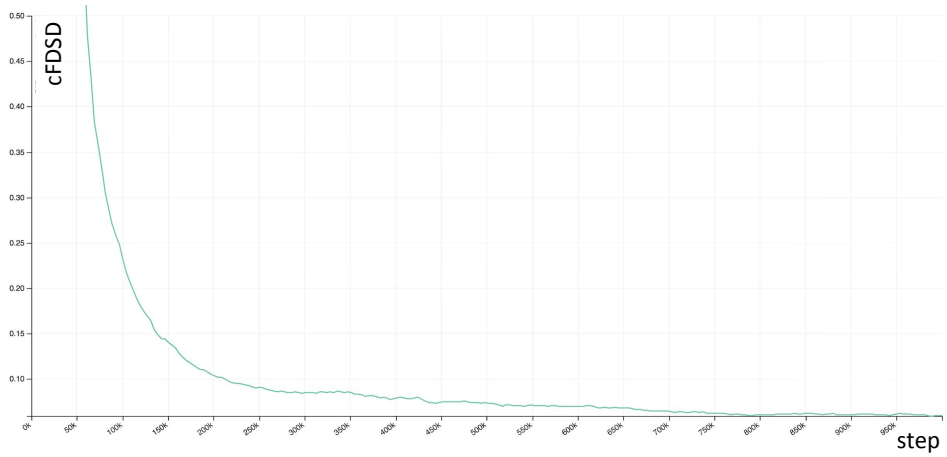


Figure 4: Learning curve for the GAN-TTS model in terms of cFDSD.

orthogonal regularisation (Brock et al., 2016; 2019) and exponential moving averaging to the generator weights with a decay rate of 0.9999 for sampling. We also use *cross-replica* BatchNorm (Ioffe & Szegedy, 2015), which aggregates batch statistics from all devices across which the batch is split and *standing statistics* during sampling. The latter means that we accumulate batch statistics from 100 forward passes through the generator before the actual sampling takes place, allowing for inference at arbitrary batch sizes. In fact, accumulating standing statistics makes the BatchNorm layers in the generator independent of any characteristics of the samples produced during inference. This technique is thus vital for sampling audio of unspecified length: producing samples that are longer than those used during training typically requires using a smaller batch size, with partially padded samples (See Appendix A.1). These smaller batches would naturally have different statistics than the batches used during training.

We trained our models on Cloud TPU v3 Pods with data parallelism over 128 replicas for 1 million generator and discriminator updates, which usually took up to 48 hours.

Figure 4 presents the stable and gradual decrease of cFDSD during training.