Generating realistic voice using Neural and Adversarial networks

Matthew Schueler [1] **Lakshay Gopalka** [1] **Pascal Bakker** [1] **Venkata Rajendra Kante** [1]

Abstract

Generational neural networks have emerged in the last couple of years as a promising technology. Great strides have been made in image generation. However, temporal data has proven to be both computationally costly and inaccurate. To overcome this, researchers have devised methods using convolutional layers and generative adversarial networks to help assuage these difficulties. This paper will discuss the efforts to reimplement two of these networks using unsupervised learning, WaveGan and WaveNet to generate novel and realistic speech audio output. The experiments are followed using small audio dataset and the results of generated speech output seem to be promising for further work.

1. Introduction

Much work has been done recently on developing realistic text-to-speech algorithm such as for use in Google's Home series of products. Such an algorithm could be useful anywhere where reading text out loud was required, such as a system for a vision-impaired person. This could also be used to imitate the voice of a specific person and be able to read with a particular style of speech. Even Google's main algorithm though does not sound extremely realistic compared to what natural speech sounds like. Within the last few years, several new architectures have been developed that allow for more realistic text-to-speech systems. Much of this work is based on the WaveNet architecture developed in 2016 (Oord et al., 2016). This work was then used to build the Tacotron 2 system which is able to create a very realistic sounding voice clip from an arbitrary text string, which sounds like a specific person it was trained on (Shen et al., 2017a). The training data was roughly 24 hours of reading lines of a single speaker from the LJSpeech dataset (Ito, 2017). However, there are still some strange sounds that are produced by quirks in the language rules, such as apostrophes.

This project tackles voice generation by comparing Generative Adversarial Networks (GANs) and Recurrent Neural Networks (RNNs). We aim to produce a voice through this method that sounds like a specific person, instead of sounding like a generic or electronic voice. For this, we limited the total vocabulary to only the audio samples of the digits 0 through 9 and used existing available data sets as well as ones from a specific person to try to produce a realistic-sounding voice that sounds like a specific individual. We performed two approaches, one using a GAN based on the WaveGAN architecture (Engel et al., 2019) and one based off of an RNN based on the WaveNet architecture (Oord et al., 2016). Thus, the aim of the project is to design neural network topology and adversarial network using topologies like RNN, WaveGAN to generate coherent sentences. Another challenge with speech generation is that there are various voice modulations depending on previous data and nature of the audio which is depicted below. We would compare the audio output quality, logical sense, voice clarity and accuracy for all the networks.

Figure 1. Visualization of generated audio for 1 second duration

2. Related Work

The inspiration of audio generation comes from text (Józefowicz et al., 2016) and image generative models. The key idea involves modelling joint probability over each pixel by learning methods to yield new distributions. The model architecture distributes it over numerous random variables like in PixelRNN (van den Oord et al., 2016). Using the notion from such generation, the WaveNet generates wideband raw audio waveform.

Generative Adversarial Network (Goodfellow et al., 2014) is used for data generation and applying it on audio datasets could is another way of using state-of-the-art in new applications. Recent work has shown that neural networks can be trained with autoregression to operate on raw audio (Mehri et al., 2016). These methods can be structured with a generator model which must be fed audio samples one at a time to allow discriminator to asses the best output audio.WaveGAN is one such way which is based upon DCGAN (Radford et al., 2016) architecture which is one dimensional and results in a model with the same number of parameters and operations as its two-dimensional ana-

log.

Our paper summarizes current work on audio generation an overview of the problem. We then describe our proposed method along with the dataset that we use to train our algorithm. We then go over the design of our network architecture that we think will provide an improvement to existing work on this area. We then discuss our evaluation metrics that are based on both objective machine evaluation and subjective user evaluation, both of which are important to consider for a problem such as speech generation. Finally, the conclusion discusses the findings of the project and the future scope of the problem and recommendations.

3. Methodology

As described, we have utilized WaveNet and WaveGAN for audio generation tasks. To facilitate our experimentation, it focuses on the Speech Commands (Warden, 2018) and LJ speech dataset (Ito, 2017) for training. The Speech command dataset consists of speakers recording individual words of the spoken digits "zero" through "nine" and refers to this subset as the Speech Commands Zero Through Nine (SC09) dataset. These ten words encompass many phonemes and two consists of multiple syllables. Each recording is one second in length, and do not attempt to align the words in time. LJ speech dataset is a public domain speech dataset consisting of 13,100 short audio clips of a single speaker reading passages from 7 non-fiction books. Clips vary in length from 1 to 10 seconds and have a total length of approximately 24 hours.

As generating time-dependent data through RNNs is slow. For large datasets, this results in slowly trained networks. To resolve this problem WaveNet utilizes one-dimensional convolutional layers. This allows the network to run a computation in parallel. Additionally, to reduce the computation time, even more, each layer is dilated until the final layer has only 1 neuron for the current timestep. This work shows that, in practice, WaveNet is capable of generating audio data quite well.

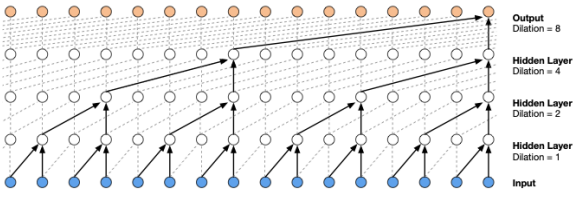

Figure 2. Visualization of a stack of dilated causal convolutional layers

The WaveNet model operates under the principle of conditional probability where the probability of the current sample is determined by the product of all previous conditional probability samples. Thus, at a given timestep, what sample to generate next is the discrete value with the largest probability.

Within each layer of the network, after dilation, the data is transformed nonlinearly by a tanh and a sigmoid function, which each sample is then combined via product as in fig. 3. This result is then combined with the tensors at the beginning of the layer via addition to the next layer. At the end of the network, the current tensor and all skipped connections are summed together and are passed through several activation functions. This model also allows the addition of a global feature, such as the current speaker, current phoneme, etc, which is connected at every layer and is passed into an upscaling convolution filter during the tanh and sigmoid activation transformation.

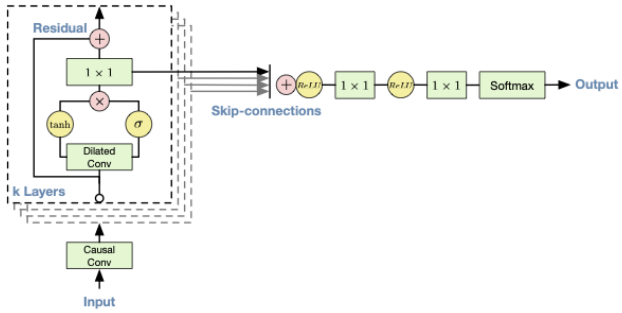

Figure 3. WaveNet architecture

WaveGAN is based off of the existing DCGAN architecture (Radford et al., 2016), which is used for 2D image generation. The same style of layers was used to create WaveGAN, except they were modified to operate on 1D data in the time domain instead of 2D data in the spacial domain. Similar to WaveNet, these networks also utilize dilated convolutional layers to expand the receptive field without severely impacting the training time by adding additional parameters. WaveGAN also utilizes regular striding and fractional striding to further increase the relationship between nearby data points over time. Finally WaveGAN also employs normalization layers to increase stability of the weight training, however this is not inlcluded at the interface between the generator and discriminator, because it was found to negatively impact the performance of the network overall.

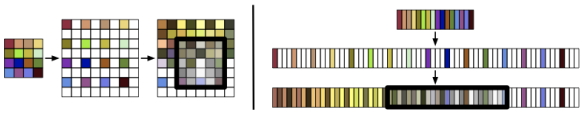

Figure 4. Representation of the transposed convolution operation for the first layers of the DCGAN (left) and WaveGAN (right) generators

4. Experiment

For WaveNet and WaveGAN, a variety of hyperparameters were experimented with to optimize the algorithm. The final sample rate was set at 16000 samples per second, as this was seen as a reasonable median between high compression and audibility. Additional hyperparameters tested include frame size, the number of samples per batch, frameshift, the offset to collect samples from the data, and the number of layers. Additionally, the team tried to implement a global condition on the network. However, getting information about the current phoneme was determined to be too difficult, and was scrapped. Due to the large size of the datasets, a smaller subset of data was used instead, with ten percent being used for validation. Different audio file subsets were experimented. However, no subset improved the accuracy of the data.

In WaveGAN model, our model contained a lot of noise, therefore, we tried to test it changing the generator. For our project, we tried to vary by introducing the WaveNet as the generator and allowing the discriminator to asses the audio quality. Unfortunately, that didn't output better audio as the WaveNet model was trained on a small dataset and it did not merge well with overall GAN architecture. We explored by using pre-trained checkpoints and tried incorporating with the GAN model. This helped us produce a better sounding audio output. As Tacotron (Shen et al., 2017b)is based on text-to-speech synthesis and has been used with neural network architectures, we tried to add a discriminator to the model. The audio output from existing tacotron2 model is better than our model as it uses a text data which guides the learning architecture and helps the GAN model to synthesize the audio better.

5. Results and Evaluation Metrics

The WaveNet algorithm, although working, was unable to get human-like sounding voices. It was, however, able to generate audio wav samples that, although are static. The network was able to generate audio data with distinct patterns, just not of that of human voice patterns. The accuracy, which utilized cross-entropy, tended to be around 60 percent with a smaller dataset. Although, considering that there were 256 'classes' for the algorithm to pick at any given time, the research team feels this adequately shows the capability of the algorithm. The fig 5 represents the source audio and generated audio waveform, which clearly shows some learning behaviour in the form of peaks and valleys.

The WaveGAN algorithm also worked but failed to output a human-like voice. This could be because of several factors like the limited dataset, training time, understanding of various subtle key hyperparameters which influence tone, pitch, bass etc of the voice. When we utilised the pre-trained checkpoints, the audio quality got better and resounded clear and recognizable. The clear distinction of the words was not seen for the whole sequence but could have been worked upon given the required computational capacity. A visual representation of how the model is learning from real input data is depicted in fig. 6. Lastly, using Tacotron which uses a text input data as well enhanced the audio quality significantly and produced a coherent and semantic speech which we desired.

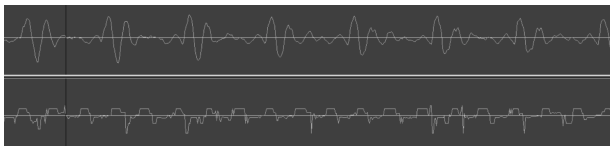

Figure 5. Representation of source audio (top) and generated audio by WaveNET

For the purpose of evaluation metrics, we relied on human user evaluation. We used a 5 point scale to asses the generated audio based on several parameters. The parameters include coherence, clarity and overall audio quality. Each user listened to 3 different audio type from the model. Since the WaveNet audio was static in nature so it received an overall score of 0. Though this seems very disappointing, the WaveGAN received most of the score in the range of 2-3 points. It had an overall average score of 2.4. For our experimental sake, we also tried to test the audio from already existing Tacotron model. As its audio quality is much clear and easily recognizable, therefore it received scores in the range of 4-5, with an overall average score of 4.5.

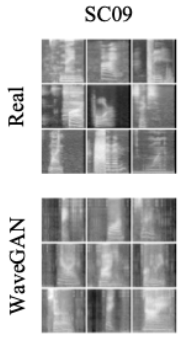

Figure 6. Samples from each of the dataset and by WaveGAN

6. Discussion

The team recommends that one should use TensorFlow's subclassing feature to develop new architectures, as it is easier to debug and to use in production. Each WaveNet

layer was placed into a Keras.layers subclass so that it could be easily looped over when creating the network. Additionally, the team highly recommends using the functional model over the sequential model as it allows for non-sequential layer connections.

In terms of hyperparameters for WaveNet, the team found that a frame size of either 128 or 256 and a frameshift of 64 or 128 performed the best. The batch size should be kept low, of values around 64 to 200. The number of layers should be at least 40 for the network, as the team found that WaveNet could not attain a high accuracy for layer sizes smaller than this. When developing code for machine learning projects, certain coding procedures must be followed. The team recommends utilizing Docker to maintain code dependencies when collaborating, as Tensorflow's versions have subtle differences that can cause a wide variety of errors on different machines. Callback functions are absolutely necessary when testing, especially the Tensorboard callback, as it allows the team to visualize the logs of the network models.

7. Conclusions and Future Work

The WaveNet and WaveGAN algorithms were properly reimplemented into Tensorflow 2.0. However, there were issues with achieving human voices. Although the network was able to produce audio data that was cyclical and contained patterns, the generated waveforms sounded like static. In the future, the team will continue to train with more dataset and optimised hyper-parameters to find a high accuracy model.

The current model only possesses the ability to pass time-series data through it. In the future, the team desires to implement global conditioning with phonemes and grammar structure. It must be noted that the basic WaveNet model cannot generate text to speech. Thus the team wants to build the additional architecture required to do so. Other than creating audio with a single model, the project goal was to combine WaveNet with a GAN to produce a novel type of network. This network would use WaveNet in its encoder. This type of network could benefit from the strengths of GANs and the computational speed of WaveNet. We also could explore more of the text-to-speech model with GAN and optimise the functionality of the overall network to produce a realistic audio waveform.

References

Engel, Jesse, Agrawal, Kumar Krishna, Chen, Shuo, Gulrajani, Ishaan, Donahue, Chris, and Roberts, Adam. GANSynth: Adversarial neural audio synthesis. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1xQVn09FX.

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

Ito, Keith. The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

Józefowicz, Rafal, Vinyals, Oriol, Schuster, Mike, Shazeer, Noam, and Wu, Yonghui. Exploring the limits of language modeling. *CoRR*, abs/1602.02410, 2016. URL http://arxiv.org/abs/1602.02410.

Mehri, Soroush, Kumar, Kundan, Gulrajani, Ishaan, Kumar, Rithesh, Jain, Shubham, Sotelo, Jose, Courville, Aaron C., and Bengio, Yoshua. Samplernn: An unconditional end-to-end neural audio generation model. *CoRR*, abs/1612.07837, 2016. URL http://arxiv.org/abs/1612.07837.

Oord, Aaron van den, Dieleman, Sander, Zen, Heiga, Simonyan, Karen, Vinyals, Oriol, Graves, Alex, Kalchbrenner, Nal, Senior, Andrew, and Kavukcuoglu, Koray. Wavenet: A generative model for raw audio. 2016. URL http://arxiv.org/abs/1609.03499. cite arxiv:1609.03499.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. 11 2016.

Shen, Jonathan, Pang, Ruoming, Weiss, Ron, Schuster, Mike, Jaitly, Navdeep, Yang, Zongheng, Chen, Zhifeng, Zhang, Yu, Wang, Yuxuan, Skerry-Ryan, RJ, Saurous, Rif, Agiomyrgiannakis, Yannis, and Wu, Yonghui. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. 12 2017a.

Shen, Jonathan, Pang, Ruoming, Weiss, Ron J., Schuster, Mike, Jaitly, Navdeep, Yang, Zongheng, Chen, Zhifeng, Zhang, Yu, Wang, Yuxuan, Skerry-Ryan, R. J., Saurous, Rif A., Agiomyrgiannakis, Yannis, and Wu, Yonghui. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. *CoRR*, abs/1712.05884, 2017b. URL http://arxiv.org/abs/1712.05884.

van den Oord, Aäron, Kalchbrenner, Nal, and Kavukcuoglu, Koray. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016. URL http://arxiv.org/abs/1601.06759.

Warden, Pete. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018. URL http://arxiv.org/abs/1804.03209.