AN AUDIO GENERATION MODEL

Eric Wang **Chase Xu** **Ziang Jiang**
Julie Jiang **Hanxiao Yu** **Siyi Zhu**
Department of Computer Science, Faculty of Engineering, University of Victoria
xhwang@uvic.ca

1. INTRODUCTION

1.1 The Problem We Are Solving

Speech synthesis is an artificial production of human speech. A text-to-speech (TTS) system, which converts normal language text into speech, has a wide range of uses in industry and academia. Currently, A TTS system mainly consists of two components, text analysis and vocoder. Text analysis is responsible for generating MEL spectrogram from input text, and vocoder is responsible for generating audios from MEL spectrogram. Traditional methods mostly use concatenative TTS to generate speech, that is, a big database of short speech components are recorded from speakers and then combined to construct complete utterances. Thanks to machine learning, recently there is a great leap in NLP field, and many publications discuss tagging emotions and plauses in text. Therefore, to improve the quality of synthesized voice, it would be feasible and valuable to model the raw waveform of an audio signal and generate more naturally-sounded speech. Our aim is to implement a model for generating raw audio waveforms (Figure 1). This model can directly generate speech waveforms close to real human voice from the given language parameters and speech parameters. Frankly speaking, We try to convert digital text into natural speech streams and build a vocoder to implement TTS.

1.2 How We Do This Project

We want to research on audio synthesis, specifically, to implement a vocoder that generates more natural sound than traditional TTS systems, reduce the gap between synthesizing voice and human voice, increase the conversion efficiency and avoid errors. An efficient end-to-end waveform audio synthesis component would be a crucial part of TTS systems.

On the one hand, deep learning based audio synthesis architectures have been widely used as vocoders to generate realistic waveform signals that are very close to the real sound in nature, because their working methods could appropriately simulate the human vocalizing system and greatly reduce the number of training parameters of neural network, thus making our work more efficient.

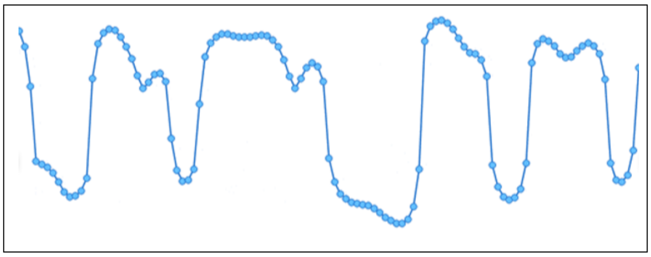

Figure 1. Sample of WaveNet: Each dot is a separately calculated sample; the aggregate is the digital waveform

The famous WaveNet provides a generic and flexible framework for many applications that rely on audio generation, including music composing, speech enhancement, voice conversion and source separation [10]. However, there still are some flaws to WaveNet, such as the long inference time. Many recently suggested papers discussed betterment of WaveNet model. Jonathan et al. found some possible optimization of WaveNet subjective quality and computational efficiency [1]. Also, D.Rethage et al. showed an end-to-end learning method for speech denoising based on Wavenet [12]. ClariNet showed real-time audio synthesis capability by incorporating inverse autoregressive flow for parallel sampling. Nonetheless, these approaches mostly need two-stage training pipelines with well-trained teacher networks and could only produce natural sound by using probability distillation along with auxiliary loss terms.

On the other hand, Ryan Prenger et al. proposed WaveGlow, a flow based model which our project implements. It only needs single-stage training procedures and a single maximum likelihood loss, without auxiliary terms. Meanwhile, it is inherently parallel due to the characteristics of generative flow [10]. Through our implement and tests, this model can efficiently sample raw audio in real-time and maintain the stability in the training procedure. As an relatively improved alternative to WaveNet vocoder, this flow based model could be used in many TTS architectures.

1.3 Approaches

This model is a hierarchical architecture that consists of context blocks as a highest abstract module and many reversible transformations inside the context block (Figure 3). The following are the general description of the structure of this model and some important generative approaches taken in order of importance. We provide the entire details in a row in section 3.

1.3.1 Context Blocks

This model has context block as the most important component and the highest abstraction module. Each block consists of the squeeze operation, and then below are the flow stacks. Multi-scale architecture and some methods similar to WaveNet would be also used in this procedure. More details about this part are discussed in 3.1 and 3.2.

1.3.2 Affine Coupling

Leveraging affine coupling layer, we could sample the raw audio signal x (in the equation 2) efficiently in parallel, because the layer could improve the efficiency of the bidirectional transformation of f. Each layer is the parametric transformation that can hold half of the channel dimension unchanged and apply the affine transformation on the other half (the equation 10).

The equation 11 demonstrates the procedure of inverse transformation involving the WaveNet affine coupling layers. To establish a flexible transformation function f, this model stacks multiple flow operations for each context block. For all channels to affect each other during subsequent flow operations, changing order operation would be used after the affne coupling of each flow. More details about this part are discussed in 2.2 and 3.6.

1.3.3 Flows

To model raw audio signals, this model uses normalizing flows. An invertible transformation function f would be used to map the given audio signal x (in the equation 2) to a prior probability, for the calculation of the log probability distribution of the signal. This parametric invertible transformation function could be established using affine coupling layers and meeting the below requirements so as for the model to achieve good efficiency:

- The Jacobian determinant of the transformation function is tractable.
- The mapping of random noise is efficient.

In the neural network based TTS pipeline, mel spectrogram is used as local condition c for the modelling of conditional probability $p(x|c)$. More details about this part are discussed in 2.2 and 3.3.

1.3.4 Activation Normalization

The use of activation normalization (ActNorm) enables the stability in the training procedure of the multiple flow operations that constitute the network. The ActNorm layer would be used at the begining of the flow. More details about this part are discussed in 3.5.

2. RELATED WORK

2.1 WaveNet

WaveNet, which has been proven effective in generating high quality synthesised speech, uses the probability distribution of the original audio signals as the estimation. This model separates the joint probability distribution of the audio signals into a multiplication of conditional probabilities.

This model typically uses deep convolutional network to simulate the long-run dependency of the conditional probabilities and thus generates a relatively fast parallel estimation of probability. By using the neural network method trained with real voice recording to directly simulate the waveform, neuro-like vocoders can generate more natural and emotional voices closer to human natural speech than traditional methods.

However, as mentioned before, the sampling method of WaveNet restricts the work efficiency during the inference time. Its autoregressive characteristics also bring many obvious shortcomings. One is the limitation to practical application. The generation speed is very slow. Because the sample points are generated one by one, it takes several minutes to generate a short audio signal [4]. In addition, the WaveNet model uses previously generated sample points as input to generate the next sample point. Therefore, when generating poor sample points, errors may continue to accumulate and affect the quality of the sound signal [11]. Those are the reasons why we finally transited to a flow based model, such as WaveGlow and FloWaveNet, to implement a vocoder. [8]. Despite of its low efficiency, we still studied on this model for a long time at the beginning of this project. We also applied a lot basic techniques in WaveNet about raw audio modelling based on probability estimation to this project.

2.2 Inverse Autoregressive Flow

A normalizing flow - Inverse Autoregressive Flow (IAF), which we use in our model, could scale well to high dimentional latent spaces. We use this flow type to construct the invertible transformations when mapping the high dimensional data to the latent variable, which we assumed exist.

Apart from WaveNet and IAF, other subtle but essential components and techniques we use are discussed in section 3.

2.3 Others

There are still some useful large work and papers that we partly reference to. We found several improved models based on WaveNet that maintain the advantages of WaveNet while increasing efficiency. FFTNet, a new architecture for parallel, non-causal and shallow waveform domain for speech enhancement. As a simplified WaveNet architecture, the initial wide extension mode is used. Such an architecture better represents the long-term correlation structure of speech in the time domain, where noise is usually highly uncorrelated, so it is suitable for speech enhancement based on the waveform domain. Compared with WaveNet, it can provide the same high-quality performance while reducing the use of parameters by 30%, thereby increasing the output speed [5]. WaveRNN is a single-layer recurrent neural network which has a dual softmax layer that matches the quality of the latest WaveNet model. The compact form of the network allows it to generate 24 kHz 16-bit audio on the GPU at a speed 4 times faster than real-time. The small number of weights in the sparse WaveRNN makes it possible to sample high-fidelity audio in real time on a mobile CPU [7].

Despite having not implemented these models, we have still learnt some useful techniques that can be used in the flow based model.

3. METHODS

A flow-based model has been chosen to finish this project. The reason is that a non flow-based model has limitations. It needs to train a WaveNet model first as the teacher network, and then it needs to train an autoregressive flow as the student network, so as to ensure the real-time synthesizing. Thus, the non flow-based model requires the exact double time of a flow-based model.

A flow-based generative model has its name "flow" because it contains many generator functions, and the input has to go through several generator functions to get the desired result. In the model, one generator does not require generating the desired output directly, but it is responsible for producing a closer output, which will be the input of the next generator function. Therefore multiple generator functions have to work together to get the desired result. The details of the flow-based generative model are shown in Figure 2.

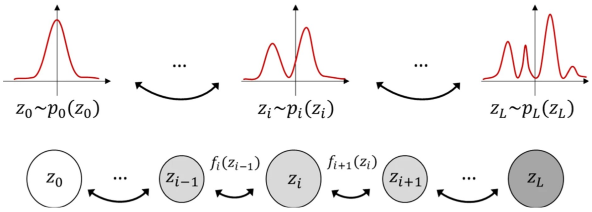

Figure 2. Diagram of a normalizing flow between a simple Gaussian distribution $z_0 \sim p_0(z_0)$ and an observed distribution $x = z_L \sim p_L(z_L)$ [15]

Inverse autoregressive flow-based generative model (IAF) is considered as the choice when real-time generation is required, as it provides parallel sampling ability. Compare with MAF, the crucial difference is that IAF models the conditional probability of the target variable with the inverse transformation function during the training process.

IAF intends to estimate the probability density function of x given that $f(z)$ is already known. The inverse flow is an autoregressive affine transformation, but the scale and shift terms are autoregressive function of observed variables from the known distribution $f(z)$.

3.1 Context Blocks

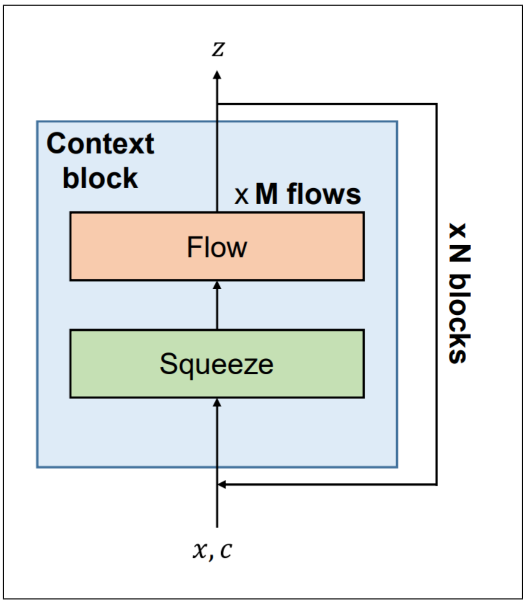

Figure 3. Context Block

The main component of this model is context block (Fig. 3). The model is constructed by n numbers of context blocks on the top of another (8 has been used in this project). Each context block acts as a container, and it contains two kind of components. The first one is a squeeze operation which can massage the data for the better of future use. The second one, the most important part, is the flow. In this project, we used 6 steps of flow to avoid long term dependencies not being captured. IAF is very strong in capturing long-term dependencies, so that is the reason why we only use half of the number of flows that WaveGlow uses. [11] Each flow is looking for an optimal value of the exact maximized likelihood by finding the value of the det part in equation 2. The details of these two components will be introduced as follow.

3.2 Squeeze operation

Squeeze operation is one of the basic operations in flow-based generative model. Due to the characteristic of the audio type data, the squeeze operations need to ensure the two output roughly at the same frame. Otherwise the result will not be accurate. This is one of biggest problems we are facing at the start of this project. As the result, this project will use the odd/even number to separate data into two sections. This operation doubles the effective receptive field per block for the WaveNet-based flow.

$$\begin{cases} y_{1:d} & = x_{\text{even}} \\ y_{d+1:D} & = x_{\text{odd}} \end{cases} \tag{1}$$

By applying the squeeze operation at the beginning of each context block, the upper-level blocks can have the potential to learn the long-term characteristics of audio, while the lower-level blocks can focus on high frequency information. (The way it behaves focuses on high frequency and then slowly studying the lower frequency).

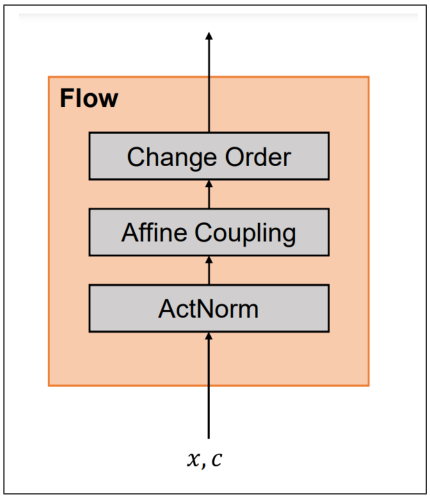

Figure 4. Flows

3.3 Flows

For each flow (Fig. 4), it calculates the exact maximized likelihood using equation (2). The flows have a goal to find a $f(x)$ (in det part) that could reach its target so that it will have the largest chance to match real output.

In order to maximize the P_x (log exact maximized likelihood of x), a few operations to the determinant part will be applied. The log determinant of the transformation f (which is also the Jacobian matrix) will be decomposed into the sum of per-flow terms (3), where N and M are the numbers of blocks and flows, respectively. The $f(x)$ can be expressed as the product of f^n_{AC} and f^n_{AN} (3). f^n_{AC} is the coupling layers and f^n_{AN} is the activation normalization.

$$logP_x(x) = logP_z(f(x)) + logdet(\frac{\partial f(x)}{\partial x}) \tag{2}$$

$$logdet(\frac{\partial f(x)}{\partial x}) = \sum_{n=1}^{MN} logdet(\frac{\partial (f^n_{AC} \cdot f^n_{AN})(x)}{\partial x}) \tag{3}$$

3.4 Jacobian Matrix

In equation 2, the det part is a matrix called Jacobian. The Jacobian matrix is the product of two matrix (eg. spectrum and output audio signals), as shown in equation eq(4).

Jacobian matrix has been used frequently in a flow-based generative model. It is useful and convenient because the Jacobian matrix of $f(x)$ can be easily calculated if the Jacobian matrix of $f^{-1}(x)$ (5) is known. The Jacobian matrix of $f(x)$ is the inverse of Jacobian matrix of $f^{-1}(x)$ in the equation 6. Remembering that the model is trained in a reverse way of generating audio, so the Jacobian's property could give the model a good flexibility to reverse itself when it is been used to generate audio.

$$J_f = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial x} \end{bmatrix} \tag{4}$$

$$J_{f^{-1}} = \begin{bmatrix} \frac{\partial f_1^{-1}}{\partial x} & \frac{\partial f_1^{-1}}{\partial y} \\ \frac{\partial f_2^{-1}}{\partial x} & \frac{\partial f_2^{-1}}{\partial x} \end{bmatrix} \tag{5}$$

$$J_f = Inverse(J_{f^{-1}}) \tag{6}$$

3.5 ActNorm

Normalization is one of the standard methods in flow-based generative model [11]. In traditional deep networks, too-high learning rate may introduce the gradient vanishing and exploding, and getting stuck in local minima [3]. The normalization step solves this problem by diminishing the small change in parameter which may induce greater change in network behavior. In GLOW, a method called ActNorm has been proposed and later used in the equation 7. ActNorm is a per-channel parametric affine transformation for each flow. It is to stabilize the training of the network composed of multiple flow operations [9].

For i-th channel (in the equation 7), s and b represent scale and bias respectively for each channel.

$$\forall i, j : y_{i,j} = s \odot x_{i,j} + b \tag{7}$$

$$logdet(\frac{\partial f^n_{AN}(x)}{\partial x}) = T * \sum_{i=1}^{C} logdet(log|s^i|) \tag{8}$$

Prior to GLOW, the normalization step was often implemented by the batch-normalization. However, this method would add activation noise to the data that is inversely proportional to the batch size per GPU [9]. Due to this problem, batch normalization is not used in this model. Instead, the mini-batch size is used in each flow. ActNorm in GLOW has the same effect as that of batch normalization, but focuses on mini-batch size normalization, which is the reason why this is used in this model.

3.6 Affine Coupling

The affine coupling layer is suggested in real NVP and later employed by GLOW. It enables the efficient bidirectional transformation of the function f by making the function bijective while maintaining computational tractability. It alters only half of the input (only x_{even} and y_{even}) each time, so as to ensure that the $logdet$ part is not going to be negative infinite while we are maximizing the $logp_x$.

The transformation method in affine coupling layer is generally originated from the masked autoregressive flow. But in our model, to achieve the ability of parallel sampling

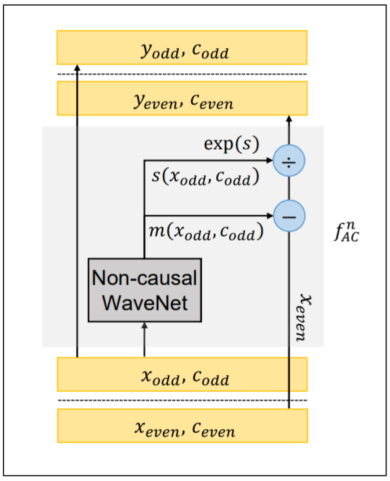

Figure 5. ActNorm

which improves the synthesizing speed, we use the transformation function originated from IAF.

Next, the affine coupling uses Non-causal WaveNet instead of causal WaveNet to find s and m. Contrary to audio synthesis in speech denoising, non-causal WaveNet can increase the filter length, which makes the model have access to the same amount of samples and inform the prediction. [13]

None causal-WaveNet also has the larger receiptive field, compared with causal WaveNet. Partnered with IAF's parallel sampling speed, we could achieve the ability of real-time audio synthesizing.

At the end of the affine coupling layer, y_{even} and y_{odd} need to be swapped around. It is to let all channels have a chance to be processed by the network and influence each other. This step, in GLOW, is replaced by an invertible 1×1 convolution layer, which shuffles the channel randomly, and promotes improving training speed. However, due to the time constrain, we have not not had this implemented but used the traditional swapping method instead.

$$logdet(\frac{\partial f^n_{AC}(x)}{\partial x}) = -\sum_{even} s(x_{odd}, c_{odd}) \tag{9}$$

$$y_{even} = \frac{x_{even} - m(x_{odd}, c_{odd})}{exp(s(x_{odd}, c_{odd}))} \tag{10}$$

$$x_{even} = y \odot exps(y_{good}, c_{odd}) + m(y_{good}, c_{odd}) \tag{11}$$

4. RESULTS AND EVALUATION

In order to measure the success and evaluate our outcomes, a set of training data with 13,100 short audio clips of a single speaker reading books was used. We found a publicly available WaveNet model as the comparison. The training data were fed to this model and our WaveGlow model. After each model had been well-trained, a set of text contents with the same sentences were provided to each model, generating speech audios. WaveNet model did cost longer time.

The audio results were also conducted with two evaluation methods: self investigation and numerical evaluation, as shown in the following.

4.1 Self Investigation

After each models had been trained, our team members randomly listened to one audio picked from each model and evaluated the speech audio by answering the following questions:

- If the audio was clear and hearable.
- If each word of the speech could be identified easily.
- If the voices sounded natural enough.

Our answers to most of the questions were yes. Only around 2% of the audio results were totally unclear and not hearable. By increasing the training time and adjusting the algorithms, we managed to obtain almost all of the results with relatively similar clearity. Despite noises existed in the audio results, the words were vaguely identified by testers. 98% of the pronunciation of the words sounded relatively natural. The results of those well-trained audio were then used for numerical evaluation.

4.2 Numerical Evaluation

A numerical analysis provides more visualized information for the better of this study. Thus, in the step of numerical evaluation, several factors of measurements were considered at the stage of final analysis. Most of those numerical evaluation methods were publicly available built-in packages. We directly used these methods to assess our models and final outcomes. Each speech audio was tested and evaluated numerically using the following factors: 5-scale Mean opinion score (MOS), conditional log-likelihoods (CLL), and inference time.

4.2.1 5-scale Mean Opinion Score

A 5-scale Mean opinion score (MOS) is a mean score of scores that are rated by humans after listening to each audio (5 represented as excellent, 4 as good, 3 as fair, 2 as poor, 1 as bad) [2]. The classic way of conducting an MOS experiment is too expensive, but there is a valid approach that could reduce the time and cost other than using a human source. MOS can be predicted by an objective quality model (MOS predictor). This model was utilized to estimate the score of each speech audio [14]. The MOS predictor was used to predict the MOS of each model for defining if an audio has a good quality.

4.2.2 Conditional Log-likelihoods (CLL)

A conditional log-likelihoods (CLL) is a natural logarithm of the likelihood that could be used to compare the fit of different coefficients, the higher the value the closer it is to the ground true sample [6]. By using these techniques, a set of random texts from test sets was used to feed the models. At the time the audios were generated, the results were used with the test sets' true audio samples to calculate CLL. This logarithm transforms a product of densities into a sum. It is very convenient because the asymptotic properties of sums are easier to analyze, and the sum is more stable than product.

4.2.3 Inference Time

The run time speed is also acting as an important factor during the process of generating speech audios. A good model should generate a good quality sound within an acceptable amount of time. Thus, the iterations and samples per seconds were used in the comparison to evaluate the models performances as well.

4.3 Final Evaluation

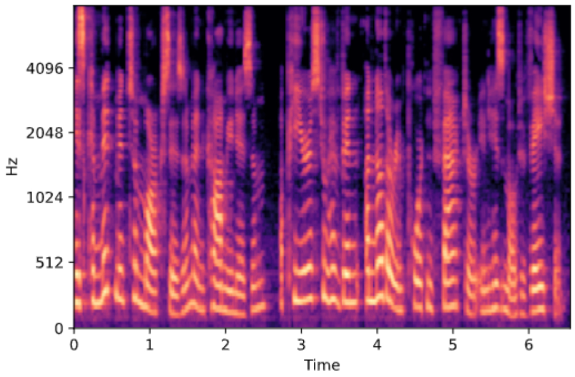

Figure 6. Spectrogram of The First Original Audio

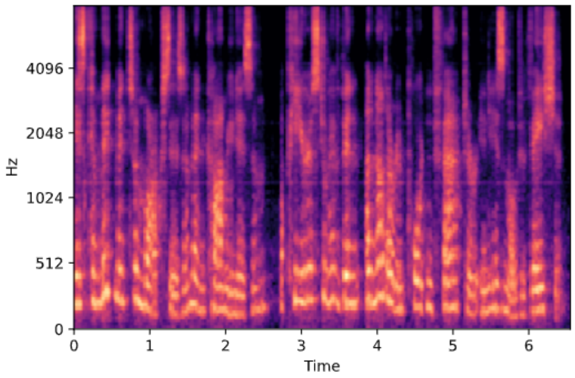

Figure 7. Spectrogram of The First Result

After self investigation, we conducted the final evaluation. Our final outcomes fell into these scenarios:

- The generated speech audios were not hearable. It was defined either the words could not be heard clearly or the voice sounded inhuman.
- The generated speech audios were of good quality. Every word was clear and the voice sounded human-like.

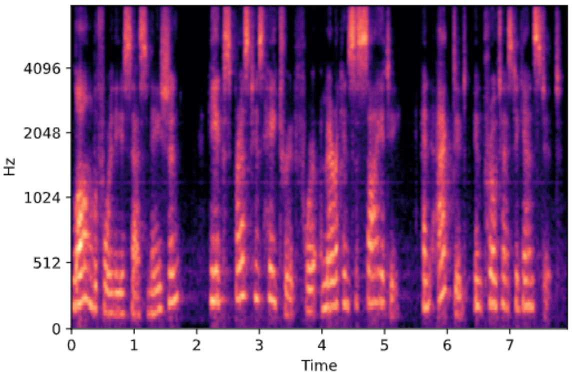

Figure 8. Spectrogram of The Second Result

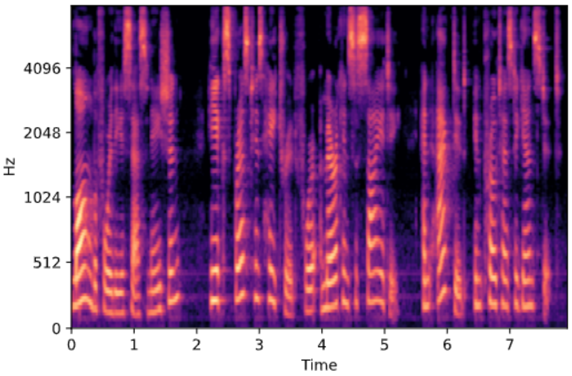

Figure 9. Spectrogram of The Second Original Audio

Most of our results fell into the second possible outcome, thus basically reaching our initial project goal. The numerical evaluation helped the study of the performance of our flow based model in comparison to the WaveNet models, and also provided a direction along which we can make a further improvement to our models.

The Table 1 below shows the comparative results from the average 5-scale MOS experiment together with conditional log-likelihoods on the test set. This shows that the WaveNet model gets the higher CCL and MOS. Our model has a slightly worse performance than the WaveNet model. Given the time limitation of our project, this evaluation result could be accepted.

The results generated by our model are provided as several figures. The random noise samples were used as inputs for our model and WaveNet models to successfully create the waveform audios, with the application of normalizing flows. The standard deviation for WaveGlow were set in proper values to theoretically generate the best quality of

Average CLL and 5-Scale MOS Comparision		
Methods	Test CLL	5-Scale MOS
WaveNet	3.90 ± 0.060	4.00
Ours	2.96 ± 0.23	3.00

Table 1. Average CLL and 5-Scale MOS Comparision

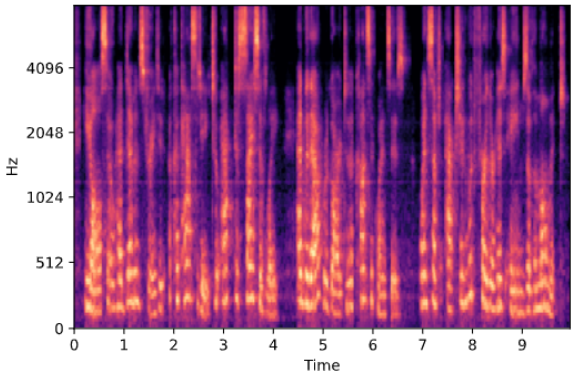

Figure 10. Spectrogram of The Third Result

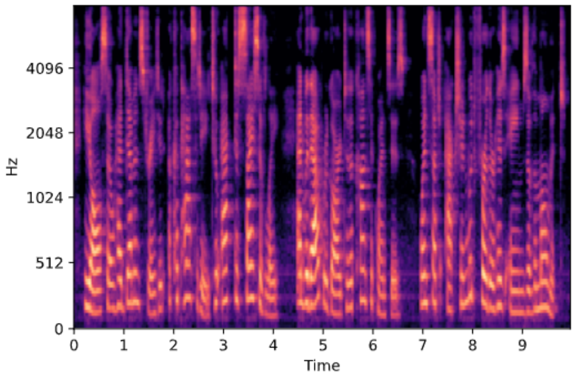

Figure 11. Spectrogram of The Third Original Audio

audio results.

In terms of those aspects hard to evaluated by numerical methods, like the sound timbre and quality, comparing our results with WaveNet model's, our model's results do not have too much glitches and white noise, thanks to the single stage training and the single maximum likelihood loss.

On the other hand, even though WaveNet seems to have the better fidelity, its ancestral sampling was relatively slower. Table 2 shows the average inference speed comparison for each model. Our model's samples/sec rate is approximately 2300 times faster than that of the WaveNet model. These are the figures of the three audio results (Fig. 7, 8, 10,) generated along with its several figures (Time-Frequency figure, and Spectrogram).

As we can see from the figures (Fig. 6, 7, 8, 9, 10, 11), for the spectrograms, each pair of graph is fairly similar to each other but the generated results have less details, and for the Magnitude-time diagrams (Fig. 12, 13, 14, 15, 16, 17) we still can see there are a lot of differences at each point of time, the original audio's magnitudes are tend to be smaller. By comparing with the original audio files to generated audio files, it is clear that the results are different but acceptable.

Comparison of Average Iters/Second and Samples/Speed		
Methods	Samples/Sec	Iteration/Sec
WaveNet	166	N/A
Ours	$350k$	0.629

Table 2. Comparison of Average Iters/Second and Samples/Speed

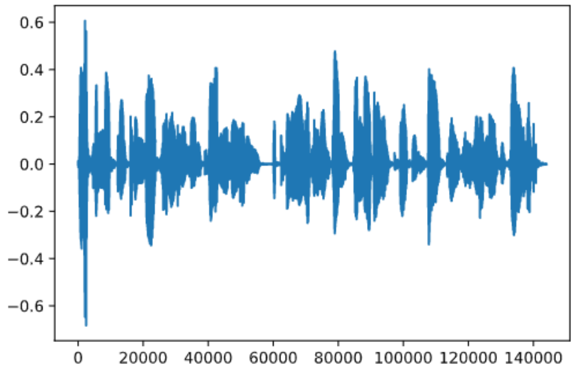

Figure 12. Magnitude of The First Original Audio

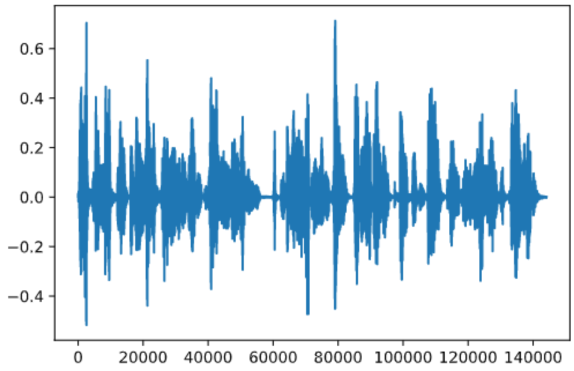

Figure 13. Magnitude of The First Result

5. DISCUSSION AND CONCLUSION

5.1 Discussion

Based on the evaluation of the results, we have basically completed the our initial goal - to implement a vocoder, achieve voice synthesizing, and produce some relatively human-like synthesizing speech. However, compared to the ideal results of WaveNet and other models, the speech result we produced is not ideal in terms of clarity and emotion. The main reasons why our model is not good enough could be classified into two points and will be further improved in future work. The first main reason is the limited time. Our current result is the result of training on the data set for several days, which is not a long enough time to generate perfect results, given the flaws of our model. We will extend the training time and then evaluate the training results in the future. Based on the results, we will decide whether to extend the training time again. The second main reason is the configuration of the model. The configuration we chose was a model with just 6 flows. We are currently unable to determine the most suitable configuration, so we will experiment with different configurations.

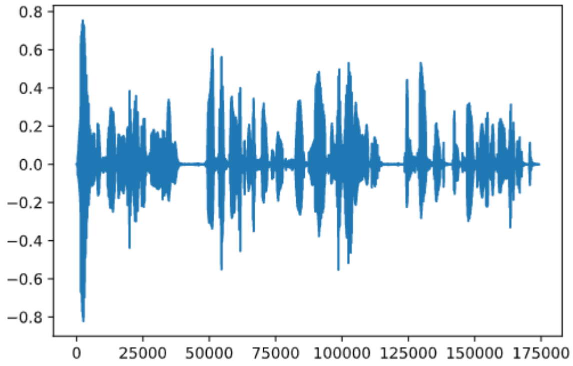

Figure 14. Magnitude of The Second Original Audio

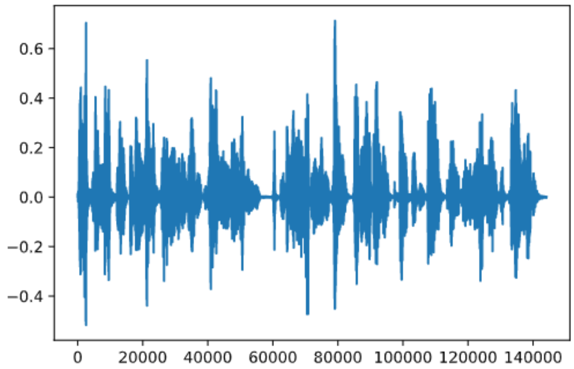

Figure 15. Magnitude of The Second Result

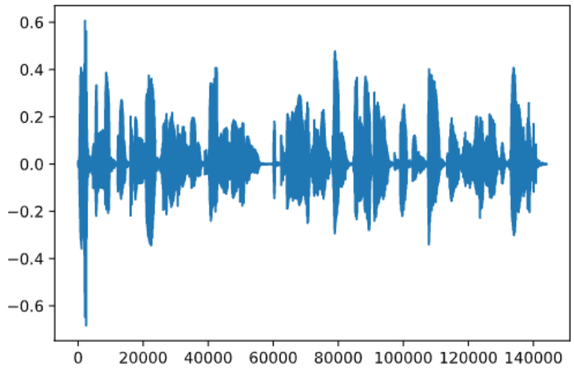

Figure 16. Magnitude of The Third Original Audio

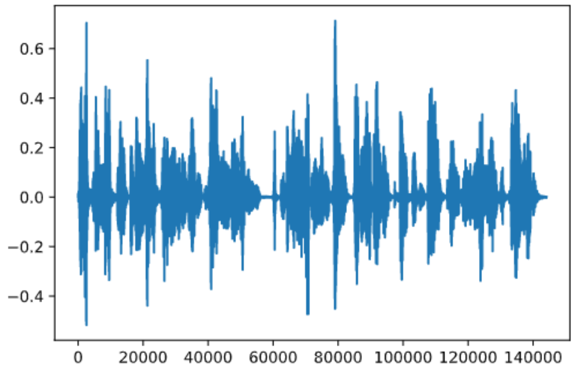

Figure 17. Magnitude of The Third Result

We will try to implement some models with different numbers of blocks and flows, compare their results to the results generated by the original model, and finally select the most suitable model configuration.

5.2 Conclusion

In this project, using WaveNet and WaveGlow as main references, we implemented a flow based generative model that can accomplish a real-time parallel audio synthesis in a reasonable time. Some important techniques including context block structure, affine coupling, flows architecture, activation normalization, inverse autoregressive flow, squeeze operation and so on are used. In the limited time, compared to WaveNet, our model shows acceptable efficiency and promising results.

5.3 Potential Area of Future Works

Given our study and experiment on WaveNet and WaveGlow models, it is safe to say that a flow based model is an excellent improved alternative to models using only neural networks. Additionally, a flow based generative model could be used to generate not only speech and music but also graphs. This field of study has great value in both academia and industry. One recent work demonstrates that by first converting 1-d ECG data to 2-d ECG images and then inputting the images as data to CNN to have higher resolution on abnormal pattern detection. We would like to try similar work by converting 1-d mel-sepctrogram to 2-d mel-spectrogram image, and having 2-d data input to the flow. We hope this could help capture more subtle change in audio, and further generate a clearer and emotional audio.

6. REFERENCES

[1] Jonathan Boilard, Philippe Gournay, and R. Lefebvre. A literature review of wavenet: Theory, application and optimization. *journal of the audio engineering society*, 03 2019.

[2] Sergio Correia, Paulo Guimarães, and Thomas Zylkin. Verifying the existence of maximum likelihood estimates for generalized linear models, 2019.

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[4] Zeyu Jin, Adam Finkelstein, Gautham J Mysore, and Jingwan Lu. Fftnet: A real-time speaker-dependent neural vocoder. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2251–2255. IEEE, 2018.

[5] Zeyu Jin, Adam Finkelstein, Gautham J. Mysore, and

Jingwan Lu. FFTNet: a real-time speaker-dependent neural vocoder. In *The 43rd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2018.

[6] J. D. Kalbfleisch and D. A. Sprott. The title of the book. 2012.

[7] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis, 2018.

[8] Sungwon Kim, Sang gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. Flowavenet : A generative flow for raw audio, 2019.

[9] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions, 2018.

[10] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio, Sep 2016.

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

[12] D. Rethage, J. Pons, and X. Serra. A wavenet for speech denoising. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5069–5073, 2018.

[13] D. Rethage, J. Pons, and X. Serra. A wavenet for speech denoising. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5069–5073, 2018.

[14] Thalles Santos Silva. Assessing audio mean opinion score with deep learning.

[15] Tomoki Uemura, Janne J Näppi, Yasuji Ryu, Chinatsu Watari, Tohru Kamiya, and Hiroyuki Yoshida. A generative flow-based model for volumetric data augmentation in 3d deep learning for computed tomographic colonography. *International journal of computer assisted radiology and surgery*, November 2020.