TECHNICAL UNIVERSITY OF CRETE

School of Electrical and Computer Engineering

DIPLOMA THESIS

Music Generation Using Deep Learning

Author:
Ileanna SOTIROPOULOU

Committee:
Assoc. Prof. Michail G. LAGOUDAKIS
Prof. Michalis ZERVAKIS
Dr. Vassilios DIAKOLOUKAS

Chania, September 2021

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημιουργία Μουσικής με Χρήση Βαθιάς Μάθησης

Συγγραφέας:
Ιλεάννα ΣΩΤΗΡΟΠΟΥΛΟΥ

Επιτροπή:
Αναπλ. Καθηγητής Μιχαήλ Γ. ΛΑΓΟΥΔΑΚΗΣ
Καθηγητής Μιχαήλ ΖΕΡΒΑΚΗΣ
Δρ. Βασίλειος ΔΙΑΚΟΛΟΥΚΑΣ

Χανιά, Σεπτέμβριος 2021

"Information contains an almost mystical power of free flow and self replication, just as water seeks its own level or sparks fly upward."

Neal Stephenson

Abstract

Machine learning and specifically deep learning methods have been applied to complex signal processing problems with remarkable results. Recent breakthroughs in audio synthesis involve the use of end-to-end deep neural networks to model speech in the auditory domain. WaveNet is one such model that is currently considered state-of-the-art in speech synthesis. In this thesis, we investigate the use of WaveNet and WaveRNN as vocoders for musical synthesis. Furthermore, we investigate WaveNet's potential to capture emotive patterns and create emotional music. Prior to choosing an optimal set of parameters for each model, it was critical to consider the spectral and structural distinctions between speech and music signals. Regarding the vocoders, we employed mel spectrograms as temporal local labels for audio reconstruction. The mood-conditional network received no structural instruction and was instead left to generate original audio, conditioned only on a specific mood tag. The models were trained intensively for a minimum of 9 days with WaveNet vocoder converging after 19 days. Synthesized waveforms were evaluated subjectively by human judges, as well as objectively with the use of the PESQ algorithm. Additionally, the respondents were asked to evaluate the mood-conditional samples by guessing the mood of each track. While WaveRNN eventually proved unfit for the nature of our problem, WaveNet-reconstructed waveforms are extraordinarily similar to the originals, with their 5-scale Mean Opinion Scores exceeding 4.0 in both subjective and objective evaluation. Also, remarkably, the majority of responders accurately predicted the moods of all four tracks. This result leads us to anticipate that with additional instruction, WaveNet will be able to respond to emotional cues and automatically create music that is clearly influenced by the range of human emotions.

Περίληψη

Η μηχανική μάθηση και ιδιαίτερα οι μέθοδοι βαθιάς μάθησης, έχουν εφαρμοστεί σε περίπλοκα προβλήματα επεξεργασίας σήματος με αξιοσημείωτα αποτελέσματα. Πρόσφατες καινοτομίες στη σύνθεση ήχου βασίζονται στη χρήση πολύ βαθιών νευρωνικών δικτύων για τη μοντελοποίηση της ομιλίας απευθείας στο πεδίο του ήχου. Στην παρούσα διπλωματική εργασία, διερευνούμε τη σύνθεση μουσικής, αξιοποιώντας τα δίκτυα WaveNet και WaveRNN ως κωδικοποιητές (vocoders). Επιπλέον, διερευνούμε τη δυνατότητα του WaveNet να αποτυπώνει μοτίβα διάθεσης και να δημιουργεί συναισθηματική μουσική. Πριν από την επιλογή ενός βέλτιστου συνόλου παραμέτρων για κάθε μοντέλο, ήταν σημαντικό να ληφθούν υπόψη οι φασματικές και δομικές διαφορές μεταξύ των σημάτων ομιλίας και μουσικής. Όσον αφορά τους κωδικοποιητές, χρησιμοποιήσαμε φασματογράμματα mel ως ετικέτες για την ανακατασκευή ήχου. Το εξαρτώμενο από τη διάθεση δίκτυο αντίθετα, δεν έλαβε καμία χωρική πληροφορία και αφέθηκε να παράγει πρωτότυπο ήχο εξαρτώμενο μόνο από μία ετικέτα διάθεσης. Τα μοντέλα εκπαιδεύτηκαν εντατικά για τουλάχιστον 9 ημέρες με τον κωδικοποιητή WaveNet να συγκλίνει μετά από 19 ημέρες. Οι παραχθείσες κυματομορφές αξιολογήθηκαν υποκειμενικά από ανθρώπινους κριτές, καθώς και αντικειμενικά μέσω του αλγορίθμου PESQ. Επιπλέον, οι ερωτηθέντες κλήθηκαν να μαντέψουν τη διάθεση δειγμάτων που είχαν παραχθεί από το μοντέλο με προκαθορισμένη διάθεση κατά το στάδιο της σύνθεσης. Ενώ το WaveRNN αποδείχθηκε τελικά ακατάλληλο για τη φύση του προβλήματός μας, οι κυματομορφές που έχουν ανακατασκευαστεί με το WaveNet είναι σχεδόν πανομοιότυπες με τις πρωτότυπες και επιτυγχάνουν Μέσες Βαθμολογίες Γνώμης σε 5-βάθμια κλίμακα (MOS) που ξεπερνούν το 4,0 τόσο στην υποκειμενική όσο και στην αντικειμενική αξιολόγηση. Αξίζει επίσης να σημειωθεί πως η πλειονότητα των κριτών προέβλεψε σωστά τις προκαθορισμένες διαθέσεις και των τεσσάρων κομματιών. Με βάση τα παραπάνω θετικά ευρήματα, αναμένουμε πως, με επιπρόσθετη καθοδήγηση, το WaveNet θα μπορεί να εντοπίζει συναισθηματικά στοιχεία και να δημιουργεί με αυτόματο τρόπο μουσική με σαφή την παρουσία ανθρώπινου συναισθήματος.

Acknowledgements

First, I would like to convey my deepest appreciation to my supervisor Associate Professor Michail G. Lagoudakis for his mentoring both throughout my undergraduate studies and during the course of this thesis.

Also, I would like to extend my heartfelt gratitude to my advisor Dr. Vasilios Diakoloukas, who not only offered precious guidance during this project, but was also the one who introduced me to the world of pattern recognition and machine learning. I would like to thank him for our great collaboration and for always responding to my questions.

I would also like to acknowledge and deeply thank Dr. Vasilios Tsiaras. Without his expertise on WaveNet, speech synthesis and his overall assistance, this thesis would not have been possible.

Last but not least, I am grateful to my family for their love and support all of these years. I am also thankful to all my friends who stayed by my side through my studies. Finally, I'm eternally grateful for Kostis, my partner in "crime", my guiding light, the person whose energy helps me overcome any challenge. I could not have done this without you.

Contents

To Matilda

Chapter 1

Introduction

Artificial Intelligence (AI) and its applications in an increasing number of scientific fields are the subject of ongoing research. This integration is necessary in an era when the amount of data available exceeds the capacity of human labor. Thus, pattern recognition was introduced as a method for AI systems to learn on their own by extracting patterns from data. This resulted in the development of neural networks (NNs), which are computing systems inspired by the function of the human brain. Deep learning via deep neural networks (DNNs), or networks with a large number of layers, is frequently used in research to improve data exploitation efficiency. The usage of deep neural networks for commercial purposes has increased dramatically in the recent years. Deep learning has become the state-of-the-art in most domains.

Neural networks have proven to be extremely effective at processing data that people generally handle intuitively, such as sound and image. Very deep generative neural networks are used in speech synthesis to generate human voices after adequate training or to emulate someone’s tone. WaveNet [1] and WaveRNN [2] are recent advances in the field of speech synthesis that perform end-to-end audio generation with amazing results.

DeepMind, the team behind WaveNet and WaveRNN have also been successful in generating music with these models. Even when not paired with additional structural or symbolic information, WaveNet can produce audio that sounds musical. WaveNet is a revelation in the field of end-to-end automatic music creation posing the challenge of making the generated audio as realistic as possible [3]. As poet Henry Wadsworth Longfellow said, "Music is the universal language of mankind", and therefore it would be fascinating to see how technology can interpret a language with such complex rules and emotions.

1.1 Motivation

Music theory constitutes a set of clear rules on how musical sequences should be structured. One way to comprehend music is as a set of notes and chords that conform to the rules of music theory. However, other aspects affecting musical works, such as human emotions, make it more difficult for non-human media to mimic the process of music synthesis.

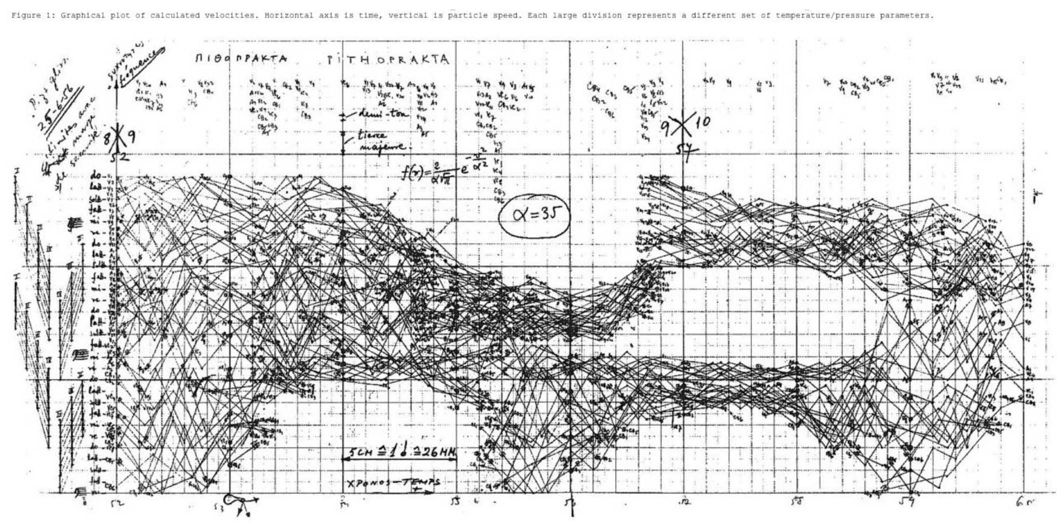


Figure 1: Graphical plot of calculated velocities. Horizontal axis is time, vertical is particle speed. Each large division represents a different set of temperature/pressure parameters.
ΠΙΘΟΠΡΑΚΤΑ
PITHOPRAKTA
demi-ton
tierce majeure
α=35
ΧΡΟΝΟΣ—TEMPS

FIGURE 1.1: Xenakis' Stochastic representations in Pithoprakta

Musical composition is inherently related to mathematics. In 1954, Iannis Xenakis introduced the use of statistics and probability concepts in musical composition to control the orchestral sound masses of Pithoprakta [4]. Later on, in 1956, he named this project *Stochastic Music* and began researching its possibilities. His peculiar notation is shown in Figure 1.1. Xenakis claimed that musical sequences are intrinsically random, hence his compositions were formulated through stochastic theory. This approach raises a question:

"*What is the minimum of logical constraints necessary for the construction of a musical process?* [5]"

Today, there is discussion around music synthesis and automated compositions involving Artificial Intelligence-driven software and deep neural networks. Yet how well would the human factor be simulated by such software? And could it also get emotional? [6]

Combining Xenakis' theory about music being stochastic and the concrete rules of music theory, neural networks should be able to model these complex and contrasting factors. In essence, the output of a deep neural network is fully deterministic, even though training methods are non-deterministic. Furthermore, unlike organized representations, such as rules and grammars, deep learning excels at processing unstructured data from which its layers will generate higher-level representations tailored to the job.

1.2 Related Work

For a long time, automatic music generation has been a topic of active research involving early digital computers [7] or, as previously discussed, pure mathematics [5]. Speech synthesis practices, such as concatenative synthesis, can also be applied in music [8],[9]. The majority of modern music synthesis tools work with symbolic musical representations like MIDI, Piano Roll, or ABC notation. A synthesizer is then used to convert these representations into audio signals.

RNNs with LSTM (Long Short Term Memory) layers are the current state-of-the-art technology for handling symbolic data. Magenta's Performance-RNN module [10] takes MIDI sequences as input and is capable of creating intricate polyphonic musical patterns. While data remains symbolic in its majority, note timings and velocities are taken into account during training, which makes the outcome sound more natural. However, due to the weakness of RNNs to capture long term dependencies, the patterns generated are not particularly interesting or original for extended duration.

Yet, pattern distinctiveness is sometimes more important than originality of a composition. DeepBach [11] is another example of an RNN symbolic music generator that uses a variation of Gibbs sampling [12] to produce notes in the style of Bach chorales. Here, the network is used to capture and reproduce a specific composing style rather than be creative.

The symbolic approach simplifies the modeling problem by operating on a lower-dimensional space. However, there are limitations on what music is generated by these models. These limitations led to the pursuit of the non-symbolic approach through direct modeling in the signal domain. WaveNet [1], as an autoregressive CNN-based model, outperformed parametric and concatenative models in terms of naturalness and fidelity of text-to-speech (TTS) synthesis despite its great computational complexity.

End-to-end music synthesis with neural networks is a rather under-explored domain in comparison to speech synthesis. Likelihood-based models like WaveNet and SampleRNN [13] are so far dominant in the field of raw audio generation both for speech and music synthesis. Apart from autoregressive models other approaches involve the use of Generative Adversarial Networks (GANs) which also have remarkable outcomes in the image domain [14]. WaveGAN[15] was the first attempt and, while the resulting samples' fidelity was not perfect, it inspired other works, like GANSynth [16], that operates on the spectral domain to produce high fidelity musical instrument timbres.

1.3 Thesis Contribution and Problem Statement

In this thesis, we aim to explore how well generative deep neural networks imitate music synthesis in terms of sound quality and naturalness. Our experiments are focused on commercially available deep neural networks for end-to-end speech synthesis.

WaveNet and its successor, WaveRNN, are dominant in text-to-speech synthesis performance. By combining important communication layers of the human voice, such as accents, emotion, and intonation, WaveNet seeks to generate voices that sound more natural than those produced by previous systems. WaveNet voices are currently used in popular Google applications, such as Google Assistant [17]. With WaveNet and WaveRNN currently being state-of-the-art in end-to-end raw audio synthesis, we decided to invest more into testing their limits in music making. WaveNet was preferred over other tools mentioned above, because of its novel network architecture and ability to capture long term dependencies over time. WaveRNN was considered a faster alternative to the long inference times of WaveNet.

Our first goal is to use WaveNet and WaveRNN as vocoders to reproduce high fidelity audio. We compare the generated result with the original piece. Secondly, we attempt to introduce the subjective aspect of human emotion into the training, to influence the generated audio. We divide the dataset into four moods, chosen subjectively by a group of human judges, and we try to enforce this trait to the generated audio. In this case, we are no longer using WaveNet as a vocoder, but rather we exploit its ability to capture long term structure and let it compose with no local information to influence structure over time. The results are highly interesting in both cases.

1.4 Thesis Outline

- Chapter 2 - **Theoretical Background:** A review of theoretical concepts explored in the thesis.
- Chapter 3 - **Deep Music Synthesis:** Detailed presentation of WaveNet and WaveRNN models in terms of architecture and operational properties.
- Chapter 4 - **Experiments:** Detailed analysis of the testing phase and an evaluation of the results.
- Chapter 5 - **Conclusions and Related Work:** A summary of the material of the thesis, discussion and suggestions for future work.

Chapter 2

Theoretical Background

2.1 Convolutional Neural Networks

Convolutional Neural Networks or CNNs are a class of deep neural networks made to imitate human vision in analyzing visual data. They were first used to identify handwritten numbers in the 1980s. However, because of the lack in data and computational resources at the time, CNNs failed to enter the world of industry and were limited to menial tasks. CNNs were revisited in 2012, when enough data and computer resources were available.

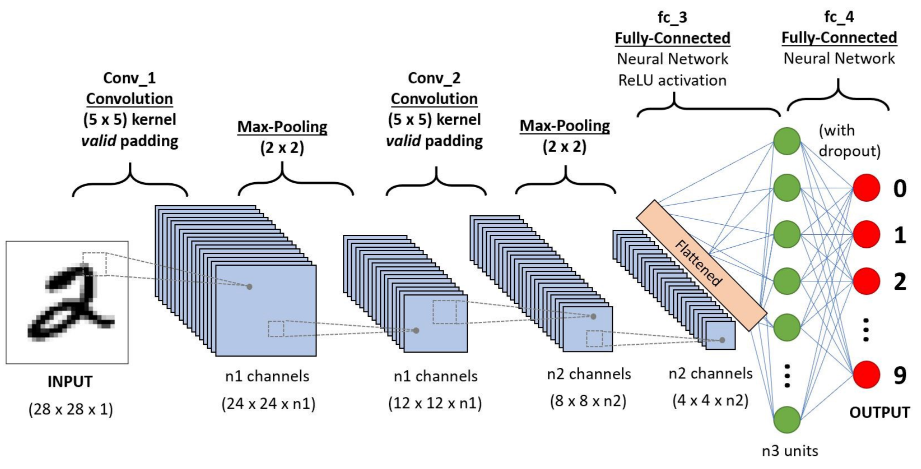


FIGURE 2.1: A basic Convolutional Neural Network.

As the name implies, the main component of convolutional networks is the convolutional layer that tries to learn the feature representation of the inputs. It is made up of many filters (kernels) that are utilized to compute the various feature maps. So, depending on the problem, a $n \times n$ filter is chosen and applied to the input data to produce the convolutional features (Figure 2.1). For instance, Figure 2.2 depicts an example with a 3×3 kernel applied to an image. After adding bias and applying an appropriate activation function, this convolution feature is passed on to the next layer.

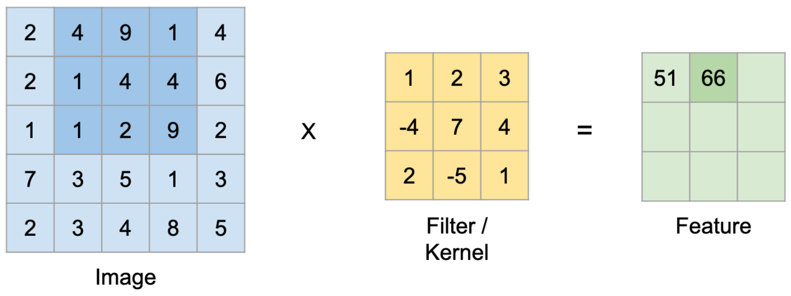


FIGURE 2.2: Convolution operation performed on image data.

In-between successive convolutional layers there is usually a pooling layer that reduces the resolution of the feature maps and the amount of parameters and computations in order to also prevent overfitting. The most common pooling operations are max and average pooling.

After feature extraction there are one or more fully connected layers. Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset.

Convolutional Neural Networks have been used mostly in image recognition and generation tasks, yet it can be fit for speech, sound or video data as well. The main difference is the representation of each kind of input; sound is represented as a one-dimensional matrix, plus one extra dimension for channels and image as a two-dimensional matrix, plus one dimension for channels. For images therefore the convolutions are two-dimensional and the process looks like Figure 2.1 and Figure 2.2, whereas for sound data one-dimensional convolution is applied with a kernel sized $1 \times n$.

The key feature that makes CNNs suitable to handle sound and image input is the compartmentalization of the signal achieved through convolutions. The notion of convolutions is not only limited to computer vision and image data. Many state-of-the art networks used for speech or natural language processing, amongst which is WaveNet [1], incorporate convolutional mechanisms at some point in their architecture.

2.2 Recurrent Neural Networks

RNNs (recurrent neural networks) are a type of neural network that can be used to model sequential data. The core property of RNNs is that they employ an internal state that functions like human memory. This way they retain important attributes of the input, which enables them to predict time-series data, such as speech, text, audio, financial time-series, the weather and so on.

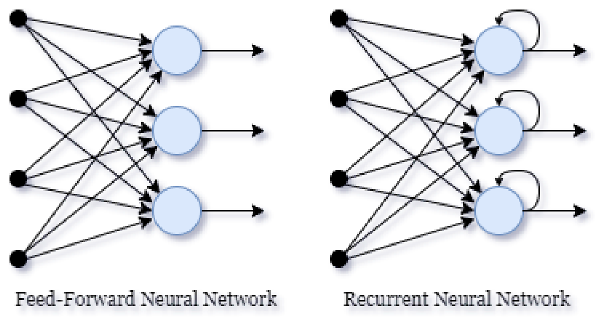


FIGURE 2.3: Visual comparison of a feed-forward Neural Network and a RNN.

Contrary to RNNs, classic feed-forward neural networks have trouble predicting the next state in a sequence due to their lack of memory. Feed-Forward networks perform poorly, when it comes to data with time dependency, as they only focus on the current input and ignore previous states. Any information about the past is assimilated through training, so there is no notion of order. In RNNs, on the other hand, data do not circulate forward, but rather in a loop where each decision takes into consideration previously acquired information (Figure 2.3)

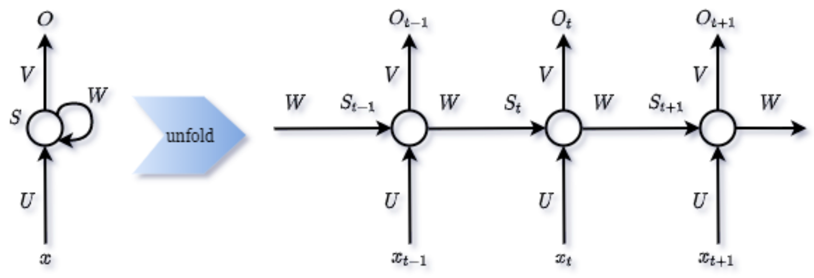


FIGURE 2.4: An expanded view of the RNN hidden layer.

Figure 2.4 shows the main structure of a RNN as well as an expanded view of its structure. RNNs consist of an Input Layer, Hidden Layer, and Output

Layer. The looping arrow in the hidden state is the recurrence, which gives the network its memory. The input at time t is x_t and s_t refers to the memory at time t:

$$s_t = f(Ux_t + Ws_t - 1)$$

where f is an activation function, W is the weight of the input, U represents the weight of the current sample input, and V is the weight of the sample output. In short, information about the immediate past is added to the current state. Unlike CNNs, these parameters are shared through the network, which greatly minimizes the number of trained and estimated parameters.

Weights in a RNN are modified both through gradient descent and through a process called Backpropagation Through Time or BPPT. For BBPT it is important to remember that the error of a timestep depends on the previous time step. With that in mind, the error is calculated for each timestep by back-propagating the error from the last step to the first and the weights are updated. Naturally, for a large number of timesteps this procedure can be computationally expensive.

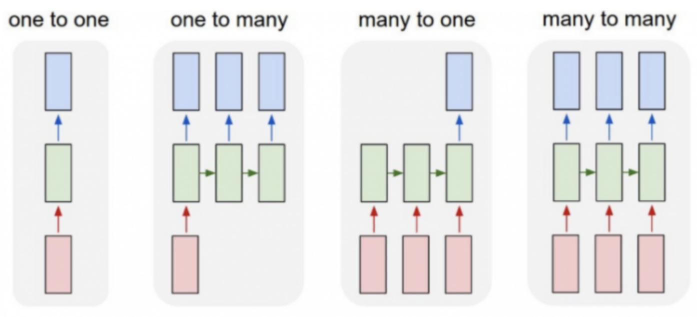


FIGURE 2.5: Mapping options of RNNs. [18]

RNNs also have the ability to map not only one input to one output, but also one to many, many to one, and many to many (Figure 2.5). This allows them to deal with a wide range of applications.

2.2.1 Drawbacks of RNNs

Vanilla RNN usually manifests the problems of either vanishing or exploding gradients. Exploding gradients is an issue commonly encountered in artificial neural networks that perform backpropagation and involve gradients in the learning process. In this case, due to the way the RNN functions, the magnitudes of the weights get extremely large over timesteps, so the gradients "explode". This issue however is usually solved by performing gradient clipping or truncation.

With vanishing gradients, weights with smaller values completely vanish, disrupting the training process, as the model is no longer learning or is learning too slowly. Fortunately, this issue can be solved by using gated RNN cells, such as LSTM.

2.2.2 The LSTM Cell

Long Short-Term memory (LSTM) networks are a helpful addition to the RNN structure, which practically extends the memory of the network. The LSTM cell is a gated cell which decides whether a piece of information is important or not, based on weight assignment. While the algorithm is learning the weights, it is simultaneously learning to differentiate between valuable and unnecessary data.

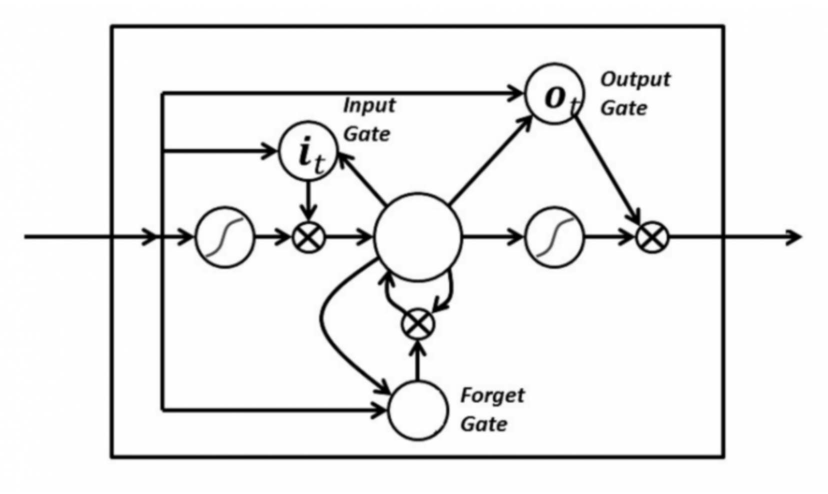


FIGURE 2.6: The gated LSTM cell.[19]

There are three gates in an LSTM: input, forget, and output. These gates determine whether incoming data should be allowed (input gate), whether it should be deleted because it isn't important (forget gate), or whether it should have an impact on the output at the current timestep (output gate). The gated LSTM cell is shown in Figure 2.6. This effective algorithm was further improved with the introduction of Gated Recurrent Unit or GRU [20]. This variant combines the forget and input gates into a single update gate, as well as the data unit state and hidden state, resulting in a model structure that is simpler than LSTM.

2.3 Speech Synthesis

Speech Synthesis is a research field related to the production of artificial human voices and speech. With the implementation of Text-To-Speech (TTS), written text can be converted to synthesized speech. Speech synthesis systems have evolved throughout time in response to recent trends and new possibilities in data collection and processing. While concatenative TTS and parametric TTS have long been the two main methods of Text-to-Speech conversion, the introduction of Deep Learning has brought a new perspective to the problem of speech synthesis, shifting the focus away from human-developed speech features and toward fully machine-obtained parameters.

2.3.1 Concatenative Synthesis

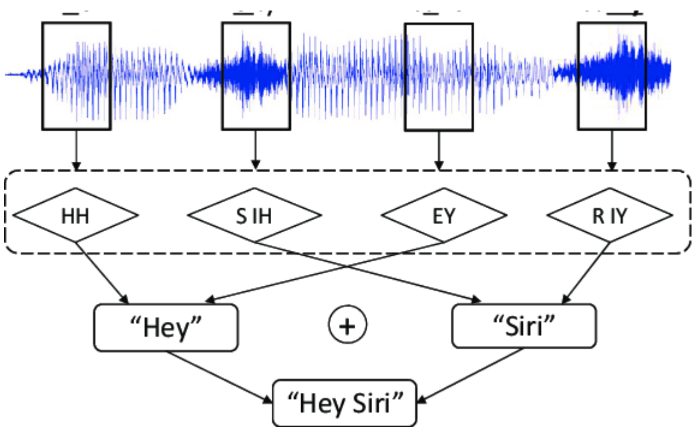


FIGURE 2.7: Concatenative Synthesis

Concatenative TTS is based on the use of high-quality audio snippets that are joined to generate the speech. Voice actors are recorded expressing a variety of speech parts, ranging from full sentences to syllables, which are then tagged and segmented into linguistic features ranging from phonemes to phrases and sentences, resulting in a large database. For speech synthesis, a Text-to-Speech engine matches the input text with relevant content from the database, concatenates them and generates an audio file (Figure 2.7).

Amongst the benefits of this method is the high audio quality in terms of intelligibility and the possibility of retaining the original actor's voice. However, concatenative systems are time consuming, as they require large databases and hard-coding the combinations to form these words. Also, this mix and match method does not guarantee that the emotion, prosody and other speech features will be preserved in the final output speech. The database may be huge, but it is extremely unlikely for the TTS engine to find recordings that match all of these factors. Therefore, sometimes the result may sound flat and unnatural.

2.3.2 Statistical Parametric Synthesis

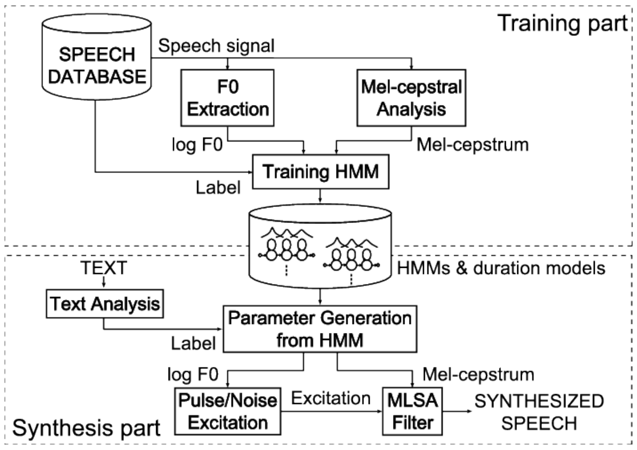


FIGURE 2.8: Parametric or HMM-based Synthesis [21]

Statistical parametric synthesis [22] is a statistical, model-based alternative to the rigid concatenative method. The goal is to generalise the statistical profile of speech in order to train a universal speech model to generate all kinds of speech. The profile of speech is described by parameters, such as fundamental frequency, magnitude spectrum, etc., which are used to train the model. These characteristics are often modeled with the use of Hidden Markov Models (HMMs) as in Figure 2.8.

The parametric method generates more natural speech samples, yet not as intelligible as the concatenative method. There are a lot of artifacts in the produced speech and voices generated may sound flat and robotic. This method is very weak in creating emotional voices, yet incorporates speaker identities and characteristics very well. Parametric systems are also easy to develop, which is why they have been so popular with TTS applications.

2.3.3 The Deep Learning Approach

Deep Neural Network synthesis is a parametric synthesis method with the ability to simulate more complicated dependencies. The audio features used to train a generative deep neural network are usually mel-filter banks or generalized spectrograms of speech segments. WaveNet [1], which debuted in 2016, transformed the area by allowing for the production of natural-sounding speech. Neural speech synthesis has been rapidly evolving since then. As an example, WaveRNN, the more compact RNN version of WaveNet was recently involved in the previously impossible project of giving people with progressive neurological diseases (ALS, Parkinson's etc.) their voices back [23].

2.4 Mel Scale

While vanilla spectrograms provide important information about how a signal's frequencies are distributed across time, they are frequently insufficient. Pairs of similarly spaced sounds picked from higher frequencies sound substantially closer in pitch than pairs of equally spaced frequencies chosen from lower frequencies, according to a psychoacoustics experiment [24]. In fact, the experiment proved that humans perceive frequency logarithmically, and not linearly as is depicted in vanilla spectrograms.

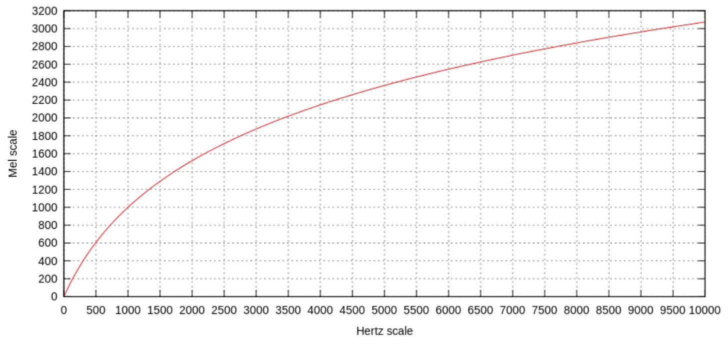


FIGURE 2.9: The mel scale.

This realization led to the mel scale ()Figure 2.9, a perceptual scale of pitches believed by listeners to be equal distance apart. As a reference, 1000 mels were assigned to 1000 Hz. Roughly, 2 octaves in Hz equal about 1 octave in mels above 500 Hz. A more intuitive representation is shown in Figure 2.10 with the help of a somewhat warped keyboard.

Obtaining the Mel representation of a signal is pretty straightforward:

1. Calculate Short-Time Fourier Transform
2. Convert amplitudes to decibels (dB)
3. Convert Frequencies to Mels

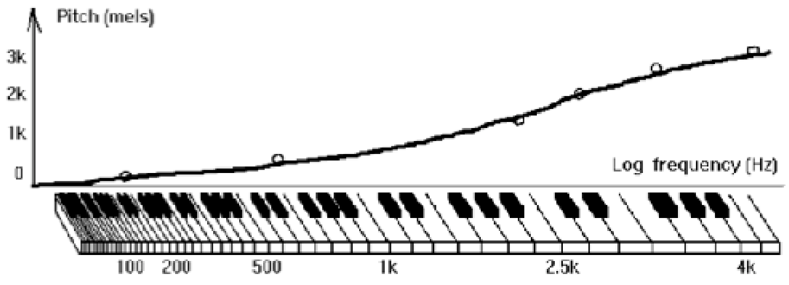


FIGURE 2.10: A warped keyboard reflecting the step distances of the mel scale.

Conversion from frequency (f) to Mels (m) and vice versa can be done according to the following two formulas:

$$m = 2595 \cdot \log_{10}(1 + \frac{f}{700}) \tag{2.1}$$

$$f = 700 \cdot (10^{m/2595} - 1) \tag{2.2}$$

Mel-based spectral representation has a number of benefits, when used as a local conditioner for speech processing in deep neural networks. First, it is a non-linear frequency analysis that approximates the way human hearing works. Also, although it is based on a rather simple formula, Mel spectrum introduces a great level of detail into synthesis and suits various different applications in sound processing and synthesis.

A useful tutorial on Mel filter banks can be found in [25].

Chapter 3

Deep Music Synthesis

In this section, we analyze the main models that were employed for deep music synthesis, namely WaveNet [1], WaveRNN [2], and variations thereof. Both models were thoroughly tested as vocoders conditioned on mel-filter bank labels.

3.1 WaveNet

WaveNet [1] is a recently introduced deep generative convolutional neural network, designed for audio synthesis. It is trained on and generates raw audio waveforms in an auto-regressive and probabilistic manner, a property that makes it suitable for dealing with more complex speech synthesis tasks. Evaluation in comparison with parametric and concatenative models indicates a radical improvement over the previous state-of-the-art in terms of naturalness and fidelity of text-to-speech (TTS) synthesis. Some samples of speech synthesis are available on Google Deep Mind's blog [26].

Despite its impressive results, WaveNet was too computationally demanding to be considered for use in real-world applications, at the time of its release. More specifically, both training and generating tasks of the initially proposed model, involve great computational complexity that eventually slows down the whole process. However, since 2017, faster, optimized versions of WaveNet have been published [27] [28], thereby facilitating the use of WaveNet voices for common TTS applications, such as Google Assistant [17]. Furthermore, WaveNet shows promising results in music modeling tasks.

The following sections offer an overview of WaveNet's main characteristics and architecture.

3.1.1 WaveNet Architecture

The design of WaveNet [1] is largely based on the PixelCNN [29], a network with the ability to predict an image's next pixel by conditioning on previously generated pixels. PixelCNN operates on the two-dimensional space of images so, with WaveNet, the same logic is applied to one-dimensional audio waveforms.

WaveNet models the probability distribution of an audio sequence conditioned on all of the previous samples. Given a sequence of previously generated samples $x = \{x_1, ..., x_{t-1}\}$, WaveNet produces the conditional probability distribution for sample x_t as a chain product:

$$p(x) = \prod_{t=1}^{t} p(x_t | x_1, ..., x_{t-1}) \tag{3.1}$$

By conditioning each sample to the previous ones, WaveNet achieves a new level of naturalness and fluidity in the modeled sound wave. The above property accounts for WaveNet's description as a fully probabilistic [30] and autoregressive [31] model.

In Figure 3.1 the basic layout of the network is shown as originally seen in [1]. Through the following sections, WaveNet is broken down into six different parts, to further clarify the network's architecture as a whole. Those parts are preprocessing, dilated causal convolutions, gated activation units, residual and skip connections, and finally post-processing.

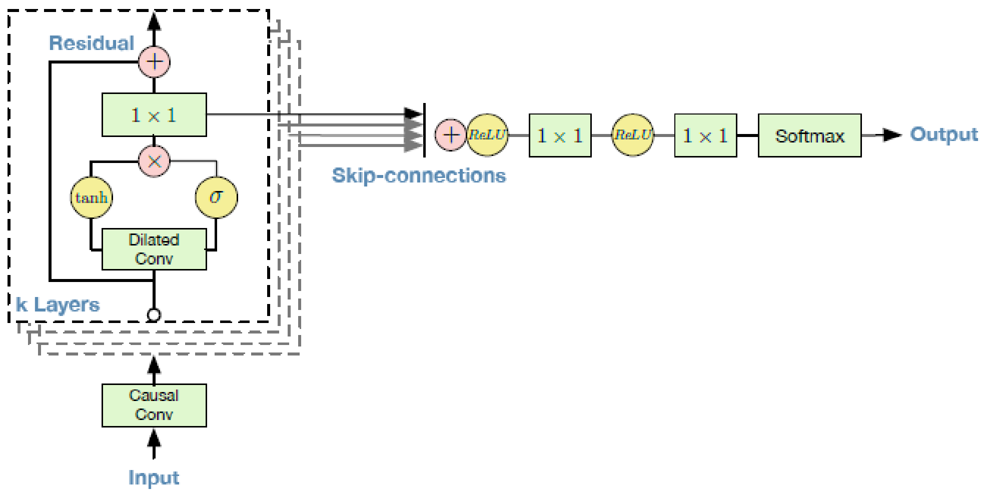


FIGURE 3.1: WaveNet Block Architecture

Preprocessing

Typically, audio referred to as CD-quality has a sampling frequency of 41000 Hz and 16-bit depth. According to the Nyquist Theorem, a sampling rate of 41000 Hz covers the entire human hearing range without aliasing. However, since WaveNet is probabilistic, input needs to be processed and encoded to minimize network complexity. As a first step, audio is converted to 16000 Hz PCM, which is a popular setup for speech synthesis applications and an adequate compromise for music. Then, audio samples are transformed using μ-law transformation

$$f(x_t) = sign(x_t)\frac{ln(1+\mu|x_t|)}{ln(1+\mu)}, \tag{3.2}$$

where $x_t \in [-1, 1]$ and $\mu = 255$. Furthermore, audio is quantized into 256 integer values. This way, audio depth is reduced to 8-bits. Finally, audio samples are encoded to one-hot vectors. As seen in the experiments section, the audio quality reduction induced by storing audio as 8-bit integers does not impact signal reconstruction negatively.

Dilated Convolutions

A key component of WaveNet architecture is the concept of dilated causal convolutions. One-dimensional causal convolutional layers capture the time dependence on the previous samples by combining the properties of causal and dilated convolution filters [32]. The difference between causal and dilated convolution layers is shown in Figure 3.2. The term "causal" indicates that only previous samples are used to predict the next one, thus preserving the time-linearity of the time series. "Dilated" refers to skipping samples by doubling a dilation factor on each filter until a set limit is reached and then repeating the dilation sequence (1, 2, 4, ..., 256, 512, 1, 2, ..., 512, 1, 2, ... 512). The dilated causal filters are then stacked to form the setup in Figure 3.2(c).

Stacked dilated causal convolutional layers are used to increase the receptive field without increasing the depth of the network, while preserving the initial input resolution. As suggested in [33], the exponential increase of the dilation factor of the layers leads to an exponential growth of the receptive field with depth. Dilated causal convolution stacks have longer receptive fields

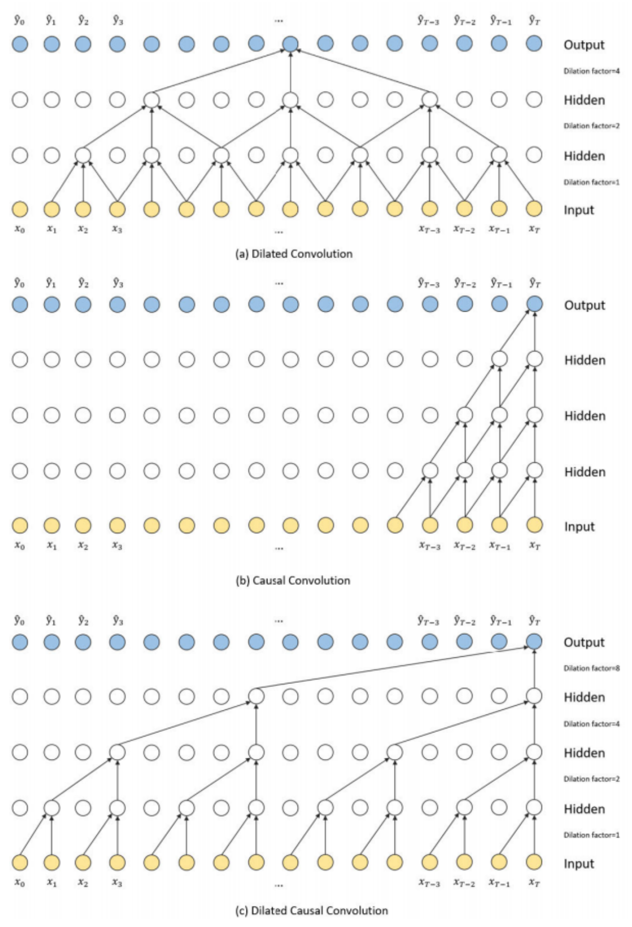


FIGURE 3.2: Visualization of a stack of convolutional layers. (a) Dilated Convolution. (b) Causal Convolution. (c) Dilated Causal Convolution [32]

in comparison with non-causal, non-dilated stacks and are considered computationally efficient and ideal for capturing the temporal characteristics of audio waveforms.

Gated Activation Units

Instead of the typical *tanh* non-linearity, WaveNet features gated activation units after the dilation stacks, which help the network model more complex operations. As first used in PixelCNN [29]:

$$z = tanh(W_{f,k} * r) \odot \sigma(W_{g,k} * r) \tag{3.3}$$

where r is the input of the residual unit, $*$ is a dilated causal convolution operation, $\odot$ denotes the Haddamard product or element-wise multiplication,

k is the index of the layer, $tanh$ is the tangent hyperbolic function, σ is the sigmoid function, $W_{f,k}$ and $W_{g,k}$ are the learnable weight matrices of the filter and the gate respectively. The activation unit essentially decides which information is valuable and worth keeping and which is irrelevant and will be discarded. Both in the WaveNet and the PixelCNN papers it is suggested that, following experiments, gated activation functions work better than rectified linear unit (ReLu) activation functions specifically for audio signal modeling [1] , [29].

Residual and Skip Connections

WaveNet also utilizes residual and skip connection logic to promote faster convergence and, overall, aid in training deeper models. Those connections can be seen in Figure 3.1 as part of the whole architecture. Shortcut connections are implemented by skip connections and identity mappings. This is clarified in Figure 3.3 where we use the nomenclature introduced in the work of He et. al. [34].

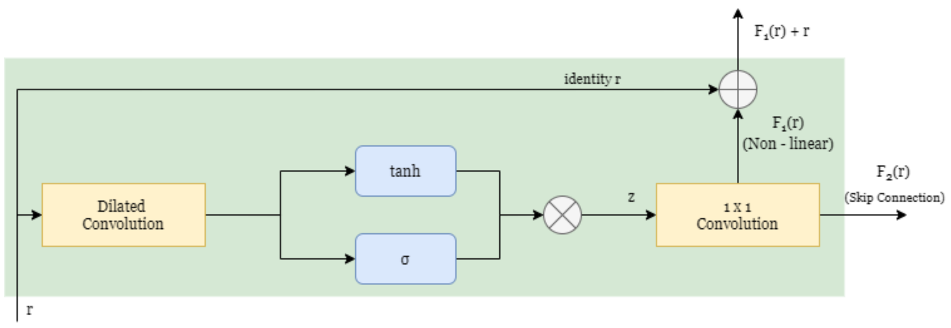


FIGURE 3.3: The Residual Network

Skip connections $F_2(r)$ bypass future residual layers, whereas identity mappings are formed by the element-wise addition of the residual input r to the non-linear output $F_1(r)$. The residual architecture ensures and facilitates gradient propagation throughout the layers, thus solving the vanishing gradient problem, a frequently encountered phenomenon in deep neural networks. Furthermore, a DNN including skip and residual connections is much faster to train than architectures without this utility [34].

Post-processing

Throughout the network, the input data are subjected to numerous operations and, as a result, the depth of the input often increases. A common way to modify the depth or number of feature maps is to implement 1x1 convolutions. 1x1 filters are often referred to as projection layers or channel pooling layers and are used to control the depth of the feature maps. Back in Section 3.1.1, one such layer was implemented to ensure that the depth of the input matches the dimension of the output for addition [32] [35]. In the post-processing section, two 1x1 convolutional layers before the SoftMax function, bring the number of non-normalized probabilities down to the initial discrete values to be classified.

The non-normalized probability distribution predicted by the network is transformed to a proper probability distribution through a SoftMax function:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.4}$$

The SoftMax function simply turns a vector of K real values into a vector of K real values that sum to 1. Therefore, by the end of the post-processing section, WaveNet predicts the normalized probability distribution.

3.1.2 WaveNet as a Vocoder

In order to use WaveNet as a music synthesis tool, temporal consistency was a major requirement. Without temporal information, the output of WaveNet would be incoherent and without structure, therefore we needed to explore the use of WaveNet as a vocoder for music.

Conditioning

Conditioning introduces parameters to the probability distribution function so that it does not only depend on previous samples, but also on some other variables that further describe the audio to be generated.

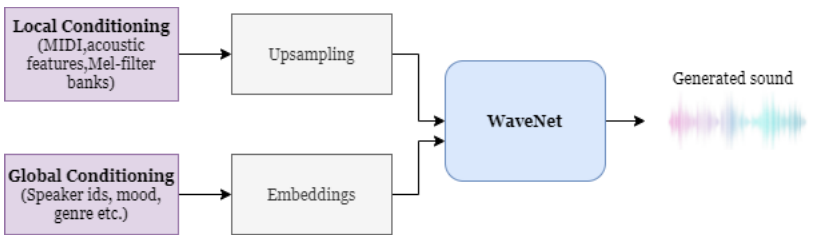


FIGURE 3.4: Conditioning in the WaveNet model

Conditioning is divided into two types, local and global. Global conditioning refers to an inherent property of an audio piece that can be applied in the form of a tag or an embedding vector and is irrelevant to time. On the other hand, temporal information in the form of an additional time series alongside the raw audio is considered local conditioning. An example of a conditioned network is demonstrated in Figure 3.4. In the field of audio synthesis, the two types of conditioning are a common way of guiding the characteristics of the generated audio.

In general, given an input **h** we can model the conditional probability distribution $p(x|\mathbf{h})$ of the audio, given the input **h**:

$$p(x|\mathbf{h}) = \prod_{t=1}^{T} p(x_t|x_1, ..., x_{t-1}, \mathbf{h}) \tag{3.5}$$

By modifying the initial probability distribution to include **h**, WaveNet can incorporate this information into the generated output audio and lead to productions with certain characteristics. In our case, we first needed a way to incorporate temporal information to get structured output audio, and then a way to embed information about the overall mood of the musical piece (happy, sad, aggressive, or peaceful). The first requirement falls into the category of local conditioning, whereas the latter falls into global conditioning.

In the following paragraphs, we demonstrate how the two methods were deployed in relation to our problem statement.

Local Conditioning

Unconditional WaveNets produce sound in terms of the timbre, temperature, and overall nature of the audio, yet they, alone, cannot provide structure to the generated piece. Likewise, Wavenet trained on voice audio without temporal data, would produce mumbling sounds resembling the speaker's tone. In our case, the musical output would sound fairly random, as if a cat were skipping on a piano.

Both of these cases require a second time-series to map the sound generated across time, a method called local conditioning. Distinct examples include the linguistic features of a TTS model, spectral information, or in a music-related scenario the scores in MIDI representation .

Assume we have another timeseries $\mathbf{h_t}$ providing temporal information about the music piece. Of course, most times $\mathbf{h_t}$ has a lower sampling rate than the corresponding audio timeseries. To resolve this, it is proposed in [1] that $\mathbf{h_t}$ is transformed, by using transposed convolution or repeated sampling, to a new time series $\mathbf{y} = f(\mathbf{h_t})$ with the same resolution as the audio. The activation function from (3.3) now is:

$$\mathbf{z} = tanh(W_{f,k} * \mathbf{x} + V_{f,k} * \mathbf{y}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k} * \mathbf{y}) \tag{3.6}$$

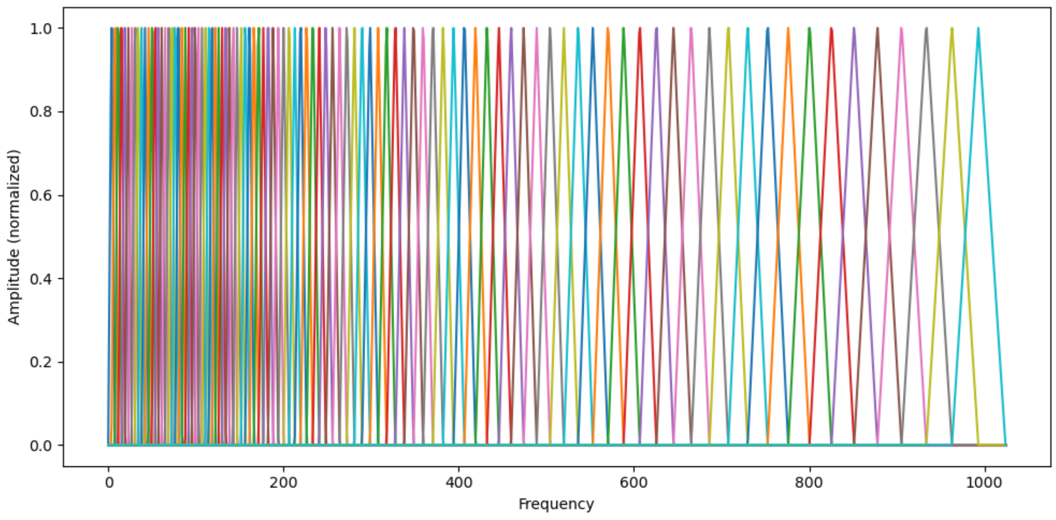
where $V_{f,k}$, $V_{g,k}$ are learnable linear projections of the filter and gate respectively. Naturally, if structure across time were the only concern, then MIDI representation would be a very accurate and flexible condition to feed to the network. A MIDI vector has the form of a binary vector, whose size depends on the quantization values of the audio in the dataset and reads 1, if a note is played in a certain key, and 0, if not.

Therefore, while the use of MIDI as local conditioning appeared fairly simple, we decided in our work to experiment with a more quality-oriented conditioning parameter, such as Mel filter banks.

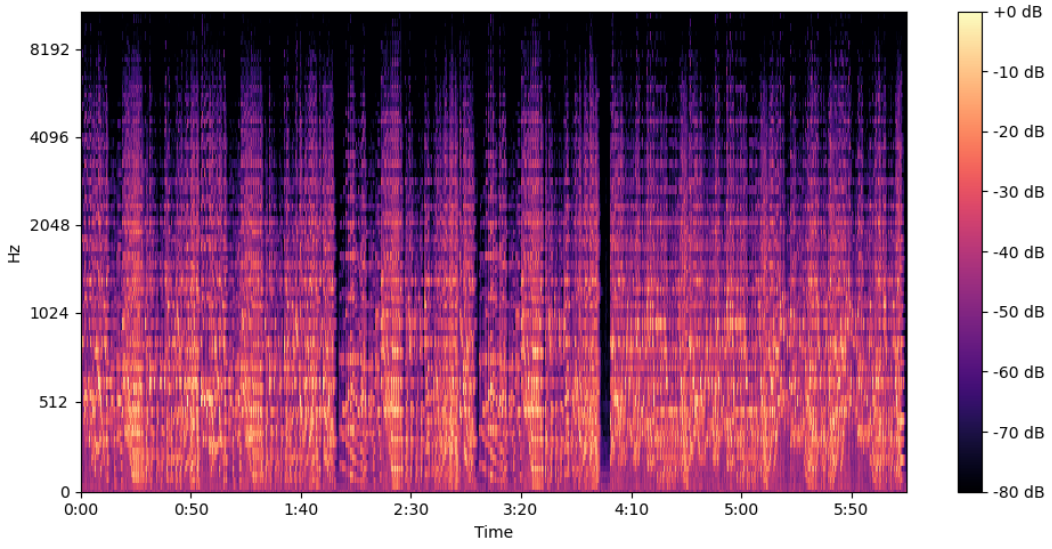
Conditioning on Mel Filter Banks

A common method of local conditioning in sound synthesis applications is the use of Mel Filter Banks. In vocoder context, filter banks are used to analyze the amplitude information of the sub-bands of a modulator signal (such as a voice) and are then used to control the amplitude of the sub-bands of a carrier signal (such as the output of a synthesizer or another voice), thus imposing the dynamic characteristics of the modulator on the carrier [36].

Mel Filter Banks are obtained by applying triangular filters, like those of Figure 3.5a on a Mel-scale to the power spectrum of the audio. The result is a snapshot of the content of each frequency sub-band for every timeframe of the audio. In Figure 3.5b we provide the resulting spectrogram of one of the songs in our dataset. Specifics about the parameters used are analyzed in Chapter 4.



(A) Mel Filter Banks.



(B) Mel Spectrogram.

FIGURE 3.5: Visualization of Mel Scale Local Conditioning information fed to the Network.

Conversion from frequency (f) to Mels (m) can be done according to the following two formulas:

$$m = 2595 \cdot \log_{10}(1 + \frac{f}{700}) \tag{3.7}$$

$$f = 700 \cdot (10^{m/2595} - 1) \tag{3.8}$$

Mel-based spectral representation has a number of benefits, when used as a local conditioner. First, it is a non-linear frequency analysis that approximates the way human hearing works. Also, although it is based on a rather simple formula, Mel spectrum introduces a great level of detail into synthesis and suits various different applications in sound processing and synthesis.

Global Conditioning

Global conditioning refers to a general identity or property that influences the output distribution across all time-steps. By implementing global conditioning, WaveNet can impose general characteristics of the input on the generated audio. Such characteristics can be speaker identity, the mood of a song, the genre of a song, etc.

Using global conditioning input **h**, equation (3.3) of the gated activation unit becomes:

$$\mathbf{z} = tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T \mathbf{h}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T \mathbf{h}) \tag{3.9}$$

where $V^T\mathbf{h}$ is a learnable transformation of **h**, which is broadcast across all steps.

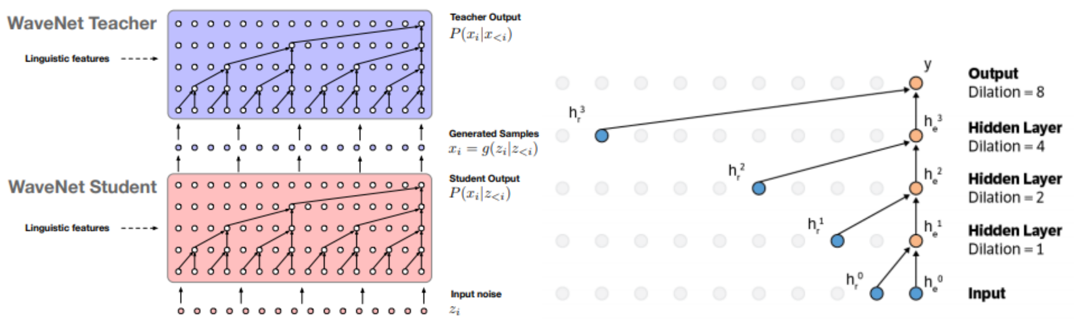
In our WaveNet implementation for music synthesis, global conditioning seems like a reasonable first step in introducing the mood of a musical piece to the training and generation processes. More specifically, we assign tags in order to characterize each song in the dataset as either happy, sad, aggressive, or peaceful.

3.1.3 Inference and Generation

The remarkably realistic results produced by WaveNet can be attributed to the exploitation of both time dependency and the auto-regressiveness of the model. However, one costly disadvantage of auto-regressive models is that inference is slow.

Parallel WaveNet [27] enabled the incorporation of WaveNet in production environments, where speed is pivotal. Parallel WaveNet is inspired by parallel architectures encountered in modern computers, and generation is done by a parallel feed-forward network from a trained WaveNet. This method is referred to as distillation and essentially involves transferring knowledge from a large model to a smaller one. This version is nearly 1000 times faster than the prototype of Oord et. al[1], creating one second of speech in 50 milliseconds.

Another helpful alteration is the Fast WaveNet [37] algorithm. The main concept of the Fast WaveNet algorithm is the implementation of queues to prevent redundant convolution calculations (Figure 3.6).



(A) Parallel WaveNet (B) Fast WaveNet Algorithm.

FIGURE 3.6: Fast WaveNet Implementations

All this research towards compactness and speed has led to WaveRNN, which is analyzed in the following section.

3.2 WaveRNN

Another model worth exploring and experimenting with is WaveRNN [2], whose performance is comparable to that of the state-of-the-art WaveNet. In fact, a WaveRNN model with a rather large number of units achieves results similar to those of the largest WaveNet.

As discussed in previous sections, sequential models imply sampling in a serial manner. However, due to its serial nature, the sampling process can grow prohibitively large. That could happen if, for example, the number of samples increases or if the network is very deep (like WaveNet[1]). Another case, common in deep neural networks, is having large computation times or overhead due to a large number of parameters or too wide layers [2].

The goal in developing WaveRNN was to enable voice synthesis models to run on minimal devices efficiently, whilst maintaining the quality of the generated samples.

3.2.1 Generation efficiency requirements

The generation speed of the sequence generation model, is expressed by the subsequent formula :

$$T(\mathbf{u}) = |\mathbf{u}| \sum_{i=1}^{N} (c(op_i) + d(op_i)) \tag{3.10}$$

$T(\mathbf{u})$ refers to the time needed to pronounce a sentence $\mathbf{u}$, there being $|\mathbf{u}|$ samples in total. As expected, high-fidelity audio has a large number of samples. There is also N representing the number of layers of the neural network. N scales with the number of layers in the network.

$c(op)$ represents the calculation time of each layer. If the network is very wide, or the network has many kernels, the calculation time will be long.

$d(op)$ represents the overhead time of the hardware execution program, including the time of calling the program and extracting the parameters.

In order to generate speech fast, each of the parameters in (3.10) has to be minimized.

3.2.2 WaveRNN Architecture

The architecture of WaveRNN relies on the core property of RNNs that a single recurrent layer applied to the previous state can produce a highly non-linear transformation of the context. It is a single-layer recurrent neural network (RNN) with a dual softmax layer, designed to predict 16-bit audio samples.

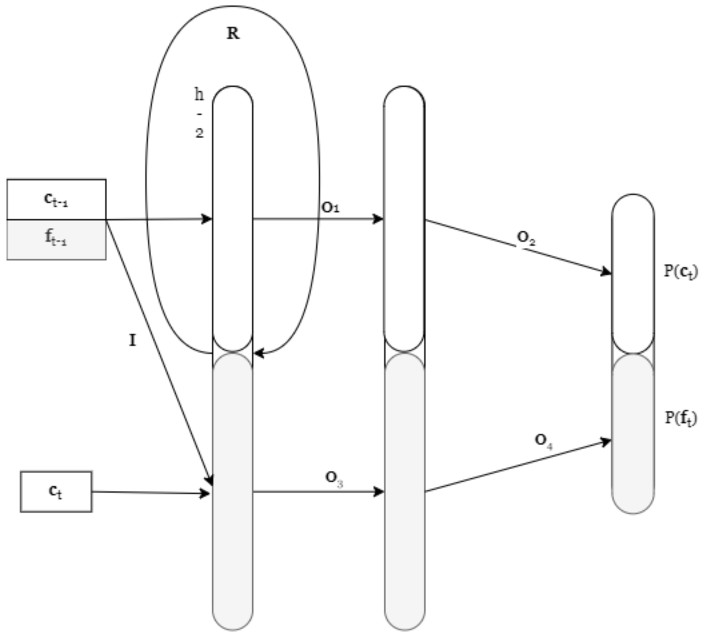


FIGURE 3.7: The architecture of the WaveRNN with the dual softmax layer. [2]

Kalchbrenner et. al [2] intended to make the sampling process faster by efficiently minimizing impactful factors, such as the number of layers, the overhead, computation time, etc. Taking into consideration that sampling is more computationally expensive as the number of parameters increases, the aim is to fully exploit a set number of parameters. This translates essentially to achieving maximum performance for the given computational resources.

This type of architecture requires a small number of operations at each step, less training data, and is faster to train.

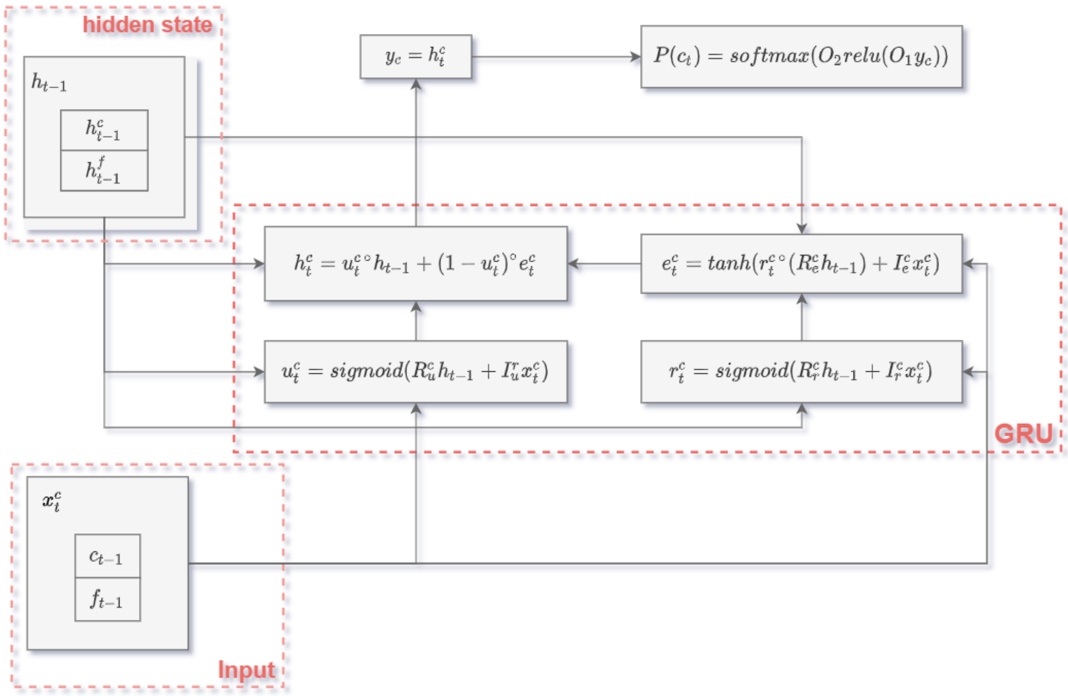
Basic computations of WaveRNN can be seen in (3.11):

$$\begin{aligned}
\mathbf{x}_t &= [\mathbf{c}_{t-1}, \mathbf{f}_{t-1}, \mathbf{c}_t] \\
\mathbf{u}_t &= \sigma\left(\mathbf{R}_u \mathbf{h}_{t-1} + \mathbf{I}_u^* \mathbf{x}_t\right) \\
\mathbf{r}_t &= \sigma\left(\mathbf{R}_r \mathbf{h}_{t-1} + \mathbf{I}_r^* \mathbf{x}_t\right) \\
\mathbf{e}_t &= \tau\left(\mathbf{r}_t \odot \left(\mathbf{R}_e \mathbf{h}_{t-1}\right) + \mathbf{I}_e^* \mathbf{x}_t\right) \\
\mathbf{h}_t &= \mathbf{u}_t \cdot \mathbf{h}_{t-1} + (1 - \mathbf{u}_t) \cdot \mathbf{e}_t \\
\mathbf{y}_c, \mathbf{y}_f &= \text{split}\left(\mathbf{h}_t\right) \\
P\left(\mathbf{c}_t\right) &= \text{softmax}\left(\mathbf{O}_2 \,\text{relu}\left(\mathbf{O}_1 \mathbf{y}_c\right)\right) \\
P\left(\mathbf{f}_t\right) &= \text{softmax}\left(\mathbf{O}_4 \,\text{relu}\left(\mathbf{O}_3 \mathbf{y}_f\right)\right)
\end{aligned} \tag{3.11}$$

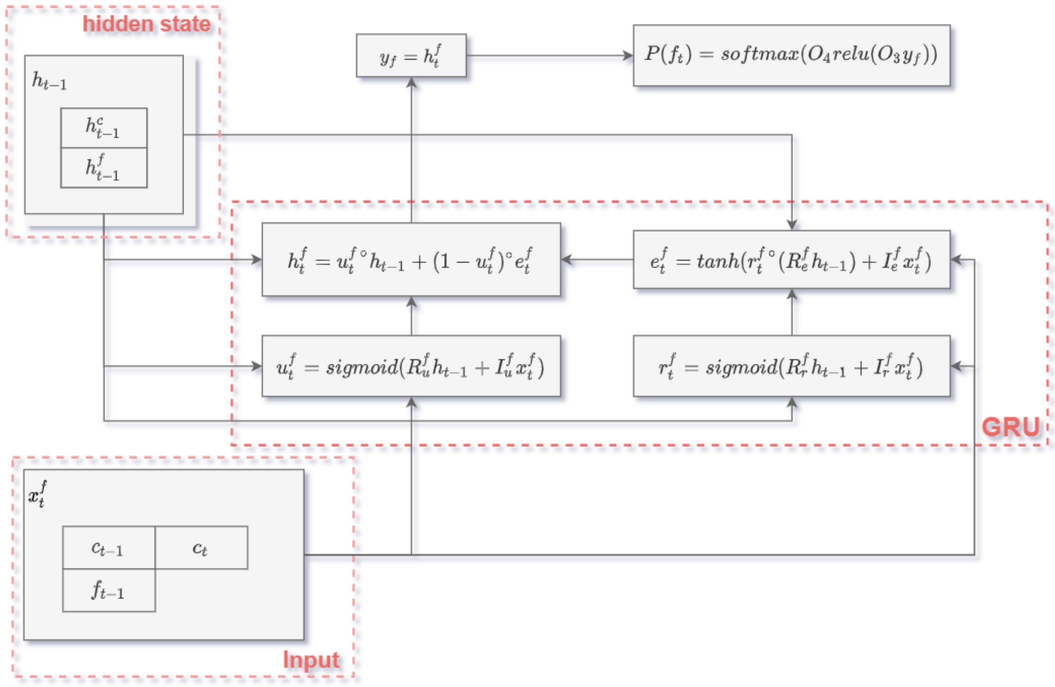
where $\mathbf{I}*$ is a masked matrix by which the last coarse input $\mathbf{c}_t$ is only connected to the fine part of states $\mathbf{u}_t$, $\mathbf{r}_t$, $\mathbf{e}_t$ and $\mathbf{h}_t$ and therefore only affects the fine output $\mathbf{y}_f$. The $\mathbf{R}$ matrices refer to the contributions to all three gates $\mathbf{u}_t$, $\mathbf{r}_t$, $\mathbf{e}_t$, as a variation of the GRU cell [38]. Also, σ and τ are known non-linear functions sigmoid and *tanh*.

The state of the RNN is divided into two parts that predict the 8 more significant (coarse) bits $\mathbf{c}_t$ and the 8 least significant (fine) bits $\mathbf{f}_t$ respectively of the 16-bit audio sample (Figure 3.7) with the split(h_t) function. Each 8-bit part passes through a SoftMax layer and the fine bits are predicted conditioned on the coarse bits. This logic is named Dual SoftMax layer in [2], and achieves more efficient prediction of the 16-bit samples by making use of two smaller spaces of 2^8 values each, rather than the more computationally expensive single large space of 2^{16} values. A visualization of the process described above is demonstrated in Figure 3.8.

Moreover, based on weight pruning techniques [40] applied during training, large and sparser models outperform short, dense models over the same number of parameters.


hidden state
h_{t-1}
h^c_{t-1}
h^f_{t-1}
$y_c = h^c_t$
$P(c_t) = softmax(O_2 relu(O_1 y_c))$
$h^c_t = u^c_t \circ h_{t-1} + (1 - u^c_t) \circ e^c_t$
$e^c_t = tanh(r^c_t \circ (R^c_e h_{t-1}) + I^c_e x^c_t)$
$u^c_t = sigmoid(R^c_u h_{t-1} + I^c_u x^c_t)$
$r^c_t = sigmoid(R^c_r h_{t-1} + I^c_r x^c_t)$
GRU
x^c_t
c_{t-1}
f_{t-1}
Input

(A) Coarse bits WaveRNN Generation


hidden state
h_{t-1}
h^c_{t-1}
h^f_{t-1}
$y_f = h^f_t$
$P(f_t) = softmax(O_4 relu(O_3 y_f))$
$h^f_t = u^f_t \circ h_{t-1} + (1 - u^f_t) \circ e^f_t$
$e^f_t = tanh(r^f_t \circ (R^f_e h_{t-1}) + I^f_e x^f_t)$
$u^f_t = sigmoid(R^f_u h_{t-1} + I^f_u x^f_t)$
$r^f_t = sigmoid(R^f_r h_{t-1} + I^f_r x^f_t)$
GRU
x^f_t
c_{t-1}
c_t
f_{t-1}
Input

(B) Fine Bits WaveRNN Generation

FIGURE 3.8: A schematic representation of generation computations used in WaveRNN [39]

3.2.3 Subscaling

In addition to the elements mentioned in the previous section, [2] proposes a highly parallel speech generation algorithm called subscaling. Longer audio segments can be generated a lot faster through parallelism offered by this algorithm.

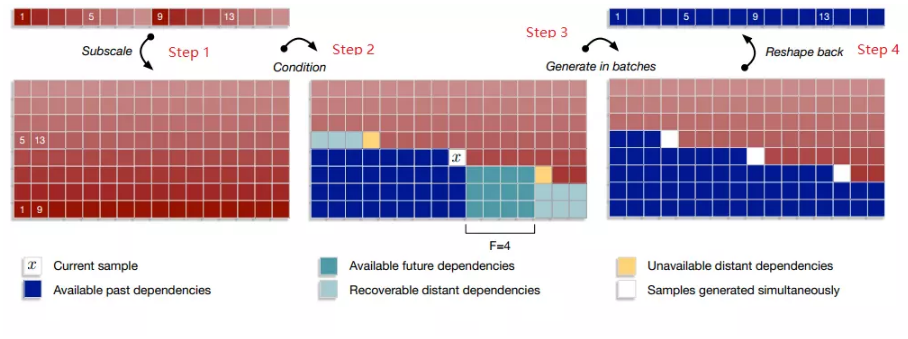

Subscale
Step 1
Condition
Step 2
Step 3
Generate in batches
Reshape back
Step 4
F=4
Current sample
Available past dependencies
Available future dependencies
Recoverable distant dependencies
Unavailable distant dependencies
Samples generated simultaneously

FIGURE 3.9: The dependency scheme of WaveRNN Subscaling method.

The main idea of subscaling is to "trade past for future", that is trade-off some past context of a few samples in exchange for some future context of other samples. Given a target batch size **B**, this is achieved by having the RNN sampling at $\mathbf{1/B^{th}}$ of the waveform sample rate clock rate, with a time offset. Each RNN of the batch sees a different context based on what has already been generated (Figure 3.9). This way, midway through the generation process, **B** samples shall be generated at the time:

$$T(\mathbf{u}) = \frac{|\mathbf{u}|}{\mathbf{B}} \sum_{i=1}^{N} (c(op_i^B) + d(op_i^B)) \tag{3.12}$$

This innovative approach manages to exploit the batch dimension in a way that efficiently exploits the system's computational capacity without sacrificing generation quality. This property renders WaveRNN an excellent choice for speech generation on small low-power mobile CPUs.

Chapter 4

Experiments

4.1 Datasets

The correct choice of dataset is pivotal in every deep learning challenge. When given enough data, both models used in this thesis generate substantially better audio. In our case, we used two high-quality raw audio datasets, which are discussed briefly below.

4.1.1 MAESTRO

MAESTRO (MIDI and Audio Edited for Synchronous TRacks and Organization), first introduced in [41], is a dataset containing about 200 hours of virtuosic piano performances. The full dataset contains around 200 hours of paired audio and MIDI recordings from the International Piano-e-Competition, recorded over a ten-year period. Uncompressed audio is of CD-quality at 44.1–48 kHz 16-bit PCM stereo.

We were interested in the raw audio data, so we did not use the MIDI. For portability reasons we isolated approximately 40 hours of audio from the full 200-hour dataset, thus obtaining a more compact, yet adequate, version of the dataset. This dataset containing single instrument recordings is ideal for music synthesis with the wavenet vocoder.

4.1.2 Harvard's MusicMood

The MusicMood dataset [42], as the name implies, is focused on music and emotion study using optimal design in factorial manipulation of musical features. It is composed of 200 short audio clips that repeat 4-5 basic musical themes played in various manners. The metadata accompanying the dataset are shown in Table 4.1:

TABLE 4.1: Harvard MusicMood Metadata overview.

Factor	**Description**
Register	6 levels (53, 59, 65, 71, 77, and 83 in MIDI pitch, see the paper for details).
Mode	2 levels (1 major, 2 minor)
Tempo	5 levels (1.2, 2, 2.8, 4.4, and 6 NPS, see the paper for details)
Sound level	5 levels (-10, -5, 0, +5, +10 dB)
Articulation	4 levels (1, 0.75, 0.5, 0.25 from legato to staccato, see paper for details)
Timbre	3 levels (1= trumpet, 2 = flute, 3 = horn)
Melody	4 categories (1 = Sad [T01.mid], 2 = Happy [G04.mid], 3 = Scary [P02.mid], 4 = Peaceful [A02.mid]

The four mood categories used (Sad, Happy, Scary, and Peaceful) were not picked at random, but rather according to Russel's psychoacoustic model [43], [44] depicted in Figure 4.1. There are also mean mood ratings from human listeners included with the metadata, which we used as subjective labels for the task of generating mood-specific music.

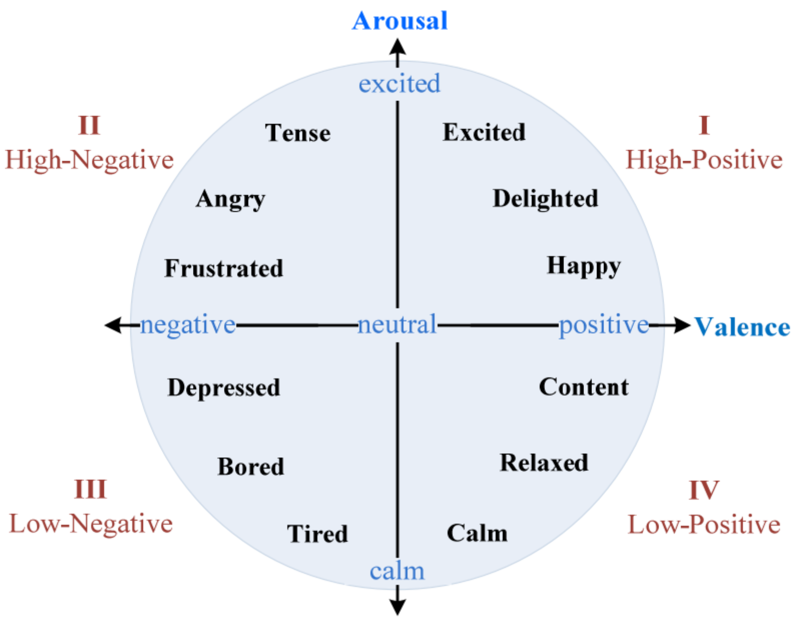


FIGURE 4.1: Russell's two-dimensional Valence-Arousal space for mood definition.

Despite the fact that this dataset only contains one hour of audio, it was chosen because the short duration of the audio clips ensures consistency in mood selection.

4.2 Experimental Setup

In general, all networks were developed in Python 3.6 and implemented with Tensorflow 1.15. Table 4.2 shows the minimal requirements that apply to all network versions.

TABLE 4.2: Minimum Project Requirements.

Package	**Version**
cudatoolkit	10.0
cudnn	7.6
pip	-
python	3.6
tensorflow-estimator	1.15.1
tensorflow-gpu	1.15
matplotlib	-
numpy	-
librosa	0.6

Initial parameter testing of all models was done on a personal computer with 8GB RAM and an NVIDIA GeForce GTX 1050Ti GPU. Within the PyCharm IDE [45] we established an Anaconda [46] environment with the technical specifications of Table 4.2. This configuration, however, quickly proved ineffective and slow, therefore all final models were trained on an HPC unit, provided by the Greek Research and Technology Network, or GRNET [47] On the HPC, we had access to Tesla K40m GPUs and 48GB RAM, which was a considerable performance gain over our previous setup and significantly accelerated the training process.

As previously stated, the research topic of this thesis can be divided into two tasks. The first is vocoder-based music synthesis, and the second is musical theme generation with mood conditioning. Because each case was treated as a separate problem, different models and setups were employed.

4.2.1 WaveNet and WaveRNN with Local Conditioning

We experimented with vocoder implementations for WaveNet and WaveRNN. Code for both WaveNet and WaveRNN vocoders is courtesy of Dr. Tsiaras [48], whom we thank deeply. The goal was to train WaveNet and WaveRNN deep neural networks providing local conditioning data along with the audio files. For the vocoder networks we used our large dataset, MAESTRO.

Preprocessing of the Musical Data

Firstly, audio tracks were mixed from stereo to mono (i.e. single audio channel) and were downsampled to 16kHz. Note that this was done to both datasets to relax computational requirements.

Next, we employed `compute_mel_fbanks_librosa.py` script to extract the local conditioning information, i.e. the Mel-Filter Banks from the MAESTRO dataset. Package librosa [49] is used for audio handling and preprocessing of the audio files. The following set of parameters for the filter banks was defined:

```
frame_length = 0.020    # in seconds
frame_shift = 0.005     # in seconds
n_fft = 2048            # number of FFT components
n_mels = 100            # number of Mel bands to generate
```

Our script builds the appropriate mel filter through the command
`librosa.filters.mel(sample_rate, n_fft, n_mels=n_mels).`
The filter generated is shown in Figure 4.2 :

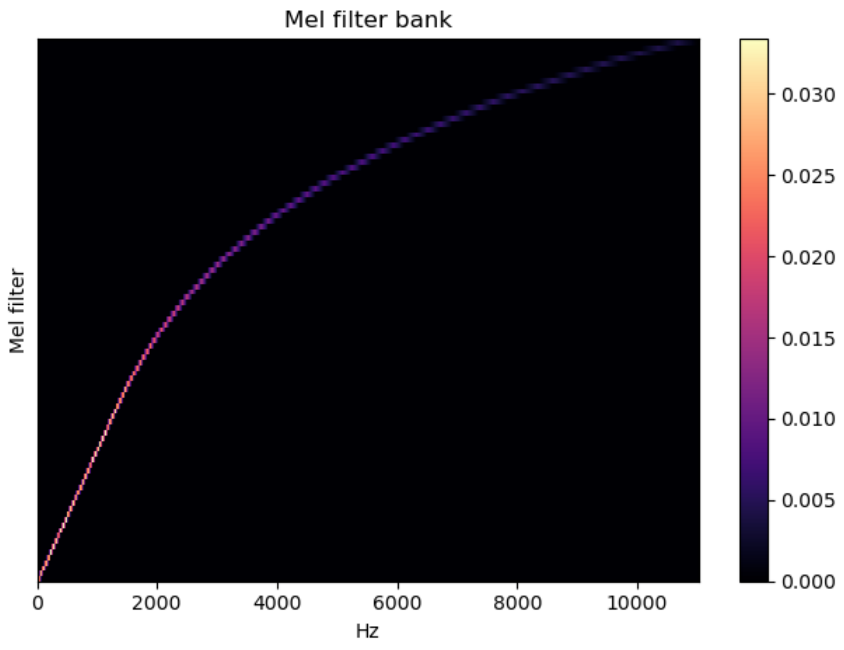


FIGURE 4.2: The Mel filter generated with Librosa.

Following the creation of the mel basis, the signal is pre-emphasized and Short-Time Fourier Transform is applied, yielding the Complex-valued matrix (numpy array) of short-term Fourier transform coefficients. Then, the mel filter of Figure 4.2 is applied to the spectrogram, amplitudes are converted to decibels (dB) and levels are normalized according to a minimum volume level in dB.

If *signal_length* is the duration of an audio clip in seconds then the number of frames of the clip's label can be calculated through (4.1) :

$$n\_frames = \frac{signal\_length - frame\_length}{frame\_shift} \tag{4.1}$$

We obtain the local conditioning labels in the form of numpy arrays of size `(n_mels x n_frames)` by repeating the process described above for the entire dataset. Naturally, the number of frames of each label is less than the total number of samples of the original audio. This means that before the label is fed to the network, it must be upsampled, so that it is alligned with the audio samples.

Network Parameters

The greatest challenge we encountered during the experimentation phase was determining which parameters to use for each network. The configuration of WaveRNN is shown in Table 4.3.

TABLE 4.3: Parameters set for WaveRNN model.

parameter	**value**
label_dim	80
sample_rate	16000
frame_length	0.025
frame_shift	0.010
segment_length	640
segment_shift	640
n_hidden_units	1024
O1_size	256
quantization_channels	256
use_ema	true

First, we attempted to set up WaveRNN with the same parameters that we would use for training on speech samples. Given the dynamic nature of musical data, we decided to greatly increase the receptive field of the network by increasing the number of hidden units from 512 to 1024 . Unfortunately, the training of WaveRNN eventually failed, despite the changes we made. Results slightly improved by lowering the learning rate and applying Gradient Clipping [50] to stabilize training. Generally, RNNs are not the optimal choice for modeling such long term dependencies, as those present in a musical piece. In our case, the issue was that the model could not converge, when trained on musical data.

On the personal computer, a single epoch, or one complete passing of the entire dataset, took about 10 hours. We discuss the results of Section 4.3.1, despite the fact that we dismissed further testing with WaveRNN in the hopes that WaveNet would yield better results.

WaveNet does, in fact, allow more flexibility over the network's density and complexity. Our initial choice of parameters can be seen in Table 4.4. As with WaveRNN, these parameters were derived from previous experience with voice synthesis models. This set of parameters corresponds to a receptive field of 1152 samples or 72 milliseconds. .

TABLE 4.4: Parameters for the small WaveNet model.

parameter	**value**
label_dim	80
sample_rate	16000
frame_length	0.025
frame_shift	0.010
quantization_channels	256
n_residual_channels	32
n_skip_channels	128
filter_width	2
dilations	[1,2,4,8,16,32,64,128,256,512, 1,2,4,8,16,32,64,128,256,512, 1,2,4,8,16,32,64,128,256,512, 1,2,4,8,16,32,64,128,256,512]
use_ema	true

These were considered to be the minimum requirements at the time. Yet, we came to a halt with this set of parameters as well, as the receptive field was too small to capture all aspects of a music clip. Because voice synthesis and music synthesis are *two distinct problems*, these minimum requirements were not sufficient in the music domain.

More specifically, the bandwidth of the music spectrum is twice that of the speech spectrum. Music can stretch past the upper limits of the ear's response at 20 kHz, whereas speech has 90 percent of its power concentrated in frequencies lower than 4 kHz (and limited to 8 kHz). In general, lower frequencies contain the majority of the signal power in music waveforms. [51]. Following the facts stated above, we scaled the model up arriving at the final set of WaveNet parameters shown in Table 4.5.

TABLE 4.5: Parameters for the large WaveNet model.

parameter	**value**
label_dim	100
sample_rate	16000
frame_length	0.02
frame_shift	0.005
n_residual_channels	64
n_skip_channels	128
filter_width	3
quantization_channels	256
dilations	[1,2,4,8,16,32,64,128,256,512, 1,2,4,8,16,32,64,128,256,512, 1,2,4,8,16,32,64,128,256,512, 1,2,4,8,16,32,64,128,256,512]
use_ema	false

By increasing the convolution filter width to 3 we obtain a receptive field of 8187 or 0.5 seconds. This time, the goal was to achieve a balance between the efficiency of the network and the quality of the produced audio. Naturally, for this denser version, training took much longer with one epoch completing in 14 hours on the HPC node. The final trained model has been trained for over 500,000 iterations, which is equivalent to 19 days uptime.

4.2.2 WaveNet with Mood Global Conditioning

The MusicMood dataset was chosen over MAESTRO for the purpose of emotion conditioning, since the mood of the audio clips in it is already rated by human judges. We first attempted to train an implementation of WaveNet both with local labels (mel filter banks) and the mood global label. Each audio clip of the dataset is labeled with one of the four moods (0 = sad, 1 = happy, 2 = scary, 3 = peaceful), and fed to the network as an embedding vector.

By using the vocoder, we aimed at reconstructing the testing clips and also altering their overall feeling through the global label. However, the first challenge we faced was that even upon convergence, the mood label did not seem to be incorporated during inference. It is not clear whether this was a bug or an issue with our initial approach of using both local conditioning and global conditioning.

Thus, the strategy was slightly diverted to letting an unconditioned model compose a few samples, while influenced by the mood label. For this reason, we chose to set up the open source version of WaveNet, which was implemented by Igor Babushkin and the wider community of GitHub [52]. This version, albeit far from perfected, does support unconditioned inference and global conditioning. We chose the parameters of Table 4.6 for our network.

TABLE 4.6: Parameters set for the WaveNet model, with global mood conditioning, but no local conditioning.

parameter	**value**
filter_width	2
sample_rate	16000
dilations	[1,2,4,8,16,32,64,128,256,512,1014,2048, 1,2,4,8,16,32,64,128,256,512,1024,2048, 1,2,4,8,16,32,64,128,256,512,1024,2048, 1,2,4,8,16,32,64,128,256,512,1024,2048, 1,2,4,8,16,32,64,128,256,512,1024,2048, 1,2,4,8,16,32,64]
residual_channels	32
dilation_channels	32
quantization_channels	256
skip_channels	512
initial_filter_width	32

This setup raises the receptive field to an impressive 20,594 samples or 1.28 seconds. The training time was similar to that of the WaveNet vocoder with the model reaching 300,000 steps in approximately 9 days. This time, the label was audible inside the generated audio clips.

4.3 Results Evaluation

The results of WaveRNN and WaveNet vocoders are evaluated with both objective and subjective rating methods. The mood conditioning network has been evaluated only subjectively through a poll.

As a subjective means of evaluation, a questionnaire was distributed blindly through social media requiring no particular knowledge of the participants. It was created in Google Forms [53] and can be found in Appendix A. The questionnaire asked the participants to compare ten original clips and their reconstructions and rate the result on a scale from 1 to 5. Participants were instructed to base their rating on the noise, clicks, closeness to the original and overall perceived quality of the synthesized audio clip.

The first part of the questionnaire included three clips generated with WaveRNN, four clips generated with WaveNet at 300,000 steps and three clips generated with WaveNet at 500,000 steps. Clip comparisons were presented in random order.

The second part comprised of four 20 second waveforms generated by the mood conditional model with one of the four mood tags specified during generation. Participants were asked to listen to each one of the four audio tracks and guess the specified mood.

The subjective evaluation is derived from 35 respondents. In the following sections, we analyze the results for both the vocoder and mood experiments. All musical samples from our experiments can be found here.

4.3.1 WaveNet and WaveRNN Vocoders

An objective evaluation method was deemed necessary for the vocoders, so we used the PESQ (Perceptual Evaluation of Speech Quality) Algorithm [54] which predicts subjective opinion scores of a degraded audio sample. The PESQ score ranges from -0.5 to 4.5, with higher scores indicating higher quality. PESQ is a software program that analyzes audio factors, such as time warping, variable delays, transcoding, and noise. It is primarily intended for use in codec evaluation and network testing, but it may be also used to any audio transmission, according to the developers. PESQ can be thought of as a set of perfect ears capable of evaluating any system. The output of the PESQ algorithm was mapped from [-0.5,4.5] to [1,5] in order to match the score range of the subjective method.

WaveRNN objective and subjective mean opinion scores (MOS) are shown in Table 4.7. As expected WaveRNN had a negative impact both on the listeners and on the objective algorithm. The poor audio quality compared to the original waveforms led to low scores.

TABLE 4.7: MOS for WaveRNN

Sample	Objective MOS	Subjective MOS
MusicMood_002	1.438	1.286
Tchaikovsky - Lullaby, Op. 16/1	2.392	1.514
Chopin Etude No. 8 in F	1.997	1.514
Overall Score	1.915	1.438

More specifically, in the case of WaveRNN, the respondents' ratings of the synthesized waveforms are much more strict than the PESQ ratings, which take into account various objective parameters, such as the noise levels. This finding however is not surprising, given how eerie and unnatural the synthesized audio actually sounds to human ears. While WaveRNN is currently considered state-of-the-art in speech synthesis, it is not yet scalable enough to deal with the much more dynamic spectrum of musical data.

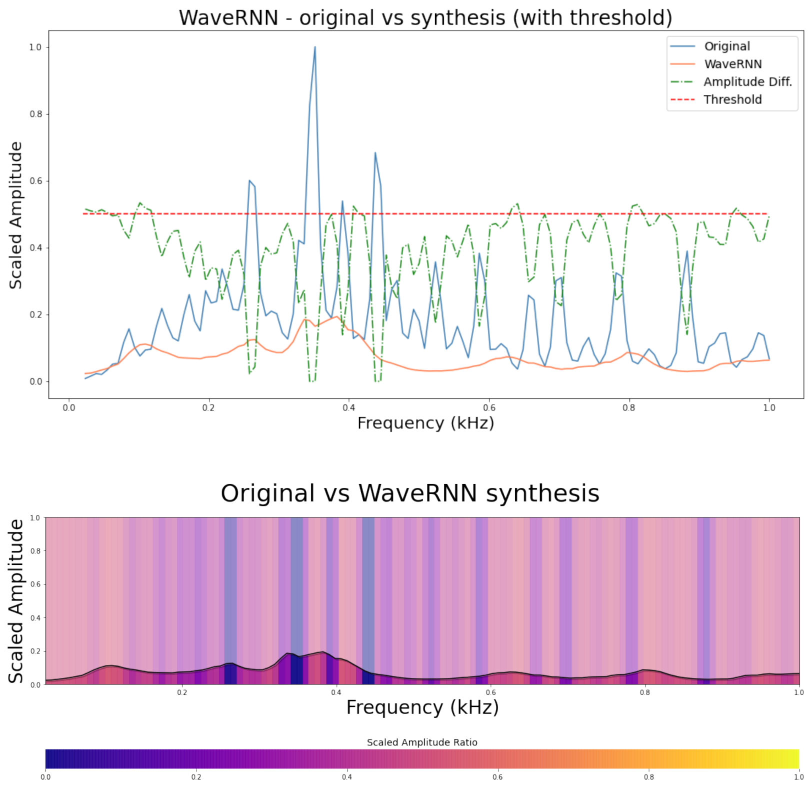


FIGURE 4.3: Spectral comparison of WaveRNN waveform reconstruction

In Figure 4.3 we present a joint plot of the spectral content of one of the audio clips of Table 4.7 and its reconstruction by WaveRNN [55]. The green dashed line shows the amplitude difference of the two signals. There is also a heatmap, where colder regions indicate larger differences in favor of the original track. WaveRNN synthesis, as seen in the graphs, follows the waveform in a broad sense, but lacks spectral detail, which explains the low MOS scores.

The WaveNet Vocoder on the other hand exceeded our expectations. For reference, we extracted results at two different stages during training, at the minimum of 300,000 iterations (Table 4.8) and upon convergence at approximately 500,000 iterations (Table 4.9).

TABLE 4.8: MOS for WaveNet (300,000 iterations)

Sample	**Objective MOS**	**Subjective MOS**
Yiruma - River Flows in You	3.082	4.543
Liszt - Prelude in C Maj	3.657	3.429
Bach - Prelude & Fugue No.16	3.424	3.829
Haydn - Sonata No.62	3.295	3.171
Overall Score	3.365	3.743

TABLE 4.9: MOS for WaveNet (500,000 iterations)

Sample	**Objective MOS**	**Subjective MOS**
Yiruma - River Flows in You	4.066	4.457
Liszt - Prelude in C Maj	3.885	4.257
Bach - Prelude & Fugue No.16	4.013	4.514
Overall Score	3.988	4.410

The first milestone at 300,000 iterations was reached after 10 days of training and was the point where the generated waveforms started to contain audibly fewer errors and clicks. Objective and subjective MOS are significantly higher than those of WaveRNN even at this early training stage. Objective scores fall close to the opinion of the respondents. The algorithm seems to be sensitive to noise crackles and overall errors of the audio that affect the objective score, while in some tracks the human ear seems to overlook minor losses and award a higher rating to the synthesis.

In Figure 4.4 the spectral content of the two waveforms is almost identical as opposed to the WaveRNN case. Although the synthesized waveform exhibits some level discrepancies at some points, there is definitely greater fidelity to the original piece with WaveNet even at 300,000 steps.

Results are nearly perfected upon convergence (Table 4.9) after 19 days of training. At 526,000 steps, differences are barely distinguishable. The PESQ algorithm obviously still spots energy discrepancies and minor noise interference, but overall there is an improvement in objective scores as well.

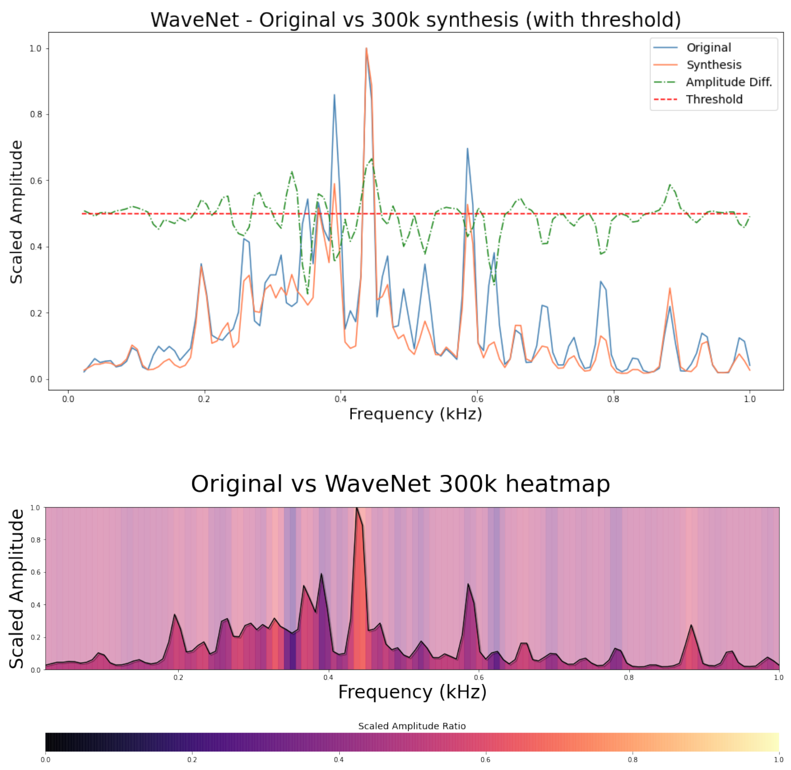


FIGURE 4.4: Spectral comparison of WaveNet waveform reconstruction at 300,000 steps

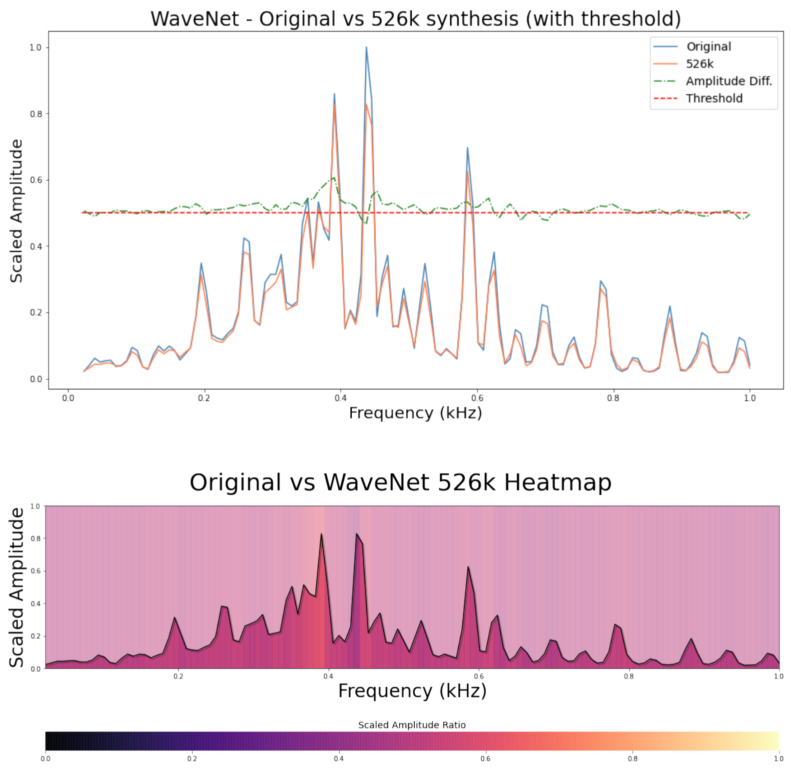


FIGURE 4.5: Spectral comparison of WaveNet waveform reconstruction at 526,000 steps

It is worth noting that out of all songs reconstructed, Yiruma's "River Flows in You" is the only track not belonging to the MAESTRO dataset, yet it achieves remarkably high objective and subjective MOS with the WaveNet model. The choice of this audio clip was a deliberate test of WaveNet's ability of generalizing the distribution learnt.

High subjective MOS is of great significance at this stage, because it means the synthesized audio could possibly deceive even keen listeners. This can be confirmed by observing the spectrum comparison of Figure 4.5. Upon convergence the synthesized waveform is almost identical to the original, with minor differences in some key frequencies. The spectral differences of the converged model, however, are not as pronounced as those in the early training stage comparison.

With WaveNet, we achieve high resolution end-to-end audio reconstruction with a subjective mean opinion score of 4.41 for the final model, which is a massive improvement over the 1.48 subjective score of WaveRNN. Therefore, the high fidelity results we achieved by experimenting with music synthesis prove that the capabilities of WaveNet, as demonstrated through DeepMind's paper, are definitely not limited to speech synthesis.

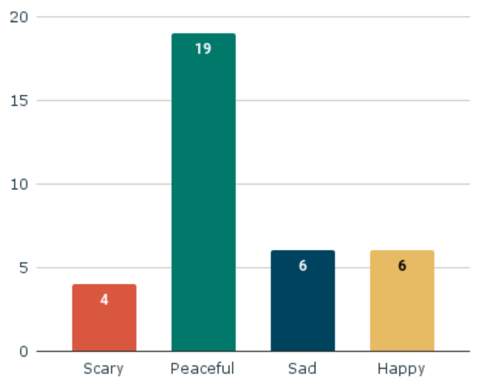
4.3.2 WaveNet with Mood Global Conditioning

For this challenge, local conditioning proved to be too strict, in that WaveNet would not compose freeform tracks.Therefore, we trained open source network on the MusicMood dataset for 300,000 iterations with only global conditioning over four mood labels (Happy, Sad, Angry, Peaceful). Then, we generated samples of approximately 20 seconds to ensure prediction consistency and specified a mood label for each generated waveform.

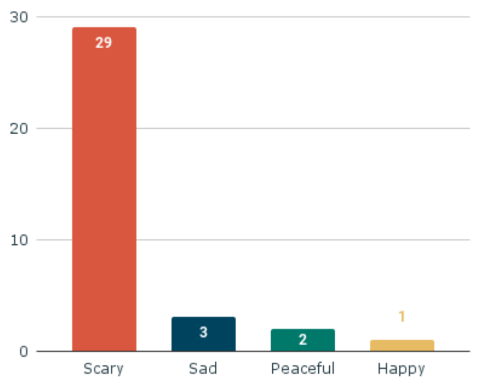
After numerous attempts the most representative sample was selected for each mood and these four samples were included in the second part of the questionnaire. Participants were asked to identify the true mood of each generated sample by listening to it, knowing that there is one of each mood.

As the generated samples are rather cacophonous, rating them proved to be a challenging task. Inference is mode seeking, i.e. it gets stuck in specific patterns, and thus yields monotonous samples. However, some of the generated audio clips incorporate the influence of the specified mood. For example, happy tracks are exhibiting more prominent patterns in major scales, or scary tracks aggregate their content in the bass frequencies.

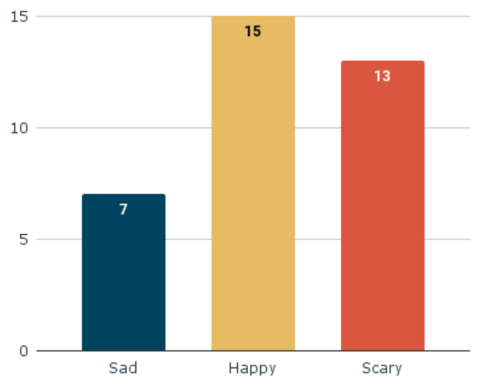
These peculiar patterns that hint the conditioned mood for each sample were apparently clear to most of the respondents as well. Figure 4.6 demonstrates the distribution of the answers obtained through the second part of the survey in the form of cumulative bar graphs.



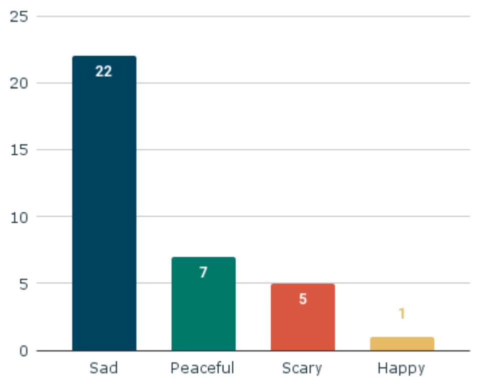
(A) Synthesis labeled "Peaceful"



(B) Synthesis labeled "Scary"



(C) Synthesis labeled "Happy"



(D) Synthesis labeled "Sad"

FIGURE 4.6: Graphical representation of the answers given by respondents asked to identify the mood of each of the four generated musical samples.

The majority of replies for each of the four queries gravitates to the true mood, as shown in the graphs, which is astounding, given the challenges discussed before. For the tracks labeled as scary (4.6b) and sad (4.6d) the poll decision was definitive with over half of the 35 respondents choosing the correct label. On the contrary, the peaceful (4.6a) and happy (4.6c) tracks were considered more obscure, so the consensus is not as clear as with the other two moods.

The key to the success of this particular experiment is first and foremost the fact that it was based entirely on subjectivity. The mood labels included in the MusicMood dataset derive from mean opinion scores of human listeners.

We chose to maintain the anthropocentric nature of this challenge, given that emotions are complex and difficult to model by artificial means.

It's also worth mentioning that these intriguing results were obtained just through experimentation in the audio domain, utilizing WaveNet's capabilities without any extra training with symbolic models or other representations, other than mood tags.

Chapter 5

Conclusions & Future Work

The purpose of this thesis was the assessment of the capabilities of deep neural networks outside the field of speech synthesis by using WaveNet and WaveRNN to synthesize musical pieces with an end-to-end approach. As a separate task, we tried to incorporate the subjective part of human emotion into the training in order to impact the audio that is generated. To achieve these goals, we first trained WaveNet and WaveRNN as vocoders, locally conditioned on mel spectrograms for music reconstruction, then we trained a WaveNet model with only global conditioning on four mood labels and let it generate short mood-based audio clips. All deep learning models were implemented in the Python programming language and utilize Tensorflow 1.15 as the main tool. After an "optimal" (within our investigation) set of parameters was acquired for each problem, the models were trained for more than nine days and were used to generate waveforms.

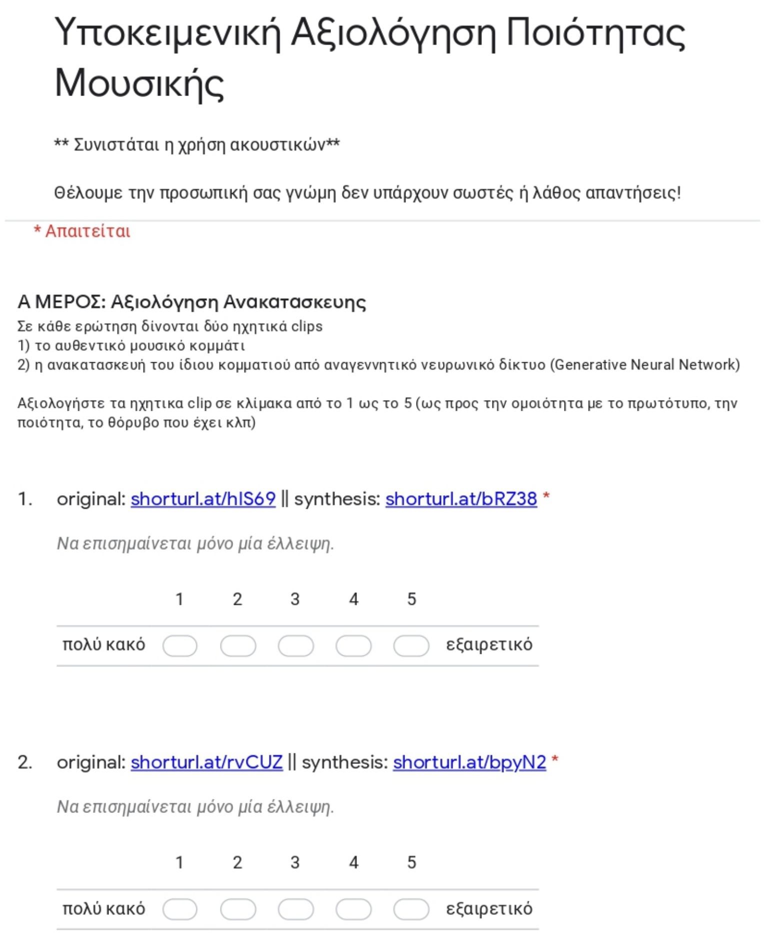
The results are very promising in both cases. All generated waveforms were evaluated by human judges in terms of clarity, noise presence, closeness to the original and overall audio quality. We also recommended that the PESQ algorithm be used as an objective evaluation metric for generic audio scenarios that are not confined to telecommunication setups. While WaveRNN was unable to produce high quality samples, WaveNet-generated waveforms achieved 5-scale MOSs above 4.0 both in subjective and objective evaluation. These scores are comparable to those presented in the original WaveNet paper. For the mood-conditioned WaveNet we evaluated the results by having listeners guess the specified mood of each generated sample. The experiment was successful as most respondents guessed correctly. Overall, the model proved to be surprisingly creative and capable of incorporating patterns intrinsic to the different moods.

As future work, we suggest further experimentation with emotional conditioning in music synthesis. To circumvent the restrictions imposed by functioning just in the auditory domain, WaveNet could be combined with symbolic models to substitute subjective mood descriptors. A possible arrangement would involve an LSTM model generating mood-conditioned original musical pieces as MIDI and utilizing these as local conditioning for a WaveNet trained in the audio domain on emotional chromas and timbres. A successful attempt at emotional music synthesis, however, would almost certainly necessitate additional research in the psychology and psychoacoustics literature. Deep neural networks have shown great promise for automatic music generation in general, so it would be interesting to try to make them emotive as well.

Appendix A

Questionnaire

All musical samples from experiments can be found here (Google Drive Link)



Υποκειμενική Αξιολόγηση Ποιότητας Μουσικής

** Συνιστάται η χρήση ακουστικών**

Θέλουμε την προσωπική σας γνώμη δεν υπάρχουν σωστές ή λάθος απαντήσεις!

* Απαιτείται

Α ΜΕΡΟΣ: Αξιολόγηση Ανακατασκευης

Σε κάθε ερώτηση δίνονται δύο ηχητικά clips
1) το αυθεντικό μουσικό κομμάτι
2) η ανακατασκευή του ίδιου κομματιού από αναγεννητικό νευρωνικό δίκτυο (Generative Neural Network)

Αξιολογήστε τα ηχητικα clip σε κλίμακα από το 1 ως το 5 (ως προς την ομοιότητα με το πρωτότυπο, την ποιότητα, το θόρυβο που έχει κλπ)

1. original: shorturl.at/hlS69 || synthesis: shorturl.at/bRZ38 *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

2. original: shorturl.at/rvCUZ || synthesis: shorturl.at/bpyN2 *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

3. original: shorturl.at/kzER5 || synthesis: shorturl.at/itAFJ *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

4. original: shorturl.at/rvCUZ || synthesis: shorturl.at/JNR15 *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

5. original: shorturl.at/tAH79 || synthesis: shorturl.at/aixCE *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

6. original: shorturl.at/zBIW1 || synthesis: shorturl.at/tAFOR *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

7. original: shorturl.at/VW138 || synthesis: shorturl.at/behrC *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

8. original: shorturl.at/pqD02 || synthesis: shorturl.at/ipA04 *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

9. original: shorturl.at/VW138 || synthesis: shorturl.at/bdhDE *

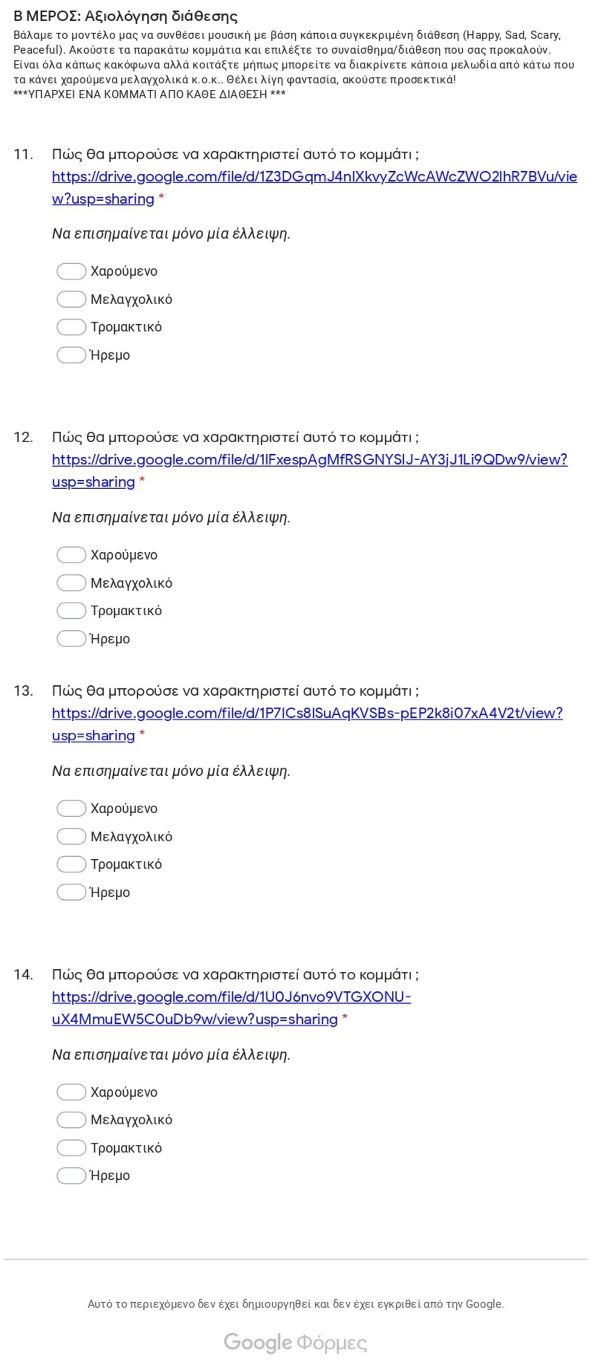
Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό

10. original: shorturl.at/kCW14 || synthesis: shorturl.at/ijCT5 *

Να επισημαίνεται μόνο μία έλλειψη.

	1	2	3	4	5	
πολύ κακό	◯	◯	◯	◯	◯	εξαιρετικό



B ΜΕΡΟΣ: Αξιολόγηση διάθεσης

Βάλαμε το μοντέλο μας να συνθέσει μουσική με βάση κάποια συγκεκριμένη διάθεση (Happy, Sad, Scary, Peaceful). Ακούστε τα παρακάτω κομμάτια και επιλέξτε το συναίσθημα/διάθεση που σας προκαλούν. Είναι όλα κάπως κακόφωνα αλλά κοιτάξτε μήπως μπορείτε να διακρίνετε κάποια μελωδία από κάτω που τα κάνει χαρούμενα μελαγχολικά κ.ο.κ.. Θέλει λίγη φαντασία, ακούστε προσεκτικά!
***ΥΠΑΡΧΕΙ ΕΝΑ ΚΟΜΜΑΤΙ ΑΠΟ ΚΑΘΕ ΔΙΑΘΕΣΗ ***

11. Πώς θα μπορούσε να χαρακτηριστεί αυτό το κομμάτι ; https://drive.google.com/file/d/1Z3DGqmJ4nlXkvyZcWcAWcZWO2lhR7BVu/view?usp=sharing *

    Να επισημαίνεται μόνο μία έλλειψη.

    - Χαρούμενο
    - Μελαγχολικό
    - Τρομακτικό
    - Ήρεμο

12. Πώς θα μπορούσε να χαρακτηριστεί αυτό το κομμάτι ; https://drive.google.com/file/d/1lFxespAgMfRSGNYSlJ-AY3jJ1Li9QDw9/view?usp=sharing *

    Να επισημαίνεται μόνο μία έλλειψη.

    - Χαρούμενο
    - Μελαγχολικό
    - Τρομακτικό
    - Ήρεμο

13. Πώς θα μπορούσε να χαρακτηριστεί αυτό το κομμάτι ; https://drive.google.com/file/d/1P7lCs8lSuAqKVSBs-pEP2k8i07xA4V2t/view?usp=sharing *

    Να επισημαίνεται μόνο μία έλλειψη.

    - Χαρούμενο
    - Μελαγχολικό
    - Τρομακτικό
    - Ήρεμο

14. Πώς θα μπορούσε να χαρακτηριστεί αυτό το κομμάτι ; https://drive.google.com/file/d/1U0J6nvo9VTGXONU-uX4MmuEW5C0uDb9w/view?usp=sharing *

    Να επισημαίνεται μόνο μία έλλειψη.

    - Χαρούμενο
    - Μελαγχολικό
    - Τρομακτικό
    - Ήρεμο

References

[1] Aaron van den Oord et al. "WaveNet: A Generative Model for Raw Audio Based on PixelCNN Architecture". In: *arXiv* (2016).

[2] Nal Kalchbrenner et al. "Efficient neural audio synthesis". In: *35th International Conference on Machine Learning, ICML 2018*. Vol. 6. 2018.

[3] Sander Dieleman, Aäron Van Den Oord, and Karen Simonyan. "The challenge of realistic music generation: Modelling raw audio at scale". In: *Advances in Neural Information Processing Systems*. Vol. 2018-December. 2018.

[4] Sergio Luque. "The Stochastic Synthesis of Iannis Xenakis". In: *Leonardo Music Journal* 19 (2009). ISSN: 0961-1215. DOI: 10.1162/lmj.2009.19.77.

[5] Iannis Xenakis. *Formalized music*. Indiana Univ. Pr., 1972. URL: https://monoskop.org/images/7/74/Xenakis_Iannis_Formalized_Music_Thought_and_Mathematics_in_Composition.pdf.

[6] Alice Baird, Shahin Amiriparian, and Björn Schuller. "Can Deep Generative Audio be Emotional? Towards an Approach for Personalised Emotional Audio Generation". In: *IEEE 21st International Workshop on Multimedia Signal Processing, MMSP 2019*. 2019. DOI: 10.1109/MMSP.2019.8901785.

[7] l. a. hiller jr. and l. m. isaacson. "musical composition with a high-speed digital computer". In: *journal of the audio engineering society* 6.3 (1958), pp. 154–160.

[8] Aymeric Zils and François Pachet. "Musical mosaicing". In: *Digital Audio Effects (DAFx)* (2001).

[9] Diemo Schwarz. *Concatenative sound synthesis: The early years*. 2006. DOI: 10.1080/09298210600696857.

[10] Ian Simon and Sageev Oore. *Performance RNN: Generating Music with Expressive Timing and Dynamics*. https://magenta.tensorflow.org/performance-rnn. Blog. 2017.

[11] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. “DeepBach: A steerable model for bach chorales generation”. In: *34th International Conference on Machine Learning, ICML 2017*. Vol. 3. 2017.

[12] Stuart Geman and Donald Geman. “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6 (1984), pp. 721–741. DOI: 10.1109/TPAMI.1984.4767596.

[13] Soroush Mehri et al. “Samplernn: An unconditional end-to-end neural audio generation model”. In: *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*. 2017.

[14] Kundan Kumar et al. “MelGAN: Generative adversarial networks for conditional waveform synthesis”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019.

[15] Chris Donahue, Julian McAuley, and Miller Puckette. “Synthesizing Audio with Generative Adversarial Networks”. In: 2019.

[16] Jesse Engel et al. “Gansynth: Adversarial neural audio synthesis”. In: 2019.

[20] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*. 2014. DOI: 10.3115/v1/d14-1179.

[22] Takayoshi Yoshimura et al. “Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis”. In: *EUROSPEECH*. 1999.

[24] S. S. Stevens and J. Volkmann. “The Relation of Pitch to Frequency: A Revised Scale”. In: *The American Journal of Psychology* 53 (3 1940). ISSN: 00029556. DOI: 10.2307/1417526.

[27] Aaron van den Oord et al. “Parallel WaveNet: Fast High-Fidelity Speech Synthesis”. In: *35th International Conference on Machine Learning, ICML 2018*. Vol. 9. 2018.

[28] Wei Ping, Kainan Peng, and Jitong Chen. “Clarinet: Parallel wave generation in end-to-end text-to-speech”. In: *7th International Conference on Learning Representations, ICLR 2019*. 2019. arXiv: 1807.07281 [cs.CL].

[29] Aäron Van Den Oord et al. “Conditional image generation with PixelCNN decoders”. In: *Advances in Neural Information Processing Systems*. 2016.

[30] Andrés R. Masegosa et al. “Probabilistic models with deep neural networks”. In: *Entropy* 23 (1 2021). ISSN: 10994300. DOI: 10.3390/e23010117.

[31] Karol Gregor et al. “Deep autoregressive networks”. In: *31st International Conference on Machine Learning, ICML 2014*. Vol. 4. 2014.

[32] Xin Zhang and Jiali You. “A Gated Dilated Causal Convolution Based Encoder-Decoder for Network Traffic Forecasting”. In: *IEEE Access* 8 (2020). ISSN: 21693536. DOI: 10.1109/ACCESS.2019.2963449.

[33] Fisher Yu and Vladlen Koltun. “Multi-scale context aggregation by dilated convolutions”. In: *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. 2016.

[34] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2016-December. 2016. DOI: 10.1109/CVPR.2016.90.

[35] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 07-12-June-2015. 2015. DOI: 10.1109/CVPR.2015.7298594.

[36] Julius O. Smith. *Introduction to Digital Filters with Audio Applications*. http://www.w3k.org/books/: W3K Publishing, 2007. ISBN: 978-0-9745607-1-7.

[37] Tom Le Paine et al. *Fast Wavenet Generation Algorithm*. 2016. arXiv: 1611.09482 [cs.SD].

[38] Junyoung Chung et al. *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. 2014. arXiv: 1412.3555 [cs.NE].

[40] Sharan Narang et al. “Exploring sparsity in recurrent neural networks”. In: *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*. 2017.

[41] Curtis Hawthorne et al. “Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset”. In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=r1lYRjC9F7.

[42] Tuomas Eerola. *Music and emotion dataset (Primary Musical Cues)*. Version V1. 2016. DOI: 10.7910/DVN/IFOBRN. URL: https://doi.org/10.7910/DVN/IFOBRN.

[43] K. R. Tan, M. L. Villarino, and C. Maderazo. “Automatic music mood recognition using Russell’s twodimensional valence-arousal space from audio and lyrical data as classified using SVM and Naïve Bayes”. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 482. 2019. DOI: 10.1088/1757-899X/482/1/012019.

[44] Liang Chih Yu et al. “Predicting valence-arousal ratings of words using a weighted graph method”. In: *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference*. Vol. 2. 2015. DOI: 10.3115/v1/p15-2129.

[48] Nagaraj Adiga, Vassilis Tsiaras, and Yannis Stylianou. “ON the use of wavenet as a statistical vocoder”. In: *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. Vol. 2018-April. 2018. DOI: 10.1109/ICASSP.2018.8462393.

[49] Brian McFee et al. “librosa: Audio and Music Signal Analysis in Python”. In: *Proceedings of the 14th Python in Science Conference*. 2015. DOI: 10.25080/majora-7b98e3ed-003.

[50] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks”. In: *30th International Conference on Machine Learning, ICML 2013*. 2013.

[51] Abdullah I. Al-Shoshan. “Speech and Music Classification and Separation: A Review”. In: *Journal of King Saud University - Engineering Sciences* 19 (1 2006). ISSN: 10183639. DOI: 10.1016/S1018-3639(18)30850-X.

[54] A.W. Rix et al. “Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs”. In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. Vol. 2. 2001, 749–752 vol.2. DOI: 10.1109/ICASSP.2001.941023.

External Links

[17] *Google Assistant*. URL: `https://assistant.google.com/`.

[18] *Basic RNN architectures (Stanford cs231 lecture)*. URL: `http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture10.pdf`.

[23] *Using WaveNet technology to reunite speech-impaired users with their original voices*. URL: `https://deepmind.com/blog/article/Using-WaveNet-technology-to-reunite-speech-impaired-users-with-their-original-voices`.

[25] Haytham M. Fayek. *Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between*. 2016. URL: `https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html`.

[26] *DeepMind.com Blog*. URL: `https://deepmind.com/blog/article/wavenet-generative-model-raw-audio`.

[39] *WaveRNN Structure Commentary - Monthly Hacker Blog*. URL: `https://www.monthly-hack.com/entry/2018/02/26/211248`.

[45] *PyCharm IDE*. URL: `https://www.jetbrains.com/pycharm/`.

[46] *Anaconda*. URL: `https://www.anaconda.com/`.

[47] *HPC ARIS GRNET*. URL: `https://hpc.grnet.gr/supercomputer/`.

[52] *A TensorFlow implementation of DeepMind's WaveNet paper*. URL: `https://github.com/ibab/tensorflow-wavenet`.

[53] *Google Forms: Free Online Surveys for Personal Use*. URL: `https://www.google.com/forms/about/`.

[55] *A Data Scientist's Approach to Visual Audio Comparison*. URL: `https://towardsdatascience.com/a-data-scientists-approach-to-visual-audio-comparison-fa15a5d3dcef`.