# Music Generation using WaveNet Architecture

Karthik Nagarajan Sundar

Chalmers University of Technology
Electrical Engineering Department

CHALMERS UNIVERSITY OF TECHNOLOGY

## Introduction

Main objective: Prediction of subsequent note/chord for a certain period of time to generate a track of music using Deep learning.

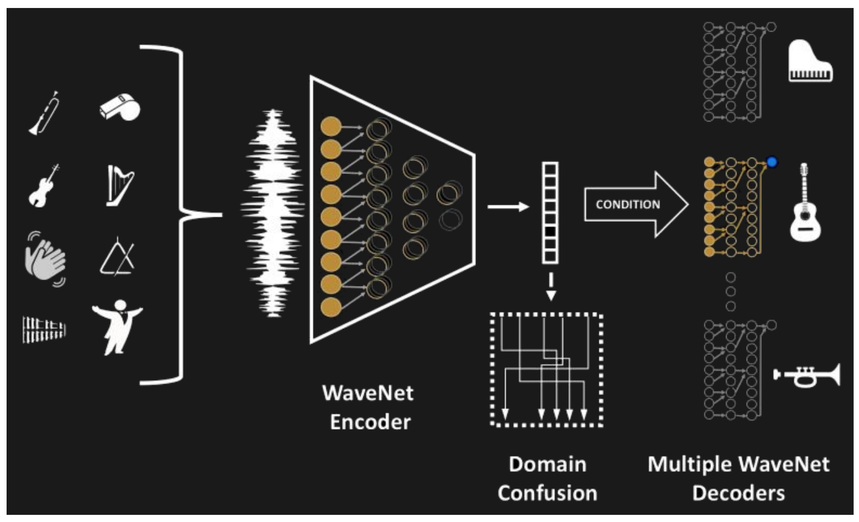


Figure 1: Music generation - basic idea

Contents:

- Music dataset(midi format)
  - Reading a midi file
  - Visualizing midi file in terms of Piano rolls.
- WaveNet architecture
  - What is Wavenet
  - Model summary
- Results and Evaluation

## Music Data set

- Self-compiled famous metal genre music (Total:65)
- Format used for processing: Musical Instrument Digital Interface (MIDI)
- Split ratio: 80/20 Training and Validation data-sets
- Reading a midi file
- Notes: sound produced by a single key
- Chord: sound produced by 2 or more keys simultaneously

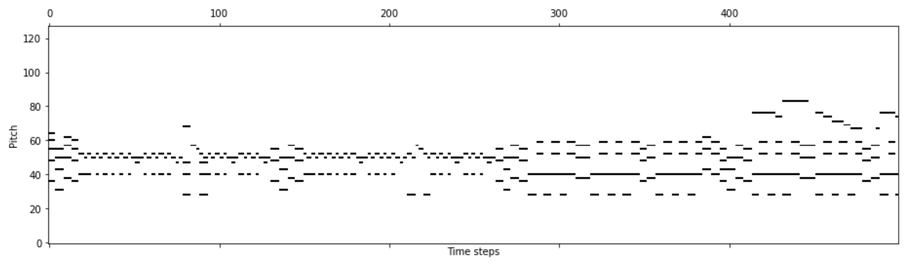


Figure 2: Visualizing midi in terms of piano rolls

## WaveNet

- Deep Learning-based generative model developed by DeepMind(Google)
- Input: a sequence of nodes and chords
- Output: new predicted samples from the sequence
- uses Dilated Convolution layer instead of Causal Convolution layers
- Major difference: all the previous inputs can have significant contribution on predicted notes

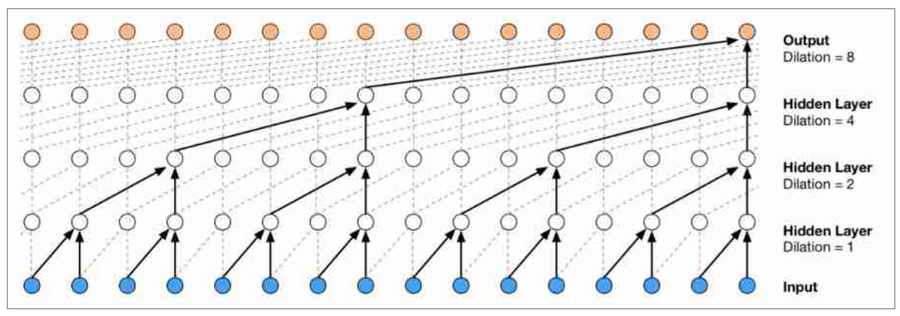


Figure 3: Diluted Convolution Layer

- Better than RNNs because
  - better long term memory
  - No vanishing gradients problem
  - Hardware friendly
- Better than LSTMs because
  - longer memory for sequences over 10k samples.

## Model Summary

- 3 1-dimensional Convolution layers
- Dropout layers: avoid over fitting
- Max-pooling layers: Down sample

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 32, 100)           23300
_________________________________________________________________
conv1d (Conv1D)              (None, 32, 512)           154112
_________________________________________________________________
dropout (Dropout)            (None, 32, 512)           0
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 16, 512)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 16, 128)           196736
_________________________________________________________________
dropout_1 (Dropout)          (None, 16, 128)           0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 8, 128)            0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 8, 16)             6160
_________________________________________________________________
dropout_2 (Dropout)          (None, 8, 16)             0
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 4, 16)             0
_________________________________________________________________
global_max_pooling1d (Global (None, 16)                0
_________________________________________________________________
dense (Dense)                (None, 256)               4352
_________________________________________________________________
dense_1 (Dense)              (None, 235)               60395
=================================================================
Total params: 445,055
Trainable params: 445,055
Non-trainable params: 0
_________________________________________________________________
```

Figure 4: Model Summary

## Results and Evaluation

- Parameters: 75 epochs run and batch size of 128
- Number of unique and random notes generated: 17 for a sequence length of 30 notes/chords
- User feedback: 5 out of 7 volunteers liked the track

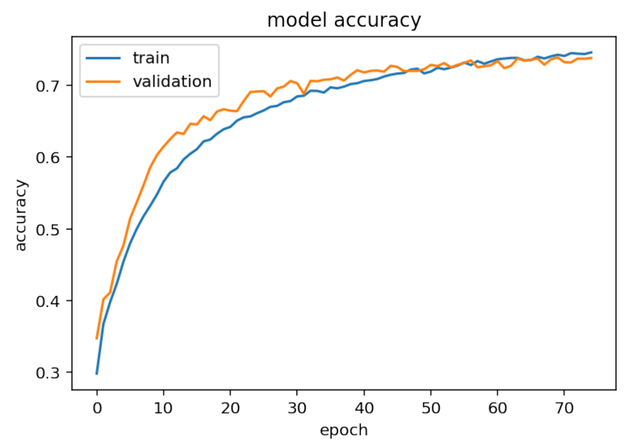


Figure 5: Accuracy

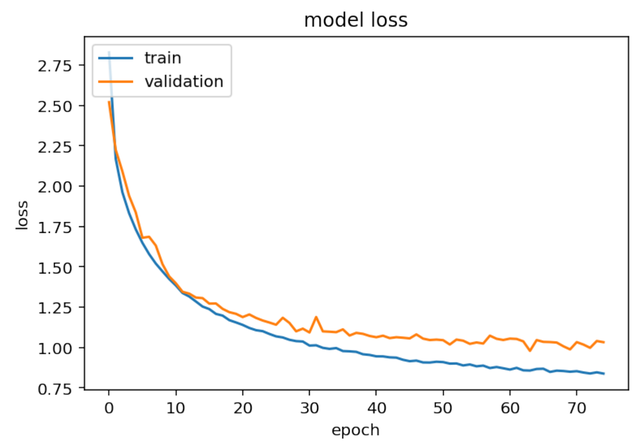


Figure 6: Loss

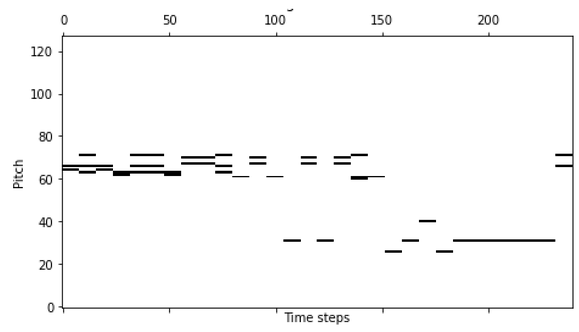


Figure 7: Visualizing output in terms of Piano rolls

## References

[1] Aravind Pai, ”Automatic music generation”, 2020, https://www.analyticsvidhya.com/blog/2020/01/how-to-perform-automatic-music-generation/

[2] J. Berntsson and A. Tonderski, ”Simple adversarial music generation”, https://github.com/jcberntsson/musicgeneration-project, 2018.



sundark@student.chalmers.se