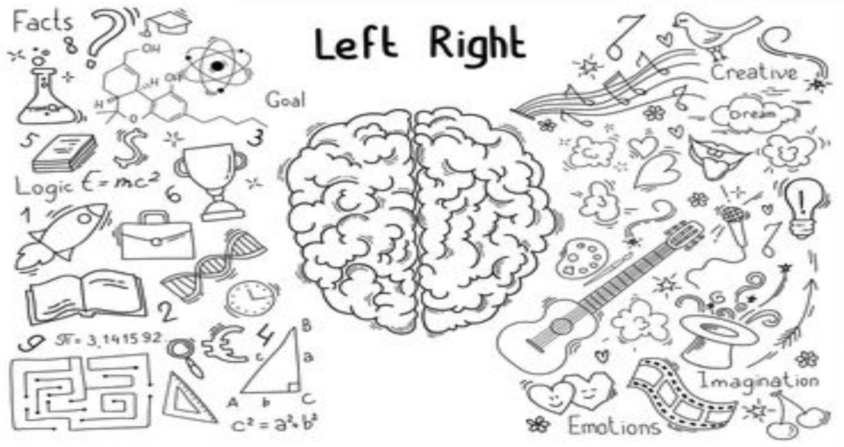
UC San Diego

MUSIC GENERATION USING NEURAL NETWORKS

Lynsey Johnson
Sushmitha Kudari
Avinash Mallavarapu

BACKGROUND

- In general, society holds a longstanding, strong distinction between art and science.
    - Art is subjective while science is objective
    - Art is an expression of knowledge while science is the practice of acquiring knowledge



- In this project, we aimed to explore an application of machine learning to explore the fusion of art and science.

- Using the analogous model of machine learning to the human brain, we used machine learning to study music and compose a song of its own, testing the limits of a machine’s ability to learn right hemisphere attributes such as style and creativity.

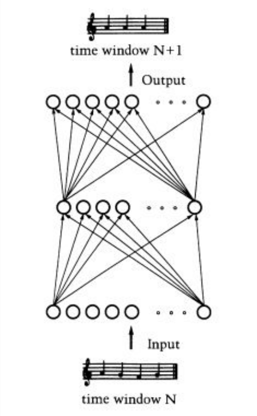
BACKGROUND

- Music composition from an algorithmic approach is not new
  - Greek-French composer and music theorist, Iannis Xenakis, discussed using Markov chains and transitional matrices that define the probabilities of certain notes being produced while composing his electroacoustic work *Analogique B* in 1958 [1]
  - Dr. David Cope of University of California, Santa Cruz describes in Experiments in Musical Intelligence writing a composition algorithm in 1981 as a method for curing a case of writer’s block [2]
- Automation of some music components could provide rewarding benefits
  - Removes the need for the painstaking work of composing snippets of music
  - Serves as an optimizer in both memory and speed for small movie scores, video games and other sound effects
  - Especially useful for any low budget production requiring original audio components

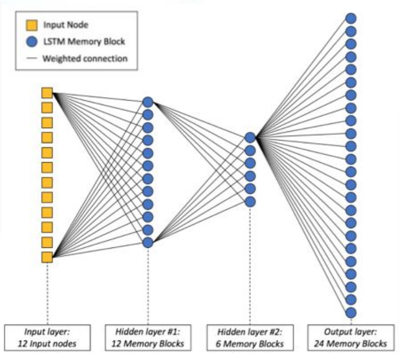
LITERATURE SURVEY

- Peter M. Todd was the first to use a regressive neural network (RNN) to generate music sequentially in 1989 [3]
  - He used a windowed method of processing successive time-periods of melodies similar to speech applications at the time

- Douglas Eck and Jurgen Schmidhuber built on the recurrent network model with Long short-term memory (LSTM) structure in 2002 [4]
  - LSTM networks utilize a cell state that windows through processing gates that measures recognizes similarities throughout different segments of a large piece of data
  - LSTM models are designed to learn long term dependencies and patterns making it especially useful to composition applications



Todd's windowed RNN from [3]


Input Node
LSTM Memory Block
Weighted connection
Input layer: 12 Input nodes
Hidden layer #1: 12 Memory Blocks
Hidden layer #2: 6 Memory Blocks
Output layer: 24 Memory Blocks

RNN and LSTM architecture similar to that of Eck and Schmidhuber in [4]

LITERATURE SURVEY

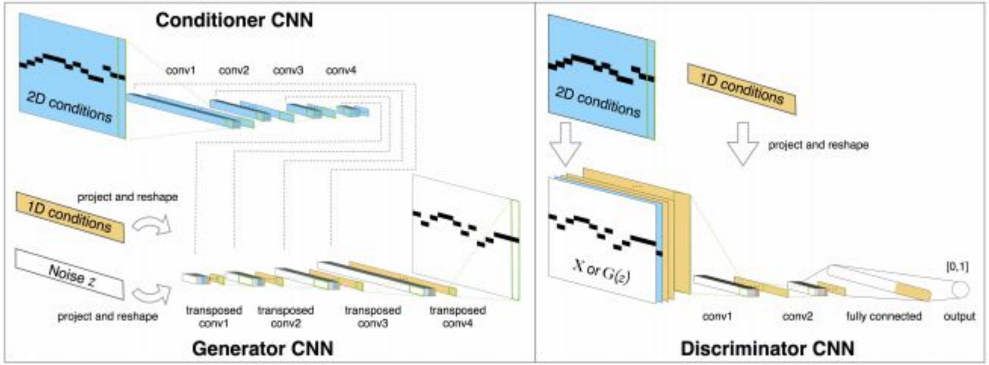


Figure 1. System diagram of the proposed MidiNet model for symbolic-domain music generation.

MidiNet architecture designed by Yang et all in [6]

- The most popular contemporary model is the deep convolutional neural network WaveNet developed by AI researchers at DeepMind introduced in 2016 [5]
- Yang et all developed MidiNet [6], a generative adversarial network based on WaveNet with an added discriminator CNN in addition to the generator CNN designed by WaveNet that learns the distribution of melodies to generate music

LITERATURE SURVEY

- Manan Oza et all refine the GAN model for music generation using a method they developed called progressive training[7]
  - A progressively trained GAN is unique to that of MidiNet in that it adds additional convolutional layers to training phases to improve periodicity and reduce note fragmentation
- HackerPoet built a neural composer that is trained on a MIDI dataset of video game theme songs [8]
  - The neural composer is based on a neural variational autoencoder and decoder model. Each song measure is converted to a feature vector, which is fed to an autoencoder that creates a global feature vector for the entire song, and is finally run through a decoder that splits the global feature vector back to independent measures.

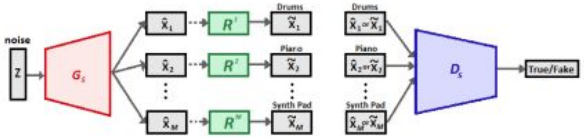
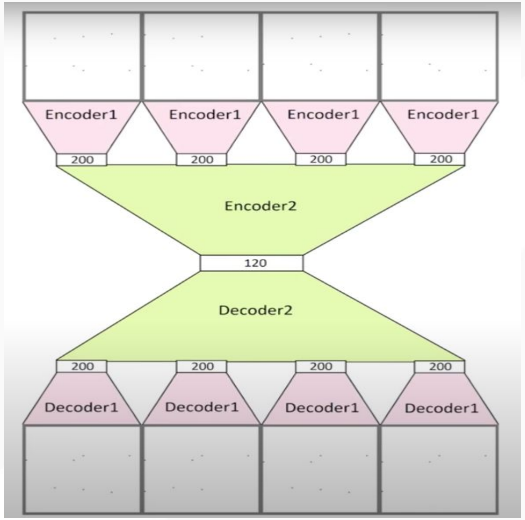


Fig. 2 Our proposed architecture. The shared generator network G_s and shared discriminator network D_s are trained progressively with a refiner network having variable tensor size in between.

GAN with progressive training architecture in [7]



Neural Composer architecture in [8]

DETAILS ON THE DATASET

- Lakh MIDI Dataset v0.1
- Generated and compiled by Colin Raffel in “Learning-Based Methods for Comparing Sequences, with Application to Audio-to-MIDI Alignment and Matching”, 2016 [9]
- Raffel generated the dataset by developing a series of learning-based methods to compare, identify and match entries in the Million Song Dataset and converted the audio files to MIDI format
- Researchers at the Music and AI Lab derived labels for the genres of audio files contained in the Lakh MIDI Dataset based on their mapping to the Mission Song Dataset and converted them to pianoroll format [10]
- The pianoroll data divides the MIDI file into five different separate tracks including drums, piano, guitar, bass and strings
- Our group wrote a script to pull all the labels identified as Jazz from the dataset, removing duplicates from different sources (lastfm, tagtraum, etc)
- We used the Jazz MIDI files for the neural composer model and the Jazz pianoroll data for the Progressive MidiNet model

MODELS

- We will be implementing two models to compare performance and results
- Each model trains on the Jazz songs extracted from the Lakh MIDI Dataset
- Each model is expected to generate music similar to the Jazz samples in which it trains on
- Using Jazz as the genre to train on was a design choice based on the models we are using
- Jazz has minimal structure and note predictability which is ideal for non-LSTM models

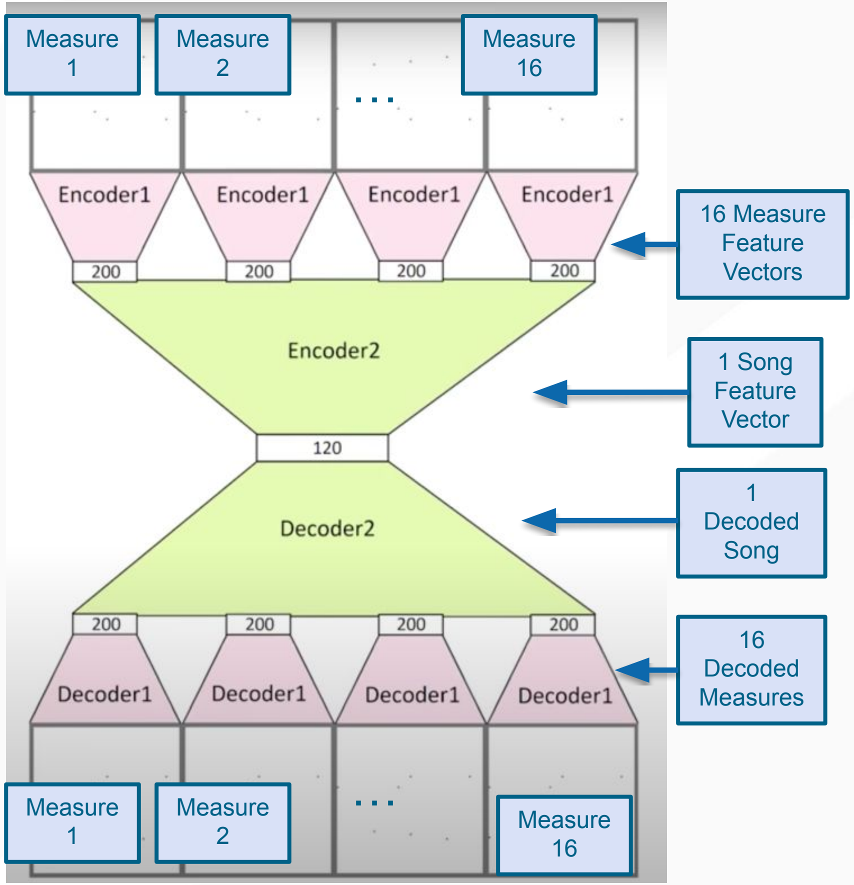
NEURAL COMPOSER MODEL

Model Details

- Decomposes single note samples into singular time steps
- Adds 16 measures of each song to the input as a third dimension
- Encodes each input measure into a feature vector
- Passes the feature vector through another encoder to generate a total feature vector for all 16 measures of the input song
- Run total feature vector through a decoder for the entire song
- Run decoded song through a second decoder to to convert back to independent measures
- The neural composure is essentially two identical encoders with opposite feed directions

Processing Details

- Each measure is of dimension 96x96.
- All 16 measures (after reshaping)are passed through 3 fully connected layers to obtain the measure feature vectors of size 200
- Encoder 2 combines all the 16 measure feature vectors and using 2 linear layers it calculates the latent representation of the song i.e a 120 dimensional vector.
- To generate a new song, we create a noise vector having a size of 120 and pass it through the decoder layer to obtain a tensor of size 16x96x96.



Loss function and Hyperparameters

The loss function is a combination of reconstruction loss and latent (VAE) loss:-

- Reconstruction loss - We use L2 loss to compare the generated song and input song
- Latent loss - Used KL divergence loss which penalizes the model if the latent vector (dim = 120) doesn’t come from a gaussian distribution with zero mean and identity covariance matrix.

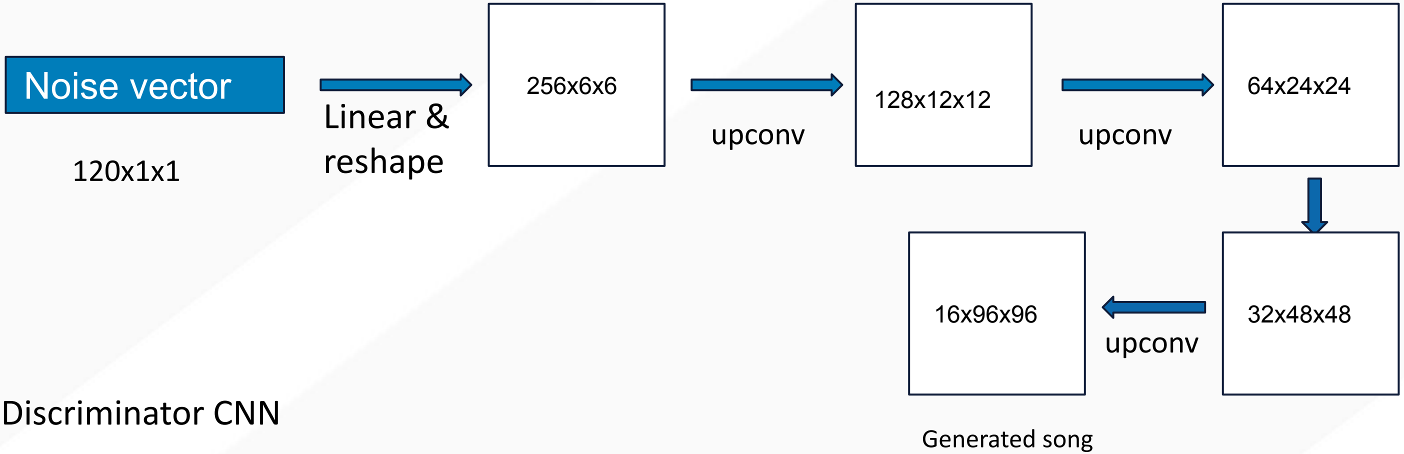
- Number of epochs - 120
- Learning rate - 0.001
- Batch size - 350
- VAE loss weights : $ƛ_1 = 0.02$, $ƛ_2 = 0.1$ (VAE loss = $ƛ_1$ *reconstruction_loss + $ƛ_2$ *latent_loss)

Convolutional GAN architecture

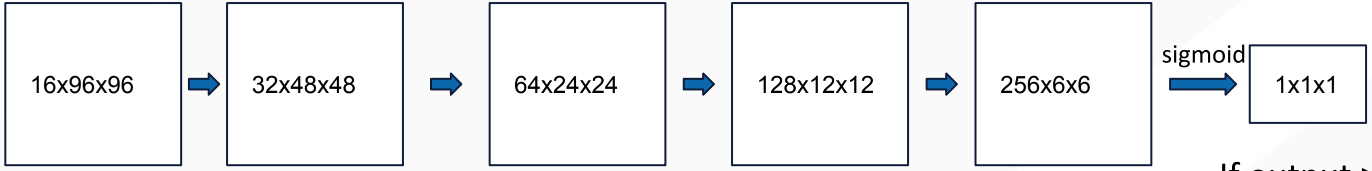
GAN Model to be integrated into Neural Composer

- Will replace Variational Autoencoder network

Generator network



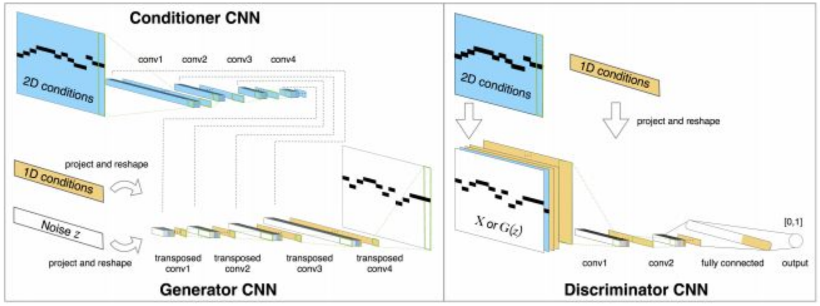
Discriminator CNN

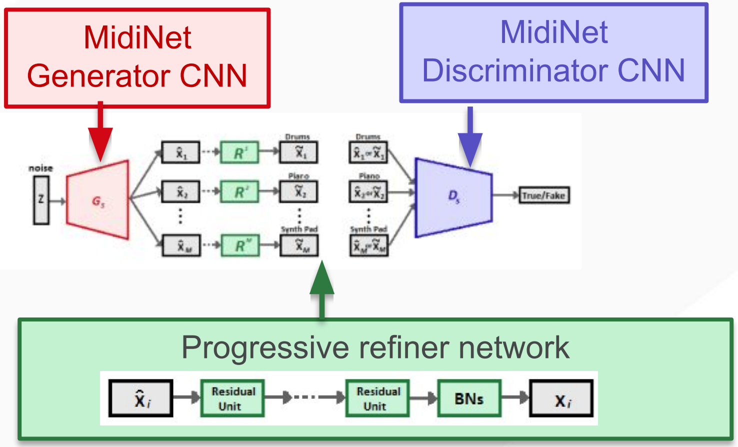


Each convolutional (strided) layer is followed by BN & ReLu

If output > 0.5 it’s a real song,
else it’s system generated

MidiNet MODEL





MidiNet Generator CNN

MidiNet Discriminator CNN

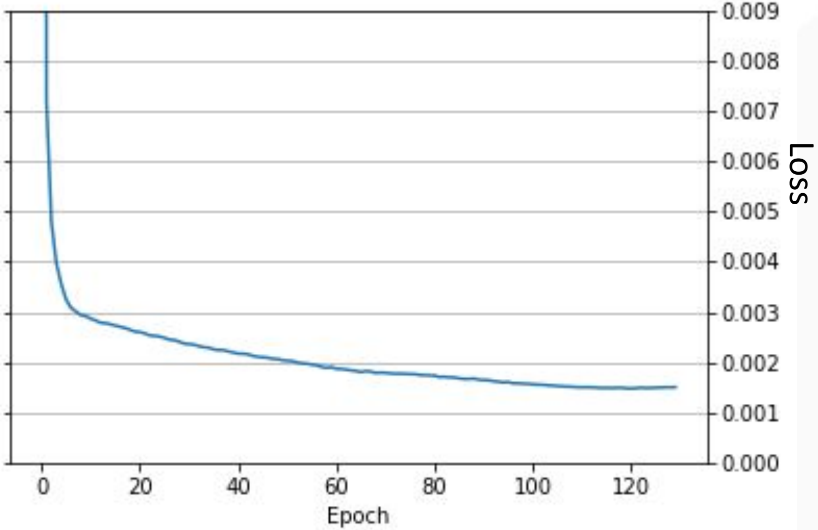
Progressive refiner network

MidiNet [6] with Oza's Progressive Training [7] feature

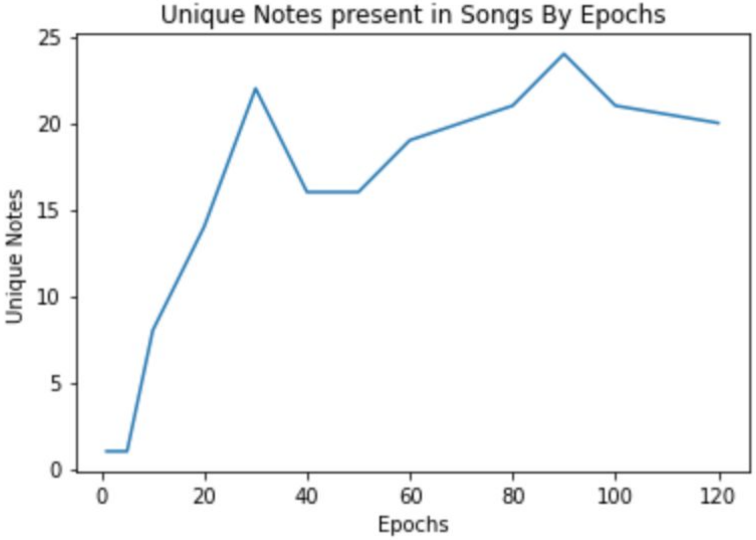
- MidiNet is a DCGAN that generates music
- MidiNet is comprised of two CNNs
    i. A generator CNN
        - Creates 2D score-like representations in the goal of appearing to be from real MIDI files
    ii. A discriminator CNN
        - Takes a 2D scores as input and predicts whether it is from a real or generated MIDI file
- The discriminator informs the generator how to produce MIDI files to appear real
- The generator improves the prediction of the discriminator by iteratively providing more challenging scores to differentiate
- We plan to adapt Manan Ozas's progressive training technique to MidiNet that require previous layers to converge prior to moving on to the next layers

RESULTS/OBSERVATIONS

Training Loss



Proof of Increasing Complexity over Epochs:



PENDING ITEMS

- Test the gan architecture for generating songs and compare its performance with the VAE architecture
- Implement progressive training procedure in the GAN architecture to further improve the stability of training
- Train for 250 + epochs

REFERENCES

1. https://monoskop.org/images/7/74/Xenakis_Iannis_Formalized_Music_Thought_and_Mathematics_in_Composition.pdf
2. http://artsites.ucsc.edu/faculty/cope/experiments.htm
3. https://abcwest.sitehost.iu.edu/pmwiki/pdf/todd.compmusic.1989.pdf
4. https://people.idsia.ch/~juergen/blues/IDSIA-07-02.pdf
5. https://deepmind.com/blog/article/wavenet-generative-model-raw-audio
6. https://arxiv.org/pdf/1703.10847.pdf
7. https://arxiv.org/pdf/1903.04722.pdf
8. https://www.youtube.com/watch?v=UWxfnNXlVy8
9. https://colinraffel.com/publications/thesis.pdf
10. https://salu133445.github.io/lakh-pianoroll-dataset/dataset