# Algorithmic Music Composition

Kai-Chieh Huang; Quinlan Jung; Jennifer Lu
Department of Computer Science, Stanford University, Stanford, CA

## Abstract / Description

Throughout history, music has for the most part, been composed by humans. Unfortunately, human labor is costly, and composing a piece of music can be a lengthy process. In this project, we created a model that can reliably generate a piece of music and lower the cost of creating music by removing the bottleneck of human labor. People who wish to listen to new music or advertising companies who want background music will be able to do so at an extremely low cost. Furthermore, with an automatic music generator, we can easily generate diverse music material with any duration in a game situation where the player can stay in a particular scene for an undetermined amount of time without looping the same background music again and again.

## Baseline

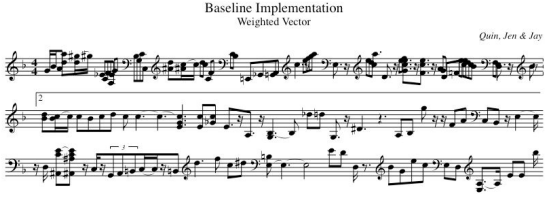
**Method:** In each abc file, a 'token' is a small set of notes delimited by whitespaces. A token usually consists of a single note, a set of notes connected by a beam, or a rest. Our baseline consists of the following steps:

**1.** Generate a weighted probability vector of each token and its frequency.

**2.** Output a piece consisting of 100 tokens. To generate a single token, we use our vector to create weighted probabilities in order to choose a random token. We repeat this process until we have chosen 100 tokens.

**3.** A piece's key and time signature is also chosen using a weighted probability vector generated at training time.

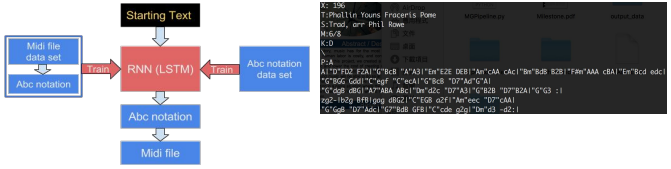
**Discussion:** While the generated music pieces are syntactically correct, they sound very different from the music seen during training. There is a more contemporary/aharmonic theme present in our baseline music, a completely different type of sound than was heard in the work J.S. Bach ever composed.




Caption 1: Musical notation of the baseline generated Classical song

## The Pipeline

Our music generator model is trained and sampled using the following pipeline. We converted midi files into abc notated text files to use as our input. After training our model, it composes music in abc notation, which we convert back to midi format for listening.



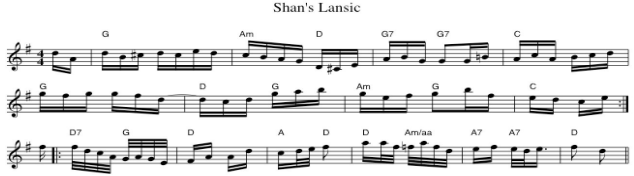

Caption 2: (Left) Pipeline of our music generator model
(Right) abc notation of raw output of the randomly generated song

## RNN (LSTM)

Our implementation is inspired by Andrej Karpathy's char-rnn, which was originally used to generate Shakespeare-like prose. We concatenated all our abc notated training songs into a large text file, then we train the RNN model on this input by feeding it into our network one character at a time. To obtain a piece of music, sample our neural network, seeding with the beginning header of an ABC file, 'X'. The output of the neural network (with enough samples) is a stream of whole music pieces, also in ABC text format.

**Implementation Details:**
Library: Tensorflow
Network: 2 128-unit LSTM cells
Activation Function: tanh
Learning rate: 0.002
Adaptive Learning Method: RMSProp
RMSProp decay rate: 0.97

These parameters were taken from Manuel Araoz's neural network that composed folk music. We ran our neural network on Amazon's EC2 g2.2xlarge instance, a machine optimized for graphics-intensive applications with 1 NVIDIA GPU.




Caption 3: Musical notation of raw output of the randomly generated song

## Oracle

We use Google Magenta as our oracle to represent the gold standard. Developed by Google Brain, Magenta is a deep learning project that creates compelling art and music. We leverage Magenta's recurrent neural network and train it with the files we used on our models. Then, we use their MIDI interface to generate Magenta's stream of music.

## Error Analysis

LISTEN FOR YOURSELF!

The final evaluation metrics of our project will be based on conducting a double blind A/B test. In each trial of the test, the listener will be presented two midi files: music from our model, and music from human. They will then decide which piece was composed by a human. If our test subjects can only label the music composed by a human correctly ~50% of the time, we can conclude that the music generated from our generator is indistinguishable from a human and is successful.

## Future Work

**Main challenge: Current LSTM doesn't learn the relationship between melodies and chords.**

One possible explanation is that in the abc notation, it represents the entire melody line of the song first, with the chords that accompany the melody on the next line. Since the RNN model can only relate texts that are close to each other in a certain range, the RNN model can lose track of the relationship of the melody and chord that is supposed to be played in the same bar if the text file represents them a line apart.

**Solution: Biaxial RNN**
The "biaxial RNN" is able to deal with both time and creating nice chords by using two axes along with a pseudo-axis for the direction-of-computation. This structure allows for patterns in both time and note space without giving up invariance.