Electronic Accompaniment Generation

ECE-4563 Intro to Machine Learning Final Project

Matthew Avallone and Anish Malhotra

The Idea:

The goal of this project is given a melody for a song, generate an accompaniment for it. The approach is to phrase this as a machine translation problem, where the melody gets passed through an encoder-decoder LSTM neural network that encodes an input matrix of notes and decodes an output matrix of notes.

The Implementation:

The implementation is going to take place in 3 stages:

The first stage is preprocessing the data to be passed into the neural network. This means reading in a corpus of MIDI files and classifying the instrument tracks in each song into 5-6 main categories. This will create several t x n matrices, where t is the tick event in the song and n is an array of notes over eight octaves (Han Qi et. al., 2016). Each note is represented as a bit (0 for noteOff and 1 for noteOn), so for eight octaves, this sets n = 102 notes. From here, we are using a word embedding algorithm to efficiently encode our data. Each row of the matrix is very sparse (only a handful of notes on at a given tick) and has clear relationships to nearby rows (chords, harmonies, etc.). This project uses major/minor chords for the target vector. The result in the end should be an embedding for the melody matrix, as well as embeddings for each accompaniment instrument class.

The second stage is training a neural network for each of the instrument classes created in the preprocessing stage. Each network will be trained on melody data and a specific accompaniment instrument class (strings, woodwinds, drums etc.). Based on research, the size of similar networks have 50 encoder neurons and 50 decoder neurons (not sure if this is too large/small). The loss function will be binary cross-entropy, where we predict whether a note is on or off at a given tick. The networks will be implemented using Keras and trained on Google Cloud.

The third stage is post processing on the output. This means taking the output matrix for each instrument class and decoding it back into a midi file to be played. We haven't exactly figured out the details for how to do this yet, but I think there is code out there for doing midi conversions.

Current Progress:

Currently, we are in the preprocessing stage of this project. We are developing a way of encoding our data in a more efficient way, using techniques similar to what you'd find in NLP or NMT. The way the inputs are currently structured, there is on the order of tens of millions of parameters

feeding into each neural network (e.g. 500 songs x 5 sequences/song x 500 ticks/sequence x 102 notes/tick). We believe word embeddings can be a useful tool for achieving this, given the similarities in structure between words in sentences and notes in ticks. With a pre-trained word embedding, we can downsample the training data to a baseline level that still captures the basic characteristics of the song and still expect good performance by the neural networks (Qi et. al., 2018).

Future Work:

After the Fall 2018 semester, we plan to pursue this project further as our senior design project. A future experiment we are considering during the preprocessing step is to first translate the long bit strings of notes per tick from the MIDI files into base 36 numbers. This is an attempt to shorten the length of the arrays from 102 to 20 in order to reduce the dimensions of the input data (and therefore help reduce the number of parameters). The could possibly negatively affect minimizing the loss function, as it would now become categorical cross-entropy (more possible choices for prediction → more complex). Next semester we will focus on setting up individual Encoder-Decoder LSTM models for each instrument class and training on our preprocessed input melody and accompaniment data. Once the models are trained, we will shift focus to post processing the output back into MIDI files. This may rely on preexisting code that does MIDI file conversions for us, otherwise we will implement this ourselves. Given the unique nature of the project and approach, a lot of information specific to the scope of this problem is limited, forcing us to experiment with it ourselves. By using electronic dance music (EDM), we have the flexibility of allowing the music to sound repetitive (as it usually does in EDM), while also the freedom to get creative with new rhythms and note patterns suitable for less structured EDM.

References:

Brownlee, Jason. “How to Develop a Neural Machine Translation System from Scratch.” Machine Learning Mastery, 10 Jan. 2018, Available at: http://machinelearningmastery.com/develop-neural-machine-translation-system-keras/.

Olah, Christopher. “Understanding LSTM Networks.” colah’s blog, 27 Aug. 2015, Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Qi, Ye, et al. “When and Why Are Pre-Trained Word Embeddings Useful for Neural Machine Translation?” Proceedings of NAACL-HLT, 2018, pp. 529–535., doi:10.18653/v1/n18-2084.

Zhu, Xuan. and Qi, Han. “Automatic Accompaniment Generation with Seq2Seq”. Cero vale todo, 2016. Available at: http://qihqi.github.io/machine/learning/music-generation-using-rnn/.