Imitative Classical Music*

APS360: Project Proposal

June 30, 2019

Introduction

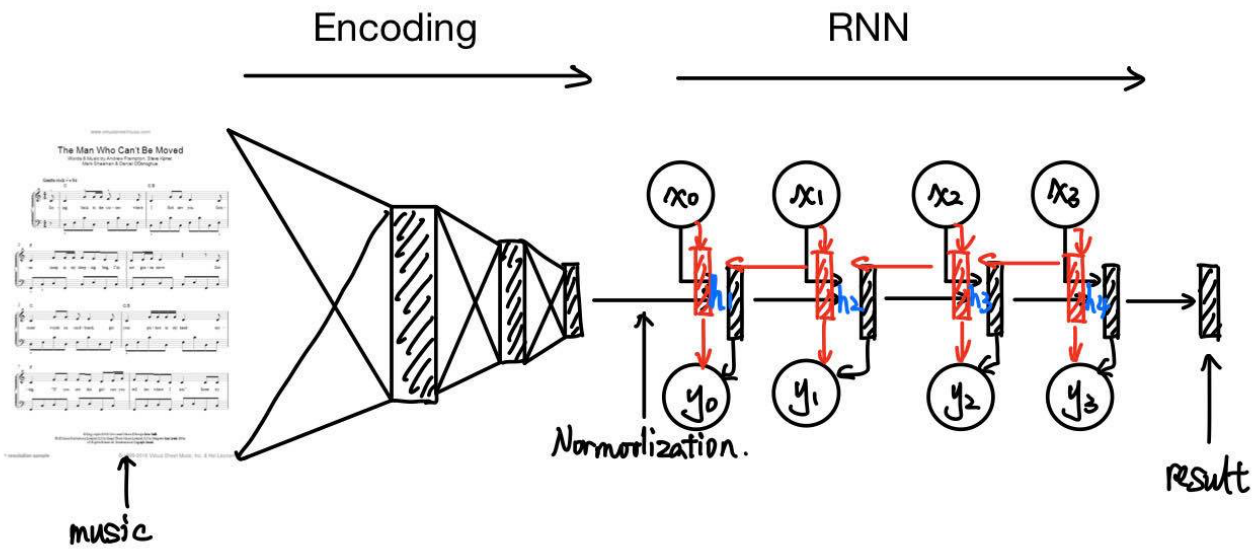
This project will investigate how musical creativity may be artificially reproduced via neural networks and signal processing algorithms. Specifically, the model's objective is to mimic the style of a certain classical music composer given a prompt, i.e., audio snippets that precede and/or succeed the blank segment. Time permitting, the complexity of the model will be circumspectly increased by accounting in finer musical niceties and by incrementally lengthening the predictive segments.

Machine learning is well-suited for this task, because the entirety of classical music theory is clearly too complex to codify into a universal algorithm. Further, emulating faux creativity is still useful for revealing insights as to the challenges involved in mathematically modelling the creative mind, and hence the frailties and deficiencies in current machine learning paradigms.

Finally, this type of "imitative composition" was demonstrated to be a valid strategy as seen in 16th-18th century parody masses such as Bach's *Weihnachtsoratorium*. Hence, ontological particulars aside, imitation may after all be a relevant facet from which creativity emerges.

*Word count: 1102. Penalty: 0%.

Illustration



- Encoding: c.f. Data Processing for details
- RNN: c.f. Architecture for details
- x_i: vector representation for audio input
- y_i: vector representation for audio output
- h_i: neural network layers

Background & Related Work

A rich profusion of papers on NN-based music generation have been published in recent years, many of which overlap in content. In this section we briefly introduce two high-impact works.

Modeling Temporal Dependencies in High-Dimensional Sequences (2012)

This influential paper Bengio et al. [1] investigates how polyphonic music may be represented in "piano roll" representation while preserving musical richness. Further, it introduces the RNN-RBM (restricted Boltzmann machine) model for extracting temporal patterns from high-dimensional sequential data.

BachBot (2016)

The BachBot [2] synthesizes harmonized chorales that are virtually indistinguishable from Bach's original compositions. The BachBot draws ideas from music theory and employs a probabilistic sequence model (LSTM-RNN). An online demo is available at `https://bachbot.com`.

Data Processing

The model will first be trained with MIDI files sourced from websites such as:

- `http://www.piano-midi.de`
- `http://musedata.org`
- `http://classicalmidiresource.com`
- `https://www.midiworld.com`

The downloaded MIDI files will be sorted by composers and normalized for one or more of candidate factors such as tempo, beat resolution, and volume. This will be further made concrete with additional research & planning as outlined in the Project Plan.

The MIDI files are then represented as an internal vector format. Previous works such as [3] used "piano roll" representation, in which each MIDI message is encoded into an independent token.

Time permitting, we will investigate downloading publicly licensed audio files from archive.org and youtube.com, using the popular command-line tool `youtube-dl`, and using the Fourier transformation as an encoding scheme to allow working with arbitrary audio files. Additional sources for raw audio files are as follows:

- `https://homes.cs.washington.edu/~thickstn/musicnet.html`

Architecture

The model will primarily employ the long short-term memory (LSTM) architecture, as satisfactory results were demonstrated with LSTM RNNs in previous related works. Using an RNN makes sense, because melodies by their very construction require harmonic congruence in conjunction with temporal proximity. The normalized, tokenized, and temporally encoded MIDI data are fed into the LSTM, as outlined in the previous section. Initially, the model will be making

unidirectional predictions in which some subsequent blank segments or notes are generated based on the temporally preceding data. After this is implemented, alternative configurations that incorporate succeeding data will be explored.

The performance will be measured by layman-judged "passability," i.e., whether the predicted sequence of notes sounds more natural compared to the baseline. For efficient division of labour, we will concurrently investigate and experiment with alternative CNN-based architectures. This warrants further research, so the details are yet to be finalized.

Baseline Model

The primary baseline model will be a rudimentary autoencoder-based predictor in which one or several (i.e., a bar) missing notes are reconstructed from a lower dimensional representation. A predetermined number of bars will be fed into the network for training, which will be a hyperparameter determined experimentally. We presume that this suffices for capturing basic musical patterns at a diminished accuracy.

Ethical Considerations

- *Copyright and ownership of training data and generated content*. In the United States and Canada, the use case largely falls under the fair use (or fair dealing) doctrine, hence legally inculpable. Thus the problem pertains to ethics than lawfulness, and one should aim to err on the side of ethical transparency when using the model. For instance, the model may be used for reconstructing portions of damaged or lost recordings, but could also be used to mass-produce infringing music that exploits the legal ambiguity for profit.
- *Superannuation of human-composed music*. While current state of the art still seems distant from inventing a musical style that matches the precision and facility of human composers', the edgy hypothetical question remains—what if AI could match or exceed human creativity? However vapid the hypothesis may seem, unimpeded numerical abstraction of human creativity may make life unfulfilling for composers and listeners alike.

Project Plan

Communication

- We will be meeting every Saturday from 12:00 to 16:00 on-campus or via online video calls to sort out any meta-issues that came up during the week and discuss everyone's progress.
- The TA meeting is yet to be scheduled, but regular communication will be maintained via the appropriate channel as per TA availability.

Tasks & Deadlines

Week	Park	Shewafera	Cheng	Description
8.0	3	3	3	Each team member reads 2 papers or web articles on music generation models.
8.5	5	5	5	Park solidifies architecture decision and programs the pipeline for preprocessing data (algorithm for encoding MIDI files into numerical representations). Cheng and Shewafera implement the baseline model.
9.0	2	2	2	Prepare for and attend the TA meeting.
9.5	5	5	5	Park starts implementing the RNN. Shewafera and Cheng investigate alternative (CNN-based) architectures.
10.0	3	3	3	Shewafera and Cheng start implementing the alternative architecture. Park continues with the RNN implementation.
10.5	6	6	6	Park finalizes RNN architecture. Shewafera and Cheng finalize the alternative architecture. Finish an initial draft of progress report.
11.0	4	4	4	Compare performance between RNN vs. alternative. Document findings in the progress report.
11.5	-	-	-	Final exam preparation.
12.0	-	-	-	Final exam preparation.
12.5	6	6	6	Time permitting, investigate further features to implement (ex. user interface, working with raw audio files).
13.0	8	4	4	Work on the final report and presentation.
13.5	-[1]	4	4	Work on the final report and presentation.
14.0	-	-	-	Final presentations.
14.5	-	-	-	Final presentations.

Table 1: Task deadlines and estimated hours of work.

[1] Park is away for DEFCON.

Risk Register

The following are the analyses of four high-likelihood risks and challenges we identified.

- *Difficulty in gathering sufficient high-quality data*. The amount and quality of data required for training the model remains unclear, and we are unsure as to what variables must be controlled (timbre, dynamics, tempo, etc.) when selecting and cleaning the audio files. To avoid spending unwarranted effort on this task, we will first build a prototype based on readily available datasets to ensure the model is functional, then explore training a model from raw audio files.
- *Encountering nontrivial flaws in architecture design*. At this point in the course, we have minimal intuition as to how generative models are implemented. Hence some additional research is warranted; and each team member will read 2 relevant papers to study industry-standard music generation models and make necessary adjustments to our preliminary architecture before proceeding.
- *Solving implementation errors*. RNNs are generally difficult to implement. To make debugging easier, we will first start with a baseline experiment, and incrementally build the RNN model with increasing topological complexity (i.e., accounting in further musical granularities and textures).
- *Long training time*. At this point the mitigation strategies are not concrete. Generally, one should set aside enough time for training, and utilize transfer learning and modular design patterns if appropriate.
- *Team member unable to finish task(s)*. Personal emergencies may come up, or a task may turn out to be much more laborious than initially anticipated. This could lead to a "snowball effect" in which progress is further impeded by lower morale. Transparent communication is crucial for identifying and stemming such issues early on.

Link to Github or Colab Notebook

`https://colab.research.google.com/drive/1Ohr5fhHgW2NNrSn7i7Pi5dtnrON6K7gf`

References

[1] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," *arXiv preprint arXiv:1206.6392*, 2012.

[2] F. Liang, "Bachbot: Automatic composition in the style of bach chorales."

[3] A. Huang and R. Wu, "Deep learning for music," *arXiv preprint arXiv:1606.04930*, 2016.