
DeepPiano: A Deep Learning Approach to Translate Music Notation to English Alphabet

Ali Hemmatifar
Department of Mechanical Engineering
Stanford University
ahemmati@stanford.edu

Ashish Krishna
Department of Mechanical Engineering
Stanford University
aak9459@stanford.edu

Abstract

We use Deep Learning Methodology to isolate musical notes and to translate them into a format suited for assigning location of keys on piano. Our model comprises of a combination of a four stage convolutional neural network (CNN) and recurrent neural network (RNN) using long short-term memory (LSTM) without forced learning and attention. Finally, we applied connectionist temporal classification (CTC) loss function to the output of LSTM block in order to determine the final sequence of musical notes translated from a given snippet of written sheet music. We achieved more than 60% accuracy on determining individual notes and 12% accuracy of perfect matches of entire sequences. We achieved this accuracy without use of localization or attention.

1 Introduction

Music notations like ♪, ♫, or ♬ are common in day to day life of a musician or those with ample musical knowledge. However, playing these notes or assigning meaning happens to be as complex as language processing itself. We aim to train neural networks to detect and isolate conventional (pictorial) music notes in piano sheet music and to translate them into their letter names. This problem is an example of optical music recognition (OMR) and is specifically useful for music instructors and learners. In addition, the output may be mapped to standard JSON, XML, or MIDI format for enhanced interoperability and conversion to audio. Moreover, the output can be easily transformed to keyboard key inputs which can be useful for learning purposes. This work is useful for music instructors and learners alike who may want to play music without needing to learn formal music language.

We used open-source code Lilypond and generated about 120,000+ random musical sequences of length 6 to 10 notes in Lilypond scripting language. Using a combination of python and shell scripts, we then created labels and image (PNG) output from PDF output of Lilypond script. An example of music score snippet is shown in Figure 1. Labels were generated in parallel with generation of Lilypond music script file. The corresponding text file lists labels of the musical notes. Automation of music sample generation and data labeling was done using shell scripting and Python. One of the most time consuming process for our project is to generate dataset and corresponding labels. The generated image files were then normalized to a fixed resolution and cropped. A simple dictionary was created that assigned integer labels to notes. This allowed us to encode both pitch and timing into a single label e.g. label value '11' represents the *quarter note 'C' of the 4th octave* (commonly known as middle-C on the piano).




Figure 1: An example of Lilypond output with eight musical notes (labels).

Figure 2 shows script files and corresponding outputs. This input is then down-sampled (30×200 pixels), re-scaled (from 0-255 to 0-1), and color-inverted (black to white and vice versa) and fed into a four-stage convolution neural network (CNN).

```
\header {
  tagline = "" % removed
}

\version "2.18.2" % necessary for upgrading to future LilyPond versions.

\score {
  {
    c''8 d'16 aes'8 ges'16
  }
  \layout {
    \context {
      \Score
      \override SpacingSpanner.base-shortest-duration = #(ly:make-moment
1/16)
    }
  }
}
```



#4 45 29 1 50 27 50 4

Figure 2: Data generation and pre-processing step

2 Related work

Optical Music Recognition is an active field of research. Attempts to transcribe musical symbols into digital format dates back to 1960's. In his PhD work in 1966, Pruslin[1] pioneered the concept of musical notes transcription. Bainbridge and Bell[2] worked on a generic OMR framework which was widely adopted by fellow researchers. Rebelo, et.al.[3] published a review paper outlining the opportunities and challenges of OMR. The problem of translation of music notations tends to follow two main approaches. One approach is to use supervised learning where feature detection such as location of ovals and stems to infer location of notes. Such methods also often include removal of staff lines during pre-processing step (Rebelo et al.[4]). Another approach is to use an end-to-end learning which requires large corpus of data. Such approach rely on the system learning by itself without many rules, thus, is generally unsupervised. Jorge et al.,[5] recently attempted OMR using an RCNN-CTC architecture. Their results have been promising. In this work, we attempt to expand on recent OMR strategies presented by Jorge, et.al.[3] using Lilypond engraver [6])

3 Dataset and Features

In our approach, we take input in graphical PNG format which represents a sequence of musical notes and output a series of integer labels that can be post-processed into XML, JSON, or other custom formats. For interoperability, we have developed a canonical vocabulary. There is not enough labeled corpus available in public domain. One of the key tasks in our work is to develop a method and dictionary to represent large set of sequenced music samples. We defined a canonical vocabulary translation that correlates labels for generated music to Lilypond syntax[6].

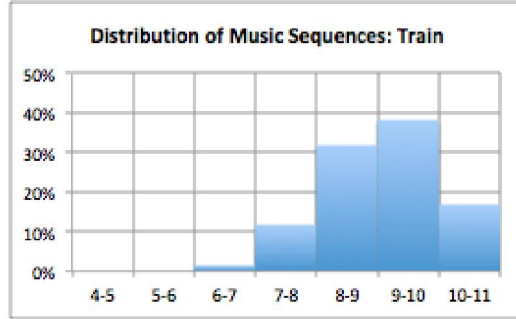


Figure 3: Histogram of lengths of music sequences randomly generated with Lilypond.

Table 1: Dataset and Steps/Epochs

Run Num	Train Size	Test Size	Num of Steps per Epoch	Epochs
Run 1	21,000	300	420	112
Run 2	38,500	300	770	60
Run 3	73,700	300	1474	32

4 Methods

4.1 Architecture

A schematic of the architecture is shown in Figure 4. We then reshape output of CNN and feed it into a recurrent neural network (RNN) consisting of long short-term memory (LSTM) units. We use connectionist temporal classification (CTC) loss function and output label sequences with maximum probability given the given ground truth labels. Table 1. shows datasets generated for experiments. However, any set of random data may be generated now that we have automated the process. It should be noted that the rate of music sample generation was about 100 samples per minute therefore, a tedious and time consuming work was required to prepare sample data.

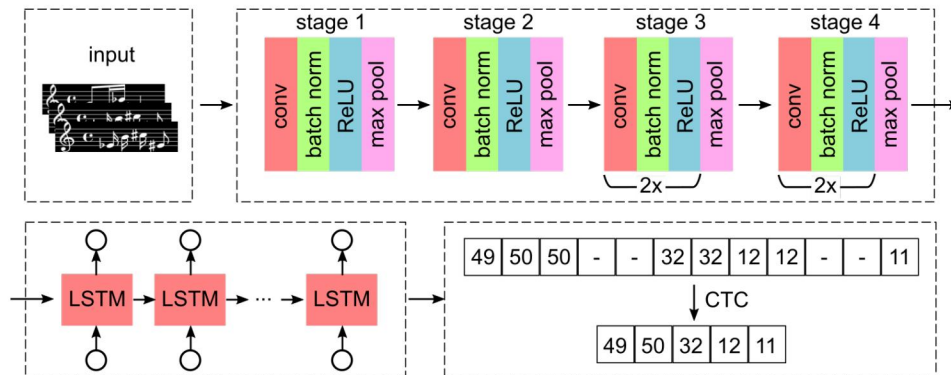


Figure 4: Schematic of convolutional recurrent neural network used for musical note translation application.

5 Experiments/Results/Discussion

5.1 Experiments Performed

We ran three different experiments using datasets shown in Table 1. Table 2 shows the list of hyper parameters for each experiment.

Table 2: List of hyperparameters

Parameter	Value
Decay rate	0.98
Decay steps	10000
Initial learning rate	0.001 or 0.01
Momentum parameter	0.9
Exponential decay rate for the 1st moment	0.9
Exponential decay rate for the 2nd moment	0.999

5.2 Results & Discussion:

Figure 5 shows results of our training set:

- Notes accuracy defined as total percentage of musical notes correctly labeled in evaluation data set
- Sample accuracy defined as percentage of examples correctly labeled
- Notes and sample accuracy significantly increase from 20k to 40k training data.
- Notes and sample accuracy fixed from 40k to 70k training data.

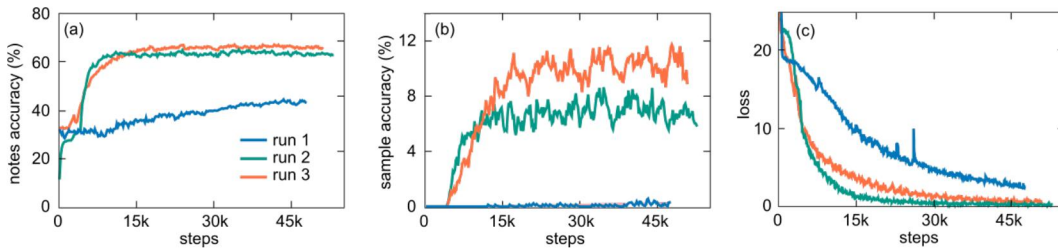


Figure 5: (a) notes accuracy, (b) sample accuracy, and (c) training loss vs steps during training phase.

Some Examples:

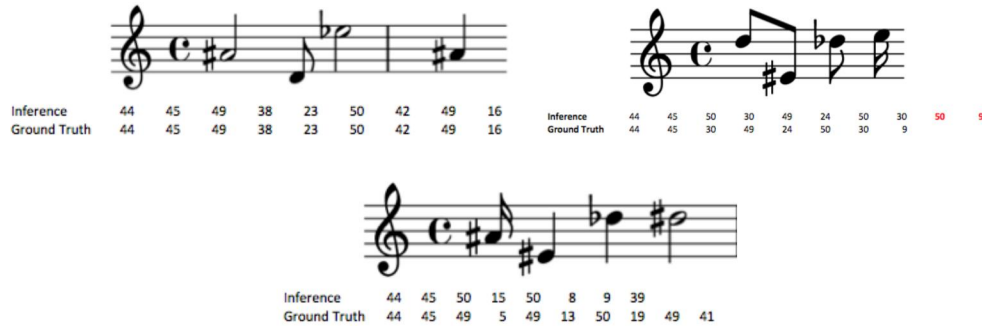


Figure 6: (a) Perfect Match, (b) Extra Notes predicted, and (c) Less notes predicted

We see that there are 3 possible outcomes for prediction (see Figure 6).

6 Conclusion/Future Work

6.1 Conclusion

1. Our model seems to have threshold for minimum dataset needed for training
2. LSTM performs better with more samples (Run 1 > Run 2 > Run 3)

3. CNN reaches a saturation (70% accuracy) around 40,000 samples
4. In order for better LSTM performance, more randomization of sequences may be needed

6.2 Future Work

Future work will include improvement of accuracy of sequence prediction. Convolution step is the first one to be inspected. Adding more stages or changing window size may help identify saturation limits. Another area to explore is to provide more randomization and longer sequence of music data. In addition, study of expanding sequence prediction accuracy for LSTM as well as generation of random sequence that respect musical scales could help speed up the process. For example, music generated in the scale of C-Major (C-D-E-F-G-A-B-C) following the rules of musical notes sequencing reduces probability of occurrence of other notes like C-sharp or G-flat, etc. Furthermore, multiple note groupings (musical chords) could be another possible extension to this project.

7 Contributions

A.H and A.K both contributed to data collection, pre-processing, and developing the algorithm. The authors contributed equally to the initial definition and learning tasks as well as data preparation and training the CNN. Running the code, refinements, debugging were responsibility of both authors. The code is inspired by this code and is available at this link.

References

- [1] Dennis Howard. Pruslin. Automatic recognition of sheet music. 01 1966.
- [2] D. BAINBRIDGE. The challenge of optical music recognition. *Computers and the Humanities*, 35(2):95–121, 2001.
- [3] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R. S. Marcal, Carlos Guedes, and Jaime S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, Oct 2012.
- [4] Ana Rebelo, G. Capela, and Jaime S. Cardoso. Optical recognition of music symbols. *International Journal on Document Analysis and Recognition (IJ DAR)*, 13:19–31, 2009.
- [5] Jorge Calvo-Zaragoza, Jose J. Valero-Mas, and Antonio Pertusa. End-to-end optical music recognition using neural networks. In *ISMIR*, 2017.
- [6] Lilypond. Lilypond music engraving.
- [7] Python API. Documentation for python.
- [8] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. 2016.
- [9] ALI HEMMATIFAR and ASHISH KRISHNA. Deepiano: A deep learning approach to music translation, 2018.