Real-time Identification of Simple and Extended Musical Chords using Artificial Neural Networks

Joachim Alfonso A. Navarro (Student 1), Lesli Natasha A. Coronel (Student 2),
Mc Jervis S. Villaruel (Research 2 teacher), and Kiel F. Granada (Research adviser)

Philippine Science High School – Main Campus, Department of Science and Technology, Agham Road, Diliman, Quezon City, 1101, Philippines

ABSTRACT

Musical chords are fundamental to musical harmony and are named due to their importance. For most people, accurately identifying or naming musical chords by ear is a difficult task requiring high levels of skill and training. A neural network that aimed to identify musical chord names from their component notes in real-time was thus programmed. A dataset of 37 chord types with 12 notes each was used to train the neural network. Two sessions of training were conducted: one with 10,000 epochs and another with 20,000. Validation and training accuracy data was obtained after every session, and response time data was obtained by testing the network on manual MIDI inputs of chords randomly selected from the dataset. Peak accuracies of 0.2% and 7.5% were obtained on the validation and training datasets respectively. Overfitting on the training dataset was observed, suggesting that the task of choosing between 444 different chords is too complex for a neural network to perform. A left-tailed T-test for one mean (n = 30, α = 0.05) was carried out on randomly selected chords from the dataset and showed that the neural network responded significantly more quickly (4.9 ms) than a standard of 40 ms for real-time use. The use of a different machine learning algorithm is recommended, as is using fewer chord types and other input formats such as audio. Such an algorithm may be implemented on devices or software for the purposes of music education.

Keywords: music information retrieval, harmony, GPU, musical note

INTRODUCTION

Chords are groups of individual notes often played together and are an integral part of harmony in many forms of music. Their names are determined by at least two parameters: the root note and the chord type. While “the general music-learning public places a high demand on chord-based representations of popular music” (Humphrey, Bello, & Cho n.d.), identifying the parameters of a certain chord takes certain skills obtained naturally or through special training (Zatorre, Perry, Beckett, Westbury, & Evans 1998), limiting the number of people who can correctly name chords by hearing them. Current neural network implementations of chord identification such as that of Perera and Kodithuwakku (2005) are limited to simpler chord types, and exploration of real-time chord identification through the use of neural networks is limited if at all existent. To explore this possibility, an Artificial Neural Network (ANN)-based real-time algorithm that aims to correctly identify the names of one of 444 unique chords over 37 simple and extended chord types using integer representations of MIDI note signals as inputs was designed. Such an algorithm could be useful in creating and/or implementing computer programs with chord recognition functions and would be particularly useful in the fields of music education and production. This paper details the design and performance of the ANN in terms of accuracy and latency.

METHODOLOGY

Procurement of materials

A graphics processing unit (GPU) was acquired and utilized to facilitate neural network computations, which are highly parallel in nature (Nickolls, Buck, Garland, & Skadron 2008). The NVIDIA GeForce GTX 1070 GPU, as prescribed by Dettmers (2017), was used.

The necessary software, including a Python distribution, Python IDE, and various libraries, were then installed according to the methods in mustgoplay (2016).

Test chord dataset preparation and randomization

A series of programs that generates a dataset of 444 chords from 37 selected chord types and all 12 notes in the Western musical scale and converts them to combinations of MIDI note numbers was written. The output of these programs was a database that contained MIDI note number combinations with each combination representing a particular chord. For example, a C major chord can be represented by the combination [60, 64, 67]. This database was then split into training and validation datasets with 60% and 40% of the chords respectively (Kolassa 2015).

Artificial Neural Network (ANN) design

After the dataset of chords was generated, the configuration of the artificial neural network (ANN) was defined. A structure of 128 input neurons, each representing MIDI notes 0 through 127, and 444 output neurons, each representing one unique chord in the dataset, was chosen.

ANN development

The neural network design was then implemented in Python using the Keras (Keras Team 2018) framework running on TensorFlow. Both input and output layers used ReLU activation functions and orthogonal matrix weight initializations to improve performance (Colina, Perez, & Paraan 2017).

ANN training, testing, and data collection

The ANN was then repeatedly trained on the training dataset and tested on the validation dataset using the built-in functions of Keras (Keras Team 2018) until the accuracy of the network on the validation dataset (called "validation accuracy") stopped significantly increasing. Training and validation accuracies were collected over the training and testing runs and recorded.

The ANN was then tested for response time. Thirty randomly selected chords were played manually on a MIDI controller, and data from the MIDI controller was automatically handled by a script that passed it to the inputs of the neural network. Response times for each chord were recorded. A statistical left-tailed T-test for one mean with Greeff (2016)'s standard of 40 milliseconds as a benchmark was carried out on the 30 collected response times.

RESULTS

Validation and training accuracies

After two training sessions of 10,000 and 20,000 epochs, the ANN showed a markedly low peak validation accuracy of 0.2%. In both training sessions, the ANN showed a sharp increase followed by a gentle decrease in validation accuracy by the 3500-epoch mark. Validation accuracy reached 0% at 7500 epochs and remained there until the end of both sessions

The ANN showed normal learning behavior on the training dataset with steadily increasing training accuracy. After 13,000 epochs, this accuracy started to level off, reaching a maximum value of 7.53%.

The following are graphs showing the validation (orange) and training (blue) accuracies of the ANN.

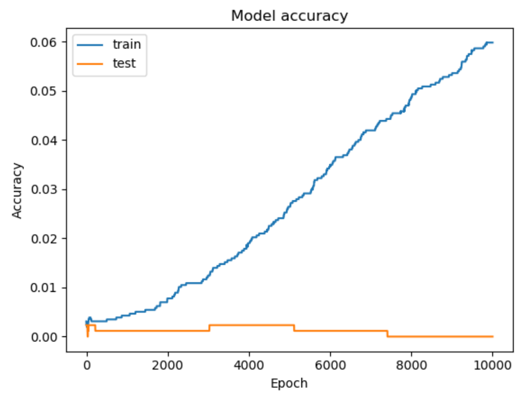

Figure 1. Training (blue) and validation (orange) accuracies of neural network from the 10,000-epoch training session

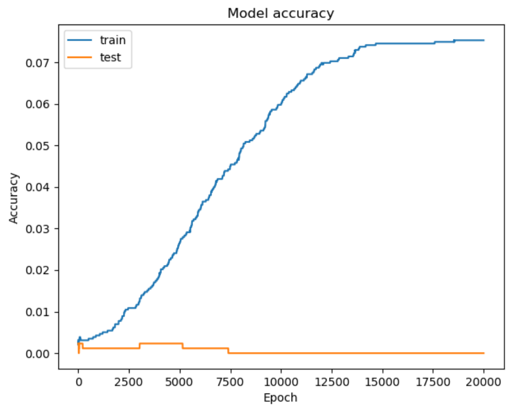

Figure 2. Training (blue) and validation (orange) accuracies of neural network from the 20,000-epoch training session

Response times

The ANN achieved a mean response time of 4.94 milliseconds with a standard deviation of 11.9 milliseconds over 30 randomly selected chords. It was found that at a significance level of 0.05, the obtained t-statistic of -3.18 was less than the critical t-value of -1.70. There was therefore not enough evidence to prove that the response time of the ANN was greater than or equal to 40 milliseconds (Greeff 2016). The following is a table with summary measures used in the t-test for one mean:

Summary measure	Value
Mean, ms	4.94
Standard deviation, ms	11.9
t-statistic of the mean	-3.18
Critical *t*	-1.70
Null hypothesis outcome	Rejected

DISCUSSION

Considering the use of a traditional neural network to accomplish a complex classification into 444 possible chords, the low obtained training and validation accuracies are of little surprise. Other neural network researches used fewer chords: Zhou and Lerch (2015) and Perera and Kodithuwakku (2005) each used 24 chords, and Osmalskyj, Embrechts, Pierard, and Van Droogenbroeck (2012) used 10. Each of these studies reported neural networks to be feasible for musical chord identification.

The steady increase in the training accuracy of the neural network despite its low validation accuracy can be attributed to overfitting, a phenomenon wherein the neural network merely learns the data in the training dataset and is not able to classify new data. In this case, it was not able to predict the chord names of arrangements of notes that it had not encountered before. This is reinforced by the comparatively low obtained validation accuracies and the lack of an increase in said parameter over the course of training. Given the complexity of the task (444-way categorization), this was once again a reasonable outcome.

While this ANN was not able to accurately identify all the chords in the database used in this study, its latency testing results show that neural networks may be suitable for a range of other tasks that require real-time inferences and/or processing.

SUMMARY & CONCLUSION

A neural network that aimed to identify 444 different simple and extended musical chords in real-time was developed. It reached markedly low accuracies and responded in satisfactory time. These results show that simple neural networks are not feasible for identifying large sets of chord types, as such a task is of notably high complexity.

These results additionally show that neural networks may be suitable for use in other real-time tasks.

Future work may consider the use of either other machine learning algorithms or a mathematical model, along with a shorter list of chord types. Audio could also be used instead of MIDI as neural network input, as audio-based datasets of chords played on different instruments are freely available online. However, doing so will restrict the list of chord types to those in the chosen dataset.

REFERENCES

COLINA NCA, PEREZ CE, PARAAN FNC. 2017. Simple techniques for improving deep neural network outcomes on commodity hardware. AIP Conference Proceedings 1871: 040001.

DETTMERS T. 2017. Which GPU(s) to get for deep learning: My experience and advice for using GPUs in deep learning. Retrieved from http://timdettmers.com/2017/04/09/which-gpu-for-deep-learning/.

GREEFF W. 2016. The influence of perception latency on the quality of musical performance during a simulated delay scenario. [MS dissertation]. Pretoria: University of Pretoria. 54p.

HUMPHREY E, BELLO JP, CHO T. n.d. Chord recognition. Retrieved from https://steinhardt.nyu.edu/marl/research/chord_recognition on May 2, 2018.

Keras Team. 2018. Keras. [GitHub repository]. Retrieved from ttps://github.com/keras-team/keras on January 25, 2018.

KOLASSA S. 2015. Dividing percentage [Online forum comment]. Retrieved from https://datascience.stackexchange.com/questions/5113/dividing-percentage.

mustgoplay. 2016. Anaconda Python, CUDA, cuDNN and Theano with PyCharm install for CPU and GPU on Windows 10. [Online forum comment]. Retrieved from https://github.com/Theano/Theano/issues/5348.

NICKOLLS J, BUCK I, GARLAND M, SKADRON K. 2008. Scalable parallel programming with CUDA. ACM Queue 6(2): 40-53.

OSMALSKYJ J, EMBRECHTS J-J, PIÉRARD S, VAN DROOGENBROECK M. 2012. Neural networks for musical chords recognition. Belgium: University of Liège.

PERERA N, KODITHUWAKKU SR. 2005. Music chord recognition using artificial neural networks. 1st Proceedings of the International Conference on Information and Automation: 304-308.

ZATORRE RJ, PERRY DW, BECKETT CA, WESTBURY CF, EVANS, AC. 1998. Functional anatomy of musical processing in listeners with absolute pitch and relative pitch. Proceeding of the National Academy of Sciences 95: 3172-3177.

ZHOU X, LERCH A. 2015. Chord detection using deep learning. 16th International Society for Music Information Retrieval Conference: 52-58.