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

Joachim Alfonso A. Navarro (Student 1), Lesli Natasha A. Coronel (Student 2),
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 is a difficult task requiring high levels of skill. With this in mind, a neural network that aimed to identify musical chord names from their component notes in real-time was programmed in Python using the Keras framework running on the TensorFlow library. Validation accuracy data was obtained after every training session, and manual MIDI inputs were used to obtain response time data. It was found that the current number of training iterations (2,400) provided an insufficient peak validation accuracy of 5.5%, but showed steadily decreasing values of the loss function. A left-tailed T-test for one mean was carried out on 30 randomly selected chords from the dataset and showed that the neural network responded significantly faster than the generally accepted standard of 10 milliseconds for real-time use. Further development of the neural network is recommended to increase validation accuracy. Such a neural network may be implemented on devices or software for the purposes of music education.

Keywords: Chord, Chord identification, Extended chord, GPU, Neural network, Real-time

INTRODUCTION

Chords are collections of two or more musical notes, often played together, and are arranged in such a way that they follow the so-called “rules of harmony” (Leino, Brattico, Tervaniemi, & Vurst 2007). These rules are recognized by humans as a response in the brain that is triggered when they are violated (Leino et al. 2007). Because of this, chords are fundamental to the harmonic integrity of any musical work.

A musical chord is commonly identified by three parameters: root note, chord type, and inversion. The root note serves as a reference point for the other notes which are played in the chord. These other notes are determined by the chord type. The inversion tells which note in the chord is played lowest. For example, a C major chord, 0th inversion has “C” as the root note, and “major” as the chord type. A major chord type includes the 1st (root), major 3rd, and 5th harmonics of the typical Western major scale, and the 0th inversion says that the root note is the bottom note. Thus, the notes of the 0th inversion of a C major chord are C (1st), E (major 3rd), and G (5th), in that order.

Humphrey, Bello, and Cho (n.d.) state that “the general music learning public places a high demand on chord-based representations of popular music” (par. 1). However, complete and accurate determination of these chords by hearing requires the use of both absolute and relative pitch, because chords utilize both an absolute reference point (root note) and a relative configuration of harmonies (chord type).

Absolute pitch is expressed when one can identify a musical note by hearing it, while relative pitch is shown when one can recognize the distances between musical notes (Zatorre, Perry, Beckett, Westbury, & Evans 1998). It is expressed in a low percentage of the human population and acquired through either favorable genes or music training at a young age, or both (Baharloo, Service, Risch, Gitschier, & Freimer 2000).

Complete chord identification is thus a rare skill found in those with mastery of both absolute and relative pitch, even though chords play an important role in any musical work. An algorithm that automatically identifies chords from individual notes in real time would be a first step towards addressing this problem.

Artificial Neural Networks (ANNs) are computational models that use a layered structure of computational units called neurons in order to learn a certain task. Using mathematical functions, the neural network is able to train itself iteratively through a process called backpropagation until its error rate is significantly low (Nielsen 2015; Sanderson 2017). Thus, machine learning modeled through ANNs is designed to be adaptable and simple through self-organization (Daniel 2013). The machine learns a given input dataset through by mapping an input through a probability distribution into its most probable class (Colina, Perez, & Paraan 2017).

This study aims to develop a neural network that quickly and correctly identifies common and extended one-root musical chords formed by playing more than two notes on a MIDI input device. Specifically, the program must identify common & extended chords and respond quickly enough to be used in live performance (Stark & Plumbley 2009). The program must be implemented in programming languages that have MIDI input-output libraries such as pygame for Python and and neural network libraries such as Keras and TensorFlow to facilitate ease of coding. Such application of real-time chord identification would be used in the field of music education, where a low proportion of music students have absolute pitch (Gregersen, Kowalsky, Kohn, & Marvin 1999) despite their demand for chordal representations of music (Humphrey, Bello, & Cho n.d., par. 1). These allow said students to learn to identify the chords they are playing more quickly and accurately, and help them develop their senses of relative and absolute pitch. They are also used in situations when musicians need to verify the chords they are playing for correctness, which usually happens when they are learning or composing a musical piece.

While automatic chord identification programs exist, they are either classical algorithm implementations that run on audio inputs in real-time (Fujishima 1999, Stark & Plumbley 2009), or neural network implementations that do not aim to run in real-time and do not include chords outside of major and minor triads (Perera & Kodithuwakku 2005, Osmalskyj, Embrechts, Piérard, & Van Droogenbroeck 2012, Zhou

& Lerch 2015). Including other chords such as 7ths and extended chords would allow identification of chords from more complex styles of music, such as jazz. The utilization of neural networks to identify extended chords from MIDI signals in real-time is largely unexplored and would provide useful data for future research.

METHODOLOGY

Procurement of materials

In order to facilitate computations that the neural network will perform, a graphics processing unit or GPU was utilized (Nickolls, Buck, Garland, & Skadron 2008).

It is useful for parallel computing applications such as artificial neural network (ANN) simulations and training. Colina, Perez, and Paraan (2017) found that the use of a consumer GPU accelerated ANN computation by approximately 30 times when compared to a regular central processing unit. The minimum recommended GPU for research work is the NVIDIA GeForce GTX 1070 (Dettmers 2017). This GPU is used in many neural network researches and papers (Colina, Perez, & Paraan 2017, Kang, Hong, & Park 2017, Kim, Hong, Nam, & Park 2017, Zorrilla et al. 2017) and was ordered online.

The necessary pieces of software to define the structure of, train, and test the neural network were acquired and installed in one of the researchers' computers using the methods in mustgoplay (2016), which uses the following components.

Keras (Keras Team, 2018) is a Python-specific neural network library that allows for swift prototyping of ANN structures. Its relative simplicity is well-suited to research. It runs on TensorFlow, a neural networks library for Python.

NVIDIA CUDA & cuDNN are libraries that allow computation on an NVIDIA GPU and ANN simulation on a NVIDIA GPU, respectively.

Anaconda is a specific distribution of the Python and R programming languages for scientific computing and data processing.

PyCharm is a piece of software known as an independent development environment or IDE, which allows the user to write and run code.

Finally, pygame, a Python library for game development, will be installed as it contains provisions for real-time MIDI signal input to Python. This input can be fed to the neural network for chord identification.

Test chord dataset preparation and randomization

A program that makes a full list of chords from the 12 possible root notes and the chosen chord types was then written. The program outputs these chords to a text file that will serve as the input dataset of the ANN. Its training and validation datasets were randomized using a Python script to be detailed in Neural network training, testing, and data collection. The input dataset was split by this script into training and validation datasets with 80% and 20% of the chords respectively (Kolassa 2015).

Neural network development

An algorithm that takes a current MIDI input and converts it into a format that can be interpreted by the neural network was written. The chords are interpreted by the neural network on a per-note basis (that is, one neuron per note spanning two octaves of 12 notes each for a total of 24 notes). Using the appropriate programming tools in *Procurement of materials*, an artificial neural network whose goal is to identify chords was written in Python using Keras (Keras Team 2018) and TensorFlow.

Neural network training, testing, and data collection

These processes are iterative by nature and thus proceeded once for every training epoch (one training “pass” through the dataset) of the neural network.

The neural network was trained using the built-in functions of Keras (Keras Team 2018). During training, the neural network was tested with the chords in the training dataset output by the script in Test chord dataset preparation and randomization. The appropriate data such as training and validation accuracy and losses were recorded. Testing proceeded in an identical manner to training using the validation dataset. After the testing phase, adjustments were automatically made by Keras to the neural network depending on the desired output. The network was trained repeatedly until its validation accuracy stopped significantly increasing.

After training, the response time of the neural network was tested against a generally accepted standard of 10 ms for real-time use using a left-tailed T-test for one mean carried out on 30 randomly selected chords from the dataset. These chords were played manually on a MIDI controller, and response time was recorded using another Python script. The statistical T-test was carried out using R (R Core Team 2017) and the Rcmdr package (Fox & Bouchet-Valat 2017).

RESULTS

Validation and training accuracies

Every neural network is trained by traversing through a training dataset, and each pass through the training dataset is called an "epoch". After every epoch, the accuracy of the network is tested separately using a different dataset called the validation dataset (Nielsen 2015).

After three training sessions of 800 epochs each, the neural network showed a markedly low peak validation accuracy, or rate of correct predictions on the test dataset, of under 5.5%. Every session of 800 epochs showed stepwise and irregular increases and decreases of validation accuracy before consistently dipping to 0% towards the end of the training session. Below are graphs showing the validation (orange) and training (blue) accuracies of the neural network.

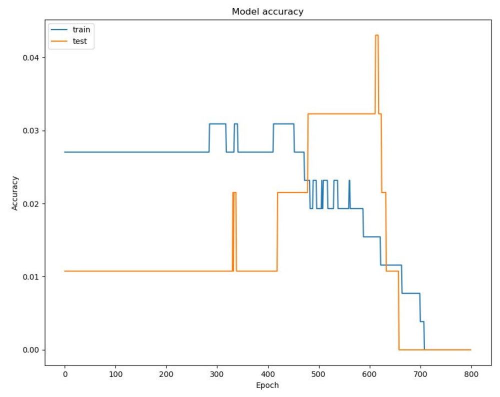

Figure 1. Training (blue) and validation (orange) accuracies of neural network, epochs 0-800

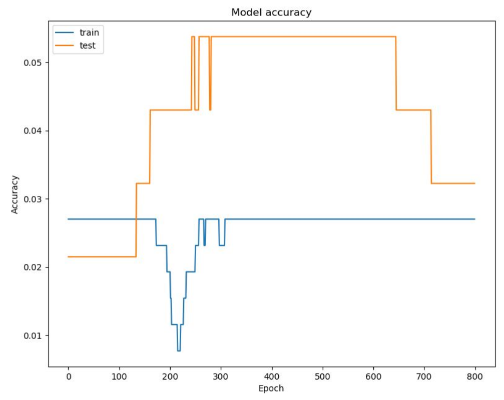

Figure 2. Training (blue) and validation (orange) accuracies of neural network, epochs 800-1600

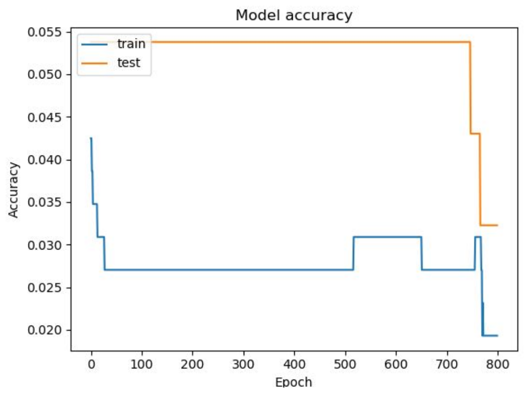

Figure 3. Training (blue) and validation (orange) accuracies of neural network, epochs 1600-2400

Validation and training losses

The loss function is a special series of mathematical operators that shows how far the neural network is from the correct prediction. As such, lower values are considered favorable (Nielsen 2015).

The same training sessions returned decreasing values of the loss function as the training session proceeded. The losses decreased in a smooth fashion, but the graphs formed were not of the same kind. Below are graphs showing the validation (orange) and training (blue) losses of the neural network.

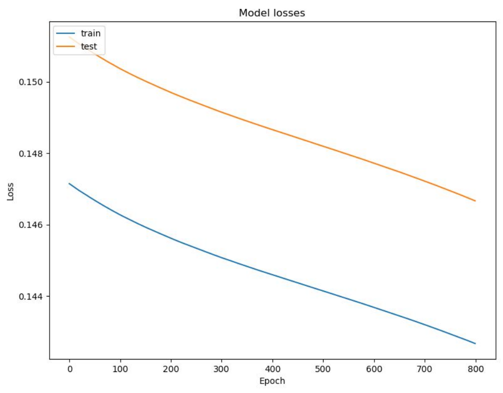

Figure 4. Training (blue) and validation (orange) losses of neural network, epochs 0-800

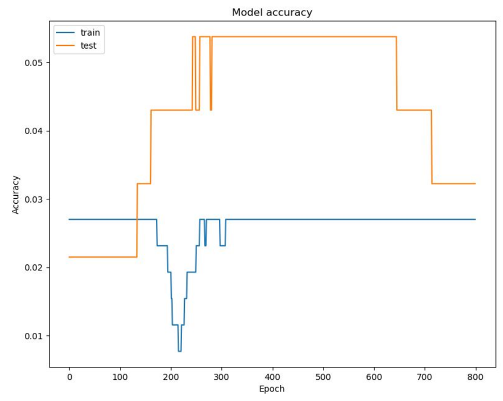

Figure 5. Training (blue) and validation (orange) losses of neural network, epochs 800-1600

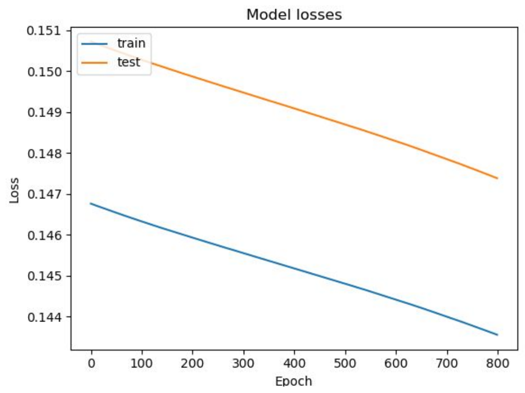

Figure 6. Training (blue) and validation (orange) losses of neural network, epochs 1600-2400

Table 7. Results of t-test for one mean on neural network response time

Summary measure	Value
Mean, ms	2.95466667
Mean t-statistic	-17.19
Critical t	-1.699
Null hypothesis outcome	Reject

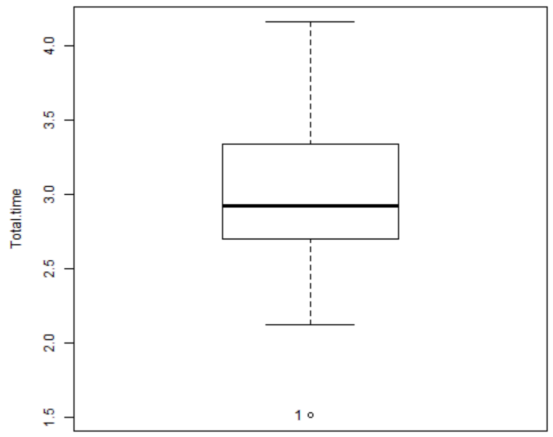

Figure 8. Boxplot of total response times of neural network to MIDI inputs in milliseconds. Note: Null hypothesis is that response time is less than or equal to 10 ms. The program used to make this boxplot does not output units on the x-axis.

DISCUSSION

The neural network performs atypically from the perspective of validation and training accuracies. It is expected that the training and validation accuracies would increase as the training proceeds. However, it is behaving as expected in terms of validation and training losses, as they decrease during training (Nielsen 2015).

Bodik (2018) suggests that this problem is caused by the input and output formats of the neural network being too complex for the neural network to properly classify input data. This can cause the neural network to underfit, i.e. not be able to adapt to the characteristics of the training data. This is manifested by consistently unfavorable values of validation accuracy. (Nielsen 2015, Bodik 2018). It is therefore recommended that steps to reduce this underfitting be taken in the future.

A neural network that aimed to identify common and extended musical chords in real-time was programmed, and did so with a markedly low accuracy. However, its satisfactory response time shows that neural networks may be used in real-time tasks.

ACKNOWLEDGEMENTS

We would like to thank Ms. Kiel Granada, Mr. Edge Angeles, and Ms. Maria April Rose A. Andaca for their invaluable guidance and support, and Mr. Alejandro Alfonso Navarro and Ms. Maria Amina Amado for their financial assistance.

REFERENCES

BAHARLOO S, SERVICE S, RISCH N, GITSCHIER J, FREIMER N. 2000. Familial aggregation of absolute pitch viruses. American Journal of Human Genetics 67: 755-758.

BODIK S. 2018. Answer to "Low loss and low accuracy. What is the reason?". [Online forum comment]. Retrieved from https://stats.stackexchange.com/posts/328889/revisions on March 5, 2018.

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/.

FOX J, BOUCHET-VALAT, M. 2017. Rcmdr: R Commander. R package version 2.3-2.

FUJISHIMA T. 1999. Realtime chord recognition of musical sound: A system using common Lisp music. Stanford, CA: Stanford University.

GREGERSEN P, KOWALSKY E, KOHN N, MARVIN, E. 1999. Absolute pitch: prevalence, ethnic, variation, and estimation of the genetic component. [Letter to the Editor]. American Journal of Human Genetics 65: 911-913.

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

KANG JK, HONG HG, PARK KR. 2017. Pedestrian detection based on adaptive selection of visible light or far-infrared light camera image by fuzzy inference system and convolutional neural network-based verification. Sensors 17(7): 1598.

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

KIM KW, HONG HG, NAM GP, PARK KR. 2017. A study of deep CNN-based classification of open and closed eyes using a visible light camera sensor. Sensors 17(7): 1534.

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

LEINO S, BRATTICO E, TERVANIEMI M, VURST P. 2007. Representation of harmony rules in the human brain: Further evidence from event-related potentials. Brain Research 1142: 169-177.

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.

NIELSEN MA. 2015. Neural networks and deep learning. Determination Press. Retrieved from http://neuralnetworksanddeeplearning.com/.

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.

R Core Team. 2017. R: A language and environment for statistical computing. Retrieved from https://www.R-project.org.

SANDERSON G. [3Blue1Brown]. 2017. But what *is* a Neural Network? | Chapter 1, deep learning. Retrieved from https://www.youtube.com/watch?v=aircAruvnKk.

STARK AM, PLUMBLEY MD. 2009. Real-time chord recognition for live performance. London: Queen Mary University of London.

ZATORRE RJ, PERRY DW, BECKETT CA, WESTBURY CF, EVANS, AC. 1998. 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.

ZORRILLA AL. DE VELASCO-VÁZQUEZ M, SERRADILLA-CASADO O, ROA-BARCO L, GRAÑA M, CHYZHYK D, PRICE CC. 2017. Brain white matter lesion segmentation with 2D/3D CNN. International Work-Conference on the Interplay Between Natural and Artificial Computation 2017 10334: 394-403.