**6CCE3EEP/7CCEMEEP**

**Individual Project Submission 2021/22**

**Name: Gustavo Masson Auriemo**

**Student Number: 19017464**

**Degree Programme: Electronic Engineering with Management**

**Project Title:**

**Supervisor: Dr. Bipin Rajendran**

**Word count:**

**RELEASE OF PROJECT**

Following the submission of your project, the Department would like to make it publicly available via the library electronic resources. You will retain copyright of the project.

I **agree** to the release of my project

I **do not** agree to the release of my project

**RELEASE OF VIDEO**

Following the submission of your project demonstration, the Department would like to make it publicly available via youtube. You will retain copyright of the project.

I **agree** to the release of my project video

I **do not** agree to the release of my project video

**Signature:** (An e-signature is acceptable, i.e., attach your signature as a picture, sign using the draw function)

**Date:03/04/2022**

**Originality Avowal**: "I verify that I am the sole author of this report, except where explicitly stated to the contrary."

**Abstract or Executive Summary**: This is a half to one-page summary of your report (rather than your project) listing your problem statement and main findings.

**Contents Page**: It is very useful to have a high-level contents page, list of figures and tables in your report as well as a list of appendices. It will also help you when you structure the report.

**Introduction**: Your introduction really sets the scene for the report. It should be a clear statement of what the project is about, a summary of the background and context and summarise what you set out to achieve.

**Background**: The background is not just a literature review but explains why your project is important and explores the theory and previous work. A good literature review synthesises and critically evaluates the existing literature identifying gaps in current knowledge that the project will fill rather than simply describing previous work.

**Report**: The next part of the report might be then broken into chapters such as methodology, findings, specification, requirements, design etc. The aim of the section as a whole is to describe the work you've done, justify your approach and explain how you arrived at your conclusion. You might also analyse the technical findings of your work.

**Conclusion**: here you can summarise the project again, make any conclusions, statements or assertions that you believe your project has achieved and offer some ways that the project might be taken forward in future.

**Professionalism and Responsibility:**It is important that you consider the professional influences on your project such as standards and competencies. You can also discuss general ethics, sustainability, cyber-security or other issues applicable to the project.

**Bibliography**: a list of all your references following the acceptable [college citation format.](https://libguides.kcl.ac.uk/reference)   
(Martin et al., 2020) or (Salas & D’Agostino, 2020) for two authors. For every in-text citation, there must be a reference list one: APA

**Appendix**: additional useful information that won't be marked but provides some completeness E.g. tables of data, additional graphs etc

# **ABSTRACT**

This report sets out a scalable method for harmony recognition utilizing the Neural Engineering Framework (NEF) developed by Eliasmith & Anderson.

A proof-of-concept is shown, along with results utilizing a restricted dataset with four different learning rates over 10 epochs.

Three different approaches to consider the seriousness of error were taken

Table of Contents

[ABSTRACT 3](#_Toc100163540)

[INTRODUCTION 3](#_Toc100163541)

[BACKGROUND 4](#_Toc100163542)

[Music Theory 4](#_Toc100163543)

[Neural Networks 6](#_Toc100163544)

[Neural Engineering Framework & Nengo 7](#_Toc100163545)

[Previous Work & General Remarks 8](#_Toc100163546)

[Report 10](#_Toc100163547)

[Methodology 10](#_Toc100163548)

[The dataset 10](#_Toc100163549)

[Framework 10](#_Toc100163550)

[Design 10](#_Toc100163551)

[Findings 10](#_Toc100163552)

[Conclusion 11](#_Toc100163553)

[Professionalism and Responsibility 11](#_Toc100163554)

[Bibliography 11](#_Toc100163555)

[Appendix 11](#_Toc100163556)

# 

# **INTRODUCTION**

Your introduction really sets the scene for the report. It should be a clear statement of what the project is about, a summary of the background and context and summarise what you set out to achieve.

It doesn’t take a specialist to be marvelled by the complexity and intricacy of some of the piano performances recorded by jazz pianist Bill Evans. Timing, voicings, and tone seem to have a life of their own, and in my own inability to emulate and transcribe what I heard, I felt powerless. That is a problem many musicians, beginners, and professionals struggle with, and something which I felt I had the tools and motivation to tackle.

Among the population of music students only about 4% have absolute pitch, which is to say they can name a note without any reference tone. This allows them, with enough training, to name the notes present in different chords (3 or more musical notes played simultaneously), and with certain ease transcribe music pieces. For the remaining 96%, it remains an arduous task which requires years of relative pitch training, technical expertise, and some trial and error. For example, in the 1970’s a group of music students at the highly prestigious Berklee College of Music attempted to transcribe and compile several jazz standards into a series of books named “Real Book”. To this day, errors persist, and differences arise regarding the best versions. This illustrates that even some of the best-trained musicians in the world have difficulty in carrying out this task.

In fact, it is arguably an example of the phenomenal capacity of human intelligence, leveraging perception of complex auditory scenes, cognitive ability, previous theoretical knowledge, and inference when testing new combinations (Benetos et al., 2018). The true issue is not about melody (i.e., one note played at a time), but rather about harmony (i.e., tones played simultaneously). The interaction between the different notes when played together create a complex set of harmonics (extra tones arising from the difference in frequency of the notes), which makes it extremely difficult for the ear to break it down into discrete parts. Consider that plus the vast number of combinations 88 keys (in a piano) can form, the poor quality of recordings, and fast changes, and you can begin to imagine the complexity of the problem of music transcription.

While undertaking a module on neuromorphic computing, we discussed how researchers are currently seeking ways to mimic the brain’s functionality and efficiency, and how these are being implemented in software and hardware. Naturally, the idea of being able to model one’s brain under different scenarios and study how accurate they were, is immensely appealing. The very “human” and complex nature of music appeared to be the perfect environment to explore these models, and further develop the knowledge within a field I am extremely passionate for.

Being presented to Nengo, a software tool that allows the implementation of biologically plausible models of cognition (Eliasmith & Anderson, 2003), proposed a new way to approach this problem and build on the work of other researchers to understand how such models would perform in face of a problem humans have difficulty in tackling.

By weaving together research in areas such as computational neuroscience, music theory, and signal processing, this report aims in providing a proof-of-concept model for harmony transcription from piano audio recordings. Moreover, it aims to do so whilst maintain a degree of fidelity to biological processes in the brain and attempting to explore the efficiency of such algorithms in relation to state-of-the-art techniques.

# **BACKGROUND**

The nature of this project is interdisciplinary, and as such, basic knowledge of each area will benefit the reader to better understand how each aspect connects to the other in a meaningful way. This section will be split into four: **Music Theory**, **Neural Networks**, **Neural Engineering Framework & Nengo**, and **Previous Work & General Remarks**.

Key words will be in bold and will be referenced throughout the report.

## Music Theory

A musical **note (or tone)** is, in its most simple terms, a frequency. For example, when a string in a well-tuned guitar is plucked, it oscillates at this frequency and produces a sound. Now, if two strings are plucked at the same time, the interaction of such frequencies creates new patterns of oscillations, and subsequently new sounds. Such sounds are dependent on how far apart the frequencies between these two notes are.

It was defined that every note with a frequency ratio of 2 (or 0.5) to a subsequent note would be denoted as an **octave**, as due to their closely related harmonics they sound extremely similar (to the human ear).For example, given a note with a frequency of 440Hz, both 880Hz and 220Hz would have the relation of **octaves** to this note.

Every octave is denoted by the same letter, accompanied by a number to denote the frequency. Considering the example above, 440Hz is denoted “A4”, whereas 880Hz and 220Hz are “A5” and “A3”, respectively.

In the Western world, the most common way to divide the range of frequencies between two octaves is in 12 equally spaced frequencies on the logarithmic scale. This frequency ratio can be written as , where is the number of notes between the two frequencies, this is called an **interval** ( can also be called the number of **semitones** between the two notes).

Figure 1 below shows **octaves** for a range of frequencies. Note that the “accidentals” (# and b) accompanying some of the notes are a matter of how the labelling system was devised, but matters not for the purpose of this report, as frequencies and ratios will be mostly used.

Table

Description automatically generated

Figure 1: Notes, their octaves, and their frequencies[[1]](#footnote-1)

For example, between C1 and D1 we have two **semitones** (), which in terms of the frequency ratio is equivalent to the ratio between G1 and A1: in both cases their ratio is .

When two or more notes are played simultaneously, we call it **harmony**. Naturally, the more frequencies played at a time, the more complex their interactions become, and the harder is it to distinguish individual notes. **Harmony** and **chords** in this context will be used interchangeably as only one instrument is considered.

Convention has set names for certain combinations of notes: one may call the chord made up of the notes C4, E4, and G4, a C Major chord, for example. However, the same name would be attributed to a chord made up of C3, E4, and G5, with notes in different octaves. The latter is called a **voicing** of C Major and is extremely important in adding flair to music, something musicians often aim to do. The importance of voicings will become clearer once the literature is discussed.

## Neural Networks

The field of Machine Learning at large is focused on developing efficient pattern recognition methods that can scale well with the size of the problem domain and of the data sets (Simeone, 2017). For this reason, many model classes, algorithms, and training methods have been devised, and are applied according to the purpose desired. One such model class is called Neural Network (Artificial Neural Network, or ANN for short). In simple terms, Neural Networks aim to extract the most useful features of the data and utilise such features to infer a result. This model is mostly used for situations in which it is difficult to determine good features of the data *a priori*. For example, if the input data is a text, the features may be determined *a priori* as a vector with the number of occurrences of given words, which can be a good way to determine the subject of the text. If the input was an image, however, the decision as to what good features are becomes less clear; and that is the main role of the Neural Network model.

Through a supervising signal, which determines whether the output of the model was correct or not, the parameters of the model (also referred to as weights) are updated to reflect such error, and hopefully yield a better result. The method with which they are updated is usually by taking the gradient of a **loss function** over a small set of randomly selected data points (called mini batch) and changing the weights according to the negative direction of such gradient (i.e., minimizing the loss). This is called Stochastic Gradient Descent.

The **loss function** is set *a priori* and depends on the nature of the problem. One such example is the detection-error loss, denoted 𝟙, which means that if prediction (i.e., correct) the loss will zero, and otherwise it will be one.

Each computational unit in a neural network is called a neuron and, except for the input neurons (denotedin Figure 2), receive a real number that is a result of the weighted sum of the outputs of the previous layer of neurons.

Diagram

Description automatically generated

Figure 2 - Neural Network model[[2]](#footnote-2)

The model in Figure 2 tackles the problem of binary classification as it can be seen by the output neuron, which denotes the probability of the target variable being equal to 1.

Consider the first layer denoted , each of the neurons is receiving as input all the outputs of the previous layer (the input layer). The arrows for each input “represent” a **weight** (a real valued number), with which the output of the previous layer will be multiplied. The neuron then sums all these inputs and passes this real valued number (denoted ) through a non-linear (activation) function (denoted ). Mathematically this can be represented as:

Where is the weight for each individual neuron-to-neuron connection.

This output will take the same role took in our example (but for the second layer), and so forth for all the layers up to . In layer , also called the classification layer, there will be only one neuron, whose activation function is a sigmoid, and will output the respective probability . Note that is the vector of all trainable parameters and symbolises that the output probability is dependent on these parameters/**weights**.

Naturally, Neural Networks take some inspiration from the brain by having interconnected neurons transmitting information, however, the similarities stop there. Such models do not take into consideration how biological neurons communicate, and neither how they process information. In an effort to explore the processes behind the world’s most power efficient computer (the brain), a new generation of Neural Networks was devised: Spiking Neural Networks (SNNs).

Notice how for each computation of the function in an ANN both multiplication and addition operations are utilised. Computationally, this is an expensive process.

Diagram, schematic

Description automatically generated

Figure 3 - ANN & SNN[[3]](#footnote-3)

The binary nature of spikes means that for any given input for the neurons, the output neuron is only required to sum the weights of the corresponding neurons that spiked (Rajendran, 2021)

## Neural Engineering Framework & Nengo

The Neural Engineering Framework (**NEF**) is a methodology that allows the construction of large-scale, biologically plausible neural models of cognition (Eliasmith & Anderson, 2003). Instead of setting connection weights between **neurons** manually or through some learning rule (such as Stochastic Gradient Descent), the NEF solves for these according to the function you desire to compute. For example, if one wants to compute , the framework will determine the connection **weights** between two ensembles of neurons that best approximates this function (the way in which it does will be discussed later). In the case where such function is not known, traditional methods can still be utilised.

*Why model according to the NEF?*

Considering that the **NEF** uses computational units based on biological neurons, any computation done by the model is forced to adhere to the basic operations that are available to neurons (Stewart, 2012). This allows insight into what sort of algorithms can and cannot be implemented in the human brain. Further, the representation of signals is done so as match the neuron ensemble behaviour seen in the brain, which is a direct effect of how neurons behave.

In 2014, Bekolay et al. developed an open-source Python package called **Nengo**[[4]](#footnote-4), implementing the NEF for building and simulating such models in a computer environment. Given the goal of this project, all the models in this report were built on **Nengo**, following the **NEF**.

## Previous Work & General Remarks

The current literature in the subject of harmony recognition is vast, but so is the complexity of the problem. Considering the several steps inherent to the problem; from the processing of the audio, to design choices of classification algorithms, to the scope of the solution, and the choice of databases, there were many unexplored aspects which were considered and utilised.

Most importantly, this model differs in its approach. That is, it is focused in utilising a biologically plausible approach for both the processing and the analysis the data, and for learning.

Fujishima (1999) presents **Pitch Class Profiles (PCP)**, a method that reduces the dimensionality of audio data in the frequency domain by restricting the representation of the signal to a 12-bin vector (i.e., one **octave**). The frequencies are effectively separated at their note-boundaries (i.e., actual frequency of the note), and the sum of their respective magnitudes is taken and assigned to a note-bin. Dixon (2003) utilises Short-time Fourier Transforms to find magnitude peaks in the magnitude spectrum and uses these as a decision criterion. Barbancho et al. obtain a probability of each note of the piano being played by applying *parallel interference cancellation* (PIC) to frequency domain signals. Tolonand & Karjalainen (2000) proposed a model with some basis on the human auditory system for multipitch analysis by separating the signals into subband channels.

Inspired by the Cochlea in the human ear, this report chose to adhere to a similar format to that of Tolonand & Karjalainen, but adapting it to range of a piano, namely with 88 bins in **PCP**-like manner ranging from 27.5Hz to 4186Hz, one for each note of the piano. This allows for a representation of the position of the notes within the instrument and addresses the issue of representing **voicings**. Dixon’s approach was also taken into consideration to pre-process the sound.

Only recently have machine learning methods (rather than relying on signal processing techniques and pattern matching), began to be used. One such research was done by Saputra et al.

# **Report**

Things I tried

What I succeeded in; and substantiate with results (and mention how significant these results are! Show why they are important) —> look at the poster for the CHD class and the questions

What I failed in and why

## Methodology

### The dataset

The UMA-piano chord database contains over 275,000 chord recordings with different levels of polyphony and playing styles. There are two main features of the database that allow it to be an efficient way to train and test the model, namely, all the recordings are from a live piano in contrast to a MIDI, which allows for those complex harmonic interactions to happen, and the range of dynamics – staccato, muted, sustained, etc., which allows for a greater fidelity to the way musicians play.

This technique is applied over an audio signal processed by the Fourier Transform, which yields a vector with 88 magnitudes, one of each note.

## Framework

For a given signal vector , the NEF uses an encoding vector to represent neuron activity of a given neuron . Assuming that the input current of the neuron is a linear function of the actual values being represented

Consequently, the learning methods within the framework are distinct from that of traditional machine learning (Stochastic Gradient Descent generally). Discuss PES learning rule

## Design

The model has two time-varying input signals, the hearing stimulus, which receives a scaled 88-dimensional vector representing a chord every 150ms, and a “correct” stimulus, which receives an 88-dimensional vector with threes in the correct position of the notes every 150ms. This allows for real-time detection of the notes once the decoder values are trained.

Due to the nature of the tuning curves of the neuron models in the NEF, which by default respond to input values between -1 and 1 (Figure 4), scaling them to respond to larger values reduces the quality of the representation. This is because the range of frequencies for the neurons are limited, and the further it is spread, the lower the resolution of the output.

Chart, diagram

Description automatically generated

Chart

Description automatically generated

Figure 4 - Neuron with Radius parameter = 1. Figure 5 - Neuron with Radius parameter = 4.

Refer to Figures 4 and 5 above, both neurons were set to have the same parameters for maximum fire rate and axis intercept. Notice how by increasing the range of input response to 4, it spreads the firing rate over a large output domain, losing resolution. The only way to fix this issue for ensembles is by increasing the number of neurons, and subsequently the computational load.

Figure 6 below shows how an actual ensemble with 20 neurons behaves in Nengo. The parameters for each neuron are set at random to provide a realistic range of responses.

Diagram

Description automatically generated

Figure 6 - Tuning Curves for an Ensemble[[5]](#footnote-5)

Considering the discussion above, the hearing input stimulus which was given by the 88-bin magnitude values of the Fourier transform was scaled with a function to restrict the values to a range between and (decimal values were set to to prevent negative numbers). This allowed for a smaller number of neurons to be used for representing the signal. Consequently, the label vector, which was initially made up of ones was scaled by 3 to match the highest magnitudes of the hearing stimulus.

Figure 7 below shows a visual representation of the model:

Diagram

Description automatically generated

Figure 7 - Learning Model

Decreasing learning rate

Reflecting seriousness of error

Both linear and sigmoid distance loss decreased as the number of training points increased. That is, the encoders were prioritising values closer to the target, and suppressing those that were further away, effectively modelling the target signal.

Findings

The base comparison for all predictions is the processed audio signal (88-dimensional vector of magnitudes). When considered against the labels, it scored 33.9% with the detection-error loss function

**often, the second note with the highest accuracy is the octave. Considering this, if the dimensionality of the data had been reduced to represent 12 notes only (one octave, as some research papers do), the total amplitude of that frequency bin would have been larger, and consequently more accurate.**

Since the model doesn’t generalise well, but does have a low training loss, the logical conclusions would be to train it with all possible combinations. The form in which this scales for a chord with n number of notes (88 choose n), however, does not make this a feasible problem, as both the computational power and time required would be significant.

On the other hand, it seems clear that there are chords played more often than others, especially in pop music. That being the case, filtering the database in such a way as to only consider chords with a probability above a certain threshold has the potential to produce an accurate model for a smaller but more likely set of outcomes; requiring significantly less computational power. This is beyond the scope of the project, but is nonetheless a direct consequence of the model and conclusions obtained.

Since the prediction is done on the processed audio (that is, on the matrix of 88 frequency bins), the structure of the data matrix can be maintained for any audio within this frequency range. In effect, the model, if trained on the full range of a piano as opposed to 30 notes, is expected to work on the harmony recognition of any piano audio.

# **Conclusion**

# **Professionalism and Responsibility**

# **Bibliography**

Eliasmith, C., & Anderson, C. H. (2003). Neural engineering: Computation, representation and dynamics in neurobiological systems. Cambridge, MA: MIT Press.

Stewart, T. C. (2012). (tech.). *A Technical Overview of the Neural Engineering Framework*.

Rajendran, B. (2021). *Brain-Inspired Computing & Hardware Design.*

Osvaldo Simeone (2017). A Brief Introduction to Machine Learning for Engineers. Available at: <https://nms.kcl.ac.uk/osvaldo.simeone/notesMLSimeone.pdf>

Heng-Tze Cheng, Yi-Hsuan Yang, Yu-Ching Lin, I-Bin Liao and H. H. Chen, (2008). "Automatic chord recognition for music classification and retrieval," 2008 IEEE International Conference on Multimedia and Expo, pp. 1505-1508, doi: 10.1109/ICME.2008.4607732.

Simon Dixon, (2003) “Extraction of Musical Performance Parameters from Audio Data”

Engineering ToolBox, (2003). *Notes, Octaves and Frequencies*. [online] Available at: <https://www.engineeringtoolbox.com/note-frequencies-d_520.html>

Takuya Fujishima (1999). Realtime Chord Recognition of Musical Sound: A System Using Common Lisp Music, ICMC Proceedings.

Barbancho et al. (2010). PIC Detector for Piano Chords. EURASIP Journal on Advances in Signal Processing. 2010. 10.1155/2010/179367.

Benetos, E., Dixon, S., Duan, Z., & Ewert, S. (2018). Automatic music transcription: An overview. IEEE Signal Processing Magazine, 36(1), 20-30. https://doi.org/10.1109/MSP.2018.2869928

T. Tolonen and M. Karjalainen, “A computationally efficient multipitch analysis model,” IEEE Transactions on Speech and Audio Processing, vol. 8, no. 6, pp. 708–716, 2000.

Saputra, F., Namyu, U. G., Vincent, ., Suhartono, D. & Gema, A. P. (2021). Automatic Piano Sheet Music Transcription with Machine Learning. *Journal of Computer Science*, *17*(3), 178-187. https://doi.org/10.3844/jcssp.2021.178.187

# **Appendix**

1. Engineering ToolBox, (2003). *Notes, Octaves and Frequencies*. [online] Available at: https://www.engineeringtoolbox.com/note-frequencies-d\_520.html. [↑](#footnote-ref-1)
2. Simeone, (2022) [↑](#footnote-ref-2)
3. Rajendran, (2021) [↑](#footnote-ref-3)
4. <https://www.nengo.ai/> [↑](#footnote-ref-4)
5. Stewart, 2012 [↑](#footnote-ref-5)