DEGREE PROJECT IN ELECTRONICS AND COMPUTER ENGINEERING,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, *SWEDEN 2020*

Categorisation of the Emotional Tone of Music using Neural Networks

A multimedia project in conjunction with Ichigoichie

JACOB HEDÉN MALM

KYLE SINCLAIR

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Abstract

Machine categorisation of the emotional content of music is an ongoing research area. Feature description and extraction for such a vague and subjective field as emotion presents a difficulty for human-designed audio processing. Research into machine categorisation of music based on genre has expanded as media companies have increased their recommendation and automation efforts, but work into categorising music based on sentiment remains lacking. We took an informed experimental method towards finding a workable solution for a multimedia company, Ichigoichie, who wished to develop a generalizable classifier on musical qualities. This consisted of first orienting ourselves within the academic literature relevant on the subject, which suggested applying spectrographic pre-processing to the sound samples, and then analyzing these visually with a convolutional neural network. To verify this method, we prototyped the model in a high level framework utilizing Python which pre-processes 10 second audio files into spectrographs and then provides these as learning data to a convolutional neural network. This network is assessed on both its categorization accuracy and its generalizability to other data sets. Our results show that the method is justifiable as a technique for providing machine categorization of music based on genre, and even provides evidence that such a method is technically feasible for commercial applications today.

Keywords

Artificial Neural Network, Emotion, Music, Audio Processing, Music Categorisation

Sammanfattning

Maskinkategorisering av känsloprofilen i musik är ett pågående forskningsområde. Traditionellt sett görs detta med algoritmer som är skräddarsydda för en viss typ av musik och kategoriseringsområde. En nackdel med detta är att det inte går att applicera sådana algoritmer på flera användningsområden, och att det krävs både god musikkunnighet och även tekniskt vetande för att lyckas utveckla sådana algoritmer.

På grund av dessa anledningar ökar stadigt mängden av forskning runt huruvida samma ändamål går att åstadkommas med hjälp av maskininlärningstekniker, och speciellt artificiella neuronnät, en delgrupp av maskininlärning.

I detta forskningsprojekt ämnade vi att fortsätta med detta forskningsområde, och i slutändan hoppas kunna besvara frågan om huruvida det går att klassificera och kategorisera musik utifrån känsloprofilen inom musiken, med hjälp av artificiella neuronnät.

Vi fann genom experimentell forskning att artificiella neuronnät är en mycket lovande teknik för klassificering av musik, och uppnådde goda resultat. Metoden som användes bestådde av spektrografisk ljudprocessering, och sedan analys av dessa spektrogram med konvolutionella neuronnät, en sorts artificiella neuronnät ämnade för visuell analys.

Nyckelord

Artificiellt Neuronnät, Känslor, Musik, Ljudprocessering, Musik Kategorisering

Acknowledgments

We offer a big thank you to the staff of Ichigoichie for the opportunity to work with them on this project and to share some good times together.

In particular we'd like to thank David Ventura and Magnus Hallin for their guidance and mentorship.

We would also like to thank Johan Montelius and Anne Håkansson for their patience and wisdom and David Gunther for his tireless feedback.

Stockholm, June 2020

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine classification of music is an area receiving significant research at time of writing. Particular attention has been paid to feature extractors that classify music based on arbitrary categories defined through for example, beat count, timbre, instrumentation or melody [3].

Less well researched is the use of generalised machine learning to categorise the emotional tone of music [4]. Emotions are vague, ambiguous and difficult to easily delimit, preventing reliable feature identification of emotional traits[4].

Of particular novelty is the application of machine learning, and more specifically artificial neural networks to the problem [5]. Neural networks have the capacity to design their own feature extractors over time in response to exposure and training to pre-classified data. Artificial neural networks are also generalizable enough to adapt their internal logic to the training data they are given. Thus the same methodology of training with different training sets allows a neural network to adapt itself to different purposes. The vagueness of emotional terms makes the task of designing feature extractors ambiguous and difficult [5]. This implies that the generalizability of neural networks could provide a useful and novel avenue into the problem.

Ichigoichie is a music game company that uses feature extraction methods to generate interactive multimedia. However, attempts to generate media effects in real time in response to difficult to define categories such as emotional content have proven difficult.

Our task has therefore revolved around developing and testing some of these various machine learning methods and structures to find an improved classification technique that can be integrated into their existing architecture. This report details our project and experimental results and is organised in the following way:

Chapter 1 (this chapter) consists of an overview of the context for the problem we are addressing. It establishes a brief background on the challenges of categorizing emotional tone of music (Section 1.1), as well as the problem faced by industry (1.2) and the purpose of our research (1.3). Our goals are made clear in section 1.4, where we discuss our research in terms of a feasibility analysis. Section 1.5 provides a brief overview of our research methodology and Section 1.6 delimits the study in recognition of the timing and other constraints faced.

The rest of this document is structured in a further 5 chapters, with each chapter introduced by a paragraph highlighting the sections it contains. The chapters in this document are: Background (Chapter 2) which provides an overview of the existing literature and approaches in the field; Method (Chapter 3), which outlines the research processes and evaluation framework used in this project; Implementation (Chapter 4), which explains the implementation of the tool following the method outlined in Chapter 3; Results and Analysis (Chapter 5), which presents our empirical findings and relates the project results back to the goals established in Chapter 1. Finally, Chapter 6 presents our conclusions, highlights limitations faced and considers avenues for future work. Some final reflections are presented before references are provided.

1.1 Background

Traditionally, music analysis has been accomplished algorithmically through the use of feature extractors that pull qualities from sound such as timbre, beat, and pitch [5]. These features can be categorised or analysed as the researcher sees fit. The company that this thesis was done at, Ichigoichie, has multiple examples of such proprietary software. However, these methods are not generalizable, and require high musical and technical knowledge in order to be successfully implemented. Furthermore, for certain tasks they prove to be difficult, due to the inherent fuzziness and subjectivity within music. Of particular focus among these issues is real time judgement of a song's emotional tone. Our project aims to determine if it is feasible to use advances in machine learning to provide such a categorization.

Recently, there has been an uptick in machine learning methodologies applied to the field of music analysis. For example, a study from 2013 compared different ML methodologies viability in extracting emotional qualities from music [6]. They found that artificial neural networks showed great promise.

Multiple other related works using neural networks were found which both demonstrated that they were a viable means of analyzing music, and which gave us a method to pursue and draw inspiration from. Our project builds on these works and extends the research by building a tool that is capable of analyzing and classifying musical inputs in real time, opening up further areas of commercial and academic application of the technique. This work was conducted together with Ichigoichie, a rhythm games company, who provided both this challenge and support to the researchers.

1.2 Problem

Ichigoichie is a rhythm game company that has developed various extractors designed to control multimedia effects in response to music. These extractors can function well for distinct key features such as beat or pitch. However, vaguer categories such as emotional content are more challenging to categorise.

In order to improve Ichigoichie's offering of games and their propietary toolkit which they offer to artists and DJs, Ichigoichie desires a tool which enables emotional tone recognition provide further multimedia responses to improvised music. The improvisational requirement means that not only must the tool be capable of accurate categorisation, it must also be able to do so

across short time frames. This is so it enables the illusion of synchronised responses to incoming audio.

This for example would allow the lighting in an installation or concert to be dynamically controlled in response to the music's emotional tone without requiring constant lighting control or a pre-programmed light show.

This paper sets out to explore if an artificial neural network can achieve reasonable success at categorising the emotional content of music in real time sufficiently fast and accurately for multimedia applications.

Scientific and engineering issues

The process of pre-processing the audio data can be computationally intensive which can make highly responsive multimedia too slow to be effective[7].

Also due to the fact that the product we develop will be used commercially, we will not be able to use certain software with more restrictive licenses.

Compounding this issue is that much multimedia developed by Ichigoichie is made through the Unity Game Engine. The engine is primarily coded in C and C sharp, with strong integrative capacity for most low level C based languages but almost no support for Python based libraries, which dominate the field of machine learning.

Furthermore, the music classification engine will need to be incorporated with the rest of the proprietary toolkit already developed by Ichigoichie, and be capable of performing live classification. This means that it needs to be able to run on embedded systems, and the live processing of music needs to be fast enough for entertainment purposes.

1.3 Purpose

Our thesis aims to categorise the emotional content of music with a tool that balances the need for accuracy with the need for a fast, real time update whose structure can be later integrated with a dominantly C Sharp based tool kit.

The primary benefit of this research will relate to entertainment and multimedia. This applied research takes place with our cooperating company, Ichigoichie. Ichigoichie is limited in its ability to make rhythm games that respond to improvised player input sounds, an area they have identified as being of value to their product offering.

Ichigoichie's commercial model is based on two pillars. The first is producing rhythm games for entertainment for consumers and the second is licensing software services to third parties, primarily artists, DJs and

museum curators for performances or installations involving light and sound. The additional categorisation capabilities will expand the potential artistic applications for these clients and thus represent a possible additional revenue stream for Ichigoichie.

The purpose of our research will be to enable Ichigoichie to make an investment decision about the technical feasibility of such a tool. We were asked to validate or invalidate an approach using machine learning in building this tool.

Ethical concerns could potentially arise if this tool were extrapolated to speech recognition which may have ramifications for political surveillance.

1.4 Goals

The main goal is to determine if it is technically feasible to build a tool that can classify the emotional content of music in real time. To achieve this the following sub-goals were established

1. Determine if advances in artificial intelligence and machine learning enable such a tool

2. Assess if artificial intelligence and machine learning provide a feasible way forward for these goals.

Requirements

These goals must be met in such a way that they satisfy the following requirements

1. Can perform categorisation on clips of audio of ten seconds duration or shorter.

2. Has operational compatibility with Ichigoichie's existing technology stack which is primarily written in Rust and C sharp

3. Must categorise music into pre-labelled categories with a success rate better than chance

4. The tool must be capable of responding with less than 5 seconds lag time in the categorisation output.

1.5 Research Methodology

Our research methodology consisted of two different phases. The first was a literature review to assess the current state of the field in using artificial intelligence to categorise music. The second phase consisted of iterative prototyping of a tool using these artificial intelligence techniques. Our literature review was constrained by the preferences of our cooperating company's preference that we assess the available methods in the field of machine learning.

The literature review began by examining potential methods of processing the raw audio data into a form that was usable with artificial neural networks. From Sahai [8], spectrographic techniques were quickly identified as the most promising of pre-processing the audio data. With this established, the literature review continued to the next goal of deciding on a neural network architecture capable of categorising these inputs. Several studies were investigated such as Zhou[4], Sahai [8], and Choi[9], detailed further below, which showed that convolutional neural networks and recurrent neural networks were possible starting points for comparative analysis.

In the second phase of our research, we iteratively prototyped a model for pre-processing the auditory data using a high-level Python framework. This was to make an initial assessment on the technical feasibility of the technique, before further investment on implementing in Ichiegochie's preferred programming languages. Once a working prototype was established, we continued to experiment with prototyping a neural network module that would interpret the pre-processed auditory data and attempt classification. This network was trained on a sample set of music files supplied by Ichiegochie. Finally, we iterated on the modules to improve both their classification accuracy, testability and to transpose them to Ichiegochie's preferred programming languages.

1.6 Delimitations

As referenced above, some decisions around implementation method were determined by the preferences of our cooperating companie. Ichigoichie's preference for a machine-learning implementation. We will not be attempting to prove that the method we develop is optimal. Our research is explicitly experimental. As a result:

- We will not be dealing with all possible types of music. We will be

dealing mainly with modern pop music.

- We will not be attempting to categorise the content of whole songs. Instead, categorisation will occur across short independent time segments.

Chapter 2

Background

This chapter provides an overview of the existing context for development and application of knowledge and techniques aimed at classifying music. This review is limited to those methods which involve our cooperating company's chosen techniques, namely neural networks to classify sound. Section 2.1 presents the context of the literature review and key concepts. Section 2.2 provides a summary of Machine Learning technologies, with subsections investing artificial neural networks (2.2.1), and convolutional neural networks (2.2.2). In Section 2.3, we introduce spectrographic processing of music, focusing on the Fourier transform (2.3.1) and Short Time Fourier Transforms (2.3.2) and the Mel transform (2.3.3). Section 2.4 presents a series of major related works that address benchmarking approaches to analysing emotional tone of music (2.4.1), the use of convolutional neural networks in recognizing emotional tone (2.4.2) and pre-processing of audio before use in a convolutional neural network (2.4.3). Finally, section 2.5 summarises these findings.

2.1 Context and key concepts

Our research indicated that much work is being done on using artificial intelligence systems, including neural networks, in the classification of music. This uptake appears to be driven largely by the success of companies such as Spotify, Google, Tidal, YouTube and other streaming services.

Particular relevance was given to discovering which structures of neural networks had been applied to the problem prior and which would be most appropriate for our particular sets of problems.

This section will explain the features needed to understand our research.

Firstly, neural networks have varied structures and properties, all of which come with their own costs and benefits. Structural variations can for example be a network wherein every neuron in a layer is connected to every neuron in the next layer (called a linear fully connected network) versus a network where connections between neurons are comparatively sparse. Examples of costs and benefits that might result from these varying structures would be the increased calculation time the more heavily connected network needs to do a full layer feed forward. The sparser network has far fewer calculations to perform at each neuron due to the reduced inputs to each node.

Structures of networks that have received particular attention in this area are convolutional neural networks (CNNs), Recurrent Neural Networks (RNNs), Long Short Term Memory networks (LSTMs) and a few relatively untested but novel Recurrent convolutional neural networks (RCNNs). Explanations for the various properties that could affect the work prior research has revealed are explored in the provided major works.

The second phenomena necessary to understand our work relates to translating the digital representation of sound into data that allows a network to properly extract features from the input. If we were to provide an image of a picturesque mountain range to a human wherein most edges had been heavily blurred, we would expect the ability of the human mind to recognise the image as a mountain range to be reduced. Conversely, stripping out colour data from a mountain range does little to reduce the ability of human eyes to recognise mountains.

What this demonstrates is that for perceptual systems built on neurons not all data from an object is equally important (at least in the human case). Instead, perceptual systems prioritise and over-rely on certain aspects of data and deprioritise others.

Relevant to our task is the human neural machinery that processes audio. Since our task involved classifying music according to subjective human emotional judgements, we needed to ensure that the data fed to the network would be analogous to how audio input to the human brain has been altered and processed by the preceding sensory machinery.

Works below thus explore how audio data can be processed to prioritise aspects of audio that human perception seems to rely on the most.

Much work in the field has focused on speech transcription, and although both neural networks and audio data processing has been done in this research it was judged as relying on features and qualities that would have little bearing on our input data.

2.2 Machine Learning technologies

Machine Learning technologies form a subset of the Artificial Intelligence field. The distinguishing characteristic that determines whether or not an algorithm is classified as machine learning is whether or not the algorithm is able to improve with experience and data. A widely accepted formalization of this definition is one stated by computer scientist Tom Mitchell: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." [10]. There are a vast array of methods that aim to provide ML algorithms with the capability to improve with experience at a given task, or learn. Roughly, these can be divided up into supervised, unsupervised, and reinforcement learning. Each of these methods is well suited to a specific class of tasks, and perhaps more importantly, a specific class of experience, or type of training data [11].

In this report, we will be singularly focused on supervised learning, due to it being well suited to classification tasks. Supervised learning is the type of learning a machine learning model undergoes when it is trained on input data where each element has a corresponding label. This means that if the model is being trained on an example of past data, that example can be mapped to a label that, ideally, the model will learn to map similar future experiences to.

In a classification problem, this label can be thought of as a class. A classic example of this is a cats vs. dogs classifier, where the training data is composed of a split of pictures of cats and dogs. Each of these pictures has a corresponding label, arbitrarily a 0 for cats and a 1 for dogs, and each time the model is trained on a dog, its output is compared to the ideal label and adjustments are made in accordance with some form of training algorithm so that in the future, the model is capable of assigning the correct class with higher probability [11].

There are multiple different algorithms for supervised classification problems Some examples are: [11]

- k-Nearest Neighbour
- Decision tree
- Random forest
- Support Vector Machine

These algorithms are generally less computationally intensive than artificial neural networks to train due to the fact that they have a comparatively small

amount of tuneable and trainable parameters. This makes them well suited to less complex classification tasks, or in situations where the training data is limited.

A comparative study [6] suggested to us that while more classical machine learning models such as the ones mentioned above had merit, more success had been achieved by the more complex structures existing within artificial neural networks for such computationally expensive tasks as recognizing emotion within music.

2.2.1 Artificial Neural Networks

Inspired by biology and with early machine learning technologies as proof of the powerful potential of algorithms that improve with experience, computer scientists sought to form an abstraction of the neuron and the network of neurons that form the mammalian brain [11].

The equivalent to the biological neuron for machine learning was termed the *perceptron* [10]. A perceptron is essentially a node in a graph. It has input edges accompanied by weights, a possible bias, an activation function, and possible output edges. As information propagates along an input channel, its value is multiplied by the weight of the edge it is flowing across, and this product term, for each input channel, is summed up, along with a bias term if there is one, to form the final input value to the perceptron. This input value is then passed into the activation function of the node, generally a function with a smooth range between zero and one. The output of this function is what propagates from the individual perceptron, and is what represents the level of activation of the biological neuron [11].

$$
\begin{aligned}
b &= bias \\
k &= numberofinputchannels \\
x &= valuepropagatingalonginputchannel \\
w &= weightofinputchannel \\
f() &= activationfunction \\
In &= \sum_{n=1}^{k} x_n * w_n + b \\
Out &= f(In)
\end{aligned}
$$

The arrangement of multiple of these perceptrons in layers, where the

layers are connected to each other in the form of edges between perceptrons, is what forms an artificial neural network. These neural networks can in theory grow incredibly large, for instance, the biological neural network consists of billions of neurons [11]. This high degree of structural complexity is what makes neural networks so capable of learning to emulate more and more complex functions, but also what makes them so difficult to understand and train.

In this report, the neural network can be described as containing an input layer, hidden layers, and an output layer. The input layer accepts a feature vector, a vector containing some form of numerical abstraction of the information we wish to perform computations on. The output layer, in the context of classification tasks, has a neuron corresponding to each possible class. An input element is predicted to belong to the corresponding class of the neuron in the output layer that activates the brightest.

Training of these networks is a two part process. A batch of feature vectors is passed to the input layer of the network and propagated through each neuron. The activations of the output layer are compared to the label the input is accompanied by (supervised training), and a loss metric is calculated. This is a measure of how far the prediction generated by the neural network is from the ideal. This measure is propagated backwards through the neural network, and by calculating the gradient of the loss function with respect to the parameters of the network, the relevant parameters of the network are adjusted in a way that follows a negative gradient in a process known as gradient descent, ideally as a way to minimize the loss function, and improve prediction accuracy [11].

2.2.2 Convolutional Neural Networks

A convolutional neural network is an adaptation of the artificial neural network that aims to be better suited to tasks relating to computer vision. This is accomplished by making the input layer to the network, and frequently some of the first hidden layers, convolutional layers [12].

A convolutional layer is a layer which accepts a multi-dimensional input, and contains a number of convolutional kernels. This kernel is typically significantly smaller than the input image.

Posit that our input is an image with pixel size 3x3

$$\begin{bmatrix} 3 & 1 & 9 \\ 4 & 3 & 7 \\ 0 & 5 & 2 \end{bmatrix}$$

and that our kernel is a 2x2 matrix:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

We begin by placing the kernel on top of the input image at index 0,0, selecting a kernel sized submatrix from the input image. The dot product of this submatrix and the kernel is then taken, and placed at index 0, 0 of the resulting output matrix.

$$3 * 1 + 1 * 0 + 4 * 0 + 3 * 1 = 6$$

After this, the kernel is moved one step, and the operation is repeated, producing the element at index 0, 1 of the output matrix. After the kernel is convolved around our input image, the output matrix will have this resulting form:

$$\begin{bmatrix} 6 & 8 \\ 9 & 5 \end{bmatrix}$$

Typically, the images are much larger than 3x3, and there are more kernels than one [12], creating multiple channels of output. Furthermore, the outputs of one convolutional layer are often fed to further convolutional layers. This method of analyzing input is well suited to visual input due to the fact that it keeps spatial trends and patterns intact, and that it allows for re-use of the tuneable parameters. When a convolutional neural network is trained, the values of the kernel are parameters that are changed to better accomplish the desired task. Ideally, this leads to each kernel selecting for and extracting some type of quality from the image, perhaps the first convolutional layer looks for specific patterns of lines present in the image, and the second convolutional layer selects for the more complicated shapes these lines or edges build. This is how a deeper neural network may be capable of greater abstraction [12]. Compared to a fully connected, or linear structure, in the case of pixel images, having one neuron directly connected to each pixel can quickly become overwhelming to train. As such, convolutions are effective ways of extracting important patterns, and discarding the noise without being unduly computationally expensive.

2.3 Spectrographic Processing of music

2.3.1 Spectrograms and the Fourier Transform

In computing sound is represented as a stream of bits. In order to apply a neural networks ability to classify images to the task of classifying sound, a method of transforming these bits into a visual form is needed.

A visual representation of audio is called a spectrogram. This spectrogram can be generated by pass filters, a wavelet transform or in this case, the Fourier transform. The Fourier transform is a mathematical method for decomposing a complex signal into many simpler sine signals that summed together reproduce the original input. The amplitudes of these component signals can be mapped on the y-axis to produce a representation of how well represented a frequency is in the original input. This represents a transformation from the time domain of a signal to the frequency domain.

This frequency density can then be mapped vertically and colour graded in order to produce a visual representation of each slice of sound. This frequency density thus describes the input signal as larger construction of many signals with different frequencies in the form of an image, where denser areas of colour at a section of the y-axis implies a greater occurrence of that frequency in the original signal.

2.3.2 Short Time Fourier Transform and Windowing

The Fourier transformation is performed over a signal's full length and shows how well represented a frequency was over the course of the entire signal[13]. This means that the visual result of a Fourier transform lacks any information about how frequencies varied over time. In practice, the task was to develop a method for producing spectrograms from an infinite feed of sound, as reflects the interactive, continuous nature of Ichigoichie's product. This involves applying the Fourier transform to small 'slices' of the input audio, producing the colour graded frequency distribution of this slice, then laying them side by side. This produces frequency densities over small slices whose summation along an axis shows changing densities across time.

This is simple enough, but it introduces issues of artifacts occurring at the boundaries of each transformed slice. Thus, a windowing method can be used to smooth out the neighbouring sections or each slice can have an overlap that helps to normalize the impact of any edge effects.

This technique is known as the Short Time Fourier Transform[13].

The developed and existing code set these respective qualities, window length and overlap, as alterable parameters. They can have a significant impact on the final spectrogram and present a possible avenue of experimentation.

2.3.3 Mel transform

Although a Fourier transform generates a representation of an audio signal, it does so by arranging all frequency density differences along the y-axis at equal intervals. This arrangement does not reflect how human perceptual machinery processes pitch intervals. [14]

Above roughly 500 Hz, increasingly large intervals in pitch are judged by listeners to produce equal pitch intervals. This biases the human auditory system towards a higher resolution of lower frequencies. [14]

Originally developed in 1937, the Mel scale (named for Melody, indicating it is based on pitch) is an empirically designed formula designed to shift frequency values towards a mapping that allocates larger intervals along the mel scale to smaller changes in frequency values at low Hz and smaller intervals on the mel scale at higher Hz. [14]

Experimental evidence for using the mel to more accurately model human hearing can be found in Steven et al. [14], Sarkar et al.[15] and more recently for the analysis of speech from Pulakka and Alku [16].

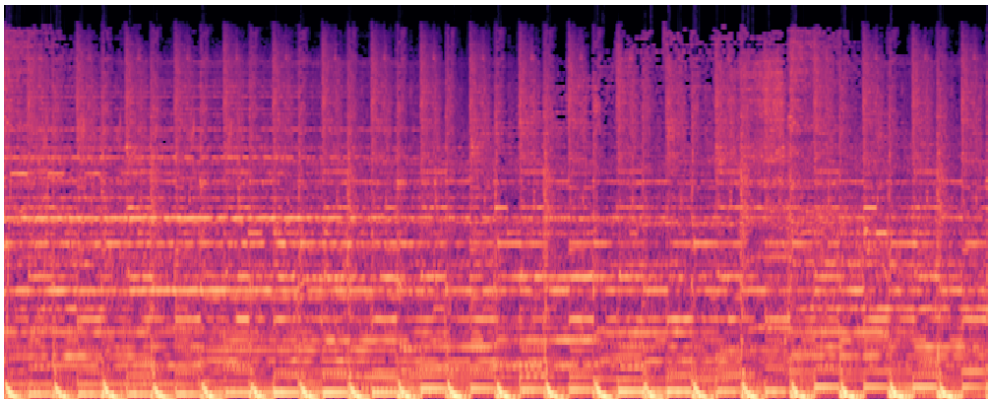

Figure 2.1: An Example of a Mel Spectrogram Generated from our Prototype

2.4 Major Related Works

2.4.1 Benchmarking different approach for analysing the emotional tone of music

Aljanaki [6] develops a common benchmark and data set for different algorithms aiming to perform a music emotion recognition task. The algorithms consisted of multiple different machine learning algorithms, there were 21 different teams taking part in benchmarking their algorithms, and the result of the study showed that artificial neural networks showed the most promise for music emotion recognition tasks[6].

Due to the similarity between music emotion recognition and music emotion classification tasks, this study provided us with an evaluation of different relevant machine learning techniques for our research project. It pointed us in the investigative direction of looking at artificial neural networks.

2.4.2 In Depth Exploration of Convolutional Neural Networks at Recognising Emotional Tone in Music

Liu et al. [5] demonstrate success in a similar task. This provided the majority of the weight behind our decision to use convolutional neural networks to solve this problem. The study sets out to map the emotional content of songs along various emotional axes. To this end a convolutional neural network was applied towards categorising spectrograms of a 1000 song labelled data. All songs were 45 seconds each. To compensate the experimenters expanded the dataset by dividing each 45 second song into 5 second chunks and using these chunks for training and validation.

This suggested to us that the mechanism employed would be sufficient to provide a network capable of only five second lag categorisation of emotional content.

2.4.3 Pre-Processing Audio before using a Convolutional Neural Network

Sarkar et al. [15] explored a method similar to our own final method. The major relevance of the work lies in its use of pre-processing data to provide methods for emulating the neural machinery of human hearing in data and demonstrates that this can improve the final categorisation accuracy.

This study used a similar convolutional neural network to our final product but provided the theoretical basis for exploring altered spectrograms to improve accuracy. This study compares several methods of altering spectrographs to mimic human perceptual systems and identifies the mel spectrum as dividing the frequency specifically into bins that represent the human ears ability of distinguish different pitches.

It was this study that suggested possible areas of exploration that our work should be capable of swapping to would come from the possible transforms that could be applied to pre-visualised data and which led us to use the mel spectrum as our first avenue of research.

2.5 Summary

In summary, one must take these points to form a general overview of the important features from our related works.

Firstly, the task of audio classification of music through AI can be solved adequately through the use of convolutional neural networks trained for image recognition

Secondly, they can perform this recognition on audio that has been transformed into a spectrogram.

Thirdly, these spectrograms can be formed in variety of ways but previous research has suggested the efficacy of the Fourier Transform in producing spectrograms that capture sufficient features from audio as to allow classification to occur.

Finally, the human auditory system's own audio pre-processing can be mimicked using transforms or filters, of which the mel filter has had a particular focus in the existing literature.

Chapter 3

Method

This chapter details the methods used to implement both the prototype and final models used to generate the results of this project. In Section 3.1, we detail the assumptions used in our research process and how this has informed the decisions taken on how to build and construct a prototype for classifying the emotional content of music (Section 3.2). In Section 3.3, we discuss how the design of the data set that will be used to train the neural network, while the details of the data (provided as validation sets) is provided in section 3.4, Data Collection. Due to the commercial applications of our project, we were required to go beyond prototyping and commence building a commercial application, as detailed in Section 3.5 and experimentally tested according to the design explained in Section 3.6. The final model and the project as a whole is understood and evaluated according the evaluation framework detailed in Section 3.7.

3.1 Research Process

We began the research process with the literature review. This was done as a way for us to orient ourselves with respect to the research that had already been conducted, and gain insight into a potential way to tackle the problem we were faced with.

This research study was started by entering combinations of relevant terms into academic search engines, the two primary search engines used were Google Scholar, and the Stockholm Royal Institute of Technology library. Examples of terms that were deemed relevant are:

- Machine learning

- Music classification
- Music analysis
- Artificial neural networks
- Emotion recognition
- Convolutional neural networks

The most significant works are included in our collection of major works in Section 2.4.

As the literature study progressed, the terms researched became more specific. This reflected the researchers narrowing in on a few relevant concepts. Furthermore, as a general implementation model was chosen, the material studied moved away from being primarily academic studies, and towards textbooks. This was to better understand the theory behind the concepts mentioned in the studies, so we could implement the concepts ourselves.

The literature review helped us to determine a framework that could be used to develop a prototype model. We decided to prototype the model in a high-level Python framework. Although licensing concerns and speed issues would require us to eventually move away from these licensed platforms and the Python interpreter, it provided initial evidence suggesting our method was worth pursuing.

3.2 Rapid prototyping

Given the time constraints, and the fact that a working integration of our prototype, were it successful, would eventually be necessary, more theoretical exploration was judged to be less appropriate than a quick empirical investigation.

Our initial prototype was to be built using well documented, user friendly, and robust tools that have received considerable investment and research.

Given the need for Ichigoichie to rapidly assess if this method held sufficient promise to warrant research, we used an experimental rapid prototyping method. Our goal was to confirm that research into this method was warranted, and thus an uncomplicated straightforward attempt to solve the problem with basic tools was our first step.

As was demonstrated by Aljanaki [6], the most promising methods were deep neural networks. As shown by Liu [5], a successful method of performing

classification tasks on music using neural networks involved the pre-processing steps of visually representing the music as spectrograms, and then using convolutional neural networks, a well proven computer vision tool, to analyze and learn relevant features of our data set.

It was decided early on that the actual architecture of the model was to be experimental. This meant that we would not try to build a theoretically ideal architecture from the start, as this was deemed to be much too difficult. Instead, we opted to try to get a standard type of convolutional neural network working for our task, whereupon we could experimentally tweak the model architecture and pre-processing steps and benchmarking the performance of the model before and after each tweak.

Inspiration for the starting convolutional neural network model architecture was drawn from a convolutional neural network classifier that had proven to work well on a cats vs. dogs classification data set provided by Kaggle [17].

This allowed us to assess the fundamental theoretical goal, proving that a neural network could perform the classification, without being concerned that the novel tools used by Ichigoichie were producing artifacts that produced a false negative.

Future researchers interested in the problem with more time or less constraints on tools used should consider more specialised networks for initial testing and a more thorough exploration of feature extraction in the creation of spectrograms. These are theoretically dense tasks and the need for a rapid prototype precluded them.

3.3 Data set

The data set that the model was trained on, and consequently, the features the model was meant to learn was proprietary, and defined by the company we were doing this thesis in conjunction with, Ichigoichie. We were handed a data set consisting of roughly 2000 songs and snippets of songs organized along two axes. The first axis was positive-neutral-negative, and the second axis was hard-neutral-soft.

We decided that in the beginning, we would abstain from including neutral as a possible category, as we wanted the distinction between labels to be as clear as possible. We also decided to focus our investigation on the positive-negative axis, as this contained more than three times as much trainable data than the hard-soft axis, and we assumed that more training data would lead to a greater ability to generalize by the model due to the training set having a larger probability to be a truly representative sample as size of data set increases.

The disparity in size arose from the fact that along the hard-soft axis, there was a massive imbalance, the majority of songs were considered to be hard. In contrast, along the positive-negative axis, there was a relatively even split amongst the songs. Due to the fact that when training our model, we wanted the training data to equally represent each possible label so that the model would not learn to exhibit bias towards the label with the lions share of the training data. This means that the effective size of the training data was limited by the category with fewest samples.

3.4 Data Collection

The benchmark we designed for the final testing of the model consisted of three different measurements.

The first was prediction accuracy on the training data set. This measurement is a percentage in the form of correct predictions out of total predictions on the training set.

We then designed two separate validation sets. For both validation sets, all of the given data set was first pre-processed. This consisted of discarding all sound samples that were less than ten seconds in length, and for all of the remaining sound samples, spectrograms were generated in ten second chunks. For example, if a given sound sample was 22 seconds long, it resulted in two spectrograms consisting of the first and the second 10 second chunk respectively, with the remaining two seconds being discarded. Then, to ensure equal representation of both labels, the class containing more samples was shortened to contain exactly the same amount of samples as the smaller data set.

In the above manner, all of the positive and negative sound samples were processed independently. This resulted in two directories containing spectrograms representing the two labels, positive and negative.

The first validation sample, hereafter referred to as VS1, was produced by randomly selecting ten percent of all images from these folders, and separating them before training began.

The second validation sample, hereafter referred to as VS2, was produced by, again, pre-processing all of the given data set in identical manner as when the first set was produced, and again, selecting approximately 10 percent of the total data set. However, this time making sure that the validation set did not contain any spectrograms that shared a common root with any spectrograms in the training set. In this context, root refers to the song the spectrograms were produced from.

When using either validation set, the training data consists of all of the songs that were not selected for assessing the validation accuracy.

Validation accuracy corresponds to the percentage of correct predictions generated by the model out of total predictions generated by the model, on either validation sets.

Sample Size

The final sample size of each classification, before separating the samples into validation sets, consisted of 688 spectrograms.

3.5 Steps taken after prototyping

The industry partner, Ichigoichie, uses the game development platform Unity. The coding for Unity is dominantly C sharp. Thus, the products developed by Ichigoichie must all be compatible and callable by this platform.

In keeping with the need for compatibility and speed, a large portion of the tools Ichigocichie currently uses for musical analysis are written in Rust and accessed using wrappers from inside Unity. Given this and the need to move away from licensed tools, the prototype's structure was re-created using PyTorch's C++ API and the audio processing was implemented by reusing existing musical analysis tools present in the already developed Ichigoichie toolkit, written in Rust.

The structure of the network and the nature of pre-processing was kept the same during this transition. This allowed us to be fairly certain that any difference in results was not due to a novel structure of the neural network or alterations in the audio processing but rather in how the tools had been used to reconstruct the successful blueprints we had used in the prototype during the rebuild into Rust and C++.

3.6 Experimental design

For assessing the success of the prototyped model, we only looked at the validation accuracy the model achieved on VS1 after each training epoch. This was because we concluded that this was a metric that, while not being without its faults, showed both that the model could learn features from the training data set, and that the model was capable of generalizing these features to an external data set, to some degree. This validation accuracy was assessed after

each training epoch, or after each run through of the entire training data set. We set number of epochs to be three, as this allowed us to notice patterns in the model's change in accuracy without taking up too much time for training on a piece of software that was never going to see the light of day.

For the final model built using PyTorch, we benchmarked its accuracy on all three metrics, training accuracy, validation accuracy on VS1 and validation accuracy on VS2. These three metrics were sampled after each epoch of training, and the number of epochs was experimentally selected to be large enough that any more epochs would not improve the accuracy of the model.

Software to be used

For prototyping, Keras was used to design the model. Keras is a high level Tensorflow API written for Python that facilitates easy development and rapid deployment of neural network models. Librosa was used for pre-processing the sound files. Librosa is a sound analysis library with an open source license, that had the functions we used already prepared.

For both the prototype and the finished product, OpenCV was used for image standardization and normalization. The Python and c++ front end respectively.

When translating and expanding the prototype into a format that was integratable, Libtorch was chosen as the framework used to build and deploy the neural network model. Libtorch is the c++ front end of Pytorch, a well documented Python framework used for neural network construction.

The pre-processing was made to build on the existing feature extractors and algorithms present in the Ichigoichie toolkit. This toolkit was built in Rust, so by necessity, as was our pre-processing.

3.7 Evaluation Framework

In order to assess the performance of the tool on the task two values were measured. The first is the categorization accuracy of the tool, and the second is a generalizability measure comparing validation sets VS1 and VS2. This ensures that the tool is both accurate and generalizable to additional contexts, key determinants of considering the tool viable for Ichigoichie.

The efficacy of the tool will be judged by looking at the categorisation accuracy that the tool achieves against the validation data sets, VS1 and VS2. The categorization accuracy is expressed as a percentage, derived from the number of members of the validation set that tool correctly categorised. In

order for the tool to be declared feasible (and enable Ichigoichie to make their investment decision in this avenue of work) the accuracy was judged to be necessarily higher than 70 percent. A merely functional benchmark would be that the tool correctly categorizes inputs at a rate better than pure chance, i.e.: with a categorization accuracy of above 50 percent. A value between 50 percent and 75 percent would mean that further research would be warranted before discarding or validating this approach to categorization.

Categorisation Accuracy(percentage)	Judgement
0-50	Worse than Chance
50-75	Slightly Better than chance
75-100	Much Better than Chance

Table 3.1: Categorisation Accuracy Rankings

To aid with determining the generalizability of the categorization measurements, the ratio between the validation accuracy and the training accuracy will be used. A ratio close to one represents a highly generalizable tool. This means that the tool would be expected to produce similar results to its accuracy rating on novel data sets.

Generalizability (Validity Accuracy / Training Accuracy)	Judgement
0-50	Low
50-75	Moderate
75-100	High

Table 3.2: Generalizability Rating

To determine what action Ichigoichie should take, and conclusively answer this project's core goal of determining the viability of the categorization tool for business applications, we can refer to figure 3.3 which maps the feasibility evaluations for the possible outcomes. This is a mapping of recommendations which compares the accuracy performance with the tool's generalizability and recommends which actions Ichigoichie should take and identifies the circumstances in which the tool is feasible.

Generalizability	*Worse than Chance*	*Slightly better than Chance*	*Much better than Chance*
Low	Not Feasible	Not Feasible	Not Feasible
Medium	Not Feasible	Continue Research	Feasible
High	Continue Research	Continue Research	Feasible

Table 3.3: Feasibility Judgements based on Final Measurements

Chapter 4

Implementation

This chapter explains the implementation of the prototype and the final models. In Section 4.1, we present the Software Overview which provides a modular description of the tools functional components and their properties. Section 4.2 shows how the model was constructed, including specific code fragments. We discuss how the model was trained in Section 4.3.

4.1 Software Overview

The project, both the prototype version, and the version that is integratable with the proprietary software provided by Ichigoichie, can be described as having two distinct parts. The first part was the pre-processing operations on the raw sound input, designed to generate a collection of spectrograms that could be saved and then used to train the neural network. The second part was the actual implementation of the neural network.

This second part could be further broken down into the packaging of the spectrogram images into a format that could be fed into a neural network, the actual implementation of the neural network, and the training of the neural network.

A demonstration of the operational logic present in the software is illustrated in figure 4.1. We start at the bubble labeled as input sound file, corresponding to the raw unprocessed audio data. If this audio data is less than 10 seconds in length, we discard the sound file. If it is not, we split the sound file at the 10 second mark, and begin processing the first 10 second snippet. The rest of the sound file is again evaluated as raw audio data.

A mel spectrogram representation is then generated of the 10 second snippet, and saved as an image. After all of the data has been processed and

saved, and if need be, split into training and validation sets, the data is read into memory. Each image is resized to have a standardized size to facilitate input into the neural network, and labeled with the class it represents.

These images are then fed into the neural network in batches, and the parameters of the neural network are tweaked to facilitate better future performance. At the end, the neural network model is benchmarked in order to provide insight into how well it meets the research goals and requirements.

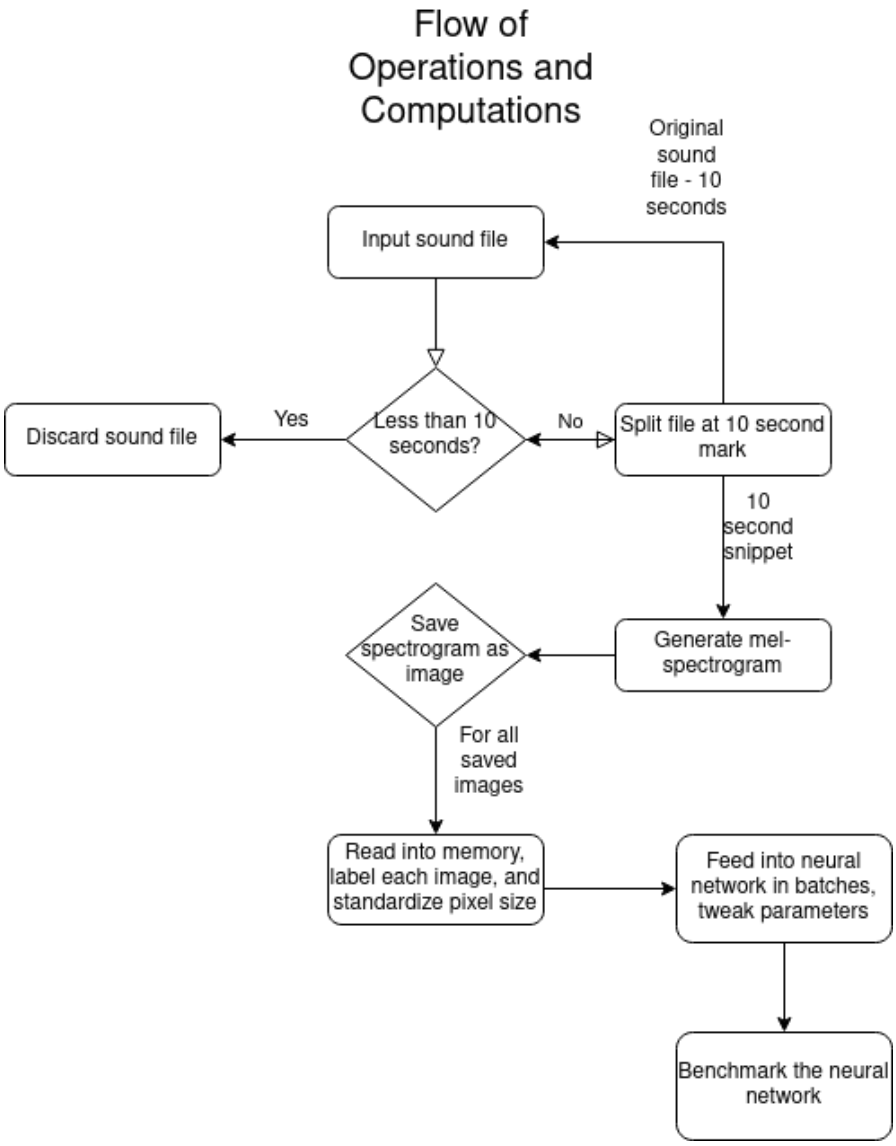

Figure 4.1: Software Overview

4.2 Audio Processing

Firstly, the audio processing was done by applying a mel spectrogram transform to ten second long audio chunks. This spectrogram was then plotted and saved as an image using matplotlib. These functions are all natively present in the Librosa Python library.

The function createSpect() takes an audio file, titled as audio_snippet and the f_name which describes how the resulting spectrogram shall be named and saved.

Of note for the reader is the parameters applied inside the mel spectrogram function in line 2. "n_fft = 2048" on line 4 defines the number of samples (each bit in the audio stream) over which the Fast Fourier Transform is to be applied and the hop length defines how much each transform 'slides' along the stream of bits. The Fourier Transform is highly optimised for powers of 2 and the small hop length helps to smooth out possible edge effects but these were ultimately arbitrary choices and their alteration represents avenues of future testing.

Listing 4.1: Spectrogram module

```
1 def createSpect(audio_snippet, fname):
2     spect=Librosa.feature.melspectrogram(
                        audio_snippet=audio_snippet,
3                                           sample_rate
                                               =44100,
4                                           n_fft=2048,
5                                           hop_length=512)
6
7
8     canvas.print_figure(fname + ".png")
```

4.3 Model Construction

There were two parts to the actual model architecture. A convolutional section of the network, and a fully connected, linear section.

The convolutional section consisted of two convolutional layers, each with a corresponding activation layer, and a subsampling layer. In figure 4.2 an illustration of how convolutional networks are commonly comprised is demonstrated. The model implemented in this project took on a similar form,

with the exception of each convolutional layer also having an accompanying activation layer.

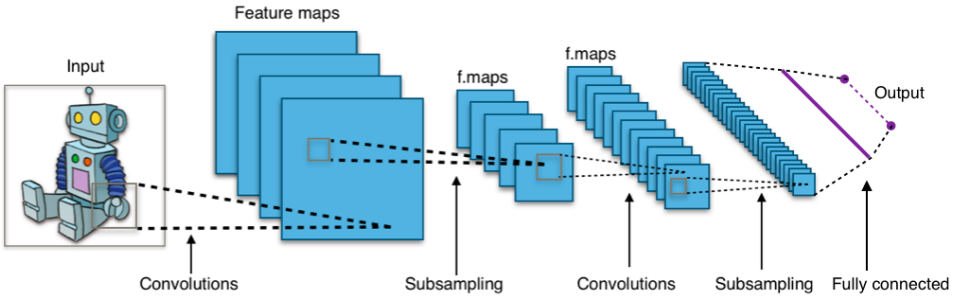

Figure 4.2: Typical convolutional neural network - Sourced from [1]

As was described in section 2.1.2, the output of a convolutional layer is one or more feature maps. Each convolutional layer was made up of separate kernels that convolved around the input image and produced an individual feature map.

The activation function used in the activation layers was a rectifier. This was chosen to be the activation function because it is commonly used with deep convolutional neural networks [12]. The rectifier is defined as follows:

$$f(x) = max(0, x)$$

The activation layer takes as input the layer before it, and applies the activation function onto the input nodes to obtain an output[12]. The purpose of the activation function is to introduce an element of non-linearity to the network, allowing it to approximate more complex functions.

This output is then fed into a subsampling layer. Subsampling is a way of further distilling the information being processed with respect to spatial locality. Max pooling, which is a variation of subsampling, looks at an arbitrarily sized subgrid of the feature map, and chooses the maximum value out of this subgrid to occupy the corresponding index in the output. As such, its output is a grid that is necessarily smaller if any subgrid other than the identity, 1x1, is chosen. Like in the case of convolutions, these subgrids are typically much smaller than the feature map itself [12].

If the input matrix to a max pooling operation is chosen as in figure 4.3, and we are performing the operation with 2x2 subgrids, we will firstly be selecting the max value out of 4, 9, 1, ad 8. Thus, 9 is selected as the first value of the output, represented in figure 4.4. This operation continues for all 2x2 sections of the input matrix.

$$\begin{bmatrix} 4 & 9 & 2 & 6 \\ 1 & 8 & 3 & 7 \\ 6 & 5 & 2 & 9 \\ 8 & 2 & 3 & 6 \end{bmatrix}$$

Figure 4.3: Max pooling input

$$\begin{bmatrix} 9 & 7 \\ 8 & 9 \end{bmatrix}$$

Figure 4.4: Resultant matrix

As stated, our model had two convolutional layers, each followed by an activation layer and a max pooling layer, the ouput of these three conjoined parts was 64 feature maps. This number was chosen arbitrarily, and further investigation may yield a more optimal choice.
In Keras, a convolutional layer was defined as follows:

Listing 4.2: Convolutional layer definition

```
1 model.add(Conv2D(64, (3, 3), input_shape =
      feature.shape[1:]))
2 model.add(Activation("relu"))
3 model.add(MaxPooling2D(pool_size = (2, 2)))
```

In libtorch a convolutional layer was implemented like this:

Listing 4.3: convolutional layer definition

```
1 torch::nn::Sequential layer1{
2     torch::nn::Conv2d(torch::nn::Conv2dOptions
          (3, 64, 3).stride(1).padding(1),
3     torch::nn::ReLU(),
4     torch::nn::MaxPool2d(torch::nn::
          MaxPool2dOptions({2, 2}).stride(2))
5     };
```

The second part of the network was comprised of a fully connected linear layer, and an output layer. An illustration of how such layers may work together is presented in 4.5. The model we defined contained one hidden linear layer which the convolutional section was connected to, and the output layer of the network, which contained two output nodes, one for each label.

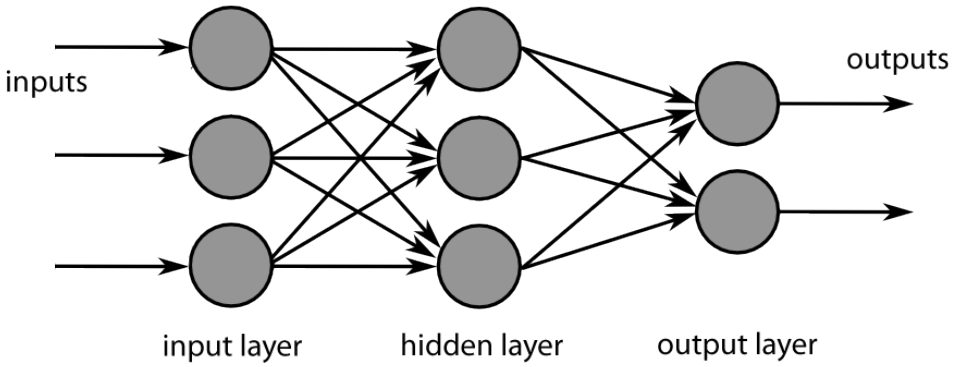

Figure 4.5: Fully Connected Neural Network - Sourced from [2]

The input size of the hidden layer was made to match the size of the output of the second convolutional layer. In order to connect these two layers of different dimensionality, a linear layer accepts a one dimensional vector whereas the output of the convolutional layer consists of multiple two dimensional feature maps, the output of the convolutional layer needed to be flattened. This meant that the columns of the feature maps were stacked on top of each other to form a one dimensional input.

Using Keras this was accomplished like so:

Listing 4.4: Connecting the two parts of the network

```
model.add(Flatten())
```

And using Libtorch:

Listing 4.5: Connecting the two parts of the network

```
x.view({-1, x.size(1) * x.size(2) * x.size(3)})
```

In listing 4.5, x is the vector flowing through the network, and the argument given to x.size() specifies the dimension. We have 2 extra dimensions because in practice we are not passing a single vector through the network, rather a batch of vectors. The -1 is a stand in for a variably sized batch. The second dimension is for color channels, since the images are RGB valued, there are three channels, and the last two dimension refer to the height and width of the image.

4.4 Training of the model

The training of the model was accomplished using standard loss functions and optimizers, provided by the two frameworks we were working with. As

suggested by the Keras documentation, the loss function used when training a binary classifier was binary crossentropy. The optimizer settled on was adam, an extension to stochastic gradient descent. These were selected based on observations from similar previous works, and proved to be good choices [17].

Upon more extensive testing after completion of the c++ version of the software product, it was experimentally found that stochastic gradient descent yielded higher training and validation accuracy, so this optimizer was adopted in the final model implementation.

Chapter 5

Results and Analysis

5.1 Major results

Our goal is to determine if it is technically feasible to build a tool that can classify the emotional content of music in real time based on the evaluation framework presented in 3.7. This chapter works sequentially through our empirical results before addressing the goals directly. Section 5.2 presents the results of the full model judged on its accuracy measures. Section 5.3 assesses the model on its generalizability scores. Section 5.4 combines the accuracy and assessibility scores into an overall assessment of feasibility. In section 5.5 we discuss several important factors that affect our result, including reliability, windowing, overfit and the role of risk in making business recommendations. Finally, a short summary of the chapter is provided.

5.2 Accuracy Measures

An accuracy measure is defined as the number of successful categorisations the tool made on data to which it had never been exposed, expressed as a percentage. For Figure 5.2 and Figure 5.3, the red line indicates the performance of the tool when supplied data that it had never been exposed to. The blue line represents its success rate when exposed to repeated data for the purpose of training.

For the first data set, selected as described in 3.4, the tool produced an accuracy rate of 85.48 percent off a sample of 124 members of the validity set. Figure 5.2 shows that the tool had an approximately linear improvement in validation accuracy (Red line) until plateauing at roughly 85 percent after epoch 10.

The training accuracy (blue line) demonstrated a similar behaviour but plateaued at a higher accuracy of almost 100 percent by the final epoch.

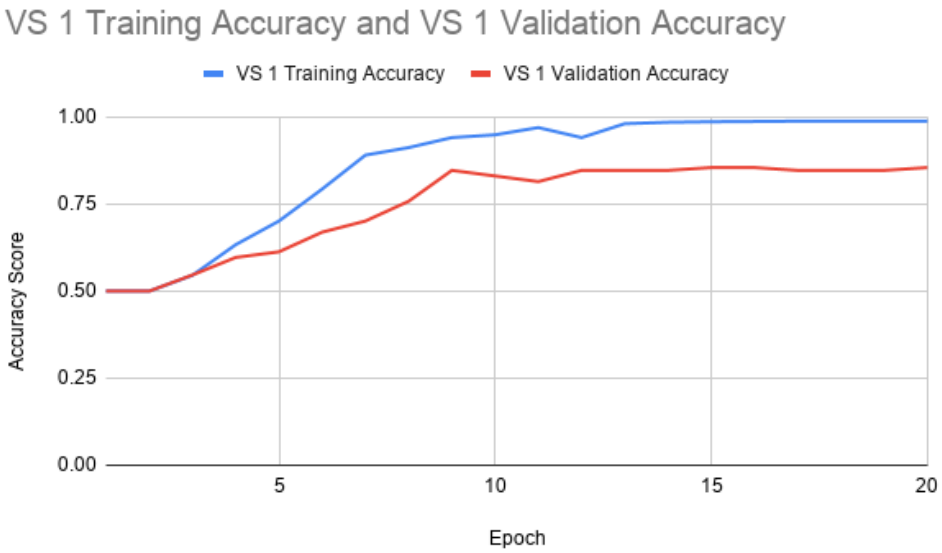

Figure 5.1: Validation set 1 results

For the second data set, selected as described in 3.4, the tool produced an accuracy rate of 62.27 percent off a sample of 89 members of the validity set.

Figure 5.3 shows that the tool produced an accuracy rating of 62 percent (red line). This accuracy rating was considerably more volatile than the linear progression seen in Figure 5.2.

The training accuracy (blue line) demonstrates a linear improvement that reaches a peak of 98 percent by the final epoch but does not plateau before reaching this point.

This contrasts with rapid of plateauing of the training accuracy in the first data set.

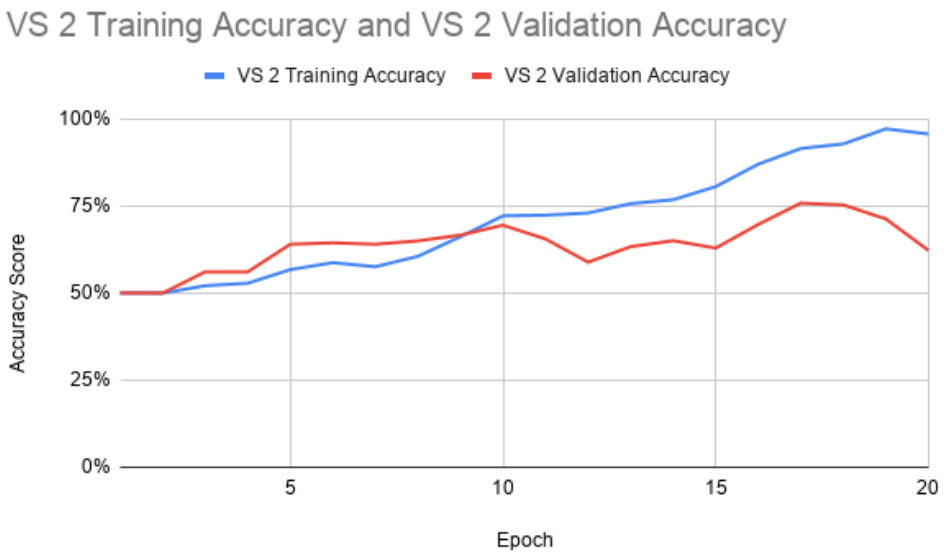

Figure 5.2: Validation set 2 results

Both models had an accuracy rating that is better than chance, with Figure 5.2 showing a much greater than chance result.

According to the evaluation metric presented in 3.7 this would suggest that the tool is at least slightly better than chance, with the larger initial data set having a much better than chance accuracy rate.

5.3 Generalizability measures

As per 3.7, generalizability is defined as the ratio between training accuracy and validation accuracy, where a higher value represents greater generalizability.

Table 5.1 shows the generalizability scores of the two models. This is taken by looking at the training accuracy (red lines) and validation accuracy (blue lines) shown in Figures 5.2 and 5.3. for data sets 1 and 2 respectively. Their raw values are recorded for transparency. In the rightmost column the generalizability score is presented. The generalizability score is derived by dividing the Validation accuracy of the model with its Training accuracy.

As per the evaluation framework established in 3.7, both models at least a moderate level of generalizability. This suggests at least a moderate level of confidence that the models would correctly categorize novel data. In particular, the model built with data set 1 shows a high level of generalizability.

Data set	Training Accuracy	Validation Accuracy	Generalizability
1	0.988019	0.854839	0.8652
2	0.918478	0.6227257	0.678

Table 5.1: Generalizability Rating of the Two Trained Models

5.4 Results in relation to project goals

This project established aimed to achieve its overall goal by reaching the following sub-goals:

1. Determine if advances in artificial intelligence and machine learning enable such a tool for categorization of emotional tone in music

2. Assess if artificial intelligence and machine learning provide a feasible way forward for these goals.

The results presented under 5.1, accuracy measures, show that the models produced are capable of categorizing the emotional tone of music with a rate that is better than chance. In this regard, the project has successfully demonstrated completion of goal 1.

To determine the feasibility of artifical intelligence and machine learning for Ichigoichie's businss applications, we must refer to the evaluation framework detailed in 3.3.

Taking each of the data sets in turn and applying the framework presented in 3.3 the following feasibility recommendations are made:

Data Set	Accuracy Rate	Generalizability	Feasibility Assessment
Data Set 1	Much Better than Chance	High Generalizability	Feasible
Data Set 2	Slightly Better than Chance	Medium Generalizability	Continue Research

Table 5.2: Feasibility Assessment based on the Two Models

In this regard, our results on goal 2, show that the project is at least worthy of further research, and based on the results from data set 1, can already be considered Feasible for investment.

5.5 Discussion

The results above represent the experimental results achieved during the project. In this discussion, we would like to raise four areas of concern where these results may differ for others pursuing similar projects. This is in the interests of transparency and reproducibility of our results. The four areas are: the reliability of the results produced by the neural networks; the windowing techniques implemented; The implications of model overfit; and finally the limitations on our recommendations in light of business risk.

5.5.1 Reliability

Assessing the reliability of neural networks is difficult. Firstly, the training of neural networks always involves a degree of randomness. The same structure of network trained on the same data set as another is not guaranteed to achieve a similar level of accuracy. This is because of the random initialization of weights that a neural network starts with, and due to the fact that the gradient descent method takes steps that are randomly sized. In recognition of this, we averaged the validation and training accuracies from two different neural networks trained with the same data set to arrive at the values presented in Figures 5.1 and 5.2. This provides a general average around which our structure of network will tend to align but does not guarantee that a trained network with an identical structure and an identical data set would reach the same accuracy rating.

5.5.2 Windowing

Several of the tools we used, such as the Short Time Fourier Transform and Mel Spectrum filters are vulnerable to windowing and edge effects. These windowing parameters that define how much overlap between short time Fourier transform frame taken on an audio stream would require testing to see if the they introduce possible compromising artifacts.

5.5.3 Overfit

Regarding VS1, due to the possibility of the same songs contributing to both the validation set and the training set, the validation accuracy derived from VS1 may be inflated. Because of the same reason, it may be difficult to detect overfitting. This could mean the generalizability is inflated, reducing the applicability of the tool.

5.5.4 Recommendations and Risk

Due to differences in the recommendations on feasibility between data set 1 and data set 2 (see 5.2) it is difficult to give a concrete, unequivocable recommendation on feasibility. What this project has shown is that there is merit in pursuing this approach and the method of using neural networks to achieve this goal appears feasible. However, any ultimate business decision lies with the level of risk a company is willing to accept. These results can at best assist in making an informed decision, they are not a decision in themselves.

5.6 Summary

This chapter has presented the results of our empirical research. We presented our results in terms of their accuracy and generalizaibility. When taken together these two measures have informed an assessment of feasibility. Overall, the results demonstrate a high generalizability and accuracy on data set 1 and a medium generalizability and accuracy on data set 2. Based on these recommendations we believe that the tool is at the very least worthy of continued research, with data set 1 showing particular promise.

We also presented some considerations of factors that could influence our results, particularly in relation to the tool's reliability, issues with windowing, overfit and general risk.

Chapter 6

Conclusion and Future Work

This chapter (6) is concerned with our conclusions. Section 6.1 presents the conclusions we make on the project as a whole, while also reviewing the results of the tool built in relation to the goals set for it. In 6.2 we present some of the limitations of our findings and core challenges that could be faced in applying this work more broadly. We also discuss some contextual challenges that arose in working on the project that may have affected our results. Section 6.3 provides two promising avenues for further work: audio transforms (6.3.1) and novel network structures (6.3.2). Finally, this project concludes (Secetion 6.4) with reflections on the implications of this work on the academy, commercial applications and broader social-political concerns.

6.1 Conclusions

This paper set out to determine the feasibility of machine categorisation of the emotional tone of music. From our literature review we established that the traditional approach to this problem has been through using specific feature extractors. We set out to expand the academy's knowledge by looking at using specific machine learning tools, due to the preference of our co-operating company Ichigoichie.

In this way, this project represents both a contribution to the academy's knowledge and a technical feasibility study for commercial applications.

As such our evaluation framework needed to address the scientific rigors of determining what knowledge was created and the business imperatives of what decisions to make.

To bridge this gap we settled on an experimental research method that saw us embedded in Ichigoichie for 8 weeks where our approaches were

validated from a commercial logic as the same time validated by the academics requirements of the project.

The result is an analysis of two data sets run through two separate iterations of our custom built tool. When assessing the results of these data sets for their technical feasibility of this research, this project concludes that the tool that was built, the method of its implementation and the way it was evaluated provides both the academy with new knowledge and our commercial partner with justifiable feasibility recommendations.

To this extent, we are satisfied that the project has met its goal of determining if it is technically feasible to build a tool that can classify the emotional content of music in real time using machine learning techniques.

During this project several insights have been gained of note for further researchers.

1. The developed tool was designed to be transposable to the lower level languages that are dominantly used by Ichigoichie. These lower level languages (Rust, C Sharp and C++) have substantially less well developed support for neural networks. The prototyping used high level Python frameworks, which have seen significant investment and research. Researchers should be aware that transitioning out of these frameworks into lower level languages can be a significant technical challenge.

2. Although the tool was developed with techniques applied from the field of artificial intelligence, our relative lack of signal and audio processing knowledge hampered our initial work. It is important not to underestimate the needed knowledge from a field when applying machine learning techniques to it from the outside.

Much of what we would do differently if given the chance is related to unfortunate current global circumstances. Given the 2020 Coronavirus pandemic lockdown, very little work and introduction could occur in proximity and it is important not to underestimate how many initial difficulties face to face work can overcome. In particular, establishing and clarifying expectations and requirements over video chats is a skill separate from normal conversation that needs to be trained.

6.2 Limitations

The largest limitation we had was the data set. We believe that with a larger data set containing more unique songs, a model capable of generalization and commercial use could be developed and trained.

We can't confirm given our limited dataset that other axes of categorisation would be equally well recognised by our tool.

Song entries were fundamentally arbitrary, selected as they were by a partner of Ichigoichie by hand and based on their subjective judgement. Songs were of different length and sample rate. There were separate distinct issues surrounding the two validation sets. Regarding VS1, due to the possibility of the same songs contributing to both the validation set and the training set, the validation accuracy derived from VS1 may be inflated. Because of the same reason, it may be difficult to detect overfitting.

However, due to VS1 being randomly selected from the entire pool of data, statistically it is more likely to be an accurate representative sample of the training data. Similarly, in a real world scenario,it is possible that there will be overlap between training set and samples the model actually will be used to classify. Further, even though there is overlap between the songs that contribute to both the training and validation sets, there is no overlap between the actual data points used. When there is overlap, different 10 second splits are used for training and validation. We also do not have a quantified measurement of which percentage of the validation set overlaps with the training set. This is a possible point of further investigation.

VS2 had other issues. The main issue with VS2 is that due to us having to make sure that all of the came from disjoint sets of songs, the number of songs that went into creating the validation set was rather small, roughly 20 unique samples. Statistically, this means that VS2 is likely not a representative sample of the greater data set supplied. Further, it also means that we expect large, discrete somewhat arbitrary changes in accuracy when testing on this validation set. This is because each song that makes up the spectrograms in the validation set corresponds to a large proportion of these spectrograms.

If the model learns to predict one song correctly that it previously did not predict correctly, due to that song likely corresponding to somewhere between 5-10 percent of the total spectrograms in the validation set, this minor improvement will lead to a significant jump in validation accuracy.This combined with the fact that the likelihood of each individual song being non-representative of the label (happy-sad) it is trying to test for(variance increases as sample size decreases, if we assume that we can model the distance each

song is oriented from being quintessentially hard or soft with some type of stochastic variable) means that it is possible that a very good model may not score amazingly on this validation set. To add on to this, it is hard to quantitatively reason about how representative each song is of the subjective quality of how representative a song is of being quintessentially hard/soft, due to the fact that it is subjective and hand picked by a person

In addition to the above limitations related to the data sets, other areas of limitation we encountered were that we were unable to perform comparative time benchmarking to gain greater insight into the inner workings of the system within the time allotted for the project. This means that we could not make a conclusion whether that the software is fast enough to use for live classification, as needed by requirement 4.

As mentioned, time as always was also an issue, and the usual constraints of limited time were exacerbated by the additional pressures of managing life and work during the corona virus pandemic. In particular, the limited face-to-face time to collaborate may have negatively impacted our overall speed and focus, and this may ultimately impact on the project's outcomes.

Finally, the necessity of working with a commercial company limited the kinds of software we could use, as our deliverables needed to be compatible with Ichigoichie's existing code base, and we were restricted in some of our architectural choices given the realities and commercial restrictions related to licensing and maintainability.

6.3 Future Work

We have divided opportunities for further work along lines similar to our modular design. This means our initial commentary is related to the area of audio processing and spectrography and the second section is related to possible alterations to the neural network's structures

6.3.1 Audio Transforms

Of interest for future work is novel research into mimicking the human auditory system's processing of audio, which in theory could lead to better feature extraction and thus spectrograms that better prioritise the same information as the human auditory system.

This work has discussed prior the research into human pre-processing of sound and pointed out that mel spectrum's log behaviour has been designed and adjusted to shift audio data into a manner where the resolution applied to

particular frequencies more closely resembles the self-reported sensitivity of the human ear to differences in frequencies.

A type of band filter known as a gammatone filter can be used to achieve a similar effect which uses a the product of a gamma distribution and a sinusoidal wave to to transform incoming frequencies. The mapping it produces thus naturally increases in sensitivity as as input values shift from one end to another.

It has been suggested and researched as a way to mimic the cochlear organ's transformation of hair's being bent by sound in the ear into neural activity.

In theory it will accomplish the same role as the mel filter was supposed to but do so through mimicking the sensory structure of the human ear rather than attempting to alter data with a function designed mimic self-reported human detection of different pitches, as the mel filter does.

6.3.2 Novel Network Structures

Research into alternate neural network structures faced with similar audio categorisation problems has often focused on networks that are capable of processing audio as a temporal experience. Although the Short Time Fourier Transform is designed to provide 'snapshots' over a time frame of an audio input's qualities, our prototype's 'forgets' prior frames immediately. This contrasts to the obvious intuition that for humans listening to music is not a snapshot but an ongoing experience [18]

One possible structure for capturing the nature of human musical experience is a combined Recurrent convolutional neural network. Zhou et al. [4] have demonstrated the efficacy of a combined recurrent convolutional neural network structure when compared to non-hybrid convolutional neural networks. Their structure used a convolutional neural network to categorise features drawn from spectrograms of audio, just as ours did, but then used a recurrent neural network to develop a temporal summarisation of these spectral features.

Rothman [19] has also suggested altering how information is fed to a convolutional neural network to reproduce this ability to understand music as a temporal experience. Rather than being fed a single spectral frame, a network is fed a buffer of frames, with each new frame being added to the buffer and the one with the oldest time stamp deleted. This leads to training a network on families of extracted features that represent an audio input's qualities stretched out over time. This method is of particular interest as it requires only the alteration of the feed in method for the network, not the network's structure itself.

6.4 Reflections

In an overall sense the contribution of this project is that it makes more of sonified world available for analysis. The implications of this could be wide ranging and they are difficult to predict. Some avenues that could be problematic could be the use of these techniques to perform emotional categorization of speech, leading to efficient identification of particular modes of speech (e.g.: anger) and may contribute to censorship tools. Conversely, the capacity to identify emotional content may be useful in medical contexts, where the tool could help individuals with social difficulties to better judge the emotional content of the conversations they have with others. As with much other research concerning the development of tools, there is a strong ethical responsibility on those who would use the tool to ensure that their intention and methods are both ethical and socially justifiable.

References

[1] Aphex34. Typical convolutional neural network. Wikimedia. [Online]. Available: https://commons.wikimedia.org/w/index.php?curid=45679374

[2] Chrislb. Multi layer neural network. Wikimedia. [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/c/c2/MultiLayerNeuralNetworkBigger_english.png

[3] J. Liu, M. Huang, and J. Liu, "Music emotion understanding by computer based on bp neural network," in *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, 2017, pp. 1–5.

[4] W. Zhao, Y. Zhou, Y. Tie, and Y. Zhao, "Recurrent neural network for midi music emotion classification," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, 2018, pp. 2596–2600.

[5] T. Liu, L. Han, L. Ma, and D. Guo, "Audio-based deep music emotion recognition," in *6TH INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN, MANUFACTURING, MODELING AND SIMULATION (CDMMS 2018)*, Busan, South Korea, 2018. doi: 10.1063/1.5039095 p. 040021.

[6] A. Aljanaki, Y.-H. Yang, and M. Soleymani, "Developing a benchmark for emotional analysis of music," *PLoS One*, vol. 12, no. 3, Mar. 2017. doi: 10.1371/journal.pone.0173392

[7] S. Mian Qaisar, L. Fesquet, and M. Renaudin, "An adaptive resolution computationally efficient short-time fourier transform," *Journal of Electrical and Computer Engineering*, vol. 2008, May 2008. doi: 10.1155/2008/932068

[8] A. Sahai, R. Weber, and B. McWilliams, "Spectrogram feature losses for music source separation," *CoRR*, vol. abs/1901.05061, 2019. [Online]. Available: http://arxiv.org/abs/1901.05061

[9] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Convolutional recurrent neural networks for music classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 2392–2396.

[10] T. Mitchel, *Machine Learning*. Singapore: McGraw-HiLL, 1997. ISBN 0-07-042807-7

[11] S. Dutt, K. D. Amit, and S. Chandramouli, *Machine Learning*. India: Pearson Education, 2018. ISBN 978-9353066697

[12] S. Afaq Ali Shah, H. Rhamani, and S. Khan, *A Guide to Convolutional Neural Networks for Computer Vision*. Morgan and Claypool, 2018. ISBN 9780080443355

[13] B. Boashash, *Time Frequency Analysis*. Elsevier Science, 2003. ISBN 9780080443355

[14] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," vol. 8, May 1937. doi: 10.1121/1.1915893

[15] R. Sarkar, S. Choudhury, Sombuddha Dutta, A. Roy, and S. Kumar Saha, "Recognition of emotion in music based on deep convolutional neural network," *Multimedia Tools and Applications volume*, vol. 79, no. 765–783, Sep. 2020. doi: 10.1007/s11042-019-08192-x

[16] H. Pulakka and P. Alku, "Bandwidth extension of telephone speech using a neural network and a filter bank implementation for highband mel spectrum," vol. 19, Sep. 2011. doi: 10.1109/TASL.2011.2118206

[17] H. Kinsley. Convolutional neural networks - deep learning basics with python, tensorflow and keras. Youtube. [Online]. Available: https://www.youtube.com/watch?v=WvoLTXIjBYU

[18] A. Wingfield, "Evolution of models of working memory and cognitive resources," *Ear Hear*, vol. 37, Jul. 2016. doi: 10.1097/AUD.0000000000000310

[19] D. Rothmann. Human-like machine hearing with ai (2/3). towardsdatascience. [Online]. Available: https://towardsdatascience.com/human-like-machine-hearing-with-ai-2-3-f9fab903b20a

For DIVA

{
"Author1": { "name": ""},
"Degree": {"Educational program": ""},
"Title": {
"Main title": "",
"Language": "eng" },
"Alternative title": {
"Main title": "",
"Language": "swe"
},
"Supervisor1": { "name": "" },
"Examiner": {
"name": "",
"organisation": {"L1": "" }
},
"Other information": {
"Year": "2020", "Number of pages": "viii,49"}
}