Instituto Superior de Engenharia do Porto

Automatic classification of music genres

Gustavo Costa Araújo Moreira

Dissertation to obtain Master's Degree in Informatics Engineering, Specialization in Software Engineering

Supervisor: Fátima Rodrigues

Jury:

Presidente:

[Nome do Presidente, Categoria, Escola]

Vogais:

[Nome do Vogal1, Categoria, Escola]

[Nome do Vogal2, Categoria, Escola] (até 4 vogais)

Porto, Fevereiro 2021

Página em branco [apagar este comentário]

Dedicatória

Página em branco [apagar este comentário]

Resumo

Palavras-chave: Palavra-chave1, …, Palavra-chave6

Página em branco [apagar este comentário]

Abstract

O documento tese deve conter um resumo em português e outro em inglês que não excedam as 200 palavras ou 1 página A4. Quando a tese é escrita em português o abstract deve ser uma tradução em inglês do resumo.

Se a tese for escrita em inglês deve conter um resumo alargado em português que não exceda as 1000 palavras ou 2 páginas A4.

Após o resumo/abstract é obrigatório colocar as principais palavras-chave/keywords do tema em que se insere o trabalho desenvolvido, sendo permitido um máximo de 6 palavras-chave/keywords.

Keywords: Keyword1, …, Keyword6

Página em branco [apagar este comentário]

Agradecimentos <opcional>

Dirigidos ao(s) orientador(es) à família, colegas, Instituições, …

Página em branco [apagar este comentário]

Table of contents

Table of Figures

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

Lista de Tabelas

Nenhuma entrada de índice de ilustrações foi encontrada.

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

Acrónimos e Símbolos

Lista de Acrónimos

IA	Inteligência Artificial

Lista de Símbolos

β	Largura de banda

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

1 Introduction

This chapter has as exclusive purpose to introduce the work that is described by this document. It contains the scope definition, the purpose that the work tries to achieve, the intrinsic value that is provided to the society by it and an overall structure of this document.

1.1 Context

Over the years that compose our humanity, music has been part of the human nature. Rhythm and sounds are fairly easily categorized into music genres. Classifying a music into genres has never been a problem for humans, as long as they previously understand the concept. It is a task that can be accomplished even by listening to only a small extract of music, even of five seconds or less. Subgenres are also easily identifiable by humans, assuming they are familiarized with the genre in question. For example, for someone who know trance music, it should also easily identify its subgenres, for instance, psy-trance, melodic-trance, hardstyle, amongst many others.

With the increasing number of songs reaching peak levels every day, it becomes humanly impossible to classify each song individually.

The music streaming service industry is growing at a fast pace, and now there is more offer for consumers than ever before.

As the consumption of music through music streaming services become more and more a standard, it also becomes imperative to guarantee that a good quality service is provided.

That implies a state of the art music genre classification, with accurate subgenre labelling. As it becomes impossible for a human, or group of humans, to classify the entire catalogue of music available, the question that this dissertation tries to answer is whether a machine learning system is capable of doing music genre classification faster and more reliable that a human does.

Nowadays, the mainstream music streaming companies like Spotify, Apple, Amazon, and many other heavily really on machine learning algorithms to automatically perform music genre classification. It is extremely important to keep researching on this topic, improving the automation of music classification and expanding the genres that are classified.

1.2 Problem

Today the worldwide music catalog, thanks to globalization, increase of accessibility to the internet and quality of life improvement, of the society in general, allows to the population to consume music by paying a monthly fee streaming service which on the other hand, should provide a service of quality with a rigorous content.

If the sentence above is not enough to convince the reader that the amount of data that is being uploaded to the internet and made available to consumers is massive, the following mathematical analysis will help statistically understand the scale of the problem.

The average duration of a song is three to five minutes. For the purpose of this study, let's assume that the average length of a song is four minutes.

The regular work length during a single day is of eight hours, in other words, 480 minutes.

If a single person spends every single minute of working hours listening to music with the single task of classifying its gender, it would be able to only classify 120 songs during a day.

In chapter 3, the amount of data related to music upload is analyzed and explored in detail, where the reader can understand why relying only on humans to perform such task is currently considered hideous.

Machine learning is the main study topic, with the purpose of providing valuable assistance to humans in order to classify music genres.

1.3 Objectives

It is the purpose of this document to present a deep learning based solution that improves the speed of music genre classification, without losing the accuracy that is obtained by humans.

This objective is wide in the spectrum regarding the meaning that can be associated to it. Therefore, defining sub objectives can help reduce the spectrum of the goal and provide valuable assistance to achieve the main objective. Having this in consideration, the sub objectives are the following:

- Study previous work developed with the single purpose of classify music genres automatically. Machine and deep learning based projects are preferrable and the biggest focus of research, but alternatives paradigms and implementation should not be discarded and should have a dedicated section to evaluate it's performance and viability.

- Study the available deep learning frameworks that are widely available and embraced by the developer community and select one to perform the practical part of the dissertation. The usage of a widely available framework deeply reduces the risk of failure of a project.
- Implement a machine learning based solution that classify a pre-selected number of music genres. This implementation should strive to always use the best development practices and reach the goal of performance better and faster than humans.
- Implement a basic application that connects to the machine learning based solution to classify songs without human assistance. This application should accept an audio file and output a music genre.

1.4 Document Structure

This document is structured in 6 different chapters, each chapter containing sections, followed by all supporting references and the appendix for further details of the process.

Each chapter is summarized below:

1. Introductory chapter that contextualizes the reader about the topic of the document, the problem that is identified, the solution that is proposed and the approach that is used to achieve the proposed goal.
2. Presents a formal definition of the value analysis that this work intends to bring to the society, going through different models, such as New Concept Development and Analytic Hierarchy Process. From a business point of view, the Canvas Business Model is used.
3. A deep dive into the meaning of Machine Learning and Deep Learning, containing a dissection of the meaning of these two concepts, to help the reader easily identify terms and notations used in the following chapters.
4. A state of the art chapter dedicated to study relevant work previously done by other researchers on the topic, presenting the achieved results, highlighted concerns and identified future work that needs to be done to improve current proposals.
5. A chapter dedicated to the system requirements design for the solution to be implemented.
6. [TBD]

Inserir página em branco apenas se necessário de modo a que o próximo capítulo comece numa página à direita

2 Value analysis

This chapter presents to the reader the value analysis made in the context of this work. Value analysis "is an organized creative approach which has as purpose the efficient identification of unnecessary cost, i.e, cost which provides neither quality nor use" (Lawrence Miles, 1946), therefore, the primary objective of this chapter is to assess how to increase the value of the work at the lowest possible cost without sacrificing quality.

In order to create value, this analysis reflects on the real problem that was introduced in the previous chapter and presents ideas that bring value to the customer. Automatic music classification is not a completely new topic, therefore, bringing value to the customer might be the increase of speed classifying one single song, increasing the accuracy of classification, or even reducing the cost of development and achieving similar results when comparing to existing solutions.

The following section of this chapter concretizes the above based on the use of the New Concept Development (NCD) model of Peter Koen (Koen et al., 2001). The New Concept Development model ideas will further be complemented with an even more business-oriented model, the business model canvas, to provide a clear vision of the value that this work brings to society.

2.1 New Concept Development Model

The New Concept Development model (NDC) is a framework developed with the single purpose of providing a concrete way of establishing new products based on an initial idea, increasing the act of creativity in the world. It was built based on the idea that there is a "widely-perceived lack of high-profit ideas entering the New Product Process Development (NPPD)" (Koen et al., 2001). To fight this issue, the New Concept Development model provides a common language (framework) of the key components that drives innovation further (Koen et al., 2001).

There are three key components that compose the New Concept Development model:

- The inner area, that defines the five key elements comprising the Front End of Innovation (FEI). These elements are:
    - Opportunity Analysis
    - Opportunity Identification
    - Concept & Technology Development
    - Idea Selection
    - Idea Genesis
- The engine, which drives the five front-end elements and is fueled by the leadership and culture of the organization
- The influencing factors, which drive product development and can be intrinsic or extrinsic, for example, organizational capabilities, outside world business strategies, among others.

The Figure1 provides a visual help to the bullet points defined above.

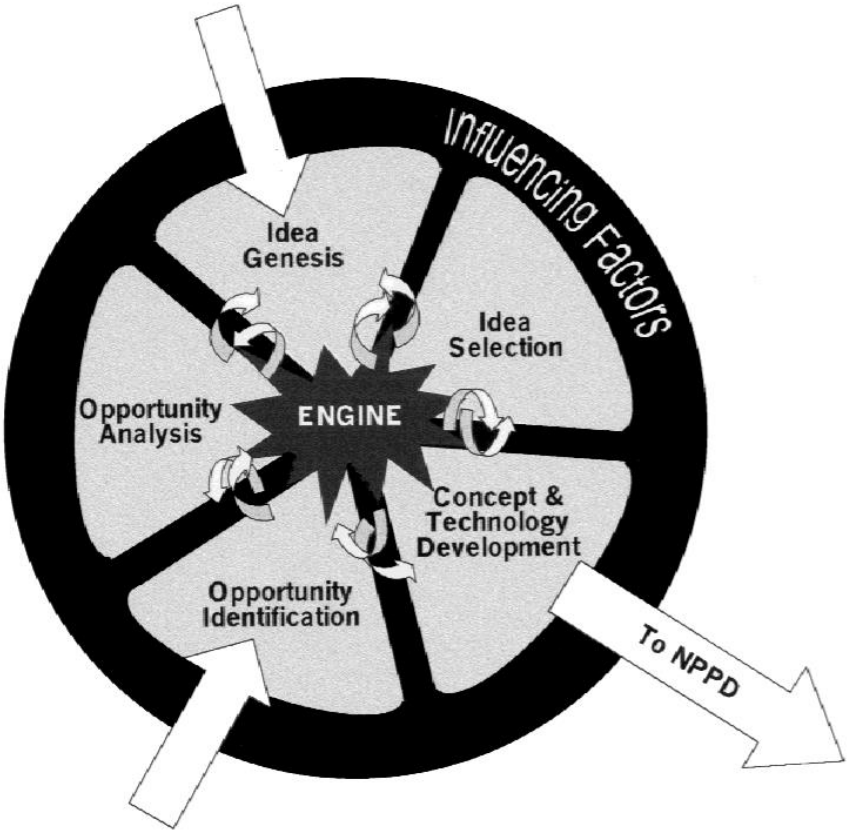

Figure 1 - The New Concept Development model as illustrated in the original paper (Koen et al., 2001)

Furthermore, several characteristics are worth noting.

“The inner parts of the NCD were specifically designated as elements rather than processes. Processes imply a structure that may not be applicable and could force a set of poorly

designed NPPD controls to be used to manage front-end activities. The circular shape is meant to suggest that ideas are expected to flow, circulate and iterate between and among all the five elements". (Koen et al., 2001)

2.2 Opportunity identification

Opportunity identification is used to identify opportunities that are worth to be investigated and studied. "Business and technological opportunities are explicitly considered so that resources will eventually be allocated" to a determined area (Koen et al., 2001).

In broad terms, an opportunity "may be a near-term response to a competitive threat, a breakthrough possibility to capture competitive advantage, or a means to simplify/speed-up/reduce the cost of operations" (Koen et al., 2001).

The New Concept Development model understands that for opportunity identification to work well, sources and methods that are used to identify opportunities are an essential element.

Typical methods may be divided into two big groups:

- Formal – opportunity identification processes that are aligned with all the influencing factors, e.g: brainstorming, mind mapping and lateral thinking, trend analysis.
- Informal – more organic processes that allow the flow of opportunity identification ideas, eg: ad-hoc sessions, individual insights, or edicts from senior management.

This work was based on trend analysis, following the trend in the technological community regarding Artificial Intelligence, Deep Learning, Machine Learning, and other technologies of the future.

Automatic music classification is a hot topic

In the Introduction chapter, it was discussed that there is an increase in data consumption regarding music content. Companies like Spotify, Apple, and Tidal provide a streaming service that must be of extreme quality to its consumers.

In 2016, Spotify provides a total of 1387 music sub-genres in its service platform. (Nick Patch, 2016). Big corporations like Spotify are already on the edge of technology and have dedicated teams working on a sub-area of machine learning called "machine listening", and classify songs based on a set of factors, including time, acoustic-ness, energy, danceability, the strength of the beat and emotional tone (Nick Patch, 2016). Spotify already builds it's platform music genres classification using a still-evolving tool, but "the process is still imperfect. At one point, the computers confused the sound of the banjo with human singers" (Nick Patch, 2016).

In 2020 and with the increase of deep learning study materials, the number of studies published in recent times shows that there is still a lot to understand and improve.

Leland Roberts developed a Convolutional Neural Network that classifies music genres with an overall accuracy of 68%. (Leland Roberts, 2020).

The above information allows identifying an opportunity, which is to improve the accuracy of automatic music classification. Streaming services benefit from a more accurate algorithm, therefore this opportunity identification allows the generation of value for the customer, once a better accuracy is achieved.

2.3 Opportunity analysis

The NDC model defines opportunity analysis as the validation of the previously identified opportunity. The first chronological moment of an opportunity is to identify it. The second is to study its viability and support the effort that might be done in the future based on facts, market needs, and business alignments. (Koen et al., 2001). Taking this into consideration, this section decomposes the opportunity to validate the value that the work will bring to society.

Why is automatic music classification important?

From 2015 to 2020, the overall Artificial Intelligence revenues grew by 20%, and this number is expected to grow exponentially in the upcoming years (UBS, 2020).

Music genre classification is only a small portion of Artificial Intelligence, but it is currently explored by many students, researchers, and big corporations that are already using Artificial Intelligence based algorithm to perform music genre classifications.

Currently, a deep investigation into the topic will not be able to find a common guideline to develop algorithms of the sort. The current documentation available is done on an individual level, within an academic context, or researchers publishing documents like this one, exposing a proposal of implementation and presenting the results.

No standard has been defined to this date.

In 2021, Spotify has 144 million paid subscribers, and this number has more than doubled since 2017. (Spotify, 2021). The numbers tell us that streaming services are reaching a never reached number of new users before, and with the increment of users, the competition between corporations gets bigger, and therefore, the quality of the service should be better to improve customer retention.

Research on the topic that is currently done is targeted at the 11 most popular music genres, contrasting with the 1387 genres that Spotify currently provides.

There is a clear need in the market to standardize music genre classification based on genre, the velocity of training, and general accuracy of the algorithm.

All of the three factors above, if improved, can bring a lot of value to the customers, and therefore, validating the value of the identified opportunity.

2.4 Idea Generation and Enrichment

In the New Concept Development model, the Idea Generation and Enrichment, or Idea Genesis, is the process of transforming an opportunity into a concrete idea. Therefore, it "represents an evolutionary process in which ideas are built upon, torn down, combined, reshaped, modified and upgraded" (Koen et al., 2001). It is not expected that the ideas, once defined, are fixed and immutable. They go through a process of continuous improvement and may go through many iterations before is finished. This process may include "a formal process like brainstorming sessions and idea banks so as to provoke the organization or individual into generating new ideas for the identified opportunity" (Koen et al., 2001).

For this work, both brainstorming and bank ideas were used when defining ideas for the identified opportunities. The bank of ideas came mostly from the author, while researching for chapters number 3 and 4, and also by the inclusion of members in informal discussions that have an interest in the topic.

The following ideas are presented with the intention of identify possible future solutions to the identified opportunity.

1. **Continue the work of an individual researcher that has publicly made his work available to the community** – The objective of this idea is to not start from scratch and take advantage of the work that someone else has already done, and improve the solution to obtain better results.
2. **Build a technical solution from scratch, using a dataset that was previously defined** – The purpose of this idea is to start a technical solution from scratch. This solution might be based on a deep learning framework, but also other types of framework. The common point here is that it bases the classification on a dataset that is freely available to the community to use.
3. **Build a technical solution and a dataset from scratch** – The objective of this idea is to fully design and implement the solution as discussed in the idea number two, but using a dataset that is manually built for the context of this work.

4. **Use previous research to identify common points of implementation and propose a guideline of implementation that is accepted by the community** – The goal of this idea is to explore the work that was already done regarding music genre classification, identify common practices and guidelines that lead to the development of a reliable algorithm. Further implementation of this idea requires starting conversation with proper identities that can validate and certify the process. In other works, the end goal of this idea is to

have a clear and define pattern of implementation for music genre algorithms, like Martin Fowler defined software design patterns previously.

5. **Build a music genre classification framework** – The objective of this idea is to develop a framework that is highly optimized to develop music genre classification algorithms, having as ultimate goal facilitate future work of the developer community by reducing speed of developing and facilitating the accuracy of the algorithm.

2.5 Idea Selection

Idea Selection is the element of NDC that allows to select the idea that brings the most value. The "selection may be as simple as an individual's choice among many self-generated options" (Koen et al., 2001). NDC does not identify a rigorous method to select the idea with the most value.

At this point, as presented in the previous section, there are 5 ideas identified, and selecting one is still left to the individual decision. To help identify the most viable solution for this dissertation, a second method is introduced, the Analytic Hierarchy Process.

2.5.1 Analytic Hierarchy Process (AHP)

The Analytic Hierarchy Process (AHP) is one of the most used methods in decision making environments. It was defined by Thomas L. Saaty, in 1980. This method uses both quality and quantity criteria in the evaluation process. The main idea is to divide the problem into decision tree levels, facilitating the comprehension and evaluation of the decision when selecting an idea.

The process describes the creation of a tree, where the first level represents the problem that is intended to be solved, and the second hierarchy level represents the factors of importance in order to achieve a problem solution. Finally, in the lowest level of hierarchy, are represented the ideas proposed to solve the problem.

The Figure 2 represents the decision tree of the AHP applied to the problem introduced in the first chapter of the document. The criteria to assess whether the idea is significant or not is the following:

- **Technical achievement** – The solution achieves a highly quality code structure and directly solves the proposed problem, by applying software engineering best practices.
- **Algorithm accuracy** – The solution achieves a good accuracy that is bigger than the current research standards.
- **Time restrictions** – If the solution is timeboxed to a specific time and needs to be delivered on a specific date
- **Community meaningfulness** – If the solution brings a more direct value to the community.

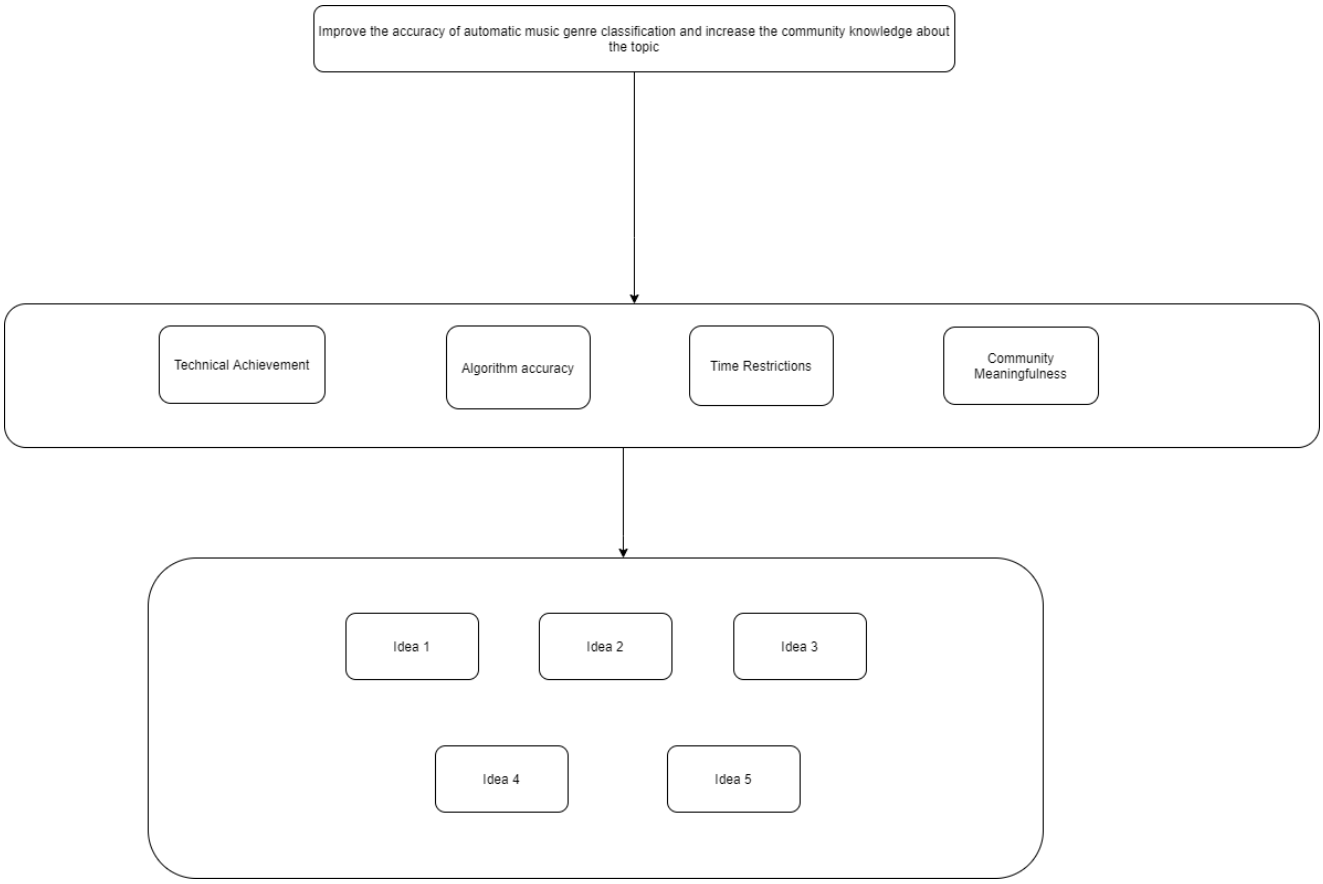

Figure 2 - The AHP hierarchy decision tree

Having the AHP tree defined, it is necessary to proceed to the next stage. The criteria on the second hierarchy level of the AHP tree is directly compared to one another in terms of relative importance, accordingly to the model proposed by Saaty, in 1990. Table 1 provides guidance to the model proposed by Saaty.

Table 1 - Scale of criteria comparison (Saaty, 1990)

Importance level	Definition	Explanation
1	Same importance	Both activities equality contribute to the goal.
3	Week importance	The experience and judgment lightly favor one activity comparing to the other one.
5	Strong importance	The experience and judgment strongly favor one activity comparing to the other one.

7	Very strong importance	One activity is highly favored compared to the other one
9	Absolute importance	The evidence favors one activity with the highest certainty level possible.
2, 4, 6, 8	Intermediate values	When the criteria is in the middle of two definitions.

Based on Table 1, the comparison between criteria for the stated problem can be achieved. Table 2 provides a comparison between the different weights of each criteria, following the AHP scale. The information in the table assembles the conclusion that Algorithm Accuracy is the most important criteria, followed by Community Meaningfulness and Technical Achievement. Time restrictions on the other hand are considered to be the less important criteria.

Table 2 - AHP Evaluation table for dissertation topic

Evaluation Criteria	Technical Achievement	Algorithm Accuracy	Time Restrictions	Community Meaningfulness
Technical Achievement	1	0.5	3	0.75
Algorithm Accuracy	2	1	4	1.5
Time Restrictions	0.33	0.25	1	0.5
Community Meaningfulness	1.25	0.75	2	1
Sum	4.58	2.5	10	3.75

The next step of decision making presented in the AHP process is to normalize the table above, so that the sum of each criteria is equal to 1 and, enable the calculation of importance of each criteria in the next step. Table 3 presents the AHP normalized table of the content presented in Table 2.

Table 3 - AHP normalized table

Evaluation Criteria	Technical Achievement	Algorithm Accuracy	Time Restrictions	Community Meaningfulness	Importance
Technical Achievement	0.2183	0.2	0.3	0.2	23%

Algorithm Accuracy	0.4367	0.4	0.4	0.4	41%
Time Restrictions	0.0721	0.1	0.1	0.1333	10%
Community Meaningfulness	0.2729	0.3	0.2	0,2667	26%
Sum	1	1	1	1	1

Table 3 is important since it provides guidance in the weights that should be considering when evaluating each idea individually. As stated before, Algorithm Accuracy is the most relevant criteria in idea selection for this work and should account for 41% of the decision. In the other hands, only 10% of the weight when assessing an idea should be related to time restrictions. Having this data, it is now possible to go through each idea and select the most valuable one.

- **Continue the work of an individual researcher that has publicly made his work available to the community –** This idea would be ideal in terms of time restrictions, improving speed of development and simultaneously provide a theoretical easy way to increase algorithm accuracy. However, there is no relevant data that suggest improved accuracy when building upon existing work. It is also not easy to determine if this idea is meaningful for the community since it does not bring a new perspective nor achieves technical details, since the implementation is already there.
- **Build a technical solution from scratch, using a dataset that was previously defined –** This idea starts from the premise that a good dataset exists and is able to classify more than 11 genres, and gives appropriate time to dedicate to build a more accurate and efficient classification algorithm. At the same time is contributing to the community by providing a new perspective of development and allows the visibility for technical achievement. It is not blocked from the beginning due to time restrictions.
- **Build a technical solution and a dataset from scratch –** This idea starts from the premise thar everything will be built from scratch. While building a dataset from scratch would definitely be helpful to the community and allowing for flexibility in terms of what can be classified, time consumed on this specific task would suck the entire time, not allowing enough time to improve algorithm accuracy, which is the most importance criteria. Therefore, this idea should not be considered.
- **Use previous research to identify common points of implementation and propose a guideline of implementation that is accepted by the community –** Although the idea would highly align with community meaningfulness, it would not provide direct response to the most important criteria, which is algorithm accuracy.
- **Build a music genre classification framework –** This idea aligns with the previously presented one. It emphases on community meaningfulness, but rejects or not direct aligns with algorithm accuracy.

Considering all the points above, the chosen idea for this dissertation in the idea "**Build a technical solution from scratch, using a dataset that was previously defined**". In chapter 5 of this document, it is presented a design proposal to the solution based on this idea. All further topics in the document may focus on the concretization of this idea and aligned with it.

2.6 Concept Definition

This dissertation has three main goals.

The first one is to identify what the community was already able to achieve regarding music genre classification. This classification might be achieved with a variety of different tools and techniques. Those are: machine learning, deep learning, machine listening, which all belong to a broader topic called Artificial Intelligence, or it might even be achieved by using techniques that are completely irrelevant for Artificial Intelligence. It is an objective of this work to identify all the possible techniques that are currently used. In parallel, a concrete report of what the community was already able to achieve, must be reported in this document. The first goal of this work should be able to answer to some questions: "What was the biggest accuracy already achieved?", "What genres are currently being considered?", "What is the cost and effort of creating a dataset capable of improve the accuracy?". Those are question that should be clearly answered in the next chapters.

The second goal of this dissertation is to provide a technical solution that allow the automatic classification of music genres. This solution should follow the best practices of software development, and should provide value to users, in the sense that is capable to do what it is supposed to do: correctly classify a music genre and with better accuracy than previous works.

Finally, the third goal is to increase the overall knowledge of the community by providing additional information regarding music genre classification. The outcomes, improvements, blocking points and breakthrough achieved during this work must be shared among the tech community to help mature the technology.

To help clarify the goals above, and to turn this work into a viable business product, the next sections of this topic will clarify the goal by using the business model canvas.

2.6.1 Value proposition

At this point the reader is familiar with music genre classification, the relation of this sub-genre as a small area of actuation of a broader topic of Artificial Intelligence, and understands that there is a deficit in accuracy in previous attempts to solve this problem.

The reader also understands that automatic music genre classification is more than just an academic research. Companies like Spotify and Apple already use machine listening to

develop their algorithm, with different goals, like providing recommendations of playlist or classifying songs are uploaded to the platform (Paul Pandey, 2018).

This work intends to help solving the current issues that have been reported in previous work, understand the failures, and provide a solution for them, contributing to the main goal of this dissertation: improve the algorithm accuracy.

In section 2.5.1, it was identified that due to time restrictions, a dataset will not be developed from scratch, so it is important to clarify that the development will not face improvements on that specific topic.

Therefore, this works intends to provide to companies, individual researchers and academics additional information on how to correctly develop music genre classification algorithms, the steps to achieve it, and additionally, how can the algorithm be adapted to other use cases, if the study becomes relevant.

In order to transform the previous statements into a concrete business idea, the Canvas Business Model for the use case will be presented. The Canvas Business Model is a business model that describes how an organization creates value (Osterwalder, A. and Y. Pigneur, 2010).

Table 4 provides detailed information of the business plan to consider when implementing this project.

The Key Partners are identified and listed. The university in which this work is inserted is a natural key partner. Furthermore, specialized knowledge platforms like Udacity and Udemy are considered key partners, as they can provide additional knowledge for the algorithm development, which otherwise would not be of easy access. Key activities include an initial research of the literature and a later implementation of a new solution to the stated problem in the previous chapters. The open source community is identified as a customer, since one of the goals of this project is to directly provide knowledge to this community. It is simultaneously clear that previous work done is the back bone and the starting point of this project and will provide invaluable knowledge. Finally, it is considered that developing the work will have no associated cost, unless there is the need to setup a cloud provider to boost the development of the solution. An optional cost is the usage of platforms like Udacity or Udemy to have access to learning content related to the topic, and therefore, contributing to the overall goal of this work.

Table 4 - Business model canvas

Key Partners	**Key Activities**	**Value preposition**	**Customer Relationships**	**Customer Segments**

<table>
<tr><td rowspan="3">- ISEP, TOWARDSDATASCIENCE, DeepLearning.ai, Udacity, Udemy

- Researchers on the subject, other academics</td><td>- Intensive review of the existing literature, and identifying common issues and problems that make the accuracy lower

- Design and implement a technical solution for automatic music genre classification

- Contribute to the community by publishing an article with the findings.</td><td rowspan="3">By identifying the most commons issues, problems and restrictions when developing the algorithm, the literature provided by this dissertation must contribute to the addition and increase of knowledge in the community. Furthermore, the develop code must also be publicly available at the end of this work.</td><td>Publish solution on Github. Find interested companies and researchers to keep working on the implemented solution.</td><td rowspan="3">- Open source community. By making the work available on Github, allow anyone interest proceeding with the work.</td></tr>
<tr><td>Key Resources</td><td>Channels</td></tr>
<tr><td>Public articles regarding automatic music genre classification.
Paid courses related to machine and deep learning in platforms like Udacity or Udemy.</td><td>Published articles
Tech blogs
Private company courses</td></tr>
<tr><td colspan="3">Cost structure</td><td colspan="2">Revenue Streams</td></tr>
<tr><td colspan="3">The results achieved by this work will be made available publicly with no costs.
To train the algorithm, the usage of a Cloud provider might be necessary and have some costs.
Additional knowledge by taking a course on Udemy or Udacity may be necessary.</td><td colspan="2">Improved accuracy
Reduced training time</td></tr>
</table>

3 Background

In this chapter, the reader is presented with the background that leads to the topic of this work. Key concepts will be presented and explained related to Artificial Intelligence, the broader topic that sits in the top of the chain (section 3.1), and subsequent topics of specialization will be further detailed, starting the Machine Learning (section 3.2), followed by Deep Learning (section 3.3), and finally explaining Machine Listening (section 3.4).

3.1 Artificial Intelligence

The term “Artificial Intelligence” was first stated by John McCarthy in 1956 and it is the science and engineering of making intelligent machines, especially intelligent computer programs (McCarthy, 2004).

Despite being first stated and researched in 1956, Artificial Intelligence remained a topic that was deeply connected to science fiction amongst the general society. In recent years though, the term started to become relevant again to the tech community since it became possible to actuate on real world problems by using Artificial Intelligence (AI) techniques. This actuation became possible by the sudden availability of large amounts of data and the corresponding development and wide availability of computer systems that can process data in achievable human time (IBM, 2020).

Artificial Intelligence is a broad topic. When applied to a specific problem, AI is concretized into a specific domain.

Figure 3 introduces a simplified representation of current layers of AI. Artificial intelligence is the most external layer, and therefore, the most generic one.

The intermediate layer is related to machine learning, one of the many categories of AI. Machine learning includes every algorithm that given a pre-determined set of inputs, produces an output based. The term machine learning exists because the output is also based on past information that was processed by the algorithm.

In the inner layer lies deep learning, a subcategory of machine learning, where, as stated before, given a pre-determined set of inputs, an output is generated. However, deep learning uses what is usually called a neural network, imitating the human brain to perform a task (Andriy Burkov, 2019).

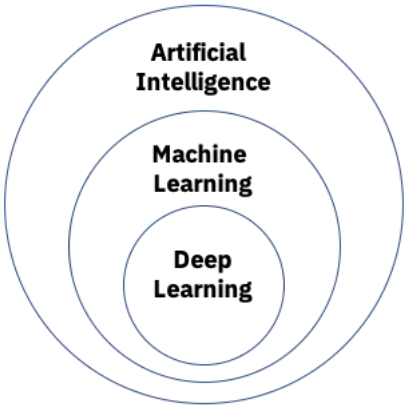

Figure 3 - A representation of Artificial Intelligence scopes

AI is not only represented by the identified categories.

In order to provide a full context, Figure 4 represents a hierarchy tree of AI and the most common categories that are currently relevant to the tech industry (Jeff Giangiulio, 2017).

Artificial Intelligence splits into two main categories: Robotic Process Automation, and Cognitive Computing. The scope of this work relies only on Cognitive Computing.

Cognitive computing can also be further split into different categories:

- Natural language processing, used for automatic text translations, text completion, etc.
- Computer Vision, used for image recognition and machine vision. This category is currently used for self-driving car technologies.
- Speech, used for speech to text and text to speech algorithms.
- Machine learning

This work relies on the exploration of Machine Learning, specially with Deep Learning techniques. The next sections will further explore these two topics separately.

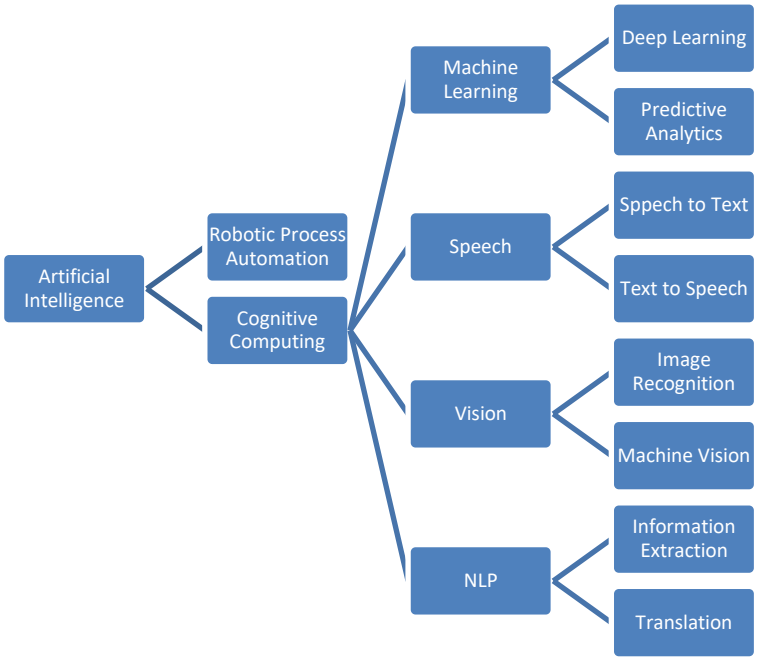

Figure 4 - Hierarchy tree of AI categories

3.2 Machine Learning

Machine learning is a subfield of computer science that is concerned with building algorithms which, to be useful, rely on a collection of examples of some phenomenon. These examples can come from nature, be handcrafted by humans or generated by another algorithm (Andriy Burkov, 2019).

3.2.1 Architecture

In Software Engineering, in order to fully understand a subfield, it is important to understand the relying architecture. Machine learning architecture is still volatile, but the community has accepted the following discussion points as concrete architectural cornerstones for this subfield.

Supervised Learning

In supervised learning, the training data is a mathematical model that consists of both inputs and respective desired outputs (Andriy Burkov, 2019). In this case, the system is able to find a relationship between the input and outputs obtained during the training data and apply the same technique for a given input afterwards. This architecture is usually related to classification and regression analysis problems.

Unsupervised Learning

Unlike supervised learning, the input feature vector that corresponds to the training data of the algorithm does not contain output. Unsupervised learning identifies relations between various features of an input, such as trends, color schemas, between others, and generates output accordingly (Andriy Burkov, 2019).

Semi-Supervised Learning

In Semi-Supervised leaning, the training data simultaneously contain labeled and unlabeled data. Usually, the quantity of unlabeled data is higher than the labeled data (Andriy Burkov, 2019). The goal of this architecture is the same as supervised learning but, given a good dataset condition, is expected to generate a better model than the one generated by relying only on supervised learning data.

Reinforcement Learning

This is used in training the system to decide on a particular relevance context using various algorithms to determine the correct approach in the context of the present state. These are widely used in training gaming portals to work on user inputs accordingly (Andriy Burkov, 2019). This topic is not relevant for the context of this dissertation, so it will not be mentioned from now on.

Components

Machine learning systems follow a component development structure.

In this work, in order to classify music genres, data collection and data generation are major underlying tasks. A classification by genre is the end goal.

From now on this document, the terms “data collection”, “data generation” and “dataset”, “training data” are used interchangeably. While chapter 4 explain in deeper details these concepts, here they are introduced for better engagement with the rest of the chapter. Figure

5 exposes the main components of a machine learning system architecture (Markus Schimtt, 2020).

1. Data Generation – Every machine learning project starts by data generation. This can be achieved by collecting the data from scratch, using previous generated data as a starting point, or by using new and previously obtained data.
2. Data Collection – After generating the data, this data should be easily accessible. This should be achieved through a well-formed database collection, cloud storage or any other kind of storage that is programmatically accessible without major difficulties evolved.
3. Feature Engineering Pipeline – The act of preparing the data for training. At this stage, raw data is selected, labeled, transformed and combined in order to become ready to be feed in the machine learning algorithm.
4. Training – Training is where the data prepared on step 3 is feed to the algorithm. A set of inputs correspond to a set of outputs that are understood by the algorithm for future classifications/predictions.
5. Evaluation/Validation – Happens after the training step. It is where the model obtained in step 4 is validated in terms of accuracy, correctness and prediction. A bad result achieved in this step should require repetition of previous steps.
6. Task Orchestration – Non mandatory step of a machine learning algorithm. It facilitates the training, validation and data consumption of big datasets, by scheduling training times. When used, it is usually associated to a cloud provider, like AWS, Azure or Google Cloud Platform.
7. Prediction – Classify an input based on the previous training. In this work, prediction is used to classify any new music and obtain an output, the corresponding music genre.
8. Infrastructure – The underlying hardware required to run the solution. This might be achieved by using proprietary hardware or using cloud providers.

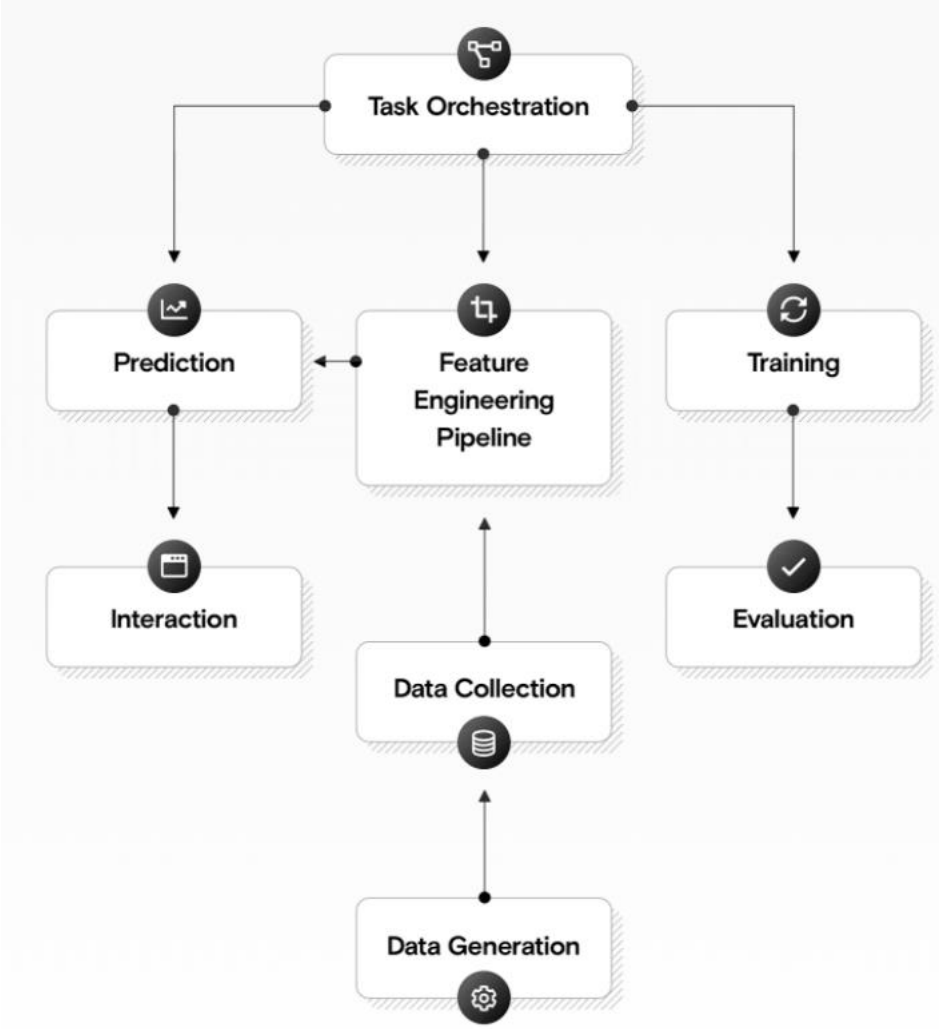

Figure 5 - Basic Machine Learning System Architecture

3.2.2 Benefits and common use cases

Machine learning gained attention in recent years, because problems stated in the past, are now solved due to the actual hardware capacity to solve them in useful time.

It is a fairly accepted statement that machine learning is capable of performing some tasks better than humans, and that is the single reason why there is an increased interest in building solutions from this subfield of computer science.

Some common use cases are spam detection, real time business decision making, creation of financial models, stock market prediction, amongst others (Nikhil Gupta, 2017).

Not so explored is music genre classification, though in recent years, some developments have been occurred. Chapter 4 focus on providing more case studies related to the current status of music genre classification.

3.2.3 Attention points

Despite growing usage of machine learning related technologies, one big attention point to have in consideration is related to data generation.

In order for a model to work correctly and achieve desirable results, the collection of good data to perform the task is an imperative step to achieve in first place.

Without a good training dataset the work is compromised from the beginning.

For this work, several datasets are considered. The GTZAN [referenciar] and A Million Song [referenciar] datasets are amongst the possibilities to be used as in the Data Generation and Data Collection steps of the machine learning system pipeline. Chapter 5 provides details about the design of the solution, and Chapter 7 focus on the solution.

3.3 Deep Learning

Deep learning is a subtype of Machine Learning and is inspired in the human brain structure. Deep learning algorithms attempt to mimic similar conclusions as humans would by continually analyzing data with a given logical structure (Artem Oppermann, 2019).

The above behavior is achieved by using a multi-layered structure of algorithms typically called neural networks.

A neural network owes its name from its comparison to a human brain. Just as the brain recognizes patterns and classifies different types of information based on what is capable of process, neural networks can be taught to perform the same task given a pre-determined set of data (Artem Oppermann, 2019).
Figure 6 represents a typical neural network used in deep learning algorithms. X vector represent the inputs, and Y vector represent the outputs.

W1, W2, W3 and W4 represent the weights associated within each layer, which are thereby represented by H1 Vector, H2 Vector and H3 Vector.

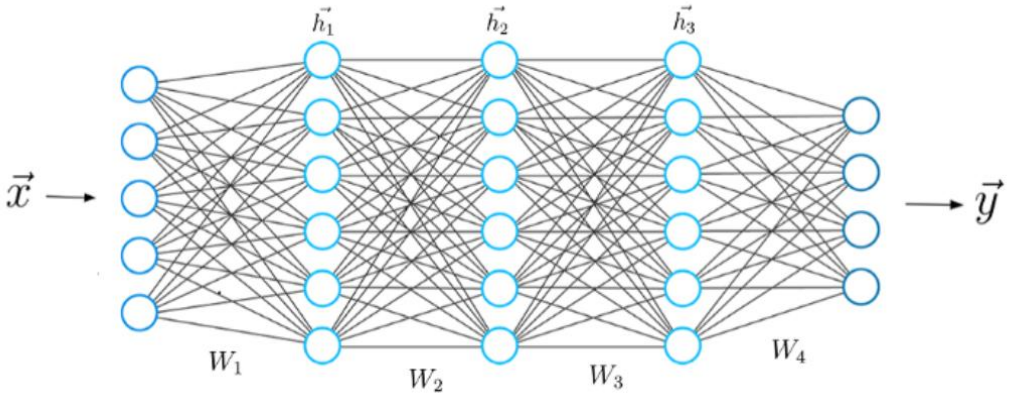

Figure 6 - Typical architecture of an artificial neural network

Feature Extraction

A common machine learning algorithm requires manual feature extraction.

Feature extraction is a term used to represent a pre-processing step that is required to be done manually to raw data so that later a machine learning algorithm can perform classification or regression tasks on the pre-processed data.

Feature extraction is usually quite complex and requires detailed knowledge of the problem domain. This pre-processing layer should be adapted and refined several times before achieving optimal results (Pier Ippolito, 2019).

This is the big differentiation of deep learning and neural networks. With neural networks, the Feature Extraction step is not required. The neural networks layers implicitly learn how to process the raw data.

This statement can be easily explained with an example. A machine learning model that intends to classify if a given image represents a car or not, feature extraction should be done manually. That is, the different features of a car, such as size, wheels, mirrors, colors should be extracted and feed into the algorithm as input data. On the other hand, with neural networks, the feature extraction step described becomes irrelevant, as the neural network recognizes unique features from the given input (images) and makes correct predictions from there.

At the time of writing, there are several identified neural network architectures that are accepted by the community as good enough to solve the majority of the problems by reusing and adapting the accepted neural network architectures.

Like any good software development practice, to develop a good deep learning system, a good architecture should be applied. The next section goes through the most common neural network architectures.

3.3.1 Types of neural networks

In this section, the most common used neural networks are described. After this section the reader should be familiar with deep learning architectures and identify the structure used in the project, described from Chapter 5 onwards.

Perceptron Networks

Perceptons are considered the first generation of neural networks.

This neural network uses feed-forward propagation, and the error is back propagated.

A limitation of perceptrons is the fact that all features are identified from the beginning, manually. A new requirement change in the feature set requires total rearrangement of the first neural network layer, which is time consuming and not practical. It is usually associated with only machine learning problems, and not commonly associated with deep learning architectures.

Convolutional Neural Networks

Convolutional Neural Networks distinguishes from perceptrons since they use unsupervised learning to classify data. These networks are primarily used for image and audio processing.

They work by given an input several convolutional layers are applied and perform feature extraction. Feature extraction is possible by transforming the initial image into different images based on similar aspects that the network identifies as a feature. That is the definition of a convolution.

A Convolutional Neural Network is able to successfully capture the spatial and temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better. (Yann LeCun, 1998).

Recurrent Neural Networks

A Recurrent Neural Network is a type of convolutional neural network which uses time series data. They are commonly used in natural language processing, speech recognition and image captioning problems (James Le, 2018). Similarly to Perceptrons and Convolutional Neural Networks, RNN use datasets to learn.

The distinguish point of RNN is that they introduce the concept of memory, as they take information from prior inputs to influence the next sequence of inputs and outputs. While traditional deep neural networks assume that inputs and outputs are independent of each other, the output of a recurrent neural network depends on the prior elements within the sequence (IBM, 2020).

Long/Short Term Memory Networks

Long/Short Term Memory networks can be considered a specialized version of RNN's, as they try to solve the memory vanishing problem of previous inputs. While RNN's introduce the concept of memory, by influencing the next input sequence with weights from the previous analysis, with time and over iterations, in RNNs that memory becomes irrelevant for newer inputs (Hochreiter & Schimhuber, 1997).

LSTM overcomes this, by defining memory cells, which store previous values and holds the data until the network explicitly forgets that same data.

These memory cells are particularly important in speech translation, where an input word at the beginning of a sentence might be more or less relevant to predict the next word in the text.

Gate Recurrent Unit Networks

Gate recurrent units are a slight variation of LSTM. The differentiation is that Gate recurrent units don't need the cell layer to pass memory along the next iteration (James Le, 2018). The calculations within each iteration ensure that the current values are being passed along either retain or discard a high amount of old information.

In general, GRU's tend to be faster. A general rule of tomb to decide between Gate Recurrent Units or LSTM is related to speed and accuracy. If speed is more important than accuracy, GRU's are usually the chosen architecture.

3.3.2 Comparison

Music genre classification can only be achieved when using the correct architecture.

Table 5 provides a comparison between different neural networks architectures, providing information related to which architecture is more suitable, according to the accuracy selection criteria, previously identified on chapter 2, as most important.

In Chapter 4 it will be described what has already been achieved in music genre classification by using some of the described architectures. Therefore, chapter 4 provides enough context to design an informed solution in chapter 5.

Table 5 - Comparison between different neural network architectures

Architecture	Date of conception	Use Cases	Speed vs Accuracy	Uses Memory	Relevant for audio processing
Perceptrons	1958	Simple classification problems. Datasets should be simple.	Speed	No	No
CNN's	1980	Image and audio processing	Accuracy	No	Yes
RNN's	1986	Speech Recognition Image Captioning Audio Processing	Accuracy	Yes	Yes
LSTM's	1995	Speech translation Audio Processing	Accuracy	Yes	Yes
GRU's	2014	Speech translation Audio Processing	Speed	Yes	Yes

3.4 Machine Listening

At the time of writing of this document (2021), the term "Machine Listening" is not yet a mainstream topic and highly accepted by the community.

There are some references to machine listening as a particular area of application of deep learning.

Machine listening is the use of signal processing and machine learning for making sense of natural / everyday sounds and recorded music. (Yoomchang Han, 2020).

Even though the architecture is still similar enough that does not justify creating a new branch of deep learning from it, investigations related to speech recognition, audio processing and sound classification suggest an emerging architecture in the upcoming years.

This section only purpose is to provide this information as an idea for the future and to encourage further investigation after the conclusion of this work.

Figure 7 suggests a redefinition of AI scopes, initially presented in this document in Figure 3.

Artificial Intelligence is still the broader topic, followed by Machine Learning and Deep Learning. Now, a new branch emerges from Deep Learning, therefore creating a new are nominated Machine Listening.

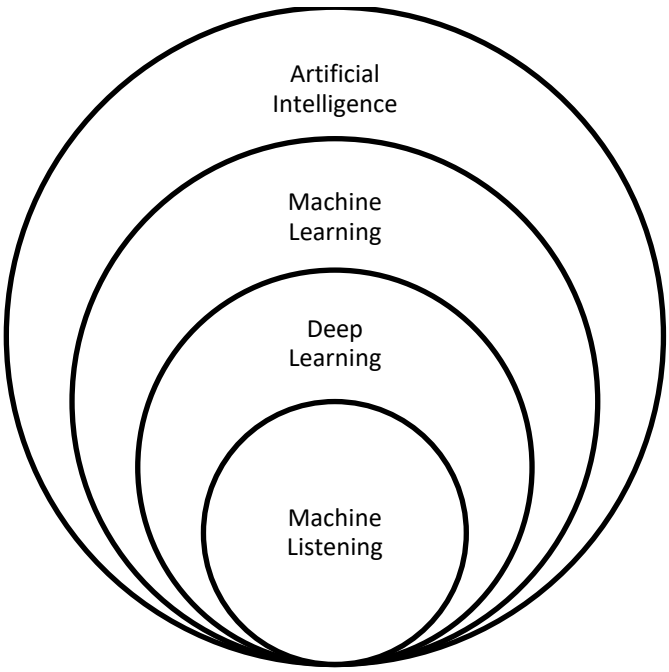

Figure 7 - Refined Venn Diagram of AI scopes

4 State of the art

First of all, this chapter intention is to provide enough context to the reader about the existing literature related to the topic, and which tools are available to start designing and develop a solution to the stated problem.

The next sections explore existing deep learning frameworks that were used to perform music genre classification tasks, as well as the underlying technologies.

Furthermore, an analysis to the literature is done to really understand what is the current state of the art of music genre classification. What is the accuracy of main projects, which genres are currently identified, what are the cornerstones to achieve a solution and what are the biggest barriers to achieve good results are some of the questions to answer in this chapter.

Previous work on music genre classification based on machine learning techniques achieved a top accuracy of 74% (Landsdown, 2019). It is also a goal of this chapter to understand common best practices and usual pit falls to consider when designing the system in chapter 5.

4.1 Deep learning frameworks research

In this section the most common deep learning frameworks are explored. The main goal is to get familiarized with the existing frameworks and together with section 4.2, provide enough information to have an informed decision about the design and solution to implement on chapter 5.

4.1.1 Existing frameworks

Even though a framework is not absolutely necessary to develop the project, it allows the focus of the work to be on the task instead of the underlying preparation work.

This section was written in January 2021. Therefore, only existing frameworks until the stated date were considered. Not all might be relevant enough for music genre classification, however, they are still identified so that a fully informed decision can be further take.

Pytorch

Pytorch is an open source machine learning framework that accelerates the path from research prototyping to production deployment.

This framework provides a rich ecosystem of tools and libraries to support development of various deep learning end projects. (Pytorch, 2020).

Pytorch also provides native cloud support, which taking into consideration big amounts of data that are necessary to train a deep learning model, might become an essential feature for product development.

The open source project is followed by 46000 people on Github, and it counts with 1733 contributors, which resembles a good community support.

Relying on this framework to develop deep learning projects provides a good community support, and challenges ahead may have been encountered by someone before.

The main development language when using Pytorch is Python and includes easy support for all the major Python libraries. (Khuyen Tran, 2020).

Literature in the topic shows that Pytorch has been used before to perform music genre classification tasks (Pankaj Kumar, 2020).

MxNet

MxNet is another open sourced deep learning framework that is backed by Apache. The benefits that this framework provides is the ability to perform distributed training, in other words, the possibility of combining local hardware with cloud hardware. (MxNet, 2020).

Another benefit of this framework is the support of eight development languages: Python, Scala, Julia, Clojure, Java, C++, R and Perl. Developers using this framework to develop deep learning models have a wider variety of choice in terms of the language they prefer to use.

The open source project is followed by 19200 people and counts with 850 contributors, which also resembles a good community support.

Literature is not clear enough to understand if previous work has been done regarding music genre classification. However, the framework has been used by Amazon to perform music recommendation to its users of the Amazon Music streaming service (Kat Ellis, 2019).

Tensorflow

Tensorflow is an end-to-end open source platform for deep learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that let researchers push the state-of-the-art in machine learning (Tensorflow, 2020).

This framework includes easy model building by making intuitive API's for immediate model iteration and debugging. It is used by several big companies like Airbnb, Google or Intel.

Tensorflow main development language is Python, and it is the only language that is fully supported by the framework. In other words, to fully take advantage of all the features provided by Tensorflow, Python is the only language that is capable to do it. However, some API's are available through other languages, such as Javascript, C++, Java, Go and Swift.

There is also non official support for languages as C#, Haskell, Julia, R, Ruby, Rust and Scala.

The open source project on Github counts with 153000 followers and around 2870 contributors.

Music genre classification has been achieved with success using Tensorflow with convolutional neural networks (Célestin Hermez, 2020).

MATLAB

Contrary to the previous frameworks presented in this section, MATLAB is not an open source framework. Nevertheless, MATLAB provides deep learning capabilities by allowing the creation, modification and analysis of deep learning architectures. Preprocessing data, ground-truth labeling of images, video and audio are amongst some of the features provided.

Integration with other deep learning frameworks such as TensorFlow, PyTorch and MxNet are also included in the MATLAB Deep Learning framework.

Literature also shows that music genre classification has been achieved using MATLAB (Mathworks, 2020).

Nvidia Caffe

This framework was originally developed by the Berkeley Vision and Learning Center, as well as community contributors, and later backed up by Nvidia. This framework is highly optimized to run faster on NVIDIA GPU's, so that is an aspect that should be taken into consideration when deciding to choose this framework.

On github is currently followed by 31300 people and counts with 258 distinct contributors.

At the time of writing, no relevant literature regarding music genre classification has been found using this framework.

Chainer

Chainer is a python based deep learning framework aiming at flexibility.

Although it is still maintained and new development efforts exist for bug fixing, no new features are developed since 2019, as announce by the maintainers. (Chainer, 2019).

The framework has followers and 250 unique contributors.

4.1.2 Comparison between frameworks

The later section provide context on what frameworks are available, which ones are open source, and the amount of unique contributors currently supporting the development of the framework.

Table 6 provides a summary of the most relevant features of the frameworks to date.

Framework	Open Source	Used for audio processing	Cloud Support	Main Language	All Supported Languages	Unique contributors
PyTorch	Yes	Yes	Yes	Python	Python	1734
MxNet	Yes	No	Yes	Python, Scala, Julia, Clojure, Java, C++, R and Perl	Python, Scala, Julia, Clojure, Java, C++, R and Perl	850
TensorFlow	Yes	Yes	Yes	Python	Python Scaka, Julia, Clojure, Java, C++, .NET	2878
MATLAB	No	Yes	Not enough information	-		-
Nvidia Caffe	Yes	No	Not enough information	Python	Python	269
Chainer	Yes	No	No	Python	Python	250

Table 6 - Comparison between Deep Learning Frameworks

Tensorflow is the framework with more unique contributors, indicating a higher support and more feature availability. On the other hand, MxNet becomes the most developer friendly framework, fully supporting different programming languages, which demonstrates flexibility. Pytorch comes second in terms of most unique contributors, and the literature is clear to indicate that this framework has been used to classify music genres.

MATLAB, given its enterprise nature, it is not open sourced. Even though, it is used by the community and it has also been used to classify music genres, accordingly to the literature.

NvidiaCaffe and Chainer are also two open-source deep learning framework. Both have less than 300 unique contributors and no literature has been found regarding music genre classification, with the later being considered deprecated and entered in maintenance mode.

All this information goal is to provide an informed decision in chapter 5.

4.2 Audio Spectogram

In order to understand the literature (see section 4.3), it is important to have an understanding of how deep learning models for music genre classification are feed.

The deep learning model is typically feed with images. Audio can also be processed in the format of an image, by studying the audio spectrogram.

A spectrogram is a representation of a signal that shows the evolution of the frequency spectrum in time (NVIDIA, 2021).

A signal on the other hand, can be represented in the format of a waveform, which is simply an image representing one air signal.

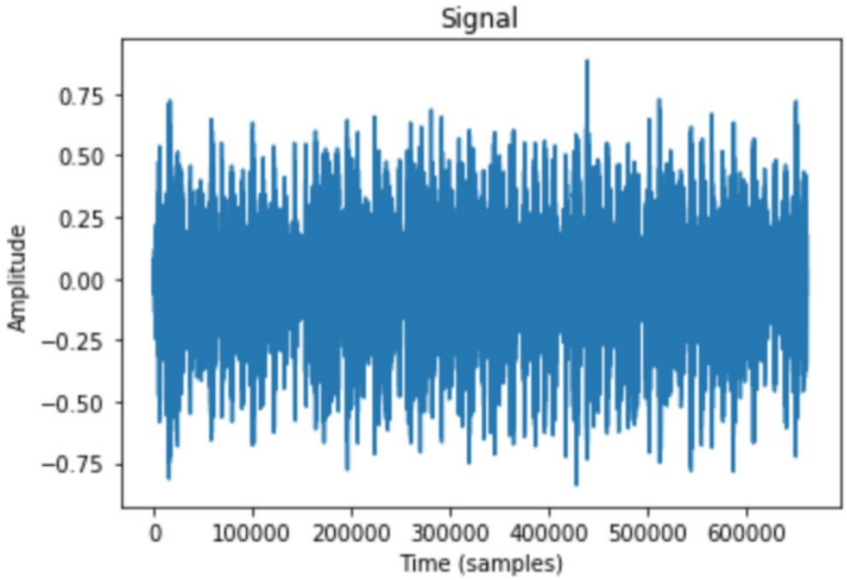

Figure 8 - Example of an audio signal represented as a waveform

Audio signal can be decomposed into smaller and more specific waveforms, that represent a particular characteristic of the audio signal. Such transformations can be applied by using a mathematical expression known as Fourier Transform. Fourier Transform is a mathematical formula that allows to decompose a signal composed of several single-frequency sound waves and isolates them into a unitary frequency. In other words, it creates an audio spectrum (Célestin Hermez, 2020).

It is outside of the scope of this work to deeply understand the mathematical formulas behind the feature extraction. Well known libraries (see section 4.3) already perform this work, and should be reused whenever possible.

Nevertheless, it is important to go through the most used spectrograms for music genre classification, in order to provide a better context to understand the rest of the document.

The Mel Scale and the Mel Spectogram

The entire spectrogram of a sound is not always relevant for human beings. Studies have shown that humans can detect frequency changes from 500 to 1000 Hz, but can hardly distinguish them between 10000 to 10500 Hz. The Mel Scale is a proposed unit of pitch such that equal distances in pitch sounded equally distant to one another (S.S Stevens, 1937).

To summarize this section, the concepted of Mel Spectogram should be defined. A Mel Spectogram is in practice an audio representation in the form of waveforms that is relevant for humans, and therefore, when applied to deep learning models, it provides a more realistic data training.

4.3 Existing applications of deep learning models for music genre classification

In this section, the main goal is to provide enough context about the existing literature regarding music genre classification. Previous work is exposed, its results are presented and major achievements are reported. Blocking points or difficult obstacles are also identified as attention points to consider during the design and implementation stage, to avoid committing the same errors, and therefore improve the existing literature.

4.3.1 Existing datasets

In this section the most common datasets used in music genre classification are presented and analysed.

GTZAN

One of the first dataset for music genre classification is GTZAN a dataset developed in 2002 by George Tzanetakis.

The dataset consists of 1000 audio tracks, each of 30 seconds long. The 1000 track are then divided into 10 different categories, each containing 100 songs. The tracks are all 22050Hz mono 16-bit audio files in ".wav" format (George Tzanetakis, 2002).

The 10 available categories are: classical, country, disco, hiphop, jazz, rock, blues, raggae, pop and metal.

The entire collection sizes approximately at 1.2GB of data.

Million Song Dataset

The million song dataset is a collection of audio features and metadata for a million songs. The dataset was first made available to the community in 2011 after being used in research (Bertin-Mahieux et al, 2011).

The entire dataset sizes 280GB. Due to its larger nature, for smaller projects and faster training, a subset of 10000 songs (around 1% of the total dataset) is provided.

This collection directly provides support for the following purposes:

- Encourage research on commercial scale algorithms
- Being the reference dataset for research
- Built-in API to work with the dataset

On contrary to the GTZAN Dataset, the labelled data includes more than just music genres.

The dataset was built for multiple purposes regarding audio processing. The collection includes a concept called “tag”, and a subset of all existing tags are related to music genres (Dawen Liang et al, 2011).

Using this dataset requires a bigger processing of the collection to use for the sole purpose of this work, music genre classification.

Free Music Archive (FMA)

The Free Music Archive is an open source collection suitable for evaluating several Music Information Retrieval (MIR) tasks (Github, 2021). One of the suitable tasks that this dataset permits is a solution for music genre classification.

The metadata for this collection includes the following:

- Unique Identifier, title, artist, genre, tags for 106574 tracks.
- 163 unique genres.

The team that maintain the collection on github provides out of the box several subsets of the data, accordingly to project scopes. The next are the available sub collections:

- Small – contains 8000 tracks of 30 seconds and 8 genres. Size is 7.2GB.
- Medium – contains 25000 tracks of 20 seconds and 16 genres. Sizes 22GB.
- Large – contains 106574 tracks of 30 seconds and 161 genres. Sizes 93GB.
- Full – contains 106574 untrimmed tracks and 161 genres. Sizes 879GB.

4.3.2 Showcase of existing applications

In this section, three different projects related to music genre classification are explored. The main goal is to identify working guidelines, common issues with implementation, and extract the accuracy results that have been achieved so far. Another important information to retain, is which music genres are currently being classified, in order to access the design in chapter 5.

Project 1

Project 1 was developed in 2018, and for the list of projects presented here, is the only one that does not use a framework to perform music genre classification (Parul Pandey, 2018).

Python was the chosen language for the implemented solution. There are three major libraries usages that need to be mentioned:

- Librosa – has the purpose of analysing audio signals geared towards music. It provides the building blocks necessary to create music information retrieval systems. (Librosa, 2021).
- IPython.display.Audio – a library that allows directly music streaming through a jupyter notebook.
- Keras – a deep learning API that amongst other functionalities, allows easy creation of Convolutional Neural Networks (CNN's), which are used in the context of this literature.

Genres classified

Given that this project uses the GTZAN dataset as a source for training, it classifies the ten following genres, which are the ones labeled by the dataset: Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae and Rock.

Data preprocessing

The challenge presented in this project for preprocessing data is related to audio formats. Audio format is an approach to store audio files on the hardware level of a machine. The GTZAN dataset provides ".au" audio files, while the mentioned IPython.display.audio library API is prepared to work with ".wav" audio files.

A conversion for each file present in training data is required in order to start the training.

Feature Extraction

Feature extraction is done manually in the context of this project.

Audio is a term used to describe any sound or noise that is within a range the human ear is capable of hearing and it is measured in hertz. (Computer Hope, 2020).

But audio has a known spectrum of features. While it is not an objective of this project to fully understand all it's feature, it is necessary to describe which ones the projects is using to train the data.

- Zero Crossing Rate – Is the rate of sign-changes along a signal, for example, the rate at which the signal changes from positive to negative or back (Parul Pandey, 2018).
- Mel-Frequency Cepstral Coefficients – Used to model the characteristics of an human voice in a sound.

- Spectral Centroid – It indicates the center of mass for a sound and calculates the weighted mean of frequencies. For example, spectral centroid will detect an equally distributed frequencies center of mass for a Blues song, while in a Metal song will detected a more weighted center of mass towards the end of a song.
- Chroma Frequencies – Chroma features are an interesting and powerful representation for music audio in which the entire spectrum is projected onto 12 bins representing the 12 distinct semitones (Parul Pandey, 2018).
- Spectral Rolloff – measures the shape of the signal.

Deep learning approach

Before starting the training of the model, the available data provided by the GTZAN dataset was prepared in order to obtain two different datasets. A training dataset, containing 80% of the available data, and a test dataset, containing the remaining 20%.

A Convolutional Neural Network was created with Keras to represent the model. The training consisted of 30 epochs with a batch size of 512. The chosen CNN has an initial layer of 512 nodes, followed by three hidden layers containing 256, 128 and 64 nodes, respectively. Finally, an output layer of 10 nodes is used, each node corresponding to a music genre.

For each hidden layer, the deep learning model uses a Rectified Linear Unit (ReLU) as an activation function.

The Rectified Linear Unit is the most commonly used activation function in deep learning models. The function return 0 if it receives any negative input, but for any positive value of “x” it returns that value back (Dans Becker, 2017).

Finally, for the output layer, it uses a softmax function, which is commonly used in deep learning to expose results in a multiclass probability for classification problems.

Accuracy achieved

A training accuracy of 82,3% was achieved. However, a validation accuracy of 62,5% was achieved for the validation dataset. The training accuracy is better when comparing to the validation accuracy, which indicates some level of overfitting of the model.

Attention points

Audio conversion needs should be taken into consideration when designing the system and choosing frameworks and libraries to work with. Depending on size of the training data that is chosen, this pre processing step can be time consuming. Furthermore, it should be validated before hand if there are libraries capable of doing the required conversion. Doing this conversion without the help of a known library might compromise the investigation of this work.

Furthermore, although very well know and easy to work with, the GTZAN dataset can only provide classification for 10 different genres. Many literatures like this one already use this dataset extensively, which might indicate that not much more value can be taken from it.

Project 2

The presented project, identified as “Project 2”, was developed in 2020 by Célestin Hermez (colocar referencia). The literature focus on a Tensorflow implementation, presenting the achieved results and also exposing bottlenecks. Nevertheless, the project also refers to machine learning techniques that were used to compare results.

Python was, as in the previous study, the chosen language to support all the development work, alongside a deep learning framework, which in this case, was Tensorflow.

Cloud usage to store the training data and to perform the training was introduced in this project, using Google Colab and Google Cloud Storage Buckets.

Genres Classified

To obtain training data, a small custom subset of data retrieved from the Free Music Archive (FMA) dataset was used, containing a total of 8000 audio clips with a duration of 30 seconds each (Célestin Hermez, 2020). Eight different genres are considered: Hiphop, Pop, Folk, Experimental, Rock, International, Electronic, Instrumental.

Deep Learning Approach

The chosen metric of the audio to use in the project is a Mel Spectogram. All required steps to prepare the training data for this approach are recorded.

Usage of Tensorflow is already mentioned. Furthermore, to build the model, Tensorflow internally uses Keras to create Convolutional Neural Networks, as described in the article.

64% of available data is used for the training data set, 16% for the validation dataset and finally for the test dataset, the remaining 20% are used.

The CNN used has an initial convolutional layer of 128 nodes, and a kernel size of 3. The model is followed by a max pooling layer. Max pooling is an operation that calculates the maximum or largest value in each feature map patch. The results of applying a max pooling layer is to help sample down the data points in the hidden layers, improve training times and reduce fighting overfitting (Jason Brownlee, 2019).

The CNN used continues with another convolutional layer with kernel size 3, followed by another max pooling layer. At this point the convolutional layer is flattened, and uses several dense layer until reaching the output layer, consisting of 8 nodes, corresponding to the 8 music genres in question.

Accuracy Achieved

The mentioned accuracy achieved was of around 40%.

Attention points

It is highlighted the importance of having a good knowledge of the business domain and feature engineering. The low accuracy achieved can be explained by an incorrect initial feature extraction. The project in study relied only in Mel Spectograms to initialize the model. Extracting the wrong set of features can compromise the goal of the study.

Another attention point to have that is also mentioned in the literature is the fact that only a small subset of the Free Music Archive dataset has been used. The compromise to make between the relationship of computational resources available vs time to train the model can compromise the results.

Project 3

For the third example in the state of the art chapter, a study lead by Aryan Khatana in 2020 is explored (colocar referencia). This example uses PyTorch as the main framework, and it also introduces the transfer learning concept.

Transfer learning is a machine learning method where a model develop for a task is reused as the starting point for a model on a second task (Jason Brownlee, 2017).

To achieve transfer learning, the ResNet model is used. ResNets are a deep learning model that have been stated and defined by several individuals and companies over recent years, and have proven excel in image classification tasks (Connor Shorten, 2019). Given that in this project classifying music genres is based in signals extraction that is later converted into an image representation, using ResNet with transfer learning improves speed training, as well as it provides a ground to achieve a good result.

Genres Classified

The study uses the GTZAN dataset, the same used in project 1. For completeness, the ten genres are enumerated here: Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae and Rock.

Deep Learning Approach and Accuracy Achieved

Instead of trying to extract audio features from the audio signal, the entire spectrogram of the audio is used, by transforming the ".wav" audio file into an image, using the Librosa library built for Python.

The available data from the GTZAN dataset is split into 90% for the training data, and 10% for the validation dataset.

For the model, the ResNet34 is used. Several training scenarios were used to train the data, compare its results and understand which hyperparameters are better suitable through the tuning technique.

Hyper-parameter tunning is a process of extensive trial and error. There are no simple and easy way to set hyper-parameters, such as learning rate, batch size, momentum and weight decay (Javaid Nabi, 2019).

With the usage of transfer learning, tuning the hyperparameters becomes the most relevant task during the training stage.

Table 7 compares the hyperparameters chosen with the accuracy achieved in the literature.

For 5 epochs, an accuracy of 48% is achieved. For a set of 15 epochs, accuracy increases to 68.75%, which results in a 20% gain. Increasing the epoch number to 100 raised the accuracy to 74.11%.

Hyper-Parameters	Epoch	Learning rate	Gradient clip	Weight Decay	Accuracy
Tuning 1	5	0.001	0.1	0.0001	48%
Tuning 2	15	0.001	0.1	0.0001	68.75%
Tuning 3	100	0.001	0.1	0.0001	74.11%

Table 7 - Comparison of accuracy vs hyper-parameters in project 3

Attention points

No major bottlenecks have been identified by reading the study. However, the limitation found is the usage of the GTZAN dataset, which is limited to 10 genres.

4.3.2.1 Comparison between existing applications

In this section, the literature chosen to be included in this document is compared.

Table 8 provides a summary of the described information in 4.2.

Table 8 - Comparison table between analysed literature

Project	**Year**	**Framework used**	**Main Language Used**	**Cloud usage**	**Accuracy achieved**	**Honourable libraries to mention**	**Dataset Used**	**Number of Classified Genres**
Project 1	2018	None	Python	No	62.5%	Librosa, IPython.display.audio, Keras	GTZAN	10
Project 2	2020	Tensorflow	Python	Yes	40%	Keras, Tensorflow	Free Music Archive	8
Project 3	2020	Pytorch	Python	No	74.11%	Pytorch, Librosa	GTZAN	10

Using the same dataset for training, the GTZAN dataset, higher accuracy results were achieved. The project that used Pytorch and Transfer Learning was able to achieve an additional 11.61% accuracy.

The project that used the Free Music Archive achieved a lower accuracy. Bottlenecks like incorrect feature extraction are the explained reason for such lower accuracy. Furthermore, exploring a bigger dataset brings additional challenges that should be tackled.

The results shown provide a guidance for the rest of this work: creating a first design implementation using the GTZAN dataset helps to create best practices guidelines and understand how to classify music genres in a first controlled and reliable attempt.

However, limitations of what genres can be classified should be taken into consideration.

A second attempt using the Free Music Archive, or even the Million Song dataset where more music genres are classified should be taken into consideration as well.

5 Deep learning for music genre classification

This chapter describes design options to implement the solution. The main goal is to provide full context on how the implementation will be achieved. At every point of the design, best practices of software development must be followed. Software Architecture is the first pilar required to achieve a reliable, extensible and future proof solution.

It is also desire of this work to contribute to community development on deep learning topics (see section 2.5), and therefore, design decisions that can contribute to that goal should be preferred, such as, preferring for library creation instead of single use implementation.

5.1 Design

Software Design is a concept that contains many moving parts. Artifacts can be developed during the design stage. Software Design can be achieved through the realisation of a Software Architecture Document (SAD).

The SAD states that two design views are always required:

- Use-Case View: used to list all possible use cases and which actors are present in the system.
- Logical View: a logical view describes all existing components of a solution, how they interact which each other and identifies all dependencies.

The SAD also allows to represent optional views:

- Deployment View: a diagram that shows the configuration of run time processing nodes and the components that live on them. Not relevant when the system run on a single CPU.
- Process View: addresses concurrency, distribution, performance and scalability of the design system.

5.1.1 Requirements

In this section, use case requirements are identified as a starting point for the design. Given that the focus is going to be on the model training, it needs to be formalized.

Figure 8 shows a representation for the Use Case View, or Use Case Diagram.

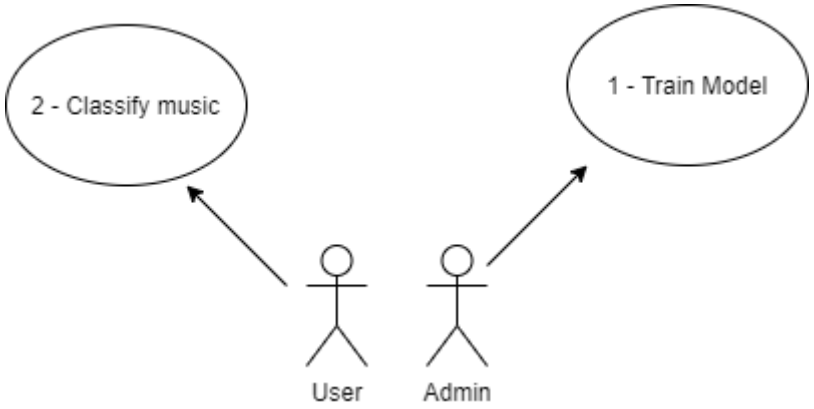

Figure 9 - Use case diagram

The project to be develop is not focused in user functionality, but rather as a platform to be included into third party application, develop by third party developers.

However, for completeness of this work and for demonstration purposes, it is a goal to show off the capabilities of the deep learning model to be developed. Taking this into consideration, the following use cases are identified:

Train Model

Training the model is something that should only be possible to be triggered by the administrator. A step by step description of the use case is provided below.

1. The administrator starts the android application and selects the option to start training the model.
2. The system accepts the request, and asks the administrator to choose a dataset from the available datasets lists to train.
3. The administrator chooses one dataset.
4. The system accepts the request, and asks the administrator to choose the genres to be trained and that are available within the chosen dataset.
5. The administrator chooses the genres.
6. The system accepts the requests, notifies the user that the training has been started and that will later notify the admin again, once the training is finished.
7. The administrator acknowledges the first notification.
8. The system finishes training the model and notifies the administrator about the result.
9. The administrator acknowledges that the training is completed.

Classify music

Classifying a music and obtaining the result, this is, the gender as an output, can be done by any user of the sample application to be developed. A step by step description of the use case is provided below.

1. The user starts the android application and uploads one song to be classified.
2. The system accepts the request, evaluates the uploaded audio file and returns a result.
3. The user acknowledges the result and closes the application.

5.1.2 Design proposal

It is a goal of this project to be community meaningful, and with that consideration, it is important to develop components in a way that can be reusable for other projects out of the box, without need for adaptation. The following design proposal measures different factors, such as: size of the dataset to use, component reusability and infrastructure costs.

Figure 9 represents the design proposal.

The presented components are expected to deliver a fully functional, end to end scenario of a user being capable of obtain a genre classification of a song based on its genre.

A brief explanation of each component is provided.

Android Application

A sample android application should be an artifact delivered at the end of this work. It is not a goal of this dissertation to show off android application capabilities, research best practices and propose new and improved ways of developing on android. However, this component should be delivered as a way of demonstrating the deep learning model capabilities in an easy and reliable way. When developing this component, best software development practices should be applied.

It should be developed using native android development kit with Android Studio.

Web-API

A Web API should also be delivered to provide an entry point for any developer who wants to take advantage of the trained model capabilities. This component should be capable of handling HTTP requests that given an audio file format, retrieve the genre of a music.

It should be developed using Django Framework.

Audio Loader

The audio loader is part of the Music Classifier library. It is the component responsible to load audio from a storage to the run time environment, which will eventually feed the training model. It should be developed in such a way that is capable to be extended to load data from different storages.

For the context of this project, at least two different storages should be provided. One local, and one loading data from a cloud provider. For this project, three different cloud providers are considered: Microsoft Azure, Amazon Web Services and Google Cloud Platform. The decision of which one to use will be decided in the following chapters.

As it is common to the Music Classifier library, all components of this library will be developed in Python.

Audio Converter

The audio converter is a component that should also be available through the Music Classifier library, and it is responsible to provide an easy to use API that is capable of converting audio file formats into another. Previous literature (see section 4.2) showed audio file format to be a challenge. There is not yet a standard way to do this, and depending on the dataset to use, it requires different types of conversion. This component should provide an extensive format conversion.

Furthermore, it will be useful to accept audio formats from the users that are not supported by the model out of the box.

DataPreprocessor

The data preprocessor component will be responsible to handle any additional requests that are necessary before feeding the model.

The model will be feed through images, so it will be up to the developers to decide which features from the audio they want to be trained (see section 4.2).

This component should also provide an API for usage, as it will be part of the Music Classifier library.

Evaluator

The evaluator is a simple component that calls the model to classify a music genre.

Storage

The dataset the be handled needs to be stored somewhere. Depending on which dataset to use and its size, local storage might not be enough. The design stage should consider different cloud providers as a way to store and retrieve data. The implementation of this component must consider best software design practices and be designed in such a way that adding a new storage type can be achieved by the Open/Closed principles of SOLID.

Model

The model will be creating by using Pytorch Framework. The decision to use Pytorch relies on the literature studied (see 4.2), where it achieved the best result in terms of accuracy.

The syntax of Pytorch was also considered easier to grasp than Tensowflow, or other presented frameworks (see section 4.1.1).

While it is still not clear which neural network architecture to use, the decision should be closed to one of the following: a Convolutional Neural Network (CNN), a Recurrent Neural Network (RNN) or a Long Short Term Memory Cell (LSTM). Only the implementation in the next chapter will clarify which architecture is the best to use for the problem in context.

Transfer learning is also considered to be used. Furthermore, the size of the dataset to train and the architecture chosen might implicate data the model is trained in a cloud provider.

Further analysis in infrastructure costs will decide if a cloud provider can be used to improve speed and accuracy of the model.

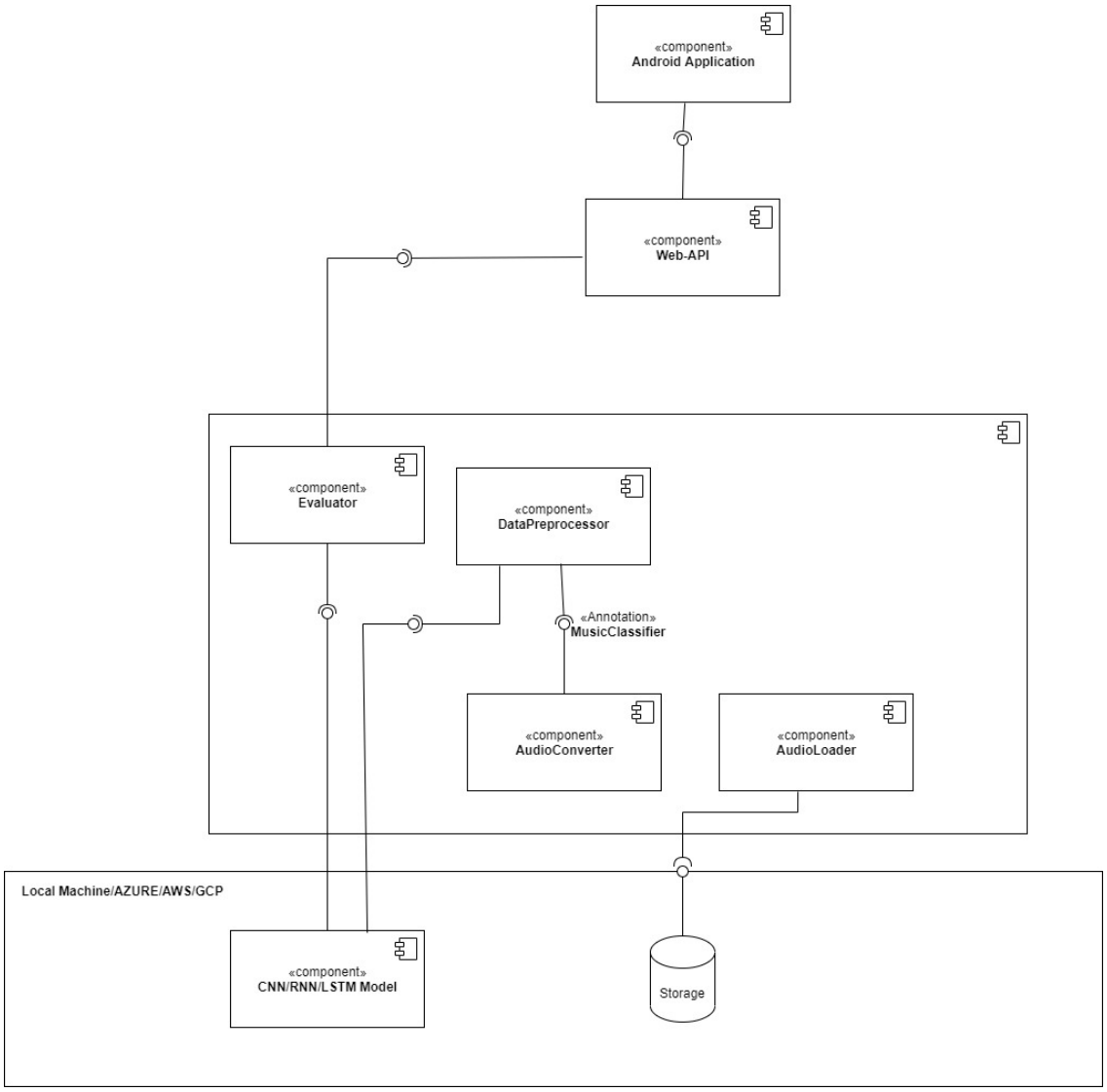

Figure 10 - Design proposal for the project

5.2 Implementation intention

Based on the presented design (see section 5.1), two different concretizations of the implementation are desired as an outcome of this project.

First Implementation

Existing literature has proven an efficient usage of the GTZAN dataset to classify music genres. Accuracy of around 70% has been presented. The disadvantage of using such dataset is that is limited in terms of genres that are created.

In order to reduce risks and not achieving any concrete results, a first implementation proposal is to use the GTZAN dataset and try to improve the accuracy presented on the lectures

For this first implementation, the following ten genres will be classified: Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae and Rock.

Second Implementation

The second Implementation should take advantage of the components already built during the first implementation and reuse them to its advantage.

Only the model should be recreated, and eventually extend the Music Classifier library to support new use cases.

This implementation should become a true research and development project. In order to provide an innovation to the community and accomplish the goal set for this work (see section 2.5), the implementation will focus on a specific genre and should classify its subgenres instead.

The Free Music Archive dataset (FMA), amongst other genres, provides Electronic music samples.

The Electronic genre is subdivided into 19 different subgenres.

It is intention of this implementation to provide a deep learning model capable of classify the following nineteen subgenres: Ambient Electronic, Breakcore-Hard, Chip Music, Chiptune, Dance, Downtempo, Drum & Bass, Dubstep, Skwee, Glitch, House, Chill-Out, IDM, Jungle, Minimal Electronic, Tecnho, Bigbeat, Trip-hop, Vaporware.

Library compatibility

Still focusing on the community goal (see section 2.5), if any additional time be left, it is desire of this project to explore how the proposed Music Classifier library can be extended to be written in different languages.

Python is the chosen implementation, but an alternative implementation in .NET is desired to be explored.

6 Evaluation and experimentation

In this section, concepts regarding the evaluation of the project are presented. After reading this chapter, a significant understanding of the techniques that are used to determine if the goal of this work is correctly achieved.

6.1 Hypothesis

Providing a guidance on how to increase the accuracy of a deep learning model algorithm to classify music genres is the main goal of this work (see section 2.5).

Using a Test of Hypothesis, it is possible to find a guideline that will determine which data exploration should be done during the implementation and training stages.

To formalize a Test of Hypothesis, it is required that the null hypothesis (H0) and an alternative hypothesis (H1), are defined. Therefore, and in order to be aligned with the main goal of this work, the following hypothesis are defined towards it

H0

The number of samples from a specific genre does not affect the accuracy of that genre.

H1

The number of samples from a specific genre does affect the accuracy of that genre.

The Test of Hypothesis above will influence the way the evaluation of the model is going to be done. Preparing training, validation and test sets (see section 6.2) will be directly related towards validation the hypothesis. Furthermore, to help validate the hypothesis, a confusion matrix should be implemented in each proposed implementation (see section 6.3).

6.2 Deep learning evaluation

Deep learning models that successfully perform an evaluation to the problem they are intended to solve, depend on a correct use of three different types of datasets. Training, validation and test datasets are the known datasets to be used when building deep learning models.

Training dataset

The training dataset is the actual dataset used to train the model. All weights and biases are applied within this model. Every epoch during training is affected by what the neural network sees from the data present in this dataset.

Validation dataset

The validation dataset is a sample of data used to provide an unbiased evaluation of a model during training (Tarang Shah, 2017). At the end of each epoch, the results obtained by the evaluation of this dataset will affect the hyperparameters tuning during training. This dataset helps machine learning engineers find the correct tuning for the hyperparameters model.

Test dataset

The test dataset is used at the end of the training, in order to access the actual performance of the model. When the model performs equally on the test and the training dataset, it indicates a correct training of the model, while a lower performance indicates an incorrect training, due to overfitting or any other potential problem.

In this work, these datasets will be originated as a subset of the original datasets presented (see section 4.3.1).

A general rule of thumb accepted by the machine learning community is presented in Figure 11. 60% of the available data is used to actually train the model, with the training dataset. 20% are used for the validation dataset, also known as the developer dataset, to help fine tuning the model hyperparameters. The remaining 20% are reserved for the test dataset.

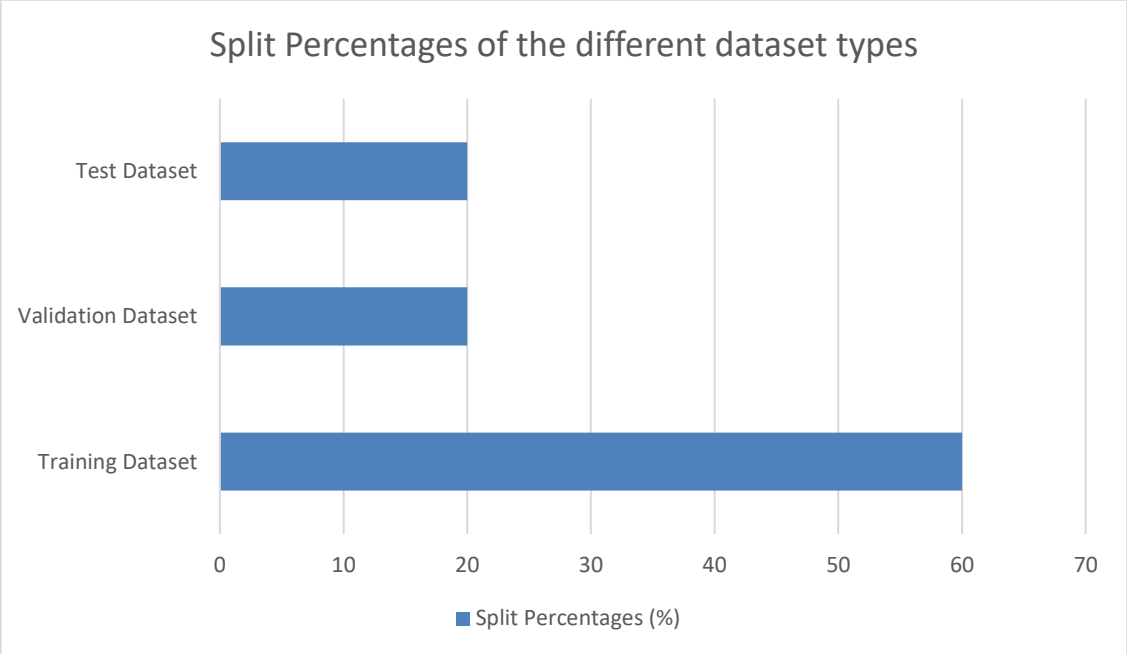

Figure 11 - Split percentages of different dataset types

To summarize this chapter, Figure 12 provides a visual help to highlight the main differences between datasets. The training dataset is used to train the model, and it learns from what is sees on the dataset during training. The validation dataset provides essential information to

the developer team to understand which hyperparameters are best suited for the problem in question. Finally, the test dataset determines that the model is fit to solve the intended problem.

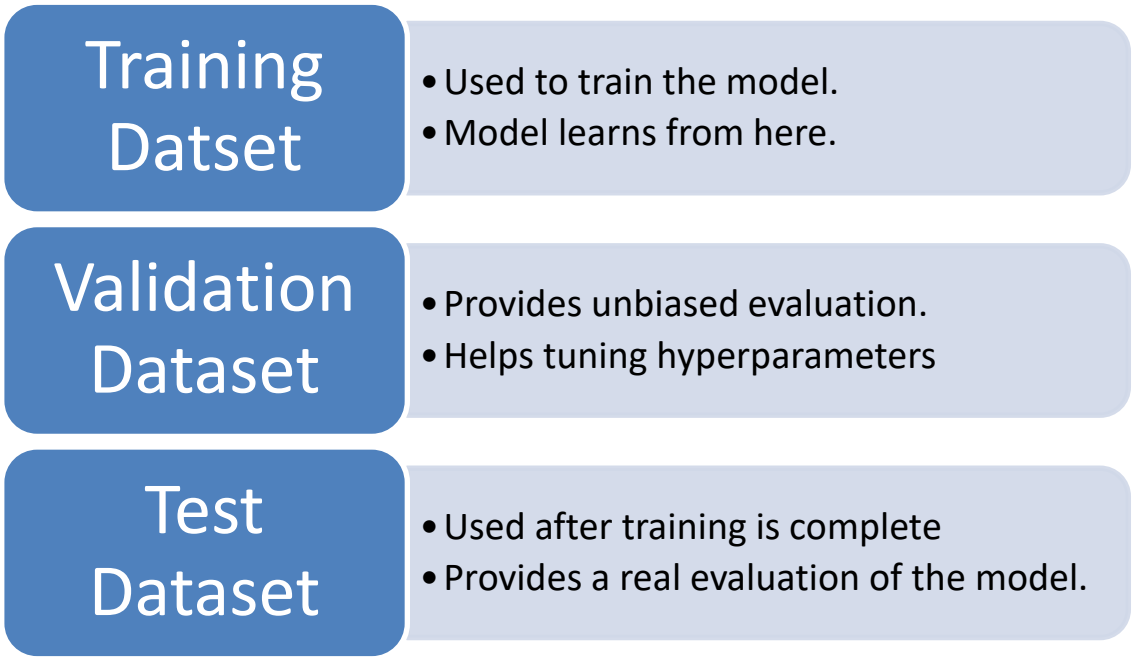

Figure 12 - Comparison between the different datasets used in deep learning models

6.3 Project classification analysis and confusion matrix

In this section, the method used to evaluate the deep learning model to be developed is analyzed.

The implementations proposed during design stage (see section 5.2), fit in a multiclass classification problem. This is a type of classification where two or more classes are classified as the output of the neural network.

Literature on music genre classification (see section 4.3.2) focus on accuracy, which is a metric that has been taken into consideration when evaluating the model, as an outcome of this work.

Presenting and studying the performance of a model not only depends on accuracy. As a matter of fact, there are some factors that might change the meaning of a high accuracy on a deep learning model.

Imbalanced dataset

The concept of an imbalanced dataset refers to a case where the labeled data on the dataset is different from class to class. For instance, if a dataset that contains ten different genres with a total of one thousand samples, and nine hundred of those samples refer to one single class, the model will be trained with imbalanced data. If the test dataset is also based on this data, there is a high chance that the model will provide a high accuracy without actually having a good performance. There is a high chance that the overfitted data will be correctly classified most of the times, but for the other nine genres in real world usage, it will perform poorly.

There are some techniques to overcome an imbalanced dataset:

Under-sampling

Remove samples from over-represented classes. This can happen when the dataset used has a reasonable amount of data.

Over-sampling

Add more samples from under-represented classes. This can happen when not enough data is available. Typically happen when datasets used for training are small.

To achieve a good evaluation of the model, the following evaluation metrics should be considered. During the next chapters, further analysis on the available model will be done, and each metric is used according to the most fit use case.

For multiclass classification, a confusion matrix is a good way to evaluate the model beyond the accuracy metric, as it shows the complete performance of the model.

The advantage of using a confusion matrix is that the accuracy can be classified per genre, instead of just relying on a general model accuracy, which might hide poorly performance in real world scenarios.

When studying the number of samples per genre, the confusion matrix helps identify potential issues with the dataset, and therefore, suggest improvements for future work.

Table 9 shows an unfilled confusion matrix for the first proposed implementation (see section 5.2), which is based on the GTZAN dataset. The black cells represent the cells of the confusion matrix that are expected to have a higher accuracy value, indicating a good model training. Lower values on the black cells might indicate that something went wrong, or that over-sampling needs to be applied in future trainings. On the other hand, higher values on the blank cells are also not desirable, and might indicate overfitting problems in the model, as well as the need of performing under-sampling.

For completeness, a similar confusion matrix should be presented for the second implementation proposal.

Blues										
Classical										
Country										

Disco										
Hiphop										
Jazz										
Metal										
Pop										
Raggae										
Rock										
	Blues	Classical	Country	Disco	Hiphop	Jazz	Metal	Pop	Raggae	Rock

Table 9 - Unfilled confusion matrix table for first implementation

7 References

Becker, D. (2017). *Rectified linear units relu in deep learning.* Fonte: Kaggle: https://www.kaggle.com/dansbecker/rectified-linear-units-relu-in-deep-learning

Bertin-Mahieux. (2011). *Million Song Dataset.* Fonte: Columbia: https://www.ee.columbia.edu/~dliang/files/FINAL.pdf

Bronwlee, J. (2019). *Transfer learning for deep learning.* Fonte: Machine Learning Mastery: https://machinelearningmastery.com/transfer-learning-for-deep-learning/

Bronwlee, J. (2020). *Train test split for evaluating machine learning algorithms.* Fonte: Machine Learning Mastery: https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/

Brownlee, J. (2019). *Pooling layers for convolutional neural networks.* Fonte: Machine Learning Mastery: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/

Culerciello, E. (2017). *Neural Network Architectures.* Fonte: Towards Data Science: https://towardsdatascience.com/neural-network-architectures-156e5bad51ba

Ellis, K. (2019). *Learning embeddings for music recommendation with mxnets sparse api.* Fonte: Medium: https://medium.com/apache-mxnet/learning-embeddings-for-music-recommendation-with-mxnets-sparse-api-5698f4d7d8

Giangiullo, J. (s.d.). *2017.* Fonte: https://social.msdn.microsoft.com/Forums/exchange/en-US/eab8b6b1-c4cf-4a75-9dfd-a95139bbd3e8/artificial-intelligence-hierarchy?forum=MachineLearning

Gupta, N. (2017). *Top benefits of mahcine learning in business.* Fonte: Hackernoon: https://hackernoon.com/top-benefits-of-machine-learning-in-business-lu3o422ig

Han, Y. (2020). *What is machine listening.* Fonte: Medium: https://medium.com/cochl/what-is-machine-listening-part-1-6fbdf2a3d892

Hermez, C. (2020). *Music genre classification with tensorflow.* Fonte: Towards Data Science: https://towardsdatascience.com/music-genre-classification-with-tensorflow-3de38f0d4dbb

Hermez, C. (2020). *Music genre classification with Tensorflow.* Fonte: Towards Data Science: https://towardsdatascience.com/music-genre-classification-with-tensorflow-3de38f0d4dbb

Hermez, C. (2020). *Understanding the mel spectogram.* Fonte: Medium: https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

Hingham, T. (2019). *NEARLY 40,000 TRACKS ARE NOW BEING ADDED TO SPOTIFY EVERY SINGLE DAY.* Fonte: Music Business Worldwide: https://www.musicbusinessworldwide.com/nearly-40000-tracks-are-now-being-added-to-spotify-every-single-day/

Hope, C. (2020). *Audio.* Fonte: Computer Hope: https://www.computerhope.com/jargon/a/audio.htm

Ippolito, P. (2019). *Feature Extraction Techniques.* Fonte: Towards Data Science: https://towardsdatascience.com/feature-extraction-techniques-d619b56e31be

Khatana, A. (2020). *Music genre classification using transfer learning pytorch.* Fonte: Medium: https://medium.com/swlh/music-genre-classification-using-transfer-learning-pytorch-ea1c23e36eb8

Kumar, P. (2020). *Music genre classification using feed forward neural network using Pytorch.* Fonte: Medium: https://medium.com/@pk_500/music-genre-classification-using-feed-forward-neural-network-using-pytorch-fdb9a960a964

Le, J. (2018). *A gentle introduction to neural networks for machine learning.* Fonte: Medium: https://medium.com/cracking-the-data-science-interview/a-gentle-introduction-to-neural-networks-for-machine-learning-d5f3f8987786

Mathworks. (2020). *Music genre classification uing wavelet scattering*. Fonte: Mathworks: https://www.mathworks.com/help/wavelet/ug/music-genre-classification-using-wavelet-scattering.html

McCarthy. (2004). Fonte: Citesserx: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1090.836&rep=rep1&type=pdf

McKinney, K. (2015). *Why are songs 3 minutes long.* Fonte: Vox: https://www.vox.com/2014/8/18/6003271/why-are-songs-3-minutes-long

Nabi, J. (2018). *Machine learning multiclass classification with imbalanced data set.* Fonte: Towards Data Science: https://towardsdatascience.com/machine-learning-multiclass-classification-with-imbalanced-data-set-29f6a177c1a

Nabi, J. (2019). *Hyper parameter tuning techniques in deep learning.* Fonte: Towards Data Science: https://towardsdatascience.com/hyper-parameter-tuning-techniques-in-deep-learning-4dad592c63c8

NVIDIA. (2021). *Spectogram.* Fonte: Docs Nvidia: https://docs.nvidia.com/deeplearning/dali/user-guide/docs/examples/audio_processing/spectrogram.html

Opperman, A. (2019). *What is deep learning and how does it work.* Fonte: Towards Data Science: https://towardsdatascience.com/what-is-deep-learning-and-how-does-it-work-2ce44bb692ac

Pandey, P. (2018). *Music genre classification with Python.* Fonte: Towards Data Science: https://towardsdatascience.com/music-genre-classification-with-python-c714d032f0d8

Schimtt, M. (s.d.). *Machine Learning Project Architecture.* Fonte: Data Revenue: https://www.datarevenue.com/en-blog/machine-learning-project-architecture

Shah, T. (2017). *Train, Validation and Test sets.* Fonte: Towards Data Science: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7

Shorten, C. (2019). *Introduction to resnets.* Fonte: Towards Data Science: https://towardsdatascience.com/introduction-to-resnets-c0a830a288a4

Sirsat, M. (2019). *Confusion Matrix.* Fonte: Manisha Sirsat Blogspot: https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Tran, K. (2020). *What is Pytorch.* Fonte: Toward Data Science: https://towardsdatascience.com/what-is-pytorch-a84e4559f0e3

Tzanetakis, G. (2002). Fonte: CMU: http://www.cs.cmu.edu/~gtzan/work/pubs/tsap02gtzan.pdf

Yoss, B. (2019). *The importance of frameworks.* Fonte: Medium: https://medium.com/@benyoss4/the-importance-of-frameworks-2c4a04d20ac5