UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

AI-Music

Presentado por José Ángel López Estrada
en Universidad de Burgos — 19 de junio de 2023
Tutores: César Ignacio García Osorio, Alicia Olivares Gil

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. nombre tutor, profesor del departamento de nombre departamento, área de nombre área.

Expone:

Que el alumno D. José Ángel López Estrada, con DNI 21071560H, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado título de TFG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 19 de junio de 2023

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. nombre tutor	D. nombre co-tutor

Resumen

El rápido crecimiento del consumo de música digital ha generado una gran cantidad de datos de audio, lo que ha priorizado la necesidad de contar con métodos de clasificación y categorización. Plataformas como *Spotify* o *Apple Music* con bibliotecas que superan los 100 millones de canciones y un crecimiento constante de decenas de miles de canciones diariamente, resaltan la importancia de contar con una clasificación musical precisa y eficiente.

En este proyecto, se utilizan métodos de inteligencia artificial (IA) con el fin de clasifiicar de forma automática diferentes géneros musicales a partir de un análisis de las características del audio.

La aplicación desarrollada en el proyecto se presenta como una plataforma web que permite a los usuarios cargar y analizar sus canciones.

Descriptores

machine learning, big data, inteligencia artificial, clasificación, música, python, flask, servidor web ...

Abstract

The rapid growth of digital music consumption has generated a large amount of audio data, which has prioritized the need for classification and categorization methods. Platforms like *Spotify* or *Apple Music*, with libraries exceeding 100 million songs and a constant growth of tens of thousands of songs daily, highlight the importance of having accurate and efficient music classification.

In this project, artificial intelligence (AI) methods are used to automatically classify different music genres based on an analysis of audio features.

The application developed in the project is presented as a web platform that allows users to upload and analyze their songs.

Keywords

machine learning, big data, artificial intelligence, classification, music, python, flask, web server

Índice general

Índice de figuras

Índice de tablas

Introduction

1.1. Context

Music is a culturally significant and important element, with a wide variety of styles and genres. The automated identification of musical styles is an emerging field of study, aiming to create intelligent systems capable of automatically recognizing and classifying the musical style of a song or music piece without any human intervention.

This field of study has important applications in areas such as music recommendation or digital audio library organization.

The application of artificial intelligence algorithms has shown considerable success in this area.

The present work aims to develop an automatic music style recognition system using audio processing and artificial intelligence algorithms.

1.2. Project Structure

The present document follows the next structure:

- **Introduction**: context and description of the problem, project structure and relevant links.
- **Project objectives**: explanation of project's main objectives.
- **Theoretical concepts**: short explanation of the main theoretical concepts that are necessary to understand the problem addressed in the project

- **Techniques and tools**: present the methodology, techniques and development tools used in the project's development.
- **Relevant aspects of project development**: important aspects about the project development.
- **Related works**: state of the art and related works in the area.
- **Conclusions and future works**: conclusions obtained after the completion of the project and possible future lines of work.

In addition, *Annexes* document contains the following structure:

- **Appendix A. Manuals**:
- **Appendix B. Requirements Specification**:
- **Appendix C. Design Specification**:
- **Appendix D. Technical Documentation**:
- **Appendix E. User Documentation**:

Project objectives

The general objective of this work is to design and develop an automatic music style recognition system using artificial intelligence. To achieve this general objective, the following specific objectives are proposed:

Data collection

The process of data collection involves gathering and organizing various types of information related to the data that we want to study. In this project, this include audio tracks and metadata obtained from FMA (Free Music Archive) dataset.

Data preprocessing

Preprocessing data in a musical dataset involves a series of steps aimed at cleaning, transforming, and preparing the data for further analysis or modeling tasks.

Data processing

Data processing in this project means extract relevant features to get relevant information to perform the classification. Features like MFCC, Spectogram or Chromagrams are suitable for this task.

Selection and implementation of models

The selection and implementation of models involve choosing appropriate machine learning algorithms and techniques for music style recognition.

Model evaluation

Model evaluation involves measuring the performance and effectiveness of the implemented machine learning models with metrics like Accuracy, Precision or F1 Score.

User interface

The development of a user interface allows the user to interact with the music recognition model in an easy way.

Deployment

Deploy the web application on a web server or a cloud platform to ensure that is accessible to users.

Testing and quality assurance

Testing the application and web interface to identify and fix any bugs or issues. This process is crucial in the entire life-cycle of the application and development process.

Theorical Concepts

Before starting with the project development, it is necessary to explain a series of theoretical concepts.

3.1. Sound

Sound is a mechanical vibration propagating through an elastic medium, such as air, water or any other material. These vibrations generate some pressure differences in the medium, which are taken by our ears and perceived as sound.

Mathematically, sound can be represented by a mathematical function $f(t)$, where t represents time. This function describes how the pressure of particles in the medium varies as time passes.

Mathematical Representation of Sound

The most common mathematical representation of sound is the sine wave. A sine wave can be described by the following equation:

$$f(t) = A\sin(2\pi ft + \phi) \tag{3.1}$$

Donde:

- A is wave amplitude, which represents the maximum deviation of the wave from its equilibrium position.
- f is wave frequency, which determines the number of complete cycles the wave makes in one second.

- t is the time.
- ϕ is the initial wave phase, which determines the horizontal displacement of the wave over time.

3.2. Music recognition

Music recognition is an area that aims on the analysis of audio features in order to extract relevant information. For example, songs identification, musical genres, or artists.

Sound features

Music recognition is based on the analysis of diverse sound features. Some of the most common are:

- **Rythm**: Rhythm is a fundamental property of music and refers to the temporal organization of sound events. In music recognition it can involve tempo detection and analysis of musical patterns.
- **Frequency**: Musical frequency is the number of vibrations or oscillations in the sound, per second. In music recognition, frequency spectrums can be analyzed to identify the musical notes.
- **Timbre**: Refers to the tonal and harmonic characteristics that distinguish different instruments and voices. Music recognition can examine the timbre of an audio signal to identify the instruments used in a song for example.
- **Music structure**: Music structure refers to the organization of a musical composition. In music recognition, structure analysis can detect changes and repetitions in the different parts of a song and identify specific musical styles.

Musical Styles and Recognition

Different musical styles often have distinctive characteristics that can be discovered in musical recognition. For example, certain genres have characteristic rhythms and harmonic patterns. These features can be identified using machine learning algorithms trained on a wide variety of labeled audio samples.

Applications of Music Recognition

Music recognition has several applications, some of which are:

- **Music recommendation**: Music recognition algorithms are used to recommend music to users based on their preferences and listening patterns. These systems analyze the features of the songs listened by the user and create a recommendation model based on their tastes.
- **Genre classification**: Music recognition is used to automatically classify songs into different music genres, making it easier to organize and search for music in large digital libraries.

3.3. Theorical examples of music features extractions

Spectrogram

A spectrogram is a visual representation of the frequency spectrum of an audio signal as a function of time. It provides detailed information about how the sound is distributed over the different frequencies.

The process for obtaining a spectrogram of a song is explained below.

- **Audio signal preprocessing**: Audio signal is divided into segments, in a process called windowing. In this way, it is possible to analyze the spectral variation at different points of the signal over time.
- **Short Time Fourier Transform (STFT)**: Each segment of the signal is applied to a short time Fourier transform (STFT). STFT divides the signal into multiple time segments and calculates the sum of different frequencies in each segment. This is accomplished by applying a time window to each segment and then calculating the Fourier transform of each window.
- **Spectrum magnitude calculation**: The STFT provides various information about the phases and amplitudes of the frequencies in each time segment. However, to construct a spectrogram, only the magnitude of the spectrum (absolute amplitude of the frequencies) is taken.

- **Visual representation**: Spectrum magnitude is represented visually in a 2D graph, where the horizontal axis represents time and the vertical axis represents frequencies. The intensity of the color or brightness at each point on the graph indicates the energy or amplitude of the corresponding frequency.

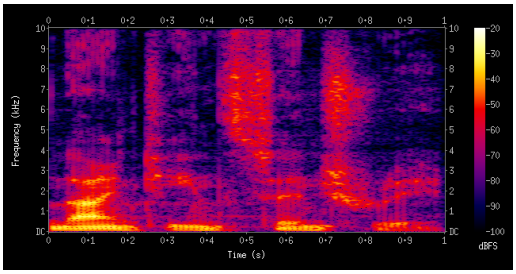

Figura 3.1: Audio track spectrogram.

Mel Frequency Cepstral Cepstral Coefficients (MFCC)

Mel frequency cepstral coefficients (MFCCs) are widely used features in audio signal processing and speech recognition. These coefficients represent the spectral characteristics of an audio signal as a function of the Mel scale, which is a frequency scale based on the response of human ear.

The process for obtaining the MFCC coefficients of a song is explained below.

1. **Pre-emphasis**: Audio signal is normalized with a pre-emphasis filter, which highlights high-frequency frequencies and compensates for the attenuation of lower frequencies. This helps to improve the signal-to-noise ratio and enhance relevant features in the spectrum.

2. **Splitting**: Pre-emphasis signal is split into short, overlapping frames or segments over time. This is done to capture the spectral variation at different points in the signal over time.

3. **Short Time Fourier Transform (STFT)**: To each frame of the signal, the Short Time Fourier Transform (STFT) is applied, which calculates the contribution of different frequencies in each frame. The STFT provides various information about the phases and amplitudes of the frequencies in each time segment.

4. **Mel filter bank**: A Mel filter bank is applied, which consists of a series of triangularly spaced filters on the Mel scale. These filters are used to represent the spectrum in terms of Mel frequency bands.

5. **Logarithm of the energy**: This is done to include nonlinear response of the human ear to frequencies.

6. **Transformada de Coseno Discreta**: Discrete Cosine Transform (DCT) is applied to the values obtained above.

7. **Extraction of the MFCC coefficients**: Finally, the most significant cepstral coefficients are selected to represent the spectral information of the audio signal. These coefficients are the ones used as features for audio processing and recognition applications.

MFCC coefficients are widely used in applications such as speech recognition, speaker identification and voice synthesis.

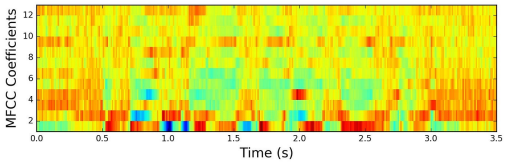

Figura 3.2: Audio track MFCC.

3.4. Artificial Intelligence

Artificial intelligence (AI) is the ability of a computer system to mimic human cognitive functions such as learning and problem solving.

Artificial intelligence systems can analyze large amounts of data, recognize patterns and make decisions based on that information. They can learn from experience and improve their performance.

There are different approaches to AI, including *machine learning*, *natural language processing*, *computer vision* and robotics, among others.

AI is used in a wide variety of applications such as recommendation systems, data analysis or medical diagnostics, for example.

Aprendizaje automático (Machine Learning)

Machine learning is a subfield of artificial intelligence that focuses on the development of algorithms and models that allow systems to learn and extract information from data, without being explicitly programmed to do so. There are several types of machine learning:

- **Supervised learning**: A set of labeled training data is provided as input to the algorithms. The model learns to make predictions or decisions based on these labeled examples. Supervised learning is commonly used in classification or regression tasks.
- **Unsupervised learning**: Algorithms work with unlabeled data sets, i.e. with no known class. The goal is to find hidden patterns or structures in the data. Unsupervised learning is used in tasks such as (*clustering*).
- **Reinforcement learning**: In this type of learning, an intelligent agent interacts with its environment and learns to make decisions according to a series of rewards or penalties. The goal is to find a policy that maximizes the rewards received. Reinforcement learning is used to train agents in video games or robotics.
- **Semi-supervised learning**: In this type of learning the dataset is not completely labeled, therefore, the objective is to maximize the performance of the model from known data.

3.5. Example of supervised learning

In this project an AI approach using supervised learning is going to be used. Therefore, an example will be detailed in more detail:

Objective

Build a machine learning model to classify emails as "spam.ºr "non-spam".

Dataset

Labeled dataset containing 1000 emails, where each email has features such as the frequency of certain words suspected to belong to spam, message length, presence of links or images, for example. **In addition to including the class to which it belongs: "Spam.ºr "Not Spam"**.

Data preparation

The data must be prepared for model training. A suitable option is to represent each e-mail as a feature vector.

Email	Suspected spam words	Lenght	Links	Class
1	1	120	0	Not spam
2	4	56	1	Spam
3	1	352	2	Not spam
4	9	174	0	Spam

Tabla 3.1: Example of training data in a supervised learning problem

Model training

Once the data is processed in a suitable format, it is fed into a supervised learning algorithm creating a machine learning model. For example, Artificial Neural Networks, Support Vector Machines or Decision Trees.

Prediction

Once the model is trained, it is fed with external data and predictions are made.

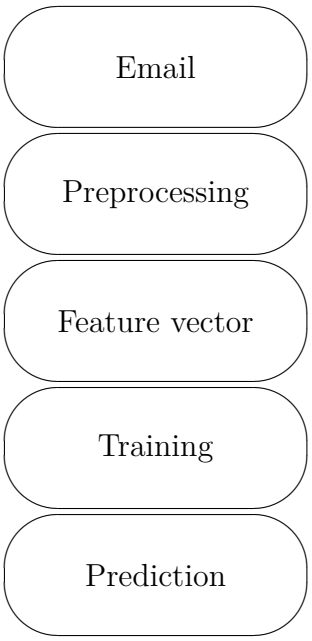

Figura 3.3: Example of supervised learning process

Techniques and tools

The objective of this part of the project is to present the methodological techniques and development tools used in the project's development.

4.1. Tools

- Python: Python is a high-level programming language widely used in the field of machine learning and artificial intelligence. It has a high number of libraries and frameworks that facilitate data processing and implementation of artificial intelligence algorithms. In this project it has been used to develop all the back-end part of the application.
- librosa: librosa is a Python library used for audio analysis and processing. It provides a wide range of methods to extract audio features in a simple way. Some examples are spectrograms, mel frequency cepstral coefficients (MFCC) or chromagrams. In this project it has been used to process and extract various audio features to feed the artificial intelligence algorithms and create the model.
- TensorFlow: TensorFlow is an open source library used in the field of machine learning. It provides a simple interface for the implementation of neural networks and other machine learning algorithms. TensorFlow has been the chosen option to train the neural network algorithms of the project.
- scikit-learn: scikit-learn is a Python machine learning library that provides a wide range of algorithms and tools for data analysis and model building. Includes functions for splitting datasets into *train* and *test*.

- NumPy: NumPy is a Python library used to perform numerical calculations on matrices and multidimensional arrays. It provides a wide range of mathematical functions and tools for efficient handling of numerical data.

- Pandas: open source Python library that provides data analysis tools. It is very used in data science fields.

4.2. Justification

Python

The following includes some of the justifications that have led to the development of the project in Python language compared to other languages.

Extensive variety of libraries and frameworks: Python has a broad range of libraries and frameworks specialized in machine learning, such as TensorFlow, scikit-learn, PyTorch, Keras and many others. These libraries provide tools and implementation of the main artificial intelligence algorithms, which facilitates the task of developing a machine learning project.

Integration with other technologies: Python easily integrates with other existing technologies in machine learning development area. For example, it can be combined with relational databases (MySQL), visualization tools (matplotlib, Seaborn), data analysis (Pandas) or in this case, music processing tools (librosa).

TensorFlow vs scikit-learn

The choice to use TensorFlow instead of scikit-learn is based on several considerations.

First, TensorFlow is suitable for projects involving deep learning and neural networks. It provides a higher number of tools for creating, training, and deploying deep learning models than scikit-learn.

TensorFlow allows implementing complex neural network architectures, such as convolutional neural networks (CNNs), which are particularly well suited for this project.

Finally, TensorFlow supporst graphics cards (GPUs) to perform the training process. This provides a very significant performance advantage,

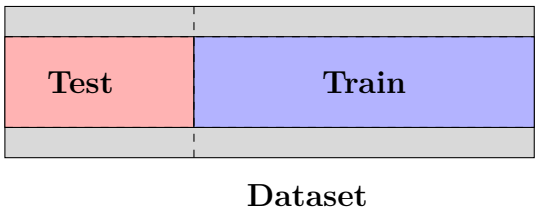

Figura 4.1: Dataset splitting

as the large number of cores in a GPU can be used to perform parallel and distributed computations. The result is a considerably faster training time compared to a GPU.

4.3. Techniques

Several techniques and methodologies have been used to develop and train the machine learning models.

(A much more detailed description of the methodological development of the project can be found in the document *Annexes*).

4.4. Processing and extracting audio features

librosa has been used for the audio processing and features extraction.

- MFCC (mel frequency cepstral coefficients): These coefficients capture the spectral features of audio and are commonly used in speech recognition and audio classification tasks. They are the main features that have been used to train the model and make predictions.

4.5. Dataset splitting

scikit-learn library has been used to split the dataset into training, test and validation data. Scikit-learn provides a function called *train_test_split* that splits the dataset into parts intended for training and model evaluation. This data set splitting technique is critical to evaluate the model's generalizability and to avoid overfitting.

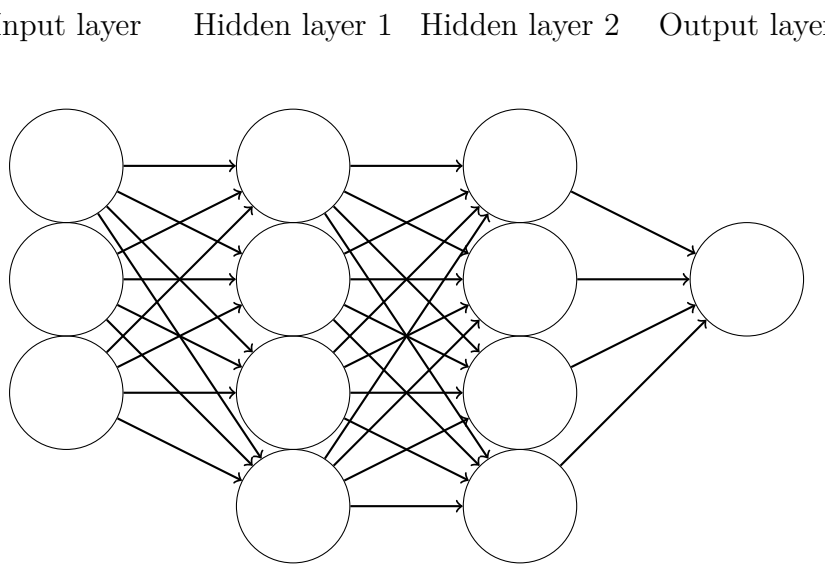

Figura 4.2: Multi-layer neural network example

4.6. Neural networks with TensorFlow

Neural networks implemented with TensorFlow library have been used to train the data.

In this project, various neural network architectures have been used, such as convolutional neural networks (CNN) and multilayer neural networks. These architectures are suitable for audio processing tasks and have proven to be effective in audio classification and pattern recognition.

Neural networks are trained by updating the weights and biases of the network iteratively to minimize *loss*. Once trained, neural networks make predictions on new audio data, classifying them into categories or musical styles.

4.7. Development with Python and Flask

Web application development part was implemented using Python and the Flask framework. Flask is a web framework that allows building web applications in Python.

The back-end of the application has been developed in Flask, which is responsible for receiving user requests, processing audio data and returning classification and analysis results. All this through the implementation of

RESTful API, which provides continuous communication between the user and the server.

Relevant aspects of the project development

5.1. Project Development

The development of this project has been conducted using an agile methodology based on Scrum. This approach provides flexibility to adapt to changes in requirements and to iteratively improve the project during its development. Agile methodologies are based on the idea of splitting the project into iterative cycles called "sprints", where planning, development, testing and review activities are done. Sprints usually have a fixed duration, in this case they have been 1 week each, with some justified exceptions.

Scrum Methodology

The project development process is explained in detail in the document *Annexes*.

In summary, the development consisted of the following iterative steps:

1. **Initial Phase**: Definition of the project's general objectives. Creation of the Product Backlog.
2. **Sprint planning meetings**: At the beginning of each sprint, a small meeting is held to review the tasks performed and select the tasks to be developed.
3. **Sprint development**: During the sprint, work is done on each of the assigned tasks.

4. **Continuous improvement**: Agile methodology promote continuous improvement throughout the development of the project.

Analysis

The objective of the project is to apply Artificial Intelligence (AI) and Machine Learning (ML) techniques for classification of musical styles. During the analysis phase, several ML models were studied, to finally choosing the Convolutional Neural Network (CNN) due to its higher efficiency.

Another factor to take into account in an AI project is the dataset. Various datasets have been thought of to train the model such as:

- **GTZAN Dataset**: dataset widely used in music classification. Compiled by George Tzanetakis in 2002, it consists of **1000 audio fragments** of **30 seconds**, distributed in **10 musical styles**.
- **Million Song Dataset**: dataset formed of *musical features and metadata* of one million popular songs. It does not include audio tracks.
- **MagnaTagATune**: dataset consisting of **25863 audio fragments** of **29 seconds**.
- **FMA (Free Music Archive)**: royalty-free music dataset formed of **106574 audio fragments** of **30 seconds**.

Finally, **FMA (Free Music Archive) dataset of 7.2 GiB**, with 8000 audio fragments has been chosen to perform the training of the model. This is because it offers a large amount of royalty-free music tracks. In addition, by including the music tracks (unlike Million Song Dataset for example) it is possible to extract the features manually using librosa or other tools.

Design

Regarding the design of the application, at first the development of a desktop application was considered. However, this idea was abandoned and it was opted to develop a web application due to the following reasons:

- **Technological evolution**: It is a reality that web applications are leading the current technological moment, so making the project on the web is a good way learn about this technology.

- **Accessibility**: The web application is accessible from any device with an internet connection, regardless of the operating system. In this way, the scope of the application is extended.
- **Updates**: When a web application is updated, users automatically receive the latest version available in production, avoiding the need to download and install updates manually.
- Maintenance: It is generally easier to maintain a web application than a desktop application, as factors such as the user's hardware or operating system disappear.

The framework chosen for the implementation of the web application was:

- **Backend**:For the backend the selected option is Flask, a Python framework, designed for web applications and APIs.
- **Frontend**: For the frontend, HTML and CSS have been chosen. HTML is a markup language that defines the structure and contents of the web pages that form the application. CSS is used to define the styles of HTML documents and add attractive elements for the user.

5.2. Extraction of audio features

This section will discuss the audio feature extraction process.

MFCC extraction using Python and librosa

MFCC extraction method is designed to loop through all files in a specified directory and extract the MFCC features of each audio file it finds. The explanation of the relevant details is as follows:

```
y, sr = librosa.load(item)

if librosa.get_duration(y=y, sr=sr) < 25:
continue

y = y[:(25 * sr)]
mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=10)
mfcc_normalized = (mfcc - np.mean(mfcc)) / np.std(mfcc)

mfccs[int(item.stem)] = mfcc_normalized.ravel()
```

1. **y, sr = librosa.load(item)**: Load audio file. Returns both the audio signal (y) and the sample rate (sr).

2. **if librosa.get_duration (y=y, sr=sr) < 25**: Audio length check. If the audio is shorter than 25 seconds, the file is skipped.

3. **y = y[:(25 * sr)]**: Trims the audio signal to the first 25 seconds.

4. **mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc =10)**: Extraction of 10 MFCC coefficients.

5. **mfcc_normalized = (mfcc - np.mean(mfcc)) / np.std(mfcc)**: Normalization of MFCC coefficients by subtracting the mean and dividing by the standard deviation.

6. **mfccs[int(item.stem)] = mfcc_normalized.ravel()**: The normalized MFCC coefficients are flattened into a 1D matrix and stored in a dictionary.

Extraction of other characteristics

To compare performance and select the best possible set of audio features to perform the classification, the process was repeated with:

1. **Spectrograms**: librosa.feature.melspectrogram(y=y, sr=sr)
2. **Chromagrams**: librosa.feature.chroma_stft(y=y, sr=sr)

5.3. Implementation of neural networks

In this section we will study the neural networks used in the project and their effectiveness.

Initial Neural Network Model

```
initial_model = tf.keras.Sequential([
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dropout(0.2),
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dense(164, activation='softmax')
])
```

First generic model used to study the quality of training and its predictions. This specific model is an example of a fully connected neural network, which is configured as a sequence of layers.

`tf.keras.layers.Dense(512, activation='relu')`

First layer of the model. Each neuron in this layer is connected to all neurons in the previous layer. It has 512 neurons and uses the ReLU (Rectified Linear Unit) activation function. The **ReLU** function is a non-linear activation function so the model can learn complex patterns.

`tf.keras.layers.Dropout(0.2)`

Dropout Layer. Dropout is a regularization technique used to avoid overfitting. During training, some neurons are randomly deactivated to avoid overfitting to the training data. In this case, 20 % of the neurons are deactivated.

`tf.keras.layers.Dense(512, activation='relu')`

`tf.keras.layers.Dropout(0.2)`:

These are the second hidden layer and the second Dropout layer, respectively.

`tf.keras.layers.Dense(512, activation='relu')`

This is the third hidden layer of the model. Like the first two hidden layers, it has 512 neurons and uses the ReLU activation function.

```
tf.keras.layers.Dense(164, activation='softmax')
```

Output layer of the model. It has 164 neurons, matching the number of musical styles present in the dataset. The activation function **Softmax** produces a probability distribution among the different classes.

TODO: ENTER GRAPHS WITH TRAINING RESULTS AND LAYER REPRESENTATION

1D Convolutional Neural Network Model (1D CNN)

```
conv_model = tf.keras.Sequential([
    tf.keras.layers.Reshape((10770, 1), input_shape=(None, 10770)),
    tf.keras.layers.Conv1D(64, 3, padding='same', activation='relu'),
    tf.keras.layers.Conv1D(64, 3, padding='same', activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling1D(pool_size=2),

    tf.keras.layers.Conv1D(128, 3, padding='same', activation='relu'),
    tf.keras.layers.Conv1D(128, 3, padding='same', activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling1D(pool_size=2),

    tf.keras.layers.Conv1D(256, 3, padding='same', activation='relu'),
    tf.keras.layers.Conv1D(256, 3, padding='same', activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling1D(pool_size=2),

    tf.keras.layers.Conv1D(128, 3, padding='same', activation='relu'),
    tf.keras.layers.Conv1D(128, 3, padding='same', activation='relu'),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.MaxPooling1D(pool_size=2),

    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(164, activation="softmax")
])
```

Convolutional neural networks are particularly effective for music classification due to local pattern recognition, in the context of music it is especially relevant as it can translate into the ability to identify patterns such as rhythm or tones efficiently in the different parts of the songs.

`tf.keras.layers.Reshape((10770, 1), input_shape=(None, 10770))`

First layer of the model. Input data is reshaped to a format accepted by the proposed convolutional layers. In this case, the input data is reshaped to a 2D matrix of 10770 rows and 1 column.

```
    tf.keras.layers.Conv1D(64, 3, padding='same', activation='relu')
tf.keras.layers.Conv1D(64, 3, padding='same', activation='relu')
```

First convolutional layers. The convolution is performed with 64 filters and a kernel size of 3. The activation function is the ReLU (Rectified Linear Unit).

```
tf.keras.layers.BatchNormalization()
```

Batch normalization layer. Important step involved in the stabilization and acceleration of the training process. The operation consists of applying a transformation that keeps the mean output close to 0 and the standard deviation close to 1.

```
tf.keras.layers.MaxPooling1D(pool_size=2)
```

Pooling layer. Reduces the spatial dimension of the input by extracting the most important features and preventing overfitting.

Subsequently, the same steps are performed with different layers of 128 neurons, 256 neurons and finally another 128 neurons. Until the last layers are reached.

```
tf.keras.layers.Flatten()
```

This layer reduces the dimensionality of the input to a 1D input. **Is the bridge between the convolutional layers and the dense layers**.

```
tf.keras.layers.Dense(512, activation='relu')
```

Fully connected layer of 512 neurons with a ReLU activation function.

```
tf.keras.layers.Dropout(0.5)
```

Dropout layer that randomly deactivates 50 % of the neurons to avoid overfitting.

```
tf.keras.layers.Dense(512, activation='relu')
```

Fully connected layer of 512 neurons with a ReLU activation function.

```
tf.keras.layers.Dense(164, activation="softmax")
```

Output layer of the model. It has 164 neurons, matching the number of musical styles present in the dataset. The activation function **Softmax** produces a probability distribution among the different classes.

This model is a 1D CNN with a structure formed by a series of blocks of convolutional layers followed by normalization and pooling, followed by a series of dense layers.

The first part of the model (the convolutional layers) is responsible for learning local features in small windows of the input data, while the second part of the model, fully connected layers, learns to combine these features to make predictions.

TODO: ENTER GRAPHS WITH TRAINING RESULTS AND LAYER REPRESENTATION

Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

Conclusions and future works

7.1. Conclusions

Through this project, the performance but also the complexity of using Artificial Intelligence (AI) for the automatic classification of musical styles has been proved.

In general terms, the system has proven to be reasonably effective in the task of classifying musical styles. However, limitations have also been identified in the process.

First, although the audio feature extraction process plays a crucial role in the development and training of musical style classification models, it might be worth considering to explore other methods or implementing more advanced techniques in order to improve the quality of the data that feeds the neural network.

It may be beneficial explore deep learning techniques such as transfer learning. Pre-trained models could be used as a starting point, which could improve the quality of the model.

The size of the dataset has a direct impact on the effectiveness of the system. Using a larger and more varied dataset could improve the system's ability to generalize.

In conclusion, this project has served to demonstrate that music style classification using AI is feasible and that artificial intelligence applied to music offers great potential for future research.

7.2. Future works

1. **Experimentation with different models**: Convolutional neural networks have been mainly used in this project, but there are many other network architectures. For example, recurrent neural networks (RNNs) or generative adversarial models (GANs) could be fine choices for experimentation.

2. **Explore semi-supervised and unsupervised learning approaches**: Given the difficulty of labeling large music datasets, semi-supervised or unsupervised learning techniques could be considered. These approaches could be useful to take advantage of unlabeled data and improve system performance.

3. **User feedback**: Allow users to correct model incorrect classifications and use that information to improve the model.

4. **Build a recommendation model**: With the music style classification model, it could be interesting to build a music recommendation system that suggests songs to the users, based on their preferences and listening habits.

TODO: ADD MORE FUTURE ENHANCEMENTS AND IMPROVE THIS SECTION!