Software Engineering Software Requirements Specification (SRS) Document

Music Analysis System

July 25, 2021

Version 2

Table of Contents

1. Introduction

1.1 Purpose: The goal of our project is to build a Music Analysis System that can predict music labels in a recording like notes, number of instruments used, composer, onset and the next note.

1.2 Document conventions: Following abbreviations have been used a couple of times in this document:
i) Artificial Intelligence (AI)
ii) Long Short Term Memory (LSTM)
iii) Gated Recurrent Unit (GRU)
iv) Exploratory Data Analysis (EDA)

1.3 Intended audience: The project can be used by music analysts who want to compose the best quality music for the people by studying the occurrences of notes. This can also be deployed as a recommendation system by music streaming services.

1.4 Scope: AI technology has the ability to learn a musician's style and sound by analyzing previous songs, common note combinations, and the creator's likely next move. So this project will help analyze instruments, composers, and notes to determine playlists, artists, and songs that fit the style of the user.

1.5 References: A list of other documents that the SRS document refers to including sources such as websites or written literature.

i)https://rua.ua.es/dspace/bitstream/10045/76991/1/Automatic_music_transcription_using_neural_networks_MINGUEZ_CARRETERO_MANUEL.pdf

ii)https://homes.cs.washington.edu/~thickstn/musicnet.html

iii)https://medium.com/@ianvonseggern/note-recognition-in-python-c2020d0dae24

2. General Description

2.1 Product perspective: Upon the completion of this project, the music analysts (end users) would be able to predict key labels to determine various characteristics of a music sample.

2.2 Product features: This project aims to design a deep learning algorithm based on LSTM and GRU, which will be capable of analysing and predicting the following characteristics of a music segment:
i) Number of notes
ii) Number of instruments used
iii) Composer
iv) Onset times of the notes
v) Next note

2.3 User class and characteristics: The end users of this project would be music analysts from companies that provide music streaming services to offer the top quality songs.

2.4 Operating environment: Considering the size of our dataset, the development team would require a cloud based virtual machine to perform all the necessary operations like EDA, training and testing of the dataset.

2.5 Constraints: i) Extremely large dataset
ii) High computational power required
iii) Need of a centralised database management system

2.6 Assumptions and dependencies: We have gathered an audio dataset that has 330 classical music recordings, assuming that the data is 100% correct (no corrupt audio files) and properly labeled.

3. System Requirements

3.1 Functional requirements
Users should be able to give an audio segment as the input and the algorithm must predict the required labels associated with different time stamps for that audio. Following are the major requirements necessary to accomplish this goal:

i) Libraries - Libraries like *Librossa* and *Mingus* will be used to study and apply EDA on the audio database. These libraries provide the necessary tools to visualise dataset features and retrieve useful information from the audio signals. Visualization of the data is important to make sure that there are no corrupt or empty files and whether all the audio files are in the same format or not.

ii) High-end System - Given an unusually large dataset, it is necessary to either go with a virtual machine or a high end system to perform pre-processing and model training in a reasonable amount of time.

4. Non-Functional Requirements

4.1 Performance requirements
To successfully integrate this Music Analysis System with a dedicated software for music analysis eligible to perform on an industrial stage, the prediction accuracy should be around 40-60%, **considering the dataset provided remains the same**.

4.2 Safety requirements

The trained model should be hosted at multiple servers, not only a single server, for the backend. So, in any case, if there is a downtime at a particular server, we can shift to another server which is not hit by downtime to prevent any interruption in the working of the program.

4.3 Security requirements

While the user is using the application, only the necessary data is gathered. The application should not execute any command embedded in the data provided by the users that forces the application to manipulate the backend in any unintended way, meaning if the application is only supposed to take wav files from the user, it shouldn't take any other file masquerading as a wav file.

4.4 Software quality attributes

Reliability: The algorithm is intended to perform at highest accuracy and to make sure that the percentage error is minimum.
Portability: Being an algorithm developed in Python language, it can be universally integrated with a software based on python framework.
Maintainability: Maintenance comes into play whenever new data is to be added in the database for routine updations.

4.5 Other requirements

These may include the legal requirements, resource utilizations, future updates etc.