

MUSIC ANALYSIS SYSTEM

Product Deployment Guide

Product Overview

Many music streaming services like JioSaavn and Apple, nowadays, optimise their content in such a way that it feels more personalised and connected to their users. To accomplish this task, they use certain tools to analyse different genres of music to predict the notes instruments used, beats duration, etc. It is extremely necessary for these companies to analyse the audio and predict these parameters correctly without affecting their current resources. Latest studies show that Spotify adds more than 5000 hours of content to be released everyday. This means it's crucial to identify which type of music is most appreciated by people of which region or culture, to offer personalised services.

This product aims to use deep learning based algorithms written in python to predict the instruments used in a classical song with the highest possible accuracy. The algorithm developed in this product allows you to check for discrepancies in the audio, pre-processing raw data to be sent to the machine learning model to train it. And then you can simply pass an input audio to check the instruments used to compose it.

How To Deploy It?

This algorithm can be deployed as a part of a fully fledged software, which can be used or sold at an industrial scale to gain better knowledge of the type and quality of a music.

However, you can also implement it by simply storing the data, running the code and then making predictions by using a single line of code.

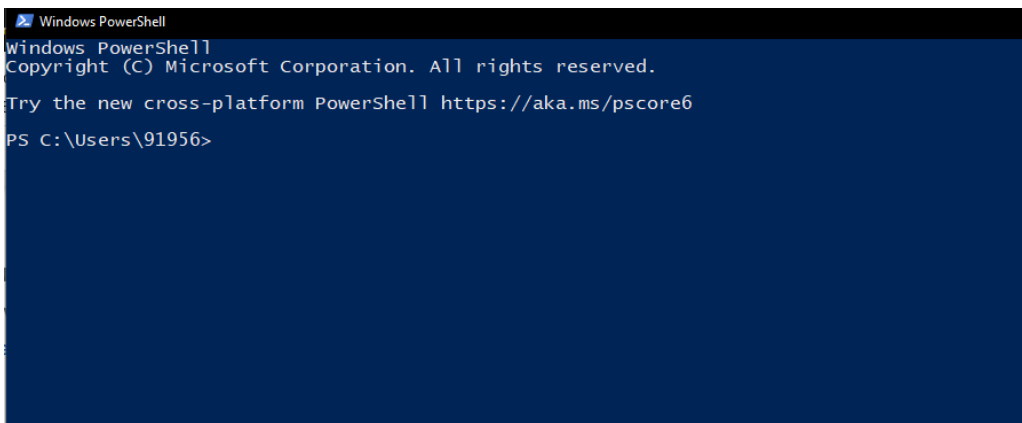
Scenario 1: If you are not good at programming or code execution, you can just simply store the audio input in the database and check for the types of instruments used in the audio. This can be done by gathering all the data at one single place and then executing the following steps:

Note: The following steps are demonstrated assuming you have set up the environment to execute the programs successfully. If you haven't done it yet, follow the user guide

Step 1: Open the command prompt

Use keyboard shortcut, press Windows+R and type powershell.

Following window will be opened.



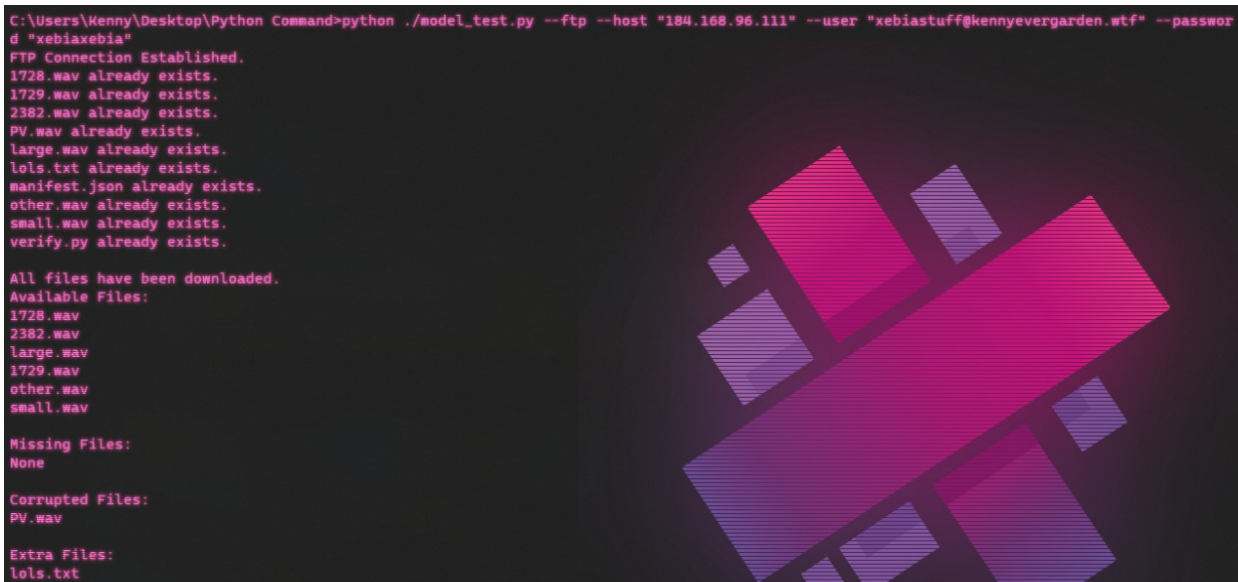
Now, if your data is located on a web hosted server, type the following command:

```
python ./model_test.py --ftp --host "type your host ip address here"
user "type your username here" --password "type your server
password here"
```

In the demonstration below, the data was stored on an FTP server. Now, if your data is located on a local drive, type the following command:

```
python model_test.py --local ./data/
```

The output is shown in the below snapshot:



```
C:\Users\Kenny\Desktop\Python Command>python ./model_test.py --ftp --host "184.168.96.111" --user "xebiastuff@kennyevergarden.wtf" --password "xebiastuff"
FTP Connection Established.
1728.wav already exists.
1729.wav already exists.
2382.wav already exists.
PV.wav already exists.
large.wav already exists.
lols.txt already exists.
manifest.json already exists.
other.wav already exists.
small.wav already exists.
verify.py already exists.

All files have been downloaded.
Available Files:
1728.wav
2382.wav
large.wav
1729.wav
other.wav
small.wav

Missing Files:
None

Corrupted Files:
PV.wav

Extra Files:
lols.txt
```

```
Instruments Used in 2382.wav:
```

```
Instrument 1: 91.14%  
Instrument 41: 0.58%  
Instrument 42: 0.28%  
Instrument 43: 7.99%  
Instrument 44: 0.0%  
Instrument 61: 0.0%  
Instrument 69: 0.0%  
Instrument 7: 0.0%  
Instrument 71: 0.0%  
Instrument 72: 0.0%  
Instrument 74: 0.0%
```

```
Instruments Used in 1729.wav:
```

```
Instrument 1: 94.06%  
Instrument 41: 1.5%  
Instrument 42: 0.25%  
Instrument 43: 4.19%  
Instrument 44: 0.0%  
Instrument 61: 0.0%  
Instrument 69: 0.0%  
Instrument 7: 0.0%  
Instrument 71: 0.0%  
Instrument 72: 0.0%  
Instrument 74: 0.0%
```

The usage of instruments in the audio sample by percentage is predicted.

Scenario 2: If you want to work on this algorithm to add more functionalities or integrate it with a software framework then you can study all the files that allow you to check for discrepancies, let you keep a track of the addition and deletion of files from the database, etc.

Once you have uploaded the files in the database, you can begin with checking for discrepancies in the files. FFmpeg command to analyse the input file:

```
PS C:\Users\91956\Downloads\NewWav\WavTest> ffmpeg -i "data\w_audio.wav"
ffmpeg version 4.4-essentials_build-www.gyan.dev Copyright (c) 2000-2021 the FFmpeg developers
built with gcc 10.2.0 (Rev6, Built by MSYS2 project)
configuration: --enable-gpl --enable-version3 --enable-static --disable-w32threads --disable-autodetect --enable-fontc
onfig --enable-iconv --enable-gnutls --enable-libxml2 --enable-gmp --enable-lzma --enable-zlib --enable-lsrt --enable-
libssh --enable-libzmq --enable-avisynth --enable-sdl2 --enable-libwebp --enable-libx264 --enable-libx265 --enable-libxv
id --enable-lbaom --enable-libopenjpeg --enable-libvpx --enable-libbass --enable-libfreetype --enable-libfribidi --enabl
e-libvidstab --enable-libvmaf --enable-libzimg --enable-amf --enable-cuda-llvm --enable-cuvid --enable-ffnvcodec --enabl
e-nvdec --enable-nvenc --enable-d3d11va --enable-dxva2 --enable-libmfx --enable-libgme --enable-libopenmpt --enable-libo
pencore-amrwb --enable-libmp3lame --enable-libtheora --enable-libvo-amrwbenc --enable-libgsm --enable-libopencore-amrnb
--enable-libopus --enable-libspeex --enable-libvorbis --enable-librubberband
libavutil 56. 70.100 / 56. 70.100
libavcodec 58.134.100 / 58.134.100
libavformat 58. 76.100 / 58. 76.100
libavdevice 58. 13.100 / 58. 13.100
libavfilter 7.110.100 / 7.110.100
libswscale 5. 9.100 / 5. 9.100
libswresample 3. 9.100 / 3. 9.100
libpostproc 55. 9.100 / 55. 9.100
data\w_audio.wav: invalid data found when processing input
PS C:\Users\91956\Downloads\NewWav\WavTest>
```

The *verify.py* checks for CRC of the files and matches it against the manifest file, allowing us to determine any missing or corrupted files.

Filename	Filesize	Filetype	Last modified	Permissions	Owner/Group
..					
.ftpquota	4	FTPQUOTA...	07/06/21 14:16:...	0777	8015460 801...
1728.wav	44,301,370	WAV File	07/06/21 14:25:...	0777	8015460 801...
1729.wav	78,409,786	WAV File	07/06/21 14:25:...	0777	8015460 801...
2382.wav	20,814,394	WAV File	07/06/21 14:25:...	0777	8015460 801...
large.wav	128,221,192	WAV File	07/21/21 13:03:...	0644	8015460 801...
lols.txt	0	Text Docu...	07/06/21 17:13:...	0777	8015460 801...
manifest.json	368	JSON File	07/21/21 13:08:...	0777	8015460 801...
other.wav	1,323,632	WAV File	07/21/21 12:56:...	0644	8015460 801...
PV.wav	11,634,030	WAV File	07/06/21 16:57:...	0777	8015460 801...
small.wav	1,761,394	WAV File	07/21/21 12:56:...	0644	8015460 801...
verify.py	1,411	Python File	07/21/21 12:33:...	0777	8015460 801...

11 files. Total size: 286,467,581 bytes

You can check for this by running the following command in the terminal:

```
Windows PowerShell
PS C:\Users\91956\Downloads\Stuff File> python .\verify.py
Available Files:
1728.wav
2382.wav
2416.wav
1729.wav

Missing Files:
2628.wav

Corrupted Files:
2556.wav

Extra Files:
PV.wav
lols.txt

PS C:\Users\91956\Downloads\Stuff File>
```

This command reads input from the manifest file and checks which file is not there in the database but is registered in the manifest file. All the files with any discrepancy, will be rejected and won't be sent to the pre-processing stage.

You can run the entire code in any IDE. In this demonstration, Jupyter Notebook was used for execution.

Once the model is trained and ready, you can check if it is predicting the instruments correctly or not, by using the commands mentioned in Scenario 1 mentioned above.