COMP4801
Final Year Project

Emotion-based Music Provider

Final Report
(Part 3)

Group: FYP22054

The University of Hong Kong
Department of Computer Science

Supervised By:
Prof. Dirk Schnieders

Submitted By:
Tsang Tsz Hin (3035685495) – BEng (CS)

Group Members:
To Hoi Lam (3035667455) – BEng (CS)
Lee Wing Fung (3035693947) – BEng (CS)
Wang Yu Tung (3035685005) – BEng (CE)

Date of Submission: 18th April 2023
Abstract

This project aims to build an Emotion-based Music Provider, which provides music of certain emotions through recommendation and generation. As the last section of the group report, this report focuses on Music Generation and the Application. Prior to Music Generation, a symbolic Music Emotion Recognition model was trained to assign emotion labels on training data. It take several music features as input to identify the emotion expressed by the MIDI. Our finalized MLP model outperformed other models with an accuracy of 70.23%. Then, we started to train two Music Generation models, which focus on monophonic and polyphonic music respectively. Models from Magenta were adopted as the baseline model. For monophonic Music Generation, extra melody information was included in the feature space. Therefore, our finalized models were able to generate music of certain emotions with good melody structure. For polyphonic Music Generation, no extra melody information was included in the feature space. Therefore, our finalized models were able to generate music of certain emotions, but the melody structure was poor. Both models have a limitation, which fails to cater information on note velocity. We then combined all the functionalities as a web application, with an user-friendly frontend design and an efficient backend design.
Acknowledgement

First, I would like to show my gratitude for the Department of Computer Science, and Dr. Dirk Schnieders, for the vulnerable support and advice they have offered to our group.

I also appreciate my groupmates, Mr. Lee Wing Fung, Mr. To Hoi Lam, and Mr. Wang Yu Tung, for their hard efforts made in achieving our project.
# Table of Contents

Abstract ................................................................................................................................. i

Acknowledgement ................................................................................................................ ii

List of Equations ..................................................................................................................... v

List of Figures ........................................................................................................................ iii

List of Tables ........................................................................................................................... iv

List of Abbreviations ............................................................................................................. v

1. Introduction......................................................................................................................... 1
   1.1 Overview ......................................................................................................................... 1
   1.2 Symbolic Music Representation .................................................................................... 1
   1.3 Application ..................................................................................................................... 2
   1.4 Outline .......................................................................................................................... 2

2. Methodology ....................................................................................................................... 3
   2.1 Introduction ................................................................................................................... 3
   2.2 Symbolic Music Emotion Recognition ......................................................................... 3
      2.2.1 Dataset .................................................................................................................... 3
      2.2.2 Preprocessing ......................................................................................................... 4
      2.2.3 Feature Extraction ................................................................................................. 5
      2.2.4 Model Training ....................................................................................................... 5
      2.2.5 Experiment and Testing ....................................................................................... 6
   2.3 Music Generation ........................................................................................................... 7
      2.3.1 Detailed Workflow .................................................................................................. 7
      2.3.2 Dataset ................................................................................................................... 8
      2.3.3 Monophonic Music Generation ............................................................................. 9
         2.3.3.1 Generation Model ............................................................................................ 9
         2.3.3.2 Preprocessing and Feature Extraction ............................................................. 10
         2.3.3.3 Experiment and Testing ................................................................................... 11
      2.3.4 Polyphonic Music Generation .............................................................................. 12
         2.3.4.1 Generation Model ........................................................................................... 12
         2.3.4.2 Preprocessing and Feature Extraction ............................................................. 12
         2.3.4.3 Experiment and Testing ................................................................................... 13

3. Result and Analysis ............................................................................................................ 14
   3.1 Symbolic Music Emotion Recognition .......................................................................... 14
3.2 Monophonic Music Generation ................................................................. 15
   3.2.1 Model Result .................................................................................. 15
   3.2.2 Music Analysis ............................................................................. 16

3.3 Polyphonic Music Generation................................................................. 18
   3.3.1 Model Result ................................................................................ 18
   3.3.2 Music Analysis ............................................................................. 19

4. Application ............................................................................................... 21
   4.1 Software and Development Workflow ............................................. 21
   4.2 Web Application Design .................................................................. 22
      4.2.1 UI Design .................................................................................. 22
      4.2.2 Features and Functionalities ...................................................... 23
      4.2.3 Application Workflow .............................................................. 29
   4.3 API Service ....................................................................................... 30
      4.3.1 Speech Emotion Recognition API .............................................. 30
         4.3.1.1 Get Model Choices ............................................................. 30
         4.3.1.2 Predict with Acoustic-based Speech Emotion Recognition .... 31
         4.3.1.3 Predict with Speech-based Text Emotion Detection .............. 32
      4.3.2 Music Recommendation API ...................................................... 33
         4.3.2.1 Get Recommended Songs from Speech Emotion ............... 33
         4.3.2.2 Get Lyrics of a Song ........................................................... 34
      4.3.3 Music Generation API ................................................................. 35
         4.3.3.1 Generate Songs given Speech Emotion ................................. 35
   4.4 Summary of Web Application ............................................................. 36
   4.5 Limitation and Mitigation .................................................................. 37
      4.5.1 Processing Time ........................................................................ 37
      4.5.2 Embedded Spotify Player .......................................................... 39

5. Conclusion ............................................................................................... 40

Appendix I ..................................................................................................... 44
Appendix II .................................................................................................... 47
Appendix III ................................................................................................... 48
Appendix IV .................................................................................................. 50
References ..................................................................................................... 42
List of Figures

Figure 2.1 Russell’s Arousal-Valence Model ................................................................. 4
Figure 2.2 Four-class taxonomy used in EMOPIA dataset [1] ........................................ 4
Figure 2.3 Workflow diagram of Symbolic Music Generation ....................................... 7
Figure 2.4 Distribution of the Lakh MIDI Dataset over popular music genre .................. 8
Figure 2.5 Distribution of the Lakh MIDI Dataset over musical instruments ................. 8
Figure 3.1 Confusion Matrix of MLP model on testing set .............................................. 15
Figure 3.2 Piano roll of a Happiness monophonic music sample .................................. 16
Figure 3.3 Violin plot on average note velocity of training MIDI ................................... 17
Figure 3.4 Box plot on average note length of training data .......................................... 18
Figure 3.5 Box plot on average note length of generated monophonic music .................. 18
Figure 3.6 Piano roll of a Happiness polyphonic music sample ..................................... 19
Figure 3.7 Box plot on pitch entropy of training data ..................................................... 20
Figure 3.8 Box plot on pitch entropy of generated polyphonic music ............................. 20
Figure 4.1 Web application development flowchart ...................................................... 22
Figure 4.2 Landing view for Music Recommendation page .......................................... 23
Figure 4.3 Landing view for Music Generation page .................................................... 23
Figure 4.4 Application with recommended songs .......................................................... 24
Figure 4.5 Application showing Overall Analysis of recommendation ......................... 25
Figure 4.6 Application showing Audio & Acoustics Analysis of recommendation .......... 26
Figure 4.7 Application showing Text & Lyrics Analysis of recommendation ................... 26
Figure 4.8 Application showing Semantics Analysis of recommendation ....................... 27
Figure 4.9 Application with generated songs ............................................................... 28
Figure 4.10 Application showing Emotion Analysis of generation .................................. 28
Figure 4.11 Application workflow summary ................................................................. 29
List of Tables

Table 2.1 Data distribution of EMOPIA dataset ................................................................. 4
Table 2.2 Feature table for symbolic Music Emotion Recognition ................................. 5
Table 2.3 Distribution of MIDI data for each emotion ..................................................... 9
Table 2.4 Summary of training and evaluation data size for monophonic music generation. 11
Table 2.5 Summary of training and evaluation data size for polyphonic music generation ... 13
Table 3.1 Result table for MER through machine learning algorithms ............................ 14
Table 3.2 Model results of Happiness monophonic generation model ......................... 15
Table 3.3 Finalized model results of Monophonic Music Generation ............................ 16
Table 3.4 Finalized model results of Polyphonic Music Generation ............................... 18
Table 4.1 Summary of all API endpoints ........................................................................... 30
Table 4.2 API endpoint for getting model choices ............................................................ 31
Table 4.3 API endpoint for predicting emotion values with Acoustic-based SER model .... 32
Table 4.4 API endpoint to predict emotion values through speech-based TED model ...... 33
Table 4.5 API endpoint to get recommended songs given a speech audio ..................... 34
Table 4.6 API endpoint to get lyrics of a song ................................................................. 35
Table 4.7 API endpoint to get generated songs given a speech audio ......................... 36
Table 4.8 Proportion of the delay in backend for Music Recommendation task ............. 38
Table 4.9 Proportion of the delay in backend for Music Generation task ..................... 38
List of Equations

Equation 2.1 The attention mechanism [7]........................................................................................................9
List of Abbreviations

CNN        Convolution Neural Network
KNN        K-Nearest Neighbors
LSTM       Long-Short Term Memory
MER        Music Emotion Recognition
MIDI       Music Instrument Digital Interface
RF         Random Forest
RNN        Recurrent Neural Network
SER        Speech Emotion Recognition
SVM        Support Vector Machine
UI         User Interface
1. Introduction

1.1 Overview

In this project, we aim to build an Emotion-based Music Provider, which provides music based on human’s speech emotions. 4 separate major objectives for achieving our goal of Emotion-based Music Provider are listed below:

Objective 1: Classifying emotion expressed by human’s speech

Objective 2: Providing existing music to the user based on a specified emotion

Objective 3: Generating symbolic music of certain emotions

Objective 4: A web application that achieves Emotion-based Music Provider

This report serves as the third section of our group report, which focuses on Symbolic Music Generation and the web application. As mentioned in Report 1, existing research focuses on generating generalized music pieces. Our project aims at a more specialized field, which is music generation of specific emotions. Our web application combines all the functionalities to achieve our ultimate goal, which is Emotion-based Music Provider.

1.2 Symbolic Music Representation

In Symbolic Music Representation, a music piece is composed of bars, which contain music notes in a fixed time interval. In each bar, there is a time signature that determines the length of bar. A time signature is composed of 2 numbers. The first number represents the maximum number of notes presented in a bar, while the second number refers to the type of notes. For example, for a 4/4 time signature, each bar can contain at most 4 quarter note. For a 6/8 time signature, each bar can contain at most 6 eighth notes.

For a machine to better understand a piece of music, quantization is carried out on bars. Each bar is further divided into fixed time intervals, called steps. A quarter note and an eighth note last for 4 steps and 2 steps respectively. Therefore, based on the time signature, the machine can calculate the number of steps in a bar.
1.3 Application

As a combination of our work, we implemented a web application, which is the Emotion-based Music Provider. It achieves our ultimate goal, which detects user’s emotion to provide corresponding music via recommendation or generation.

1.4 Outline

The report is structured into five chapters. The first chapter offers an overview of the Emotion-based Music Provider, as well as a brief introduction on Symbolic Music Generation and the web application.

Chapter two analyses the methodology used in the project. Implementation of Symbolic Music Generation will be explained in detail. Testing of various models were carried out to achieve better performance.

Chapter three presents the results of Symbolic Music Generation. The finalized models will be explained. There is also a discussion of the generated music and the limitations of our model.

Chapter four explained the implementation of our web application. Details on frontend and backend design, application flow and functionalities, will be presented.

Chapter five concludes the report. It summarizes our work in developing a Emotion-based Music Provider.
2. Methodology

2.1 Introduction

This chapter presents the technology and platform we used to implement our application (To generate new music with specific emotion). It also describes the testing method to achieve the best performance.

2.2 Symbolic Music Emotion Recognition

To identify the emotion expressed in a symbolic music piece, we built a symbolic Music Emotion Recognition (MER) model. The music piece must undergo preprocessing, feature extraction, and be fitted into the trained model. Our model then assigns emotion labels onto music data. This model was used in creating dataset with emotion labels for music generation.

2.2.1 Dataset

Currently, two kinds of datasets are commonly used in Music Emotion Recognition. They store data as either raw audio or MIDI file, which is a symbolic representation of music pieces. As we were focus on symbolic music generation, we adopted MIDI dataset to train our symbolic MER model.

For symbolic MER, the EMOPIA dataset [1] is adopted, which contains 1087 music excerpts from 387 songs in MIDI format. The dataset adopts Russell’s Arousal-Valence Model [2] for emotion classification. Figure 2.1 gives an overview on how emotion is defined under the arousal-valence model. Instead of providing music excerpts with exact annotation, EMOPIA tries to map each music excerpts onto the four quadrants of the arousal-valence model.
Figure 2.1 Russell’s Arousal-Valence Model

Figure 2.2 shows the four-quadrant classification of EMOPIA, with Q1 as high valence high arousal (HVHA), Q2 as low valence high arousal (LVHA), Q3 as low valence low arousal (LVLA) and Q4 as high valence low arousal (HVLA). Table 2.1 generalizes the data distribution of EMOPIA. Throughout the project, we generalized the four quadrants into four emotions. Q1 to Q4 represents Happiness, Anger, Sadness and Calmness respectively.

<table>
<thead>
<tr>
<th>Quadrant</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>246</td>
</tr>
<tr>
<td>Q2</td>
<td>263</td>
</tr>
<tr>
<td>Q3</td>
<td>253</td>
</tr>
<tr>
<td>Q4</td>
<td>309</td>
</tr>
</tbody>
</table>

Table 2.1 Data distribution of EMOPIA dataset

2.2.2 Preprocessing

The importation of the MIDI files is done through the MusPy library [3] as Music class. Only some subclasses, such as Track and Note, contain useful information for symbolic music emotion recognition. In EMOPIA, all music pieces contain one track only because they are limited to piano music. So, only one sequence of notes is extracted from each music excerpt.
2.2.3 Feature Extraction

Features of the music piece are extracted through the MusPy library [3]. Table 2.2 summarizes all features extracted from the music piece. To ensure all features lie within a comparable scale, standardization is carried out on all features. Justifications behind each selected feature are discussed in section 2.2.5.

<table>
<thead>
<tr>
<th>Arousal-related</th>
<th>Valence-related</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Feature</strong></td>
<td><strong>Explanation</strong></td>
</tr>
<tr>
<td>Note Density</td>
<td>Number of notes per beat</td>
</tr>
<tr>
<td>Average Note Length</td>
<td>Average of the length of all notes</td>
</tr>
<tr>
<td>Average Note Velocity</td>
<td>Average of the velocity of all notes</td>
</tr>
<tr>
<td>Note Density S.D.</td>
<td>Standard Deviation of Note Density</td>
</tr>
<tr>
<td>Note Length S.D.</td>
<td>Standard Deviation of Note Length</td>
</tr>
<tr>
<td>Note Velocity S.D.</td>
<td>Standard Deviation of Note Velocity</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.2 Feature table for symbolic Music Emotion Recognition

2.2.4 Model Training

To build a symbolic MER system with better performance, different models are built for performance comparison. Both machine learning and deep learning algorithms are taken into consideration.

For machine learning algorithms, all models are built and trained through the Seikit-learn package [4]. Logistic Regression model acts as the baseline model for finalizing the feature space. Afterwards, we built and trained different models on the finalized feature space, which include Logistic Regression, Kernel Support Vector Machine (Kernel SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and AdaBoost.
For deep learning algorithms, we experimented on models including Multi-Layer Perceptron (MLP) on the finalized feature space. We used Scikit-learn and TensorFlow to build our models respectively.

### 2.2.5 Experiment and Testing

At first, we followed research Livingstone et al. [5] and extracted Note Density, Average Note Length and Average Note Velocity as arousal features. Our baseline model has achieved satisfactory results on classifying degree of arousal with an accuracy of 90.5%. We further extracted Average Pitch, Scale and Scale’s starting notes as valence features. We presented scale and scale’s starting notes in three different ways, including 24 values through one-hot encoding, 1 value summarizing scale and starting notes, and 2 values explicitly describing scale and starting notes. The third approach gives the best accuracy of 66.5%

To improve the performance on classifying degree of valence, we inspected the pitch-related and rhythm-related metrics provided by MusPy [3]. By observing the correlation between metrics’ components over the 4-quadrant music pieces, we concluded that Pitch Range, Polyphony, Pitch Entropy, and Groove Consistency provides necessary information for MER. Yet, there was no improvement in the performance of our baseline model. We discovered that the current pitch range is highly sensitive to outliers, which might poorly affect the performance of our model. Therefore, we adopted “soft” pitch range instead, which ignored 20% of the outer pitch. The result after adjustment was satisfactory, as the valence accuracy increased to 67.7%.

We also incorporated features like the standard deviation of Note Pitch, Note Velocity, Note Length, and Note Density into the model. The result is significantly better, where the valence accuracy further increased to 70.1%. We adopted this feature space as finalized setting for all models.

In addition, to find the best configurations for MLP, we experimented with various parameters, including the number of neurons, the number of hidden layers and the activation functions for each layer. While increasing the number of neurons and layers for better results, we kept track on the validation accuracy to avoid overfitting. Throughout testing, we discovered that softmax as output layer activation function can effectively improve the model performance as compared with the ReLU activation.
2.3 Music Generation

Besides recommending existing music to our users, generating new music is an alternative way to provide emotional music. This section presents the method to achieve the third objective (Generating music and melodies of certain emotions) mentioned in Section 1.1. The detailed milestones are as follow:

Objective 3.1: To generate monophonic symbolic music with specific emotions

Objective 3.2: To generate monophonic symbolic music with recognizable melody

Objective 3.3: To generate polyphonic symbolic music with specific emotions

Objective 3.4: To generate polyphonic symbolic music with recognizable melody

2.3.1 Detailed Workflow

Figure 2.3 shows the detailed workflow of Symbolic Music Generation. After collecting speech emotion through Speech Emotion Recognition (SER), suitable primers and generation model will be selected to generate music of corresponding emotion. Symbolic MER model will be used to verify the emotion of the generated music.

![Figure 2.3 Workflow diagram of Symbolic Music Generation](image)

In our project, we focus on two types of music generation, which are monophonic and polyphonic music generation. In monophonic music generation, at most 1 note is allowed within a time step, while multiple notes are allowed in polyphonic music generation.
To generate music of the four emotions mentioned in section 2.2, four separate models were trained independently, which represents a different emotion. The training data was first classified by our symbolic MER to obtain an emotion label. It was fitted into corresponding models after preprocessing and feature extraction. In addition, a primer will be provided when we generate music. A group of primers were prepared for each emotion, which are collected from the first 25 steps of the music pieces. The emotion of the primers was verified by the symbolic MER model as well.

As the final step, we use the symbolic MER model to classify the emotion of the generated music. If it fails to pass the test, a different primer will be selected, and the generation will be restarted.

2.3.2 Dataset

Throughout building our music generation model, the Lakh MIDI Dataset was adopted for training [6]. The Lakh MIDI Dataset is a large-scale music dataset containing 176,581 unique MIDI files. Among all MIDI files, 45,129 of them are matched and aligned with the data in the Million Song Dataset. Extra information, mainly music genre, can be obtained. Figure 2.4 and Figure 2.5 shows the distribution of the Lakh MIDI Dataset over popular music genre and instrument.

![Most common tags (20)](image)

**Figure 2.4** Distribution of the Lakh MIDI Dataset over popular music genre

![Instrument classes](image)

**Figure 2.5** Distribution of the Lakh MIDI Dataset over musical instruments
Music generation with recognizable melody is one of our main objectives. Music of different genres and different instruments may have different melody structures, which may poorly affect the quality of the generated music. Therefore, the genre and instrument of the training data are restricted to pop and piano respectively. In addition, to identify the emotion expressed by the music data, all training data was passed into our symbolic MER system for preprocessing.

Table 2.3 summarizes the final size of training data adopted for each emotion. It was observed that the data size of pop piano music for Sadness and Calmness was relatively small, which was insufficient to train a music generation model. Although the quality of the dataset may be affected, pop guitar music was included as extra training data.

<table>
<thead>
<tr>
<th>Emotion</th>
<th>No. of MIDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angry</td>
<td>3811</td>
</tr>
<tr>
<td>Calmness</td>
<td>1612</td>
</tr>
<tr>
<td>Happiness</td>
<td>4090</td>
</tr>
<tr>
<td>Sadness</td>
<td>3029</td>
</tr>
</tbody>
</table>

Table 2.3 Distribution of MIDI data for each emotion

2.3.3 Monophonic Music Generation

2.3.3.1 Generation Model

For Monophonic Music Generation, we adopted the Attention RNN model from Magenta [7]. Attention RNN is a double-layered LSTM model with an attention mask. Given a preceding note sequence, the model will suggest the following notes based on the probability distribution and combine all generated notes to become a music piece. Taking advantage from the Long-Short Term Memory (LSTM) cells, it can effectively model long-term sequences with time dependency. In addition, this model is capable of recognizing the melody of the music by introducing the attention mask and specific features.

\[
\begin{align*}
    u_t^i &= v^T \tanh(W'_i h_t + W'_c c_t) \\
    a_t^i &= \text{softmax}(u_t^i) \\
    h_t' &= \sum_{i=t-n}^{t-1} a_t^i h_i
\end{align*}
\]

Equation 2.1 The attention mechanism [7]
Equation 2.1 explains the core principle of the attention mechanism. Given learnable parameters $v$, $W'_1$ and $W'_2$, the attention mechanism takes the last n outputs ($h_i$) and the current cell state ($c_j$) into consideration to obtain an attention mask $u_i^j$. The attention mask is then normalized ($a_i^j$) and applied onto $h_i$ to obtain a new vector $h'_i$. $h'_i$ is then combined with the current output to create a new output. The attention mechanism can help the model to learn the long-term dependencies of the data, such as melody structure.

Besides applying the attention mask, extra information on melodies is included as features. Details of the specific features will be explained in the next section.

2.3.3.2 Preprocessing and Feature Extraction

In order to be fitted into training models, all midi data must undergo transformation through Magenta’s pipeline. There are 4 essential processes, which are NoteSequence Transformation, Time Splitting, Quantization and Melody Extraction.

First, all midi data was transformed into notesequence format, which is an array storing all notes. Each note is represented in the format of a list of (pitch, starting time, ending time), where the pitch value ranges from 1 to 127, each representing a unique musical key (i.e. 84 is C6, 104 is G#7). Next, as Magenta does not support MIDI file with different time sequence (meaning different tempo at different section of the song), time splitting was carried out for each note sequence. It splits note sequence into two parts, when a time signature change is detected. Note sequences were then quantized by fixing the number of steps per quarter note and the number of quarters per minute (tempo of the music). Instead of using starting and ending time, the starting and ending of all notes were mapped to the closest quantized steps. After performing all the MIDI processing, we extract melodies from the note sequence as the inputs of the generation model. Conditions were set to limit the length of a melody and split the melody by detecting silence.

For monophonic music, at most 1 note exists in each time step. 3 events were used to formulate a melody. When a note starts to be played, the pitch of the note is recorded in the first step as note-on. A no-event (-2) represents that a note or a silence is continued in the current step. A note-off (-1) represents silence, which no notes are played in the current steps. For example, to formulate a note of pitch 80 for 2 steps and then silence for 2 steps, it is presented as [80,-2,-1,-2].
To keep track of the melody structure of the music, additional information was extracted from the melody itself as extra features for each note event. For pattern recognition, note events from 1 and 2 bars ago were included in the input. Also, the last note event of the melody was checked for repeating the note event from 1 and 2 bars ago. A custom label was added to the input as the result of checking. Lastly, the current position of the note event in the current bar was added in the input. It was presented by 5 values in binary format. All features were one-hot encoded and fitted into the models for training.

Table 2.4 shows the total number of melodies extracted from all midi data for each emotion.

<table>
<thead>
<tr>
<th>Emotion</th>
<th>No. of melody</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Training</td>
<td>Evaluation</td>
<td></td>
</tr>
<tr>
<td>Angry</td>
<td>4128</td>
<td>453</td>
<td></td>
</tr>
<tr>
<td>Calmness</td>
<td>2212</td>
<td>218</td>
<td></td>
</tr>
<tr>
<td>Happiness</td>
<td>6414</td>
<td>781</td>
<td></td>
</tr>
<tr>
<td>Sadness</td>
<td>3433</td>
<td>363</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.4 Summary of training and evaluation data size for monophonic music generation

### 2.3.3.3 Experiment and Testing

In order to find out the best setting for our model, we tried with different hyperparameter values when training our models. Experiments and testing were carried out on happiness model, and the finalized setting was extended to all generation models. It was because the data size of happiness melody is the largest, which gives the most representative result among all emotions.

In terms of layer size, our generation model was test with size [64,64], [128,128] and [256,256]. We also adjusted the dropout rate of the model to examine the effect of dropout rate for improving potential overfitting issue. Originally, we would like to test with different size of attention length, which may affect the model performance in preserving melody structure. However, the training process kept crashing when we tried to increase the attention length. It was caused by the out-of-memory issue, which was a hardware limitation. Therefore, we kept the value of attention length to be 40 steps throughout testing, which was the maximum attention length for stable training.
We were also interested in the effect of training data size on our generation model. Therefore, we trained a model with layer size [64,64] on a reduced training dataset for comparison.

Apart from finding the best model setting, we also inspected some generated samples to analyse the melody structure. Furthermore, we used the trained symbolic MER model to check the emotion expressed by the generated samples, obtaining an emotion accuracy for each model.

### 2.3.4 Polyphonic Music Generation

#### 2.3.4.1 Generation Model

For Polyphonic Music Generation, we adopted the Polyphonic RNN model from Magenta [8]. Similar to Attention RNN, Polyphonic RNN is a multi-layered LSTM model. On one hand, it is capable of handling multiple notes with a single time step. On the other hand, it does not contain extra information regarding melody structure, such as attention mask and note position.

#### 2.3.4.2 Preprocessing and Feature Extraction

In order to be fitted into training models, all midi data must undergo transformation through Magenta’s pipeline. Same as Monophonic Music Generation, NoteSequence Transformation, Time Splitting and Quantization was carried out first. The principles of the 3 processes are the same as Monophonic Music Generation, which is mentioned in section 2.3.2.2.

After performing all the MIDI processing, we extract polyphony track from the note sequence as the inputs of the generation model. Conditions were set to limit the length of a melody and detect the end of the track. For each note sequence, only 1 polyphony track was extracted. The remaining part of the note sequence was discarded.

For polyphonic music, multiple notes can exist in each time step. There, a new set of events were used to formulate the polyphony track. The start of a polyphony track is marked by the START event. In each time step, a NEW_NOTE event, following with the pitch value, represents a note that starts to play. A CONTINUED_NOTE event, following with the pitch value, represents a note that continues to play in the current time step. A STEP_END event is used to separate each time steps. If there are no note events in between 2 STEP_END, it means that it is a silence step. The end of the track is marked by the END event.
Table 2.5 shows the total number of polyphony tracks extracted from all midi data for each emotion.

<table>
<thead>
<tr>
<th>Emotion</th>
<th>No. of polyphony track</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Training</td>
<td>Evaluation</td>
</tr>
<tr>
<td>Angry</td>
<td>11664</td>
<td>1332</td>
</tr>
<tr>
<td>Calmness</td>
<td>4500</td>
<td>396</td>
</tr>
<tr>
<td>Happiness</td>
<td>15576</td>
<td>1647</td>
</tr>
<tr>
<td>Sadness</td>
<td>8262</td>
<td>855</td>
</tr>
</tbody>
</table>

Table 2.5 Summary of training and evaluation data size for polyphonic music generation

2.3.4.3 Experiment and Testing

Although we would like to test our model with different layer sizes, the training process kept crashing when we set a larger layer size. For stability, we fixed the layer size to be [64,64] for all models.

As mentioned in the last section, no extra information of melody structure was included in the model input. Therefore, besides inspecting some generated samples to analyse the melody structure, we also made comparisons with the generated sample from monophonic generation models. We also used the trained symbolic MER model to obtain an emotion accuracy for each polyphonic generation model.
3. Result and Analysis

3.1 Symbolic Music Emotion Recognition

Table 3.1 summarizes the results of our machine learning models with respective hyperparameter tuning. All machine learning models give satisfactory results on Arousal Accuracy. Among all the models, Kernel SVM gives the best results with a valence accuracy of 74.0% and a 4-Quadrant Accuracy of 69.5%. We believe that the projection of data onto high-dimensional space helps establish a clear margin over degree of valence, which results in a significant improvement on valence accuracy. On the other hand, AdaBoost gives the poorest result, which is even worse than LR. It may be due to the overfitting problem when defining decision boundary on overlapping samples from different quadrants. Out of our expectations, Random Forest achieves the second-best results. Given a reasonably large feature space, the decision trees in our RF model successfully split data at each node with satisfactory information gain.

<table>
<thead>
<tr>
<th>Model</th>
<th>4Q Accuracy</th>
<th>Arousal Accuracy</th>
<th>Valence Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR</td>
<td>0.645</td>
<td>0.908</td>
<td>0.701</td>
</tr>
<tr>
<td>Kernel SVM</td>
<td>0.695</td>
<td>0.908</td>
<td>0.740</td>
</tr>
<tr>
<td>RF</td>
<td>0.664</td>
<td>0.909</td>
<td>0.716</td>
</tr>
<tr>
<td>AdaBoost</td>
<td>0.613</td>
<td>0.909</td>
<td>0.687</td>
</tr>
<tr>
<td>KNN</td>
<td>0.654</td>
<td>0.891</td>
<td>0.707</td>
</tr>
</tbody>
</table>

Table 3.1 Result table for MER through machine learning algorithms

For MLP, the finalized setting is [14 (ReLU), 200 (ReLU), 100 (ReLU), 50 (ReLU), 4 (softmax)], with learning rate = 0.00005. Figure 3.1 shows the confusion matrix of our MLP model. Our MLP model has achieved a 4-Quadrant Accuracy of 73%, which slightly outperformed machine learning models. We decided to adopt the MLP model as finalized symbolic MER model.
3.2 Monophonic Music Generation

3.2.1 Model Result

Table 3.2 shows the result of generation models for happiness. It is observed that higher layer size gives better results in terms of loss and accuracy. However, serious overfitting exists in models of layer size [128,128] and [256,256]. For overfitting models, we tried to increase the dropout rate, but it only had a slight effect on improving the overfitting issue. Furthermore, when we trained the model of layer size [64,64] on a reduced dataset, serious overfitting issue also existed. Therefore, we concluded that the overfitting issue was caused by the lack of training data.

<table>
<thead>
<tr>
<th>Emotion</th>
<th>Layer Size</th>
<th>Loss</th>
<th>Accuracy (%)</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Train</td>
<td>Eval</td>
<td>Train</td>
</tr>
<tr>
<td>Happiness</td>
<td>64, 64</td>
<td>1.046</td>
<td>1.378</td>
<td>70.36</td>
</tr>
<tr>
<td></td>
<td>64, 64</td>
<td>1.026</td>
<td>1.107</td>
<td>71.80</td>
</tr>
<tr>
<td></td>
<td>128, 128</td>
<td>0.7633</td>
<td>1.029</td>
<td>77.92</td>
</tr>
<tr>
<td></td>
<td>128, 128</td>
<td>0.8249</td>
<td>1.038</td>
<td>76.71</td>
</tr>
<tr>
<td></td>
<td>256, 256</td>
<td>0.5893</td>
<td>0.9851</td>
<td>83.15</td>
</tr>
</tbody>
</table>

Table 3.2 Model results of Happiness monophonic generation model

At last, the model setting was finalized with a layer size of [64,64] and a dropout rate of 0.5. Table 3.3 listed the results of all finalized models for all emotions. All models achieved an accuracy of around 70%.
### Table 3.3 Finalized model results of Monophonic Music Generation

<table>
<thead>
<tr>
<th>Emotion</th>
<th>Layer Size</th>
<th>Loss</th>
<th>Accuracy (%)</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Train</td>
<td>Eval</td>
<td>Train</td>
<td>Eval</td>
<td>Train</td>
<td>Eval</td>
</tr>
<tr>
<td>Anger</td>
<td>64, 64</td>
<td>0.9421</td>
<td>0.9829</td>
<td>74.78</td>
<td>73.47</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calmness</td>
<td>64, 64</td>
<td>0.9007</td>
<td>1.121</td>
<td>74.51</td>
<td>69.94</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Happiness</td>
<td>64, 64</td>
<td>1.026</td>
<td>1.107</td>
<td>71.80</td>
<td>69.36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sadness</td>
<td>64, 64</td>
<td>0.9549</td>
<td>1.050</td>
<td>75.62</td>
<td>73.33</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.2.2 Music Analysis

To analyze the performance of our finalized models on melody structure, we randomly selected some primers to generate some samples. Figure 3.2 shows the piano roll of a generated sample. The primer was included as the beginning of the music. In both the primer part and the generation part, similar patterns can be observed. Melody structure from primer is preserved. Furthermore, the pattern is repeating in the generation part, with slight changes in note composition. The generation part has achieved a clear and good melody structure. In some samples, a change in melody structure can be observed as well. Appendix I includes more piano rolls of the generated samples for all emotions. We believed that our models are capable of generating music with good melody structure.

![Figure 3.2 Piano roll of a Happiness monophonic music sample](image)

In addition, we used the symbolic MER model to verify the emotion expressed by the generated music. Our symbolic MER model successfully verified generated songs of Happiness and Anger for over 80% accuracy, but failed in verifying generated music of Sadness and Calmness. After investigation, we found out that the major reason was about note velocity, which is the loudness of a note. Figure 3.3 shows the average note velocity plot of all training MIDI for each emotion. The average note velocity for Sadness and Calmness music is below 100. One of the limitations of our generation model is that it fails to cater the note velocity of each note as it is not included in the feature space. During generation, the generated
music was set to have a constant note velocity of 100, which failed to provide useful information on the mean and standard deviation (S.D.) of note velocity.

![Note Velocity Average Violin Plot](image)

**Figure 3.3** Violin plot on average note velocity of training MIDIs

To solve the above problem, we manually set the note velocity of all notes to be 20 and 80 respectively for Sadness and Calmness music, which is the median of the average note velocity for that emotion. After augmentation, our symbolic SER model was able to classify generated music for Calmness, but not Sadness. The above augmentation fails to complete the feature space of the generated music, as the S.D. of note velocity remained unchanged. This part is left as a future work.

In addition, comparison was made on several music features between the training dataset and the generated MIDIs. **Figure 3.4** and **Figure 3.5** show the graph of average note length for training data and generated music. Similar plots are observed, showing that feature characteristics are preserved in generation. More graphs are listed in Appendix II. Evidence shows that our generation models can successfully generate music of suitable melodies and pitch for certain emotions.
3.3 Polyphonic Music Generation

3.3.1 Model Result

Table 3.4 summarizes the finalized model results for all emotions. Among the 4 emotions, the training results for Happiness and Sadness models were satisfactory, while Anger and Calmness encountered overfitting issue. To solve the overfitting issue for the two emotions, we tried to reduce the layer size. However, the result was even worse, with no improvement in the evaluation accuracy throughout the training process. Therefore, we kept the layer size setting for Anger and Calmness model.

<table>
<thead>
<tr>
<th>Emotion</th>
<th>Layer Size</th>
<th>Loss</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Train</td>
<td>Eval</td>
</tr>
<tr>
<td>Anger</td>
<td>64, 64</td>
<td>0.7965</td>
<td>1.093</td>
</tr>
<tr>
<td>Calmness</td>
<td>64, 64</td>
<td>0.7945</td>
<td>1.082</td>
</tr>
<tr>
<td>Happiness</td>
<td>64, 64</td>
<td>1.056</td>
<td>1.149</td>
</tr>
<tr>
<td>Sadness</td>
<td>64, 64</td>
<td>0.8322</td>
<td>0.9545</td>
</tr>
</tbody>
</table>

Table 3.4 Finalized model results of Polyphonic Music Generation

In addition, it is noted that the overfitting issue of the Anger model is more serious than that of the Sadness model, but the training data size of Anger model is larger. We think that the quality of training data may be the major reason. This will be explained in the next section.
3.3.2 Music Analysis

Similarly, we first analyzed the melody structure of the generated samples. Figure 3.6 shows the piano roll of a generated sample. The melody structure of the music is loose. Neither primer melody is preserved, nor similar patterns are observed in both primer and generation part. More piano rolls are listed in Appendix III. As no extra information on melody structure was included in the feature space, our model failed to generate music of good melody structure. In addition, our model performed poorly in note composition. In many generated samples, it was observed that long notes, which were out of the pitch range of the original melody, was often included in the song irrationally, breaking the whole melody structure. Also, an unreasonably long silence was often recorded, with a sharp change in melody afterwards. In addition, multiple notes seldom existed within a time step. The quality of training data may be the cause of the above issue, which failed in providing polyphony tracks with good melody.

![Piano roll of a Happiness polyphonic music sample](image)

**Figure 3.6** Piano roll of a Happiness polyphonic music sample

Besides analyzing the melody structure, we also tried to verify the emotion of the generated music with our symbolic MER model. Note velocity augmentation, which was mentioned in section 3.2.2, was applied to generated songs for Sadness and Calmness. The verification of generated music for Happiness, Anger and Calmness was successful, with an accuracy of over 80%. The verification of generated music for Sadness remained failed.
Last but not least, comparison was made on several music features between the training dataset and the generated MIDI files. As the generated music was polyphonic, polyphony was included in the music features for comparison. Like Monophonic Music Generation, similar plots were also observed in several music features. Figure 3.7 and Figure 3.8 show a comparison of pitch entropy between the training data and the generated music. More graphs are listed in Appendix IV. Evidence shows that our model was capable of generating music for certain emotions, but the melody structure was loose and poor.

![Figure 3.7 Box plot on pitch entropy of training data](image1)

![Figure 3.8 Box plot on pitch entropy of generated polyphonic music](image2)
4. Application

We have completed the analysis of the whole Emotion-based Music Provider process. In order to host the service, we developed a web application allowing user to record a speech or upload a speech audio file, subsequently recommend or generate corresponding music given the emotion detected from their speech input. In addition, the web application allow user to display more information about how and why the selected music is being recommended or generated given your speech input.

Detailed objectives in this section are provided below.

**Objective 4.1:** Application achieving Emotion-based Music Provider services

**Objective 4.2:** An intuitive user experience and professional looking user interface

**Objective 4.3:** Backend API services allowing Emotion-based Music Provider services to be used without the frontend application

**Objective 4.4:** Deployment of web application and API services

4.1 Software and Development Workflow

Figure 4.1 shows the full web application development flow. Given that Python is used in developing the SER, Music Recommendation, and Music Generation models, we remained using Python as the environment for backend. Specifically, the Flask framework in Python is adopted to facilitate the API connection from the frontend. The frontend is developed with React JS, HTML, CSS, and JavaScript. Material UI and NIVO are the two main libraries used for UI design and diagrams, keeping the design of the application consistent.

Frontend and backend are packed in separate docker images, capturing the requirements and environments needed. Both images are then pushed to Heroku (a cloud application platform hosting websites) for web application deployment. To achieve objective 4.3, we deployed the frontend web application and the backend Emotion-based Music Provider services separately, allowing our Emotion-based Music Provider to be used as an API services, not be limited to our web application only. Finally, version control and collaboration on the web application and API development is facilitated through GitHub.
4.2 Web Application Design

As our Emotion-based Music Provider involves two major methods of “Music Providing”, we separated them into two different tabs: Music Recommendation page and Music Generation page. Section 4.2.1 illustrates the User Interface design of the application. In Section 4.2.2, the features and functionalities of the web application are presented. Section 4.2.3 summarizes the frontend with an application flow chart.

4.2.1 UI Design

Figure 4.2 and Figure 4.3 shows the User Interface (UI) of the Music Recommendation and Music Generation landing page respectively. The design of the webpage is kept in a simple and intuitive manner, letting the user to focus on the music recommended and generated for them instead of on a fancy UI. Color gradient is heavily used in the application to counterbalance the simplicity style of design.

Design between the Music Recommendation page and Music Generation page is kept consistent with same positioning of speech recording and control section (left side) and same color scheme. The webpage background color is the only major difference in UI between Music Recommendation and Music Generation page, giving user a clear indication of which page, they are in. With the music recommendation page having a bluish green to red color gradient background, and music generation page having a purple to pink color gradient background.
4.2.2 Features and Functionalities

In both recommendation and generation, users can either click to upload an audio speech file or record an audio at that moment. Recommendations and generations will be based on the emotions and information detected from the speech.
In music recommendation (Figure 4.2), since we only analyzed emotion information for songs on the Spotify dataset, all the recommended songs are linked to Spotify. There are two modes of recommendations available: audio only and audio with text. In audio only mode, recommendation is based on the audio analysis of the speech and songs (comparing similarities between emotion detected from the human speech through the SER model, with the emotion from songs through the acoustic MER model, as mentioned in paper 2). In audio with text mode, text emotion analysis and semantics analysis between human speech and music is also compared, giving a more accurate recommendation from the emotional state in the input audio speech. However, a limitation with using audio with text mode is that it only supports English speech audio, as all our text models are strictly trained with English datasets. Furthermore, user can also filter the recommended songs by genres, where the genre information of a song is provided along with the rest of the information in the Spotify API.

After the user click “recommend music”, a list of recommended songs will be returned in the order of similarity to the user’s speech emotion (Figure 4.4). User can play to listen to a 30 second preview of the recommended songs or redirect to open up the song on Spotify. The similarity is displayed besides the recommended music, user can also click “more” to open an info window, illustrating a detailed statistics and information telling why the song is recommended to the user given their speech audio.

![Figure 4.4 Application with recommended songs](attachment:application_with_recommended_songs.png)
Figure 4.5, Figure 4.6, Figure 4.7, and Figure 4.8 shows the expanded information of the analysis between speech and selected audio. If user selected audio mode, only “Overall Analysis” (Figure 4.5) and “Audio & Acoustic Analysis” (Figure 4.6) is shown. If user selected audio and text mode, “Text & Lyrics Analysis” (Figure 4.7) and “Semantics Analysis” (Figure 4.8) is also included.

In overall analysis (Figure 4.5), the selected song name, artist, and audio is shown once again. Your emotion and percentage, as well as similarities between emotion of speech and songs for all applicable components (overall, acoustic, text, and semantics) are displayed in this section.

![Overall Analysis](image)

**Figure 4.5 Application showing Overall Analysis of recommendation**

For the Audio & Acoustic Analysis (Figure 4.6), two charts are displayed. A donut chart showing the emotion percentages from the acoustics-based Speech Emotion Recognition (SER) model (explained in paper 1), and a scatter graph showing the distance of valence and arousal between the speech audio and the selected song from the acoustic-based Music Emotion Recognition (MER) model (explained in paper 2).
If user selected audio with text mode, Text & Lyrics Analysis (Figure 4.7) will be shown. In Text & Lyrics analysis, left side shows a donut data of the emotion percentages recognized from the speech audio with speech-based text emotion detection (TED) model (explained in paper 1) and speech transcript of the input audio (explained in paper 1). While right side shows the emotion percentages recognized from the lyrics of selected song using lyrics-based TED model (explained in paper 2). The lyrics of the selected song is also retrieved from Genius [9] and is presented in this section.
Finally, Semantics Analysis (Figure 4.8) is also shown if user selected audio with text mode as recommendation. In this section, the left-side list all the keywords detected from the speech audio with the corresponding significance of the keyword in the speech, by using the keywords extraction model (explained in paper 2). The right-side list the same information extracted from the lyrics of the selected song. Between the two list of keywords, a confusion matrix is presented, plotting the similarities between each keywords (model explained in paper 2).

![Semantics Analysis](image)

Figure 4.8 Application showing Semantics Analysis of recommendation

In music generation, user can choose either monophonic or polyphonic music generation. After user recorded or uploaded a speech audio and clicked generate music, a list of 3 generated music will be shown in the right with the corresponding similarity percentage and a button to show more information about the generated music.
After user clicking more information, three donut charts representing the emotion percentages are displayed, including the emotion percentages from speech audio detected through acoustic-based SER model (explained paper 1), emotion percentages of primers MIDI detected through Symbolic MER (explained in paper 2), and emotion percentages of generated music recognized from Symbolic MER.
4.2.3 Application Workflow

The flowchart below (Figure 4.11) summarizes the flow of the application for all features demonstrated in Section 4.2.2.
4.3 API Service

As we deployed the frontend and backend separately, our Emotion-based Music Provider service can be accessed through API without having to use our application. This is to facilitate any future development that utilizes only the Emotion-based Music Provider services. As the corresponding details on how the music recommendation and music generation works are illustrated in Section 2.4 of paper 2 and Section 2.3.1 of paper 3 respectively, this section only provide the details of the API services available.

Table 4.1 presents a summary of all the API endpoints conducting Emotion-based Music Provider services, including tasks of SER, Music Recommendation, and Music Generation.

<table>
<thead>
<tr>
<th>Type</th>
<th>Path</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/speech-emotion-recognition/models</td>
<td>Get SER models</td>
</tr>
<tr>
<td>POST</td>
<td>/speech-emotion-recognition/predict</td>
<td>Predict speech emotion with acoustic-based SER</td>
</tr>
<tr>
<td>POST</td>
<td>/speech-emotion-recognition/predict-by-text</td>
<td>Predict speech emotion with speech-based TED</td>
</tr>
<tr>
<td>POST</td>
<td>/music-recommendation/getsongs</td>
<td>Get recommendation of songs given speech audio</td>
</tr>
<tr>
<td>GET</td>
<td>/music-recommendation/getlyrics/&lt;id&gt;</td>
<td>Get lyrics of song given Genius ID</td>
</tr>
<tr>
<td>POST</td>
<td>/music-generation/generate</td>
<td>Get generated songs given speech audio</td>
</tr>
</tbody>
</table>

Table 4.1 Summary of all API endpoints

The following subsections talk in detail of the request and response format of all API endpoints.

4.3.1 Speech Emotion Recognition API

The following API endpoints are for Speech Emotion Recognition tasks.

4.3.1.1 Get Model Choices

Table 4.2 shows the API endpoint in getting the major SER models available to predict emotion values. Available models include the final CNN model and the final CNN-LSTM model.
4.3.1.2  **Predict with Acoustic-based Speech Emotion Recognition**

Table 4.3 illustrates the API endpoint that takes in a list of speech audio and the choice of model and returns with the emotion result of the speech audio, through analyzing the speech audio acoustically with the acoustic-based SER model (discussed in paper 1). Options of model choices can be retrieved through the “models” endpoint (Section 4.3.1.1). List of audio and model choice are packed in form data when calling API.

```json
{
  "data": [
    {
      "id": 0,
      "name": "Best CNN Model"
    },
    {
      "id": 1,
      "name": "Final Model (CNN-LSTM)"
    }
  ],
  "errMap": {
    "": "ok"
  },
  "status": "ok"
}
```
### Table 4.3 API endpoint for predicting emotion values with Acoustic-based SER model

<table>
<thead>
<tr>
<th>Type</th>
<th>POST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>/speech-emotion-recognition/predict</td>
</tr>
<tr>
<td>Response Format</td>
<td>JSON</td>
</tr>
<tr>
<td>FromData</td>
<td>modelChoice: “0” for CNN Model “1” for Final Model</td>
</tr>
<tr>
<td></td>
<td>&lt;audio_name&gt;: Audio file in Binary</td>
</tr>
</tbody>
</table>

API endpoint for predicting emotion values with Acoustic-based SER model.

#### 4.3.1.3 Predict with Speech-based Text Emotion Detection

Table 4.4 illustrates the API endpoint that takes in a list of speech audio and returns with the emotion result of the speech audio, through analyzing the text of the speech with speech-based TED model (discussed in paper 1). List of audio are packed in form data when calling API.
Type | POST  
--- | ---  
Path | /speech-emotion-recognition/predict-by-text  
Response Format | JSON  

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;audio_name&gt;</td>
<td>Audio file in Binary</td>
</tr>
</tbody>
</table>

**Table 4.4** API endpoint to predict emotion values through speech-based TED model

### 4.3.2 Music Recommendation API

The following API endpoints are for Music Recommendation tasks.

#### 4.3.2.1 Get Recommended Songs from Speech Emotion

Table 4.5 illustrates the API endpoint that takes in a single speech audio and mode of SER detection, then returns the recommended songs given the emotion value analyzed through the SER process. There are 3 types of mode to perform recommendation. “Audio”, which only analyze the acoustic aspect of the speech by using only acoustic-based SER model. “Combined”, which analyze in term so both acoustically through acoustic-based SER and textually through speech-based TED. Finally, “All”, which involve not only the acoustic and textual aspect, but also through semantics analyses as well. The whole process of music recommendation is discussed in depth in paper 2. This endpoint fixed the model choice to be the final CNN-LSTM model in prediction emotion values. The list of recommended songs is returned with in-depth details to illustrate how and why the song is recommended given such speech audio.
4.3.2.2 Get Lyrics of a Song

Table 4.6 shows the endpoint of getting the lyrics of the song. It takes in Genius song ID as the parameter, and backend will scrape the Genius song webpage indicated by the song ID and return the corresponding lyrics to front end.
### 4.3.3 Music Generation API

The following API endpoints are for Music Generation tasks.

#### 4.3.3.1 Generate Songs given Speech Emotion

Table 4.7 shows the API endpoint that takes in a single speech audio and mode of generation (monophonic or polyphonic), then returns three generated songs given the emotion value analyzed through the SER process. The whole process of music generation is discussed in depth in this paper. This endpoint fixed the model choice to be the final CNN-LSTM model in prediction emotion values. The list of generation songs is returned with in-depth details to illustrate how and why the song is generation given such speech audio.
### Type

| POST |

### Path

```
/musice-generation/generate
```

### Response Format

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FromData</td>
<td></td>
</tr>
<tr>
<td>mode</td>
<td>&quot;monophonic&quot; or &quot;polyphonic&quot;</td>
</tr>
<tr>
<td>&lt;audio_name&gt;</td>
<td>Audio file in Binary</td>
</tr>
</tbody>
</table>

Zipped file containing the generated audio(s) and info.json of information in the picture below

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
{ "data": { "generatedMusic": { "players_list": { "player": "7", "emotion": "Anger", "emotion_percentage": { "Anger": 0.0000000000000000, "Calmness": 0.0000000000000000, "Happiness": 0.0000000000000000, "Sadness": 0.0000000000000000 }, // Other information in JSON } } }, "speech_data": { "emotion": "Anger", "confidence": "0.0000000000000000", "emotion_percentage": { "Anger": 0.0000000000000000, "Calmness": 0.0000000000000000, "Happiness": 0.0000000000000000, "Sadness": 0.0000000000000000 }, // Other information in JSON } }, "status": "Ok" } |

```

### Success Response

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
{ "data": { "generatedMusic": { "players_list": { "player": "7", "emotion": "Anger", "emotion_percentage": { "Anger": 0.0000000000000000, "Calmness": 0.0000000000000000, "Happiness": 0.0000000000000000, "Sadness": 0.0000000000000000 }, // Other information in JSON } } }, "speech_data": { "emotion": "Anger", "confidence": "0.0000000000000000", "emotion_percentage": { "Anger": 0.0000000000000000, "Calmness": 0.0000000000000000, "Happiness": 0.0000000000000000, "Sadness": 0.0000000000000000 }, // Other information in JSON } }, "status": "Ok" } |

```

### Failed Response

```
1 2 3 4 5
{ "data": [], "error_msg": "Mode is not selected! Please select a mode from dropdown!", "status": "failed" } |

```

Table 4.7 API endpoint to get generated songs given a speech audio

### 4.4 Summary of Web Application

The completed web application achieved all the objectives listed in start of this section.

Objective 4.1, application achieving Emotion-based Music Provider services. Our application fully supports all functionalities in our Emotion-based Music Provider, recommending and generating music based on user’s speech input.

Objective 4.2, An intuitive user experience and professional looking user interface. We used constant theme, colors, and font to encourage a more professional look. To maintain
professionalism, we keep the design element to be simple and minimal, while using color gradient to prevent application look dull. User experience like loading screen, hover effect, auto scrolling down to info section after clicking “MORE” button, as well as interaction on the diagrams are implemented. In addition, to prevent confusion, we clearly separates the music recommendation and music generation tasks into different tabs, with different background color indicating which task the user selected.

Objective 4.3, backend API services allowing Emotion-based Music Provider services to be used without the frontend application. As explained in Section 4.3, API are deployed separately, so user can access our Emotion-based Music Provider through the API services without going to our application. To facilitate easy use of API services, API documentation is also provided and published.

Objective 4.4, deployment of web application and API services. Although, we have successfully deployed the application on Heroku (details given in Section 4.1), the application crashes when performing recommendation and generation tasks. This is due to the memory limitation of Heroku, where a higher memory and processing power Dynos requires expensive charges. Therefore, we cannot argue that we have satisfied this objective.

To conclude, our development of application successfully achieved our first three objectives under Objective 4, but faced limitation when achieving objective 4.3.

4.5 Limitation and Mitigation

There are some limitations in the application both frontend and backend, where we have tried to minimize the limitation as much as possible.

4.5.1 Processing Time

The first and major issue in the application is the delay due to processing time in the backend. In recommendation task, without incorporating the speech-based TED model, it takes approximately 10 to 15 seconds to get a response. However, if incorporating the speech-based TED model for recommendation (i.e., including textual analyses), it takes around 2 minutes to get a response. For generation task, it takes an average of 1 minute to retrieve three generated audios. Table 4.8 and Table 4.9 shows the approximate proportion of what and how much
Each sub-task in the backend takes up the processing time in Music Recommendation and Music Generation respectively.

<table>
<thead>
<tr>
<th>Delay (Music Recommendation) Task</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>4%</td>
</tr>
<tr>
<td>Audio file I/O</td>
<td>2%</td>
</tr>
<tr>
<td>Acoustic-based SER</td>
<td>2%</td>
</tr>
<tr>
<td>Speech-based TED</td>
<td>90%</td>
</tr>
<tr>
<td>Mapping and Sorting Recommendation</td>
<td>2%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Delay (Music Generation) Task</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>10%</td>
</tr>
<tr>
<td>Audio file I/O</td>
<td>5%</td>
</tr>
<tr>
<td>Acoustic-based SER</td>
<td>5%</td>
</tr>
<tr>
<td>Generation, MER, Re-generation</td>
<td>80%</td>
</tr>
</tbody>
</table>

Table 4.8 Proportion of the delay in backend for Music Recommendation task

Table 4.9 Proportion of the delay in backend for Music Generation task

Obviously, the causes of a lot of the delay is due to the prediction from a large model generally takes time. But also, because we are using Heroku to host our web application, both frontend and backend, and the processing power with the Heroku free version is far from enough for what we need to have a quick backend response. Despite this, we still adopted many different modifications in order to mitigate the delay as much as possible.

From Table 4.8, it is evident that the processing delay for music recommendation task is greatly limited by the Speech-based TED task, taking up to 90% of the time. To alleviate the problem, we selected a slightly less performative model with much fewer model parameters, but greatly increase the prediction speed. With one such simple change, we reduce the time spent in predicting through speech-based TED model by 50%.

For music generation, the only method to reduce the time spent in generating music is by reducing the duration of the music generated, as well as the number of music to be generated at each API call. Therefore, we change our application from initially allowing user to set how long of a music to generated and how much music to generate, to fixing the generation number to three, and generation duration to 70 steps (which is 45 seconds long). In addition, we change from initially requiring the backend to forever re-generate music until the emotion predicted on generated music and speech by symbolic MER model aligns, to a maximum of 3 re-generation before returning to frontend. Despite this, these are all mitigation methods to prevent generation and re-generation in taking too much of a time, and not a solution to reduce the processing time. However, by these modifications, we can guarantee that the processing time in the backend for music generation would not take longer than 1 minute 30 seconds.
Some other adjustment made that slightly improves the processing time includes loading all necessary json and model into the code before receiving any API calls, as well as storing audio file in buffer instead of performing IO operation to store audio file into backend storage.

With all the above modifications, we successfully reduce the music recommendation processing time from around 2 minutes to approximately 70 seconds and capping the time of music generation task to be always less than 2 minutes.

### 4.5.2 Embedded Spotify Player

The seconds limitation is in the application is the embedded Spotify player. In order to allow the web application to play songs from Spotify recommended by our Emotion-based Music Provider, we must use the provided Spotify player through iframe. An issue with using iframe is that it has additional overhead in rendering and maintaining this iframe instance in the frontend.

The only solution to solve this problem is by not using iframe. However, without using iframe to directly retrieve the song player from Spotify, we must first download all full-length audio of songs in our recommendation sets, store them in backend, and create our own player that plays the stored song file. With this method, not only does it require significant amount of time in downloading song audios, but also requiring a powerful server to host and store all the songs in the backend, which is not something we can currently acquire. Therefore, the mitigation to this problem is left at future work.

### 4.5.3 Limitation in Deployment

Finally, the last limitation is at deployment. As mentioned in Section 4.4, we faced memory limitation when trying to deploy our application. Heroku free plan only provides 1 Dyno, which gives limited processing power and a memory of only 256MB. As our backend requires heavy computation for recommendation and generation, the application crashes on the deployed webpage. To increase processing power and memory, larger number of Dyno must be purchased. Therefore, our application is limited by the memory and processing power provided from Heroku in its free plan.
5. Conclusion

As a whole, we have successfully completed our project as the Emotion-based Music Provider, which provides user with music of certain emotions through recommendation and generation. The project separates into three stages, Speech Emotion Recognition, Music Recommendation, and Music generation, where 3 reports are written corresponding to each stage of the work. This section serves as the conclusion of all three stages, wrapping up the whole project.

For Speech Emotion Recognition (SER), we adopted a CNN-LSTM model involving 4 convolutional maxpooling block, and one Bi-LSTM block. The data is being augmented and the dataset size is multiplied by 7 times. The SER model predict four emotions, including Happiness, Calmness, Sadness, and Anger. The model achieved a testing accuracy of 62%. It is adopted as our finalized model.

For Music Recommendation, two approaches, Acoustic Music Emotion Recognition (Acoustic MER) and Text Emotion Detection (TED), are considered to classify music emotion. Utilizing the acoustic features extracted from music audio, the Acoustic MER applies deep learning techniques to predict the valence-arousal value of the music. TED applies Natural Language Processing techniques to analyse the emotion of the lyrics. To recommend songs, the system will find a suitable song for the user by comparing the similarity between the user’s speech emotion and the songs’ emotion. Additionality, the semantics of the speech from the user are also analysed. They will be compared with the semantics of the lyrics to provide better recommendation results.

For Music Generation, we adopted the Attention RNN and Polyphonic RNN models from Magenta [7] [8] to achieve monophonic and polyphonic music generation respectively. Symbolic MER models were applied to assign emotion labels onto training data. Both models can generate symbolic music of specific emotions. Our monophonic models are capable of generating music of good melody structure, while the polyphonic one fails to do so.

The application separates the recommendation and generation into two tabs, uniquely providing the services of each part. User first record an audio of speech or upload the audio file, then the application recommends or generates corresponding songs based on the emotion and semantics values. User can expand each song to retrieve details on reasons why the song is being recommended or generated.
Music is a form of creativity. For machines to provide self-composed music as human beings, there is still a long way to go. Yet, advancement of technology has proven this to be possible. Good progress is shown in both emotion recognition and music generation. We are looking forward to the appearance of the first ideal “Emotion-based Music Provider”.
References


Appendix I

Happiness:

Anger:
Anger:

Sadness:
Calmness:
Appendix II

<table>
<thead>
<tr>
<th>Music Feature</th>
<th>Training dataset</th>
<th>Generated Monophonic MIDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note Density</td>
<td><img src="image1" alt="Note Density Average Box Plot" /></td>
<td><img src="image2" alt="Note Length Average Box Plot" /></td>
</tr>
<tr>
<td>Average Pitch</td>
<td><img src="image3" alt="Average Pitch Average Box Plot" /></td>
<td><img src="image4" alt="Average Pitch Box Plot" /></td>
</tr>
<tr>
<td>80% Pitch Range</td>
<td><img src="image5" alt="80% Pitch Range Box Plot" /></td>
<td><img src="image6" alt="80% Pitch Range Box Plot" /></td>
</tr>
<tr>
<td>Pitch Entropy</td>
<td><img src="image7" alt="Pitch Entropy Box Plot" /></td>
<td><img src="image8" alt="Pitch Entropy Box Plot" /></td>
</tr>
</tbody>
</table>
Appendix III

Happiness:

Anger:
Sadness:

Calmness:
Appendix IV

<table>
<thead>
<tr>
<th>Music Feature</th>
<th>Training dataset</th>
<th>Generated Polyphonic MIDI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note Density</strong></td>
<td><img src="image1.png" alt="Note Density Average Box Plot" /></td>
<td><img src="image2.png" alt="Note Density Average Box Plot" /></td>
</tr>
<tr>
<td><strong>Note Length</strong></td>
<td><img src="image3.png" alt="Note Length Average Box Plot" /></td>
<td><img src="image4.png" alt="Note Length Average Box Plot" /></td>
</tr>
<tr>
<td><strong>Average Pitch</strong></td>
<td><img src="image5.png" alt="Pitch Average Box Plot" /></td>
<td><img src="image6.png" alt="Pitch Average Box Plot" /></td>
</tr>
<tr>
<td><strong>80% Pitch Range</strong></td>
<td><img src="image7.png" alt="80% Pitch Range Box Plot" /></td>
<td><img src="image8.png" alt="80% Pitch Range Box Plot" /></td>
</tr>
</tbody>
</table>
Polyphony