Song Recommendation Based on User's Facial Emotion

COMPSCI 9542B Artificial Intelligence - II

Neha Mary Thomas
University of Western Ontario
London, Canada
nthoma54@uwo.ca

Hanumanthu Susmitha
University of Western Ontario
London, Canada
shanuman@uwo.ca

Pavani Madhur
University of Western Ontario
London, Canada
pmadhur@uwo.ca

Abstract— **Music plays a significant part in one's life. It is known to uplift the mood of a person. Analyzing the expression of an individual would help to understand the emotion one is going through. Many users have a playlist so large that they are unable to decide which song to play, thus a recommendation system would suggest songs based on the mood of the individual. The user image is given as an input to our system which predicts the emotion of the image and according to the emotion songs are recommended. The songs are classified into happy, sad, calm and energetic based on the features from the Spotify API. Three models were applied to the classified song data set namely Support Vector Machine, Random Forest and K-Nearest Neighbor Algorithm of which Random Forest showed the maximum test accuracy of 85%. The algorithms used for expression detection includes Convolution Neural Network, Support Vector Machine, K-Nearest Neighbor Algorithm of which Convolution Neural Network showed the maximum test accuracy of 64%. Thus, we used Convolution Neural Network and Random Forest for our project. We also have a provision to add new songs, the song input should be the trackid from the Spotify API. These songs would be dynamically classified and added to the playlist.**

I. INTRODUCTION

Music plays an integral part in our life. Research in the field of music has shown that music induces a clear emotional response in its listeners [1] and also said to have a therapeutic effect. Hence, it is necessary to play a song based on the emotional state of a person. Existing music players would satisfy a user's basic requirement but the user would have to browse his/her song manually from a large set of songs from the playlist according to the mood [2]. Uplifting the mood of a person could help overcoming mental trauma and not push a person to the extent of depression.

A user's response to a piece of music would depend on many factors like age, personal preferences, emotion and culture. However, keeping aside these external factors, humans are able to categorize songs as happy, sad, energetic or calm [1]. Facial Expression is the most common way of understanding the emotion of a person. Emotion detection has many applications. For example, it can be used to understand customer experience instead of filling out the customer experience survey. This project has divided expressions as happy, sad, neutral, angry, surprise, contempt, fear.

This project has three modules namely **emotion detection module** where the user inputs an image and with the help of a deep learning algorithm the image is classified. The second module is the **music classification module** which classifies the song based on a supervised algorithm into happy, sad, energetic or calm. A user can also add new songs to the already existing playlist by giving the trackid from the Spotify API. This new song would get classified into one of the 4 different classes. The third module is the **music recommendation module** which combines the output of emotion module and song classification module to display the final list of songs to the user [1]. This module takes the emotion of the image as input and displays 10 songs randomly from the playlist. If the emotion is happy, then songs from the playlist happy would be returned. If the emotion is surprise or neutral then energetic song playlist would be displayed. If the emotion is sad or disgust then songs from the sad playlist would be displayed. If the emotion is fear or angry then the songs from the calm playlist would be displayed.

The rest of the paper is organized as follows: Section II presents the existing work done on the face recognition and song classification modules. Section III describes the data used in the project. Section IV explains the methods implemented as a part of this project. Section V presents the results of the project along with the proposed application. Section VI talks about further enhancements while section VII concludes the paper.

II. RELATED WORK

Various Methods have been proposed to classify the emotional state of the user. Mase et al. focused on using movements of facial muscles while Tian et al. attempted to recognize Actions Units [1]. As years passed, researchers wanted to use deep learning algorithms like Convolutional Neural Network (CNN) as this technique reduces the dependency on the physics of the face model which means no extra feature extraction step is required for these types of models [6]. Shlok Gilda et al. used CNN for emotion detection and classified songs based on the audio features of the song. The recommender system used to display songs based on the emotion of the image [1]. In 2019, Ahlam et al., [4] proposed a system which to detect the face and capture the expression of the user using PCA and plays music accordingly, but only four emotions are considered.

Music can be classified by using lyrical analysis. Although this method is easier to implement but it not advised for classification of songs because of the language barrier which restricts the classification to a single language [1]. Another method is using acoustic features like tempo, pitch to identify the sentiment conveyed by the song. This method involves extracting a set of features and using those feature vectors to find patterns characteristic to a specific mood [7].

III. DATA

The dataset we used for training the model is from a Kaggle Facial Expression Recognition Challenge, FER2013 [11]. The data consists of 48×48-pixel grayscale images of faces. Each of the faces are organized into one of the 7 emotion classes: angry, disgust, fear, happy, sad, surprise, and neutral. There is a total of 22074 images which are classified into train data and test data. The images of the train data folder are further split into train dataset and validation dataset in a 80:20 split.

TABLE I
FER2013 DATASET

Emotion	Train Data	Validation Data	Test Data
Happy	5772	1443	1774
Sad	3864	993	1233
Neutral	3972	966	1247
Surprise	2536	634	831
Fear	3277	819	1024
Disgust	348	87	111
Angry	3196	799	958

Another data set used for image classification is a ck-48+ data set downloaded from Kaggle. This is a subset of the Cohn Kanade Extended (CK+) dataset [12]. This data set also consists of grayscale images. There is only a single folder and the images can be classified into one of the 7 emotion classes: happy, sadness, surprise, fear, disgust, anger, contempt. The total number of images in the data set are 976. The images are split into train dataset and test dataset with a total of 783 images for the train data and 193 images for the test data.

TABLE II
CK- 48+ DATASET

Emotion	Train Data	Validation Data
Happy	165	42
Sadness	67	17
Surprise	199	50
Fear	60	15
Disgust	141	36
Anger	108	27
Contempt	43	11

The data set used for song classification is an already classified dataset consisting of 648 songs divided into 4 classes namely happy, sad, neutral, energetic [10]. This dataset consists of columns related to the audio features of the song. The data set is divided into train and test data set in the ratio 80:20. The features includes :

TABLE III
Spotify Song Features

Acousticness	Measure of whether the track is acoustic
Danceability	How suitable a track is for dancing
Energy	Perceptual measure of intensity and activity
Instrumentalness	Predicts whether a track contains no vocals
Liveness	Presence of audience in a recording
Valence	Musical Positiveness conveyed by the track
Loudness	Overall loudness in decibels(dB)
Speechiness	Presence of spoken words (Identify podcasts, live show)
Tempo	Overall tempo of a track in beats per minute (BPM)

IV METHODS

A. *Landmark Detection Algorithm*

This is a pre-processing step in analysing an image to detect faces and extract facial features from the images. This is supported by Dlib which is an advanced machine learning library. A face has several distinct features like eyes, nose that can be identified by mapping a set of points around a feature.

The algorithm is divided into 2 parts. The first part is the face detection. Dlib library supports a function get_frontal_face_detector() whose input should be a grayscale image and returns a rectangle(x, y, w, h) comprising of points related to a face in the image. The next part is the facial landmark detection which takes the output of the face detection and passes through a Dlib function shape_predictor() along with a pre-trained model as an argument to get the 68 landmark points. This pretrained model is available as a dat file shape_predictor_68_face_landmarks which was downloadable from the internet.

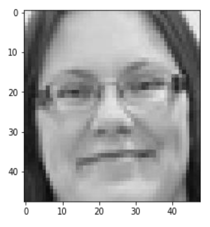


Fig. 1: An image from FER data set

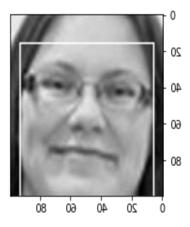


Fig. 2: Image after Face Detection

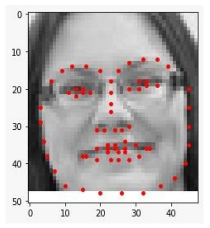


Fig. 3: Image after Landmarkdetection

Hyperparameter of an algorithm controls the capacity of a model. It includes flexibility, degrees of freedom it has in fitting the data. Proper control of model capacity can prevent overfitting, which happens when the model is too flexible, and the training process adapts too much to the training data losing predictive accuracy on new test data. Thus, a proper setting of the hyperparameters is important [3]. The different methods for hyperparameter tuning include:

- GridSearchCV - This function comes under Scikit-learn’s model_selection package. It tries all combinations of values passed in the dictionary and using cross-validation evaluates the model and the hyperparameter with the maximum accuracy is returned.

- RandomizedSearchCV- This function comes under Scikit-learn's model_selection package. It works similar to GridSearchCV except that not all combinations are tried out but a fixed number of parameter settings are sampled from the dictionary hence it is quicker than GridSearchCV.

The hyperparameter tuning used for this project is GridSearchCV as RandomizedSearchCV returns different hyperparameter for every execution as the combinations taken for hyperparameter tuning may vary with each execution and only a limited number of combinations can be tried for every execution.

B. Algorithms

Support Vector Machine (SVM)

This supervised learning algorithm works on the principle of finding a hyperplane in an N-dimensional space where N is the number of features that can distinctly classify the data points [4]. The hyperparameters used in SVM are 'C' which controls the amount of regularization in data, gamma which controls the distance of influence of a single training point. Finding an optimal hyperplane is relatively easy for linearly separable data but for non-linear data this algorithm makes use of kernel. The main idea is that the data points are mapped on to a different dimensional space in which they are linearly separable. A kernel function shows a similarity measure between the data points of the original and new dimensional space. There are different types of kernel:

- Linear Kernel –The hyperparameter for Linear Kernel is 'C'. Training an SVM with Linear Kernel is faster than other kernels.

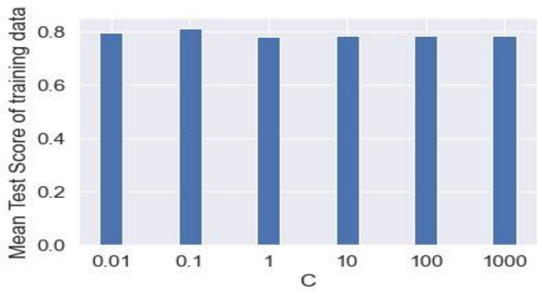


Fig. 4: Song Classification - Hyperparameter tuning for Linear Kernel

- Polynomial Kernel – It takes an additional parameter 'degree' that controls the model complexity and computational cost.

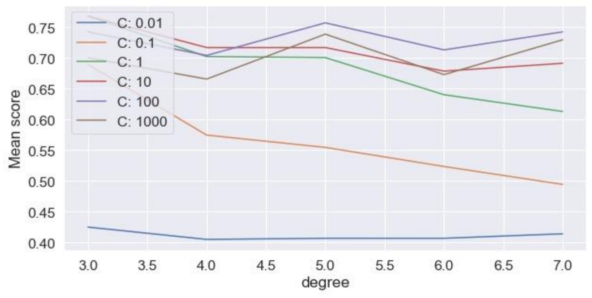


Fig. 5: Song Classification - Hyperparameter tuning for Polynomial Kernel

- Radial Basis Function Kernel (RBF) - The hyperparameters used are 'C' and gamma. The similarity between two points (X1, X2) in the transformed feature space is an exponentially decaying function of the distance between the vectors and the original input space as shown below [5]:

$$K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$$

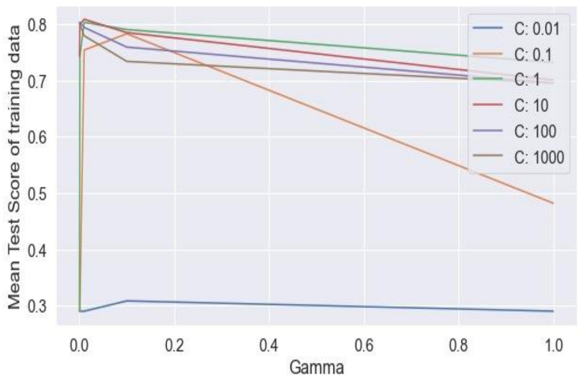


Fig. 6: Song Classification - Hyperparameter tuning for RBF Kernel

- *Comparision of Validation Accuracy of all the kernels*

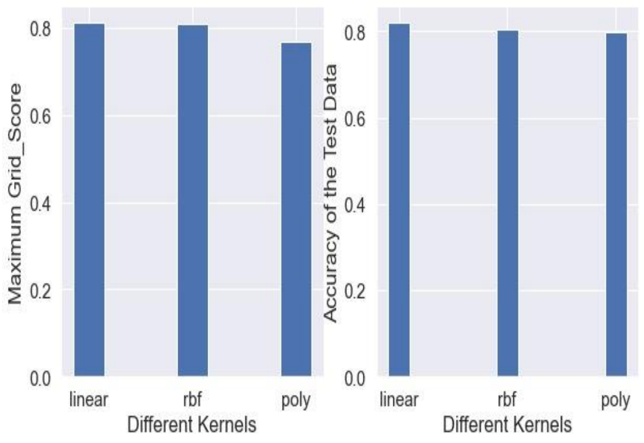
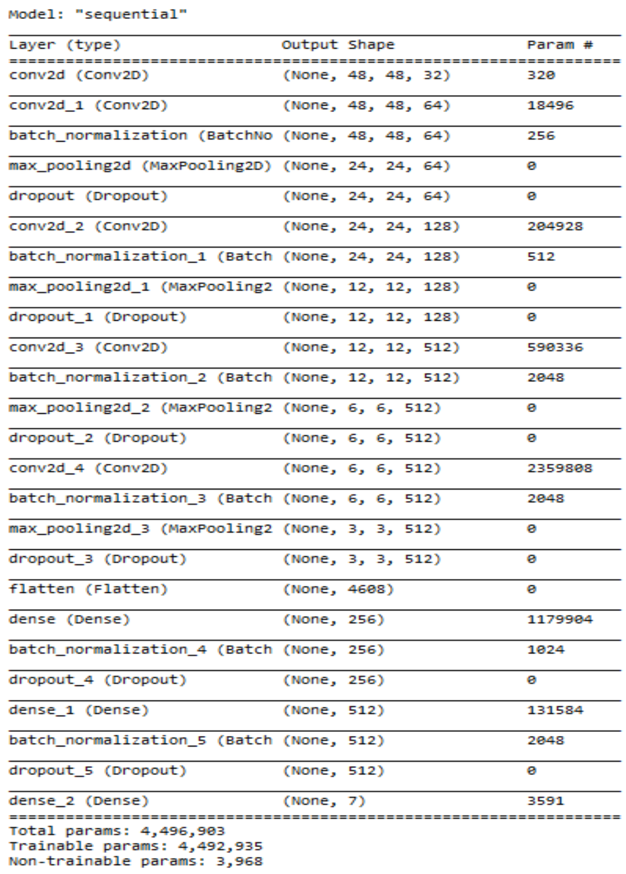


Fig. 7 : Accuracy vs kernels

Convolution Neural Networks (CNN)

This is deep learning neural network algorithm known for stimulating human brain when analysing images [1]. A multi layered neural network has been implemented to evaluate the image features. It contains five convolutional layers and three fully connected layers with pooling and batch normalization layers in between. Relu has been used as the activation function at all layers except the last dense layer (output) where softmax function was used. Fig 8 provides the summary of the model used.

The hyperparameters used in CNN are epochs, batch_size, learning rate. Epochs represents the number of the times the model has to run for the given data. Batch_size represents the number of samples to the network after the weights are updated. and learning rate refers to the speed at which the network updates its parameters, Adam has been used as optimizer in this model. Manual search method is used to find the best parameters.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 48, 48, 32)        320
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 48, 48, 64)        18496
_________________________________________________________________
batch_normalization (BatchNo (None, 48, 48, 64)        256
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 24, 24, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 24, 24, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 24, 24, 128)       204928
_________________________________________________________________
batch_normalization_1 (Batch (None, 24, 24, 128)       512
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 128)       0
_________________________________________________________________
dropout_1 (Dropout)          (None, 12, 12, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 12, 12, 512)       590336
_________________________________________________________________
batch_normalization_2 (Batch (None, 12, 12, 512)       2048
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 512)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 512)         0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 6, 6, 512)         2359808
_________________________________________________________________
batch_normalization_3 (Batch (None, 6, 6, 512)         2048
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 3, 3, 512)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 3, 3, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 4608)              0
_________________________________________________________________
dense (Dense)                (None, 256)               1179904
_________________________________________________________________
batch_normalization_4 (Batch (None, 256)               1024
_________________________________________________________________
dropout_4 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               131584
_________________________________________________________________
batch_normalization_5 (Batch (None, 512)               2048
_________________________________________________________________
dropout_5 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 3591
=================================================================
Total params: 4,496,903
Trainable params: 4,492,935
Non-trainable params: 3,968
```

Fig. 8: CNN model Summary

K-nearest Neighbors Algorithm (KNN)

This supervised learning algorithm classifies data points according to how its neighbours are classified. It is used for classification and regression problems. The most common hyperparameter is n_neighbors (k) which is the number of neighbours that needs to be considered while taking a majority vote to classify the data point. For regression problems, the Euclidian distance between the data point to be classified and already classified data points are calculated and depending on the number of neighbours we need to rank the distance in ascending order and fetch the mean of first k Euclidian distance.

Song Classification Module:

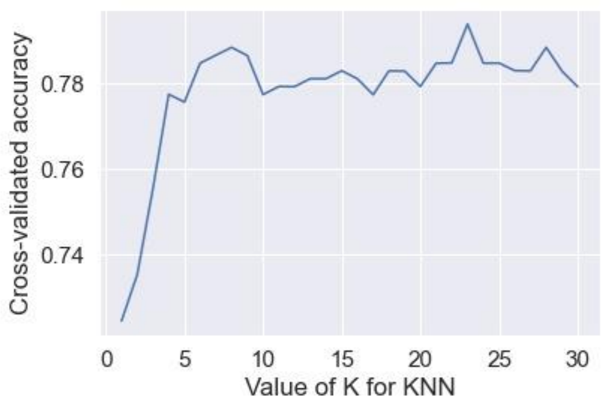


Fig. 9: KNN- Hyperparameter Tuning for Spotify data

Emotion Detection Module:

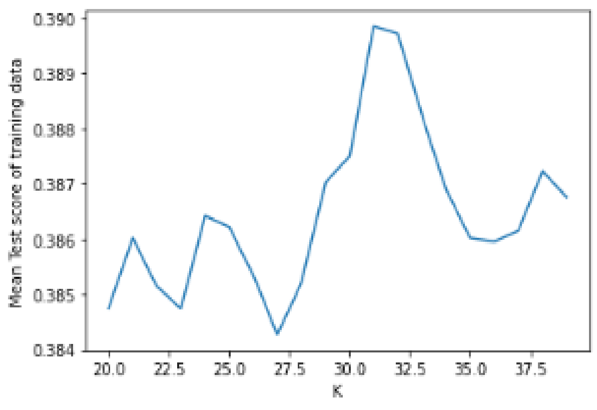


Fig. 10: KNN-Hyperparameter Tuning for FER Dataset

Random Forest

This supervised learning algorithm is used for classification and regression tasks. It utilizes multiple decision tree and the output of every decision tree is cumulated to predict the final result of the data point. The hyperparameters used for this project includes:

a) n_estimators – Denotes the number of trees in the forest

b) max_features Maximum number of features considered for splitting a node

c) max_depth - Maximum number of levels in each tree

d) min_samples_split- Minimum number of data points placed in a node before split

e) min_samples_leaf- Minimum number of data points allowed in a leaf node

f) bootstrap – Sampling of data should be done with or without placement

TABLE IV
Hyperparameter Tuning for FER Dataset

Algorithm	Hyperparameter	Input	Best Parameter
SVM	C	1-10	8.95
	Gamma	Reciprocal (0.001,0.1)	0.03
KNN	N_neighbors	1-30	23
CNN	Epochs Batch Size Learning Rate	40,50,60,75,100 32, 64 0.0001, 0.001	100 32 0.0001

TABLE V
Hyperparameter Tuning for CK+ Dataset

Algorithm	Hyperparameter	Input	Best Parameter
SVM	C	0.1, 1, 10,0.01, 0.001, 0.0001	0.01
	Kernel	Linear	
KNN	N_neighbors	1-30	1
CNN	Epochs Batch Size Learning Rate	40,50,60,75,100 32, 64 0.0001, 0.001	100 32 0.001

TABLE VI
Hyperparameter Tuning for SongClassification

Algorithm	Hyperparameter	Input	Best Parameter
KNN	n_neighbors	1-30	23
Random Forest	n_estimator	100, 200, 300, 400, 500, 600, 700, 800, 900, 1000	100
	max_features	auto, sqrt	Sqrt
	max_depth	1-15	8
	min_samples_split	2,5,10	5
	min_samples_leaf	1,2,4	2
	Bootstrap	True, False	False
SVM	Linear o C	0.01, 0.1, 1, 10, 100, 1000	0.1
	RBF o C o gamma	0.01,0.1, 1, 10, 100, 1000 1, 0.1, 0.01, 0.001, 0.0001	1 0.1
	Poly o C o degree	0.01,0.1, 1, 10, 100, 1000 3,4,5,6,7	100 5

V. RESULTS

A. *Emotion Detection Module*

CNN, KNN, SVM have been applied to FER data set and CK+ data set and the following accuracies were obtained.

TABLE VII
Accuracy table for FER Dataset

Algorithm	Validation Data Accuracy (%)	Test Data Accuracy (%)
CNN	65.95	65.90
SVM	52.98	53
KNN	46	45

TABLE VIII
Accuracy table for CK+ Dataset

Algorithm	Validation Data Accuracy (%)
CNN	81.7
SVM	91
KNN	90

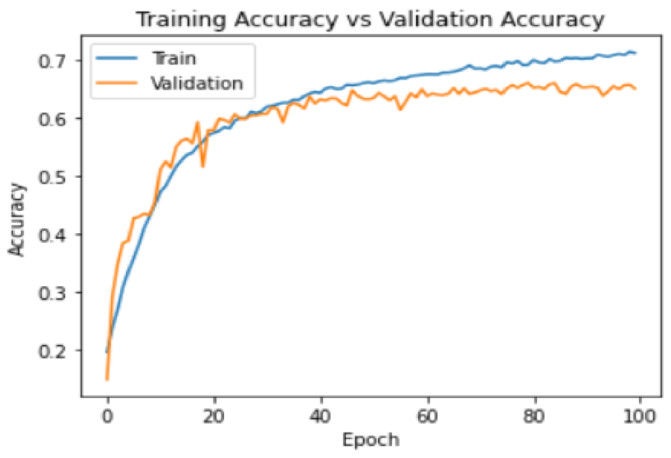


Fig. 11: Training and validation accuracy of trained CNN model

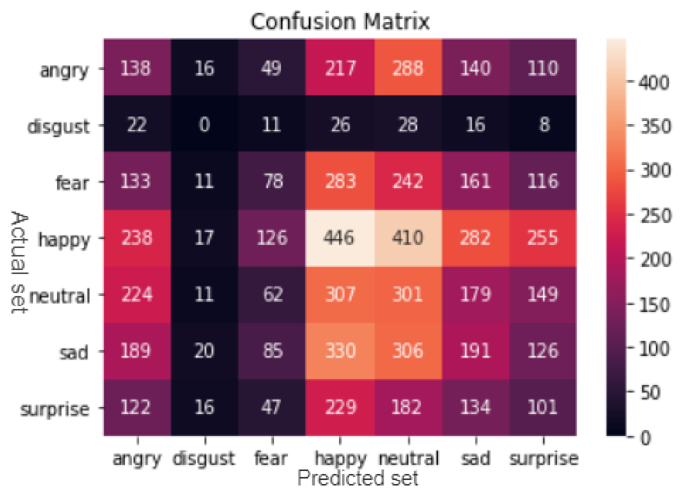


Fig. 12: Confusion matrix of CNN for Fer2013 dataset

B. *Song Classification Module*

SVM, KNN and Random Forest have been applied to song dataset and the following accuracies were obtained:

TABLE IX
Accuracy table for Song Dataset

Algorithm	Validation Data Accuracy (%)
SVM	80.43
KNN	79
Random Forest	85

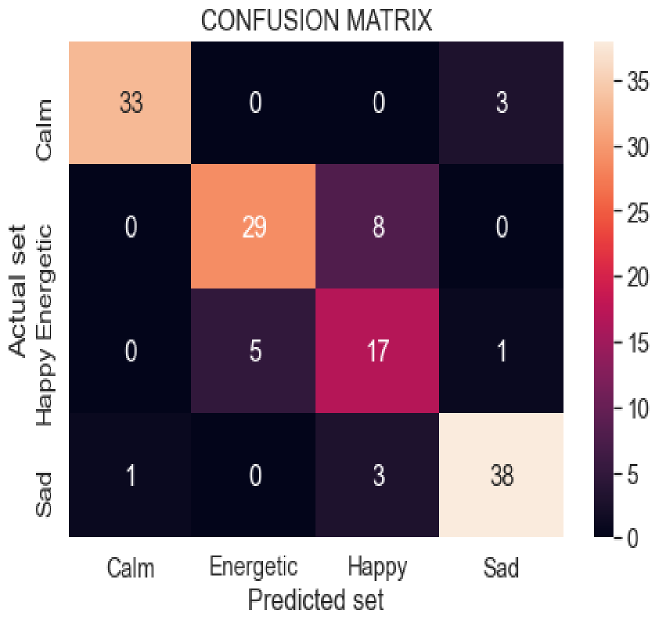


Fig. 13: Confusion matrix of Random Forest for Song Classification Dataset

Accuracy is the metric we have chosen to score the algorithms and decide the best to integrate with our application. For the song classification module, Random Forest gave a maximum accuracy of 85%.

Our project has developed an application using the CNN algorithm used in FER dataset for the facial expression detection and Random Forest algorithm for the song classification. Even though SVM gave the highest accuracy for face recognition, we have chosen CNN as it worked well with huge data as compared to SVM, so in deciding the algorithm for facial recognition both accuracy and size of the input data are considered. The user interface of the application was developed using HTML, JavaScript and web application framework Flask. There are 2 user inputs in this application; one can add a new track to an existing playlist, this is done by integrating our web application with the Spotify API. Through the Spotify Web API, external applications retrieve Spotify content such as album data, playlists, audio features of the track. To get the data through Web API, an application must be authorized by the user to access that particular information. The primary steps involved in retrieving the information from the Spotify web API are register an application with Spotify, authenticate a user and get authorization to access user data. Once the authorization is successful, audio-features [8] web API endpoint can be used to retrieve the audio features of the track. After fetching the audio features, the track is classified into one of the 4 different classes using the already saved trained Random Forest.

The second user input is the image. The image is classified using the already saved trained CNN model and based on the output of the expression, it displays 10 songs randomly from the playlist according to the type of emotion. Along with this, we gave a provision to play the track by using Spotify Widgets which provides an embeddable view of a track within the web project [9].

The following figures shows the user interface of the application

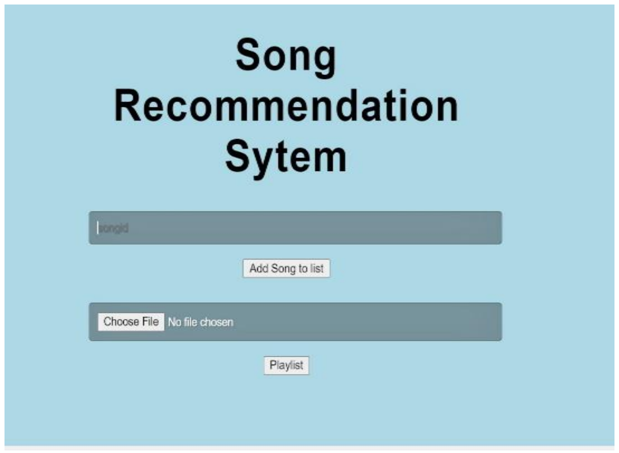


Fig. 14: UI of the application

Fig. 15: Sample input image

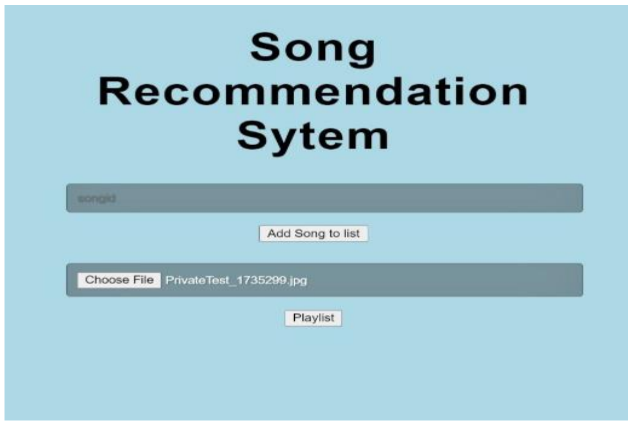
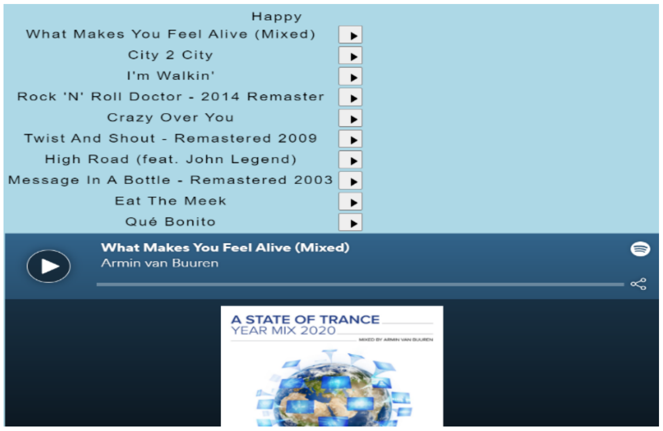


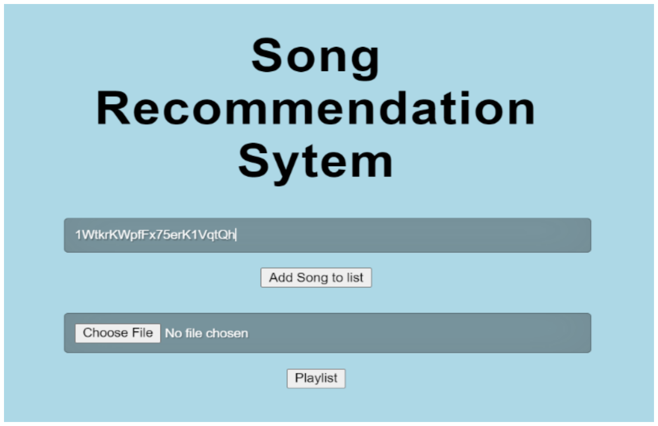
Fig. 16: User inputting an image



Happy
What Makes You Feel Alive (Mixed)
City 2 City
I'm Walkin'
Rock 'N' Roll Doctor - 2014 Remaster
Crazy Over You
Twist And Shout - Remastered 2009
High Road (feat. John Legend)
Message In A Bottle - Remastered 2003
Eat The Meek
Qué Bonito

What Makes You Feel Alive (Mixed)
Armin van Buuren

Fig. 17 : List of recommended songs for happy face



Song Recommendation Sytem

1WtkrKWpfFx75erK1VqtQh

Add Song to list

Choose File No file chosen

Playlist

Fig. 18: Adding new track to playlist

VI. FUTURE WORK

In the current implementation the user is expected to upload an image which can be further enhanced by developing the system such that it captures a real time picture with a webcam or other interactive device. Integrating the application with users own Spotify account and creating a personal playlist will benefit the application even more. Automated classification and addition of newly released songs using the API's from Spotify will keep the system up to date and enhance the user experience. Further the application can be integrated with other music applications like tidal, Apple music. The current accuracy of face recognition module using CNN for testdata is 68% when all the seven emotions are included. More work can be done on improving the accuracy of that particular module.

VII. CONCLUSION

In this project, we wanted to learn and implement some of the most common Machine Learning algorithms for different datasets and use cases. By comparing the accuracies of the models, we picked the algorithm with highest accuracy for each module, Convolutional neural network and Random Forest for emotion recognition and song classification respectively. The proposed system was able to detect the emotion of the user and song and successfully suggested a playlist for the user based on their mood.

REFERENCES

[1] Smart Music Player Integrating Facial Emotion Recognition and Music Mood Recommendation – Shlok Gilda, Husain Zafar, Chintan Soni, Kshitija Waghurdekar IEEE WiSPNET 2017 conference.

[2] Mood Based Music System – A.S. Mali, A.A. Kenjale, P.M. Ghatage, A.G. Deshpande ISROSET Research paper

[3] Evaluating Machine Learning models – Hyperparameter Tuning, Alice Zheng https://www.oreilly.com/library/view/evaluating-machine-learning/9781492048756/ch04.html

[4] Support Vector Machine – Introduction to Machine Learning Algorithms, Rohith Gandhi, 2018 https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[5] Introduction to Support vector Machines (SVM), 2020 https://www.geeksforgeeks.org/introduction-to-support-vector-machines-svm/

[6] Music Recommender System for users based on Emotion Detection through Facial Features

[7] Bo Shao, Dingding Wang, Tao Li, and Mitsunori Ogihara, "Music recommendation based on acoustic features and user access patterns," IEEE Transactions on Audio, Speech, and Language Processing, vol. 17, no. 8, Nov. 2009.

[8] https://developer.spotify.com/console/get-audio-features-track/

[9] https://developer.spotify.com/documentation/widgets/generate/embed/

[10] Spotify music data to identify the moods, https://www.kaggle.com/musicblogger/spotify-music-data-to-identify-the-mood

[11] Fer 2013, https://www.kaggle.com/msambare/fer2013

[12] CKPLUS, https://www.kaggle.com/shawon10/ckplus