Music Recommendation based on Facial Emotion Detection

Sonika Malik
Assistant Professor
Department of Information Technology
Maharaja Surajmal Institute of Technology, Delhi, India

ABSTRACT

Human face expressions directly express what's going on inside a person. Detecting someone's emotions is not a difficult task for a human but for a machine it can be difficult. Also when it comes to sear a song according to our mood, is very difficult and confusing task. So, keeping in mind here we have designed a model which can easily detect what's going on inside a human and recommend him few songs according to his mood using facial expressions. On the basis of the prediction of emotions, the goal is to play the song that best fits the mood reflected by our expression. The major task in the paper involves the detection of human face, extract the features of face and detect emotion, predict the emotions of new face, and play song on the basis of that emotion.

Keywords

Facial Emotion Detection

1. INTRODUCTION

Music has the ability to excite powerful emotional responses. Listening to music is an easy way to change mood or relieve stress. There a lot ofdifferent form of music which reflects different emotions / moods of a person? People tend to listen music according to theirchanging moods. But they have to select the music manually by going through playlist of songs and play song on the basis of theircurrent emotional state whenever they need so. But to avoid the difficulty to select a song, some people might play any random song which may be not a fit to their current mood and may disappoint the user. The idea of recommending a song based on thecurrent mood of a user will not only relieves the stress and enlighten the mood of the user but at the same time will save their time and remove overhead of selecting a song from their playlist. People express their emotions through facial expressions. Humans make use of facial expression to express clearlywhat they want to say. With the help of recommendation system, it could help a user to decide the music one should listen therebyhelping the user to reduce his/her stress level. In other words, this feature lightens up the mood of the user by playing those songs that match the requirements of the user by capturing image of the user. Facial expression is the best form of expression analysis that is being known to human kind. People can conclude the emotions/thoughts of another person with the help of their facial expressions [1, 2].

The aim of this paper is to capture the image through webcam and display the emotion and play song according to the displayed emotion. The methodology used here is CNN (Convolutional Neural Network). A CNN is most commonly applied to analyzing visual imaginary. So CNN turns out to be the best way to figure out the tasks in computer vision. Working on human emotion detection using CNN is the most preferable way of doing it as the chances of error is very less here. In this article five-layered CNN is used. The project focuses on brighten the mood of the user by playing the song that match mood of the user. The facial expression is broadly categorized into seven types like happy, angry, sad, disgusted, fear, surprise and neutral [3]. This project also makes use of the C++ dlib library which increases the overall performance of the project [4]. On further expansions of this paper we can help study human face and form lie detector using facial expressions. Not only this but by adding more features to it, may help in us to achieve many things like detection of human behavior in suspicious circumstances, which may help in reduction of many bad elements of society and also in huge reduction of terrorism.

The following are the contributions:

- To implement the ideas of machine learning, deep learning and neural networks for entertainment.
- To provide a platform with new features for music lovers and an interface between the music system and users.
- To reduce the headache of searching music and provide a new entertainment for the users.

2. RELATED WORK

With Emotion Detection and Characterization Using Facial Features [5], faces can be detected from any given image, their features (eyes and lips) extracted, and their emotions classified into six categories (happy, fear, anger, disgust, neutral, sadness). Using Grid Search, the training data is refined after it passes through several filters and processes. A classification report is then generated based on the testing data and its labels. The best results have been obtained by passing the training images through HOG followed by SVM characterization, resulting in an average precision of 85%.

An approach to improving music recommendation systems is presented in a novel music recommendation system using deep learning [6]. Various platforms and domains could benefit from the proposed solution, such as YouTube (videos), Netflix (movies), and Amazon (shopping). As more variables are introduced, current systems become inefficient. As input to a deep learning classification model, Tunes Recommendation System (T-RECSYS) combines content-based and collaborative filtering. Based on Spotify Recsys Challenge data, the authors achieve precision scores of 88% at a balanced discrimination threshold.

Researchers developed a model to identify a model that recognizes facial micro-expressions and recommends music based on mood in Research on Automatic Music Recommendation Algorithm Based on Facial Micro-

Expression Recognition [7]. In this model, convolutional neural networks are combined with micro expression recognition technology in order to identify facial micro expressions and recommend music accordingly. 62.1% of the songs were recognized using this model. Content-based music recommendation algorithms are used to extract the feature vector of the song and cosine similarity algorithms are used to make the music recommendation.

An analysis of the recognition of seven emotional states from facial expressions (neutral, joy, sadness, surprise, anger, fear, disgust) [8]. In the process of recognizing emotions, k expressions play a significant role in determining features of a person. The face is the most exposed part of the body, which makes it possible to analyse the image of the face for recognizing emotions using (typically cameras). Due to its low price and simplicity, the authors used Microsoft Kinect for 3D face modelling in this experiment.

Using deep learning, face detection has advanced significantly over the past few years, often outperforming traditional computer vision methods. A face detection algorithm was automatically learned and synthesized using CNN by Zhan et al. [9]. In a multi-task discriminative learning framework, Li et al. [10] combined a ConvNet with a 3D mean face model for detecting faces in the wild. As a state-of-the-art generic object detector, Faster R-CNN was applied by authors in [11, 12], achieving promising results. A lot of work has also been put into improving Faster R-CNN. In [13], end-to-end optimization was realized through the joint use of CNN cascades, region proposal networks (RPNs) and Faster R-CNNs. [14] Wan et al. successfully enhanced the detection performance of FDDB by combining Faster R-CNN face detection algorithm with hard negative mining and ResNet.

3. DATA SET USED

The data consists of 48x48 pixel gray scale images of faces. The task is to categorize each face based on the emotion shown in the facial expression of the person into one of these seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral). This dataset was prepared by Pierre-Luc Carrier and Aaron Courville, for an ongoing research project [15].

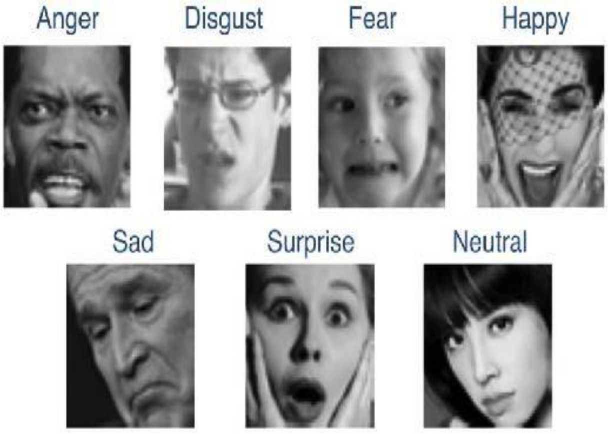


Fig 1. Dataset Images

Before making model, we observed that dataset was not of a good size and had unbalanced data. Therefore, we decided to use image augmentation. We augmented images of lesser in size up to a scale of 2.5x where they were nearly equal to the size of major emotion category which is “Neutral” or “Happy”. Now again putting all data together we again augmented the image to meet a fairly large amount of data in order to get the desirable size. Now our data was ready to get inserted to a model. In this paper we have used a seven-block model for the prediction. The first block starting with a convolution layer with filter size of 32 and kernel size of 3*3. We used He_normal kernel initializer with same padding as ReLu Activation Function is used and the best outcome is through this type of initializers. The input size of this block is same as dataset image size (i.e. 48x48). Same Convolution Layer is again added to same before and after applying Batch Normalization and at last the pooling payer is added with pool size of 2x2. The output we get now will be of size 64 to make model more robust we added a dropout layer with dropout value of 0.2. This now completes the block for us.

We then created 2nd, 3rd & 4th block with same architecture. Every time due to adding pooling layer we had to increase the input filter size by a factor of 2 so, 2nd, 3rd,4th block had filter of size 64, 128,256 respectively.

The fifth block was the ANN layer (flattened). This too was used ReLu activation function and we used a dropout layer with dropout factor of 0.5 this time.

Typical similar sixth block is used. The only difference is that this time no dropout layer is used as this is the last layer with connects to the predictor layer.

The Final block in the model where the prediction is summed up is in probabilistic form. This Layer has the sigmoid activation function which helps to define the value of our probability in the form in which our result can be obtained in such a way that using argmax function we can find our desired output in categorical form.

4. PROPOSED WORK

Based on the user's facial expressions, the proposed system extracts facial landmarks, which are then classified based on a particular emotion. Songs matching the user's emotions will be displayed once the emotion has been classified. Basically We have used four steps:

- Face Detection
- Emotion Detection
- Emotion Recognition
- Song Playing

Face Detection

Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class. Examples include upper torsos, pedestrians, and cars. Face-detection algorithms focus on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit. Image matches with the image stores in database. Any facial feature changes in the database will invalidate the matching process. A reliable face-detection approach based on the genetic algorithm and the Eigen-face technique: Firstly, the possible human eye regions are detected by testing all the valley regions in the gray level image. Then the genetic algorithm is used to generate all the possible face regions which include the eyebrows, the iris, the nostril and the mouth corners. Each possible face candidate is normalized to reduce both the lighting effect, which is caused by uneven illumination; and the shirring effect, which is due to head movement. The fitness value of each candidate is measured based on its projection on the Eigen-faces. After a number of iterations, all the face candidates with a high fitness value are selected for further verification. At this stage, the face symmetry is measured and the existence of the different facial features is verified for each face candidate.

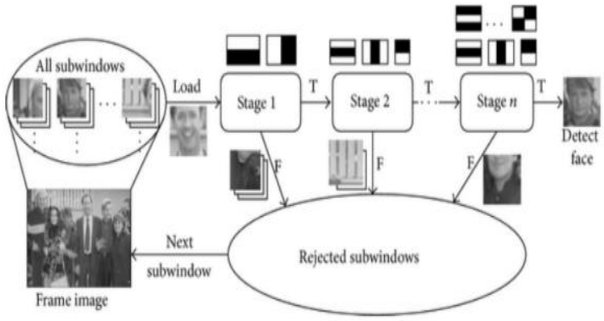
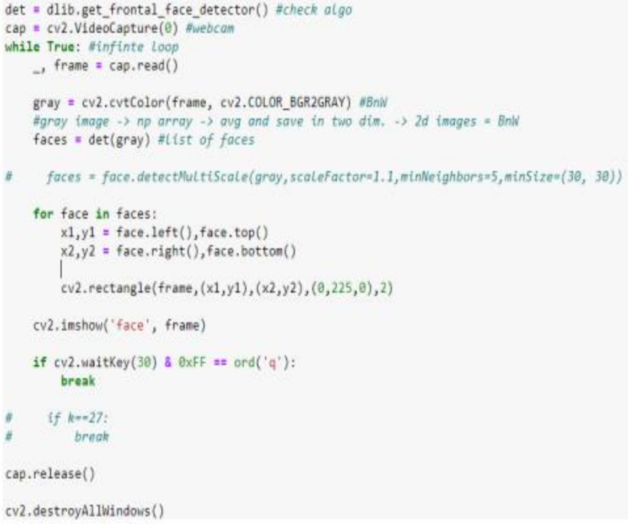


Fig. 2. Face Detection



```
det = dlib.get_frontal_face_detector() #check algo
cap = cv2.VideoCapture(0) #webcam
while True: #infinte loop
    _, frame = cap.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY) #BnW
    #gray image -> np array -> avg and save in two dim. -> 2d images = BnW
    faces = det(gray) #list of faces

#     faces = face.detectMultiScale(gray,scaleFactor=1.1,minNeighbors=5,minSize=(30, 30))

    for face in faces:
        x1,y1 = face.left(),face.top()
        x2,y2 = face.right(),face.bottom()
        |
        cv2.rectangle(frame,(x1,y1),(x2,y2),(0,225,0),2)

    cv2.imshow('face', frame)

    if cv2.waitKey(30) & 0xFF == ord('q'):
        break

#     if k==27:
#         break

cap.release()

cv2.destroyAllWindows()
```

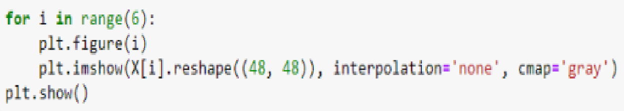
Fig. 3. Face Detection Code

Emotion Detection

People intuitively understand that emotions are more complex than this, says Barrett. When I say to people, _Sometimes you shout in anger, sometimes you cry in anger, sometimes you laugh, and sometimes you sit silently and plan the demise of your enemies, 'that convinces them, she says. I say, Listen, what 's the last time someone won an Academy Award for scowling when they 're angry? 'No one considers that great acting. These subtleties, though, are rarely acknowledged by companies selling emotion analysis tools. In marketing for Microsoft 's algorithms, for example, the company says advances in AI allow its software to recognize eight core emotional states ... based on universal facial expressions that reflect those feelings, which is the exact claim that this review disproves. This is not a new criticism, of course. Barrett and others have been warning for years that our model of emotion recognition is too simple. In response, companies selling these tools often say their analysis is based on more signals than just facial expression. The difficulty is knowing how these signals are balanced, if at all. One of the leading companies in the $20 billion emotion recognition market, Affective, says it's experimenting with collecting additional metrics. Last year, for example, it launched a tool that measures the emotions of drivers by combining face and speech analyses. Other researchers are looking into metrics like gait analysis and eye tracking. In a statement, Affective CEO and co-founder Rana el Kaliouby said this review was much in alignment‖ with the company's work. Like the authors of this paper, we do not like the naiveté of the industry, which is fixated on the 6 basic emotions and a prototypic one-to-one mapping of facial expressions to emotional states, said el Kaliouby. The relationship of expressions to emotion is very nuanced, complex and not prototypical.

Barrett is confident that we will be able to more accurately measure emotions in the future with more sophisticated metrics. I absolutely believe it's possible, she says. But that won't necessarily stop the current limited technology from proliferating.

With machine learning, in particular, we often see metrics being used to make decisions, not because they 're reliable, but simply because they can be measured. This is a technology that excels at finding connections, and this can lead to all sorts of spurious analyses: from scanning babysitters 'social media posts to detect their attitude to analyzing corporate transcripts of earnings calls to try to predict stock prices. Often, the very mention of AI gives an undeserved veneer of credibility. If emotion recognition becomes common, there's a danger that we will simply accept it and change our behavior to accommodate its failings. In the same way that people now act in the knowledge that what they do online will be interpreted by various algorithms (e.g., choosing to not like certain pictures on Instagram because it affects your ads), we might end up performing exaggerated facial expressions because we know how they will be interpreted by machines. That wouldn't be too different from signaling to other humans. Barrett says that perhaps the most important takeaway from the review is that we need to think about emotions in a more complex fashion. The expressions of emotions are varied, complex, and situational. She compares the needed change in thinking to Charles Darwin's work on the nature of species and how his research overturned a simplistic view of the animal kingdom. Darwin recognized that the biological category of a species does not have an essence, it's a category of highly variable individuals, says Barrett. Exactly the same thing is true of emotional categories.



```
for i in range(6):
    plt.figure(i)
    plt.imshow(X[i].reshape((48, 48)), interpolation='none', cmap='gray')
plt.show()
```

Fig. 3. Emotion Detection Code

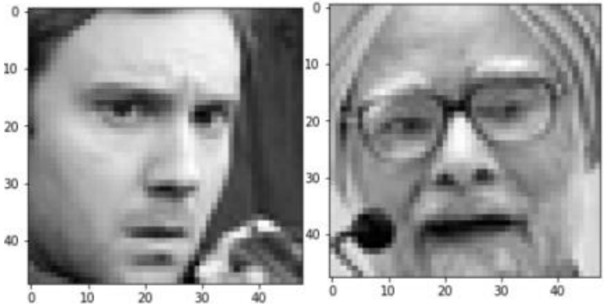
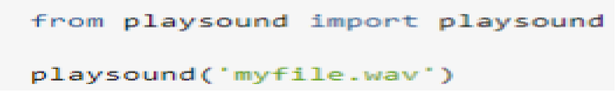


Fig 4. Emotion Detection

Song Playing

WAV files contain a sequence of bits representing the raw audio data, as well as headers with metadata in RIFF (Resource Interchange File Format) format. For CD recordings, the industry standard is to store each audio sample (an individual audio data point relating to air pressure) as a 16-bit value, at 44100 samples per second. To reduce file size, it may be sufficient to store some recordings (for example of human speech) at a lower sampling rate, such as 8000 samples per second, although this does mean that higher sound frequencies may not be as accurately represented. A few of the libraries discussed in this tutorial play and record bytes' objects, whereas others use NumPy arrays to store raw audio data. Both correspond to a sequence of data points that can be played back at a specified sample rate in order to play a sound. For a byte's object, each sample is stored as a set of two 8-bit values, whereas in a NumPy array, each element can contain a 16-bit value corresponding to a single sample. An important difference between these two data types is that byte's objects are immutable, whereas NumPy arrays are mutable, making the latter more suitable for generating sounds and for more complex signal processing. For more information on how to work with NumPy, have a look at our NumPy tutorials. Simple audio allows you to play NumPy and Python arrays and bytes objects using simpleaudio.play_buffer(). Make sure you have NumPy installed for the following example to work, as well as simple audio. (With pip installed, you can do this by running pip install NumPy from your console.)



```
from playsound import playsound

playsound('myfile.wav')
```

Fig. 5. Song Play

Methodologies Used:

First we have done face detection which can detect face on live web cam video. We have used Dlib lib and OpenCV for face detection.

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. This algorithm is hugely used in both industry and academic in a wide range of domains like robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib's open source licensing allows you to use it in any application, free of charge which makes it cost efficient.

OpenCV is a highly optimized, open-source library for computer vision, machine learning, and image processing. It supports wide variations of programming languages like Python, C++, Java, etc. OpenCV majorly process images and videos to identify objects, faces, and even the handwriting of a human. When it is integrated with various libraries, such as NumPy which is a highly used library for numerical operations, then the number of operations one can do in NumPy can be combined with OpenCV. Then we have detected the emotion of the image using the models based on CNN.

We used the basic CNN neural network method for emotion detection. CNN is a feed forward network to process and recognize image data with the grid vision.

Layers in CNN

1. Convolutional Layer -
   - Converts images into an array
   - First Layer of CNN
   - Store the pixelated values of image into an array
   - Used for extracting the features of the image and reducing its dimensionality
2. Activation Function: ReLu
   - Converts negative values into zero
   - Relu is a half rectifier
   - f(y) = 0 when y < 0
   - f(y) = y when y >= 0
3. Pooling Layer
   - Used to reduce dimensionality Helps to control overfitting
4. Fully Connected Layer
   - Combines all the features together to create a final model
   - Takes the output of the previous layers, "flattens" them
   - Takes the inputs from the feature analysis and applies weights to
   - Predict the correct label.
   - Gives the final probabilities for each label
5. Final Prediction (Not a layer)
   - Final prediction is done using functions like Softmax.
   - The category with highest probabilistic value is considered as our resultant output.

The flowchart of the working model is shown in Fig. 2. Using CNN, all of these layers were built for the better performance and better accuracy of the model. Emotions on the human face was detected using this seven-layer model. The most probabilistic emotion in the time period will be detected as the result of that particular part. After the emotion detection, in the end we have played the song on the basis of the output that is on the displayed emotion. For that output, songs were also categorized into different libraries and each library consist songs of different emotion. As we have used seven different emotions, seven different libraries will be created. songs will be recommended according to that emotion.

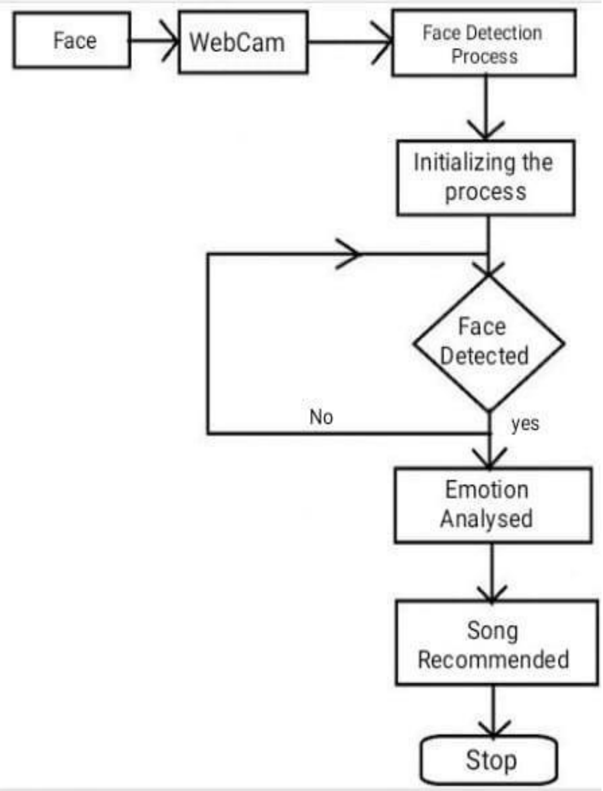
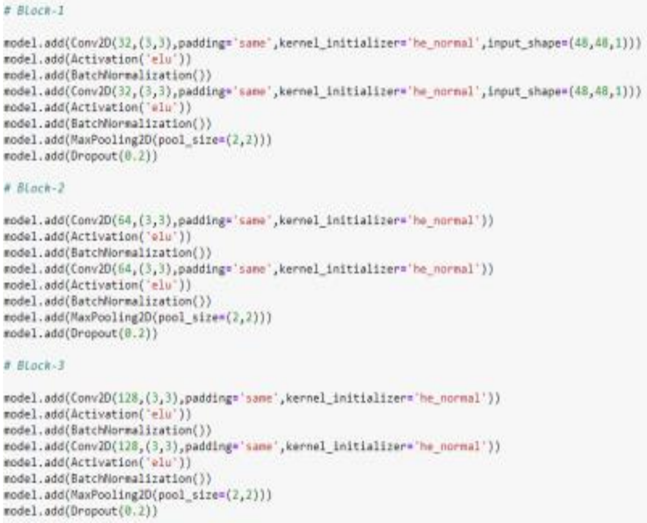


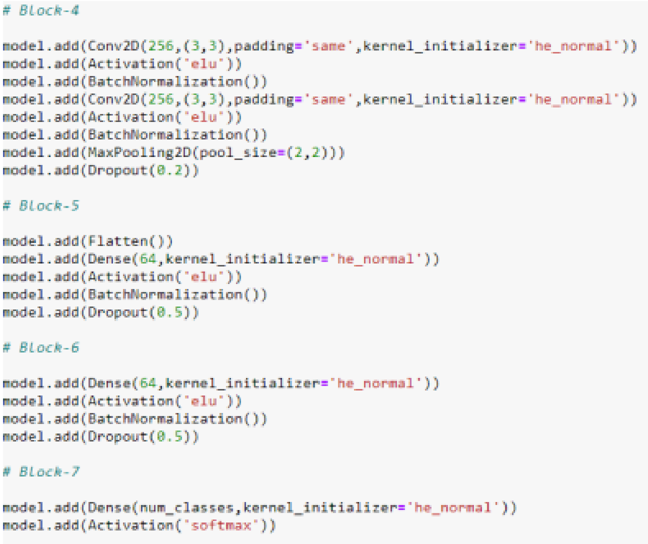
Fig. 6: Flowchart of the Working Model

Model & Outcome

For emotion detection we have used CNN and tried to achieve the best accuracy that we could. We have therefore used the 7 Layered CNN-ANN network.

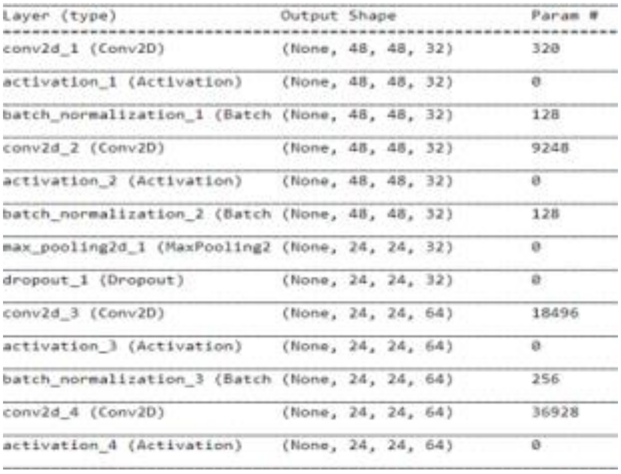


```
# Block-1

model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_shape=(48,48,1)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_shape=(48,48,1)))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-2

model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-3

model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
```

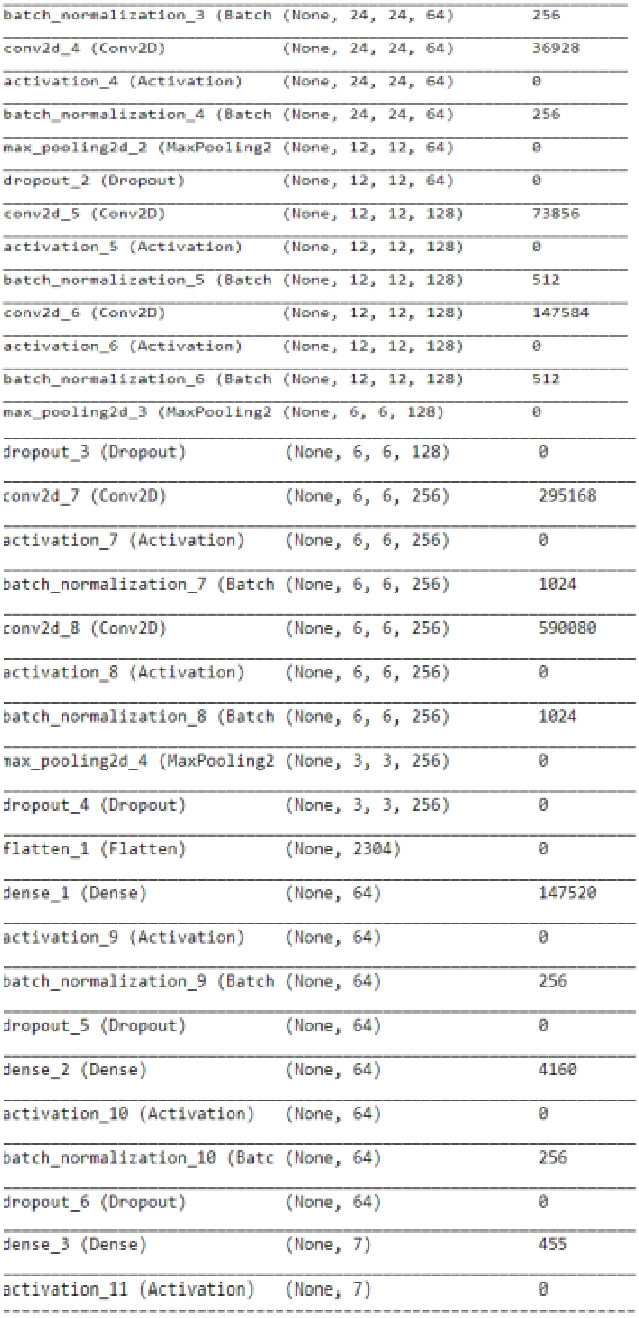


```
# Block-4

model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

# Block-5

model.add(Flatten())
model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

# Block-6

model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))

# Block-7

model.add(Dense(num_classes,kernel_initializer='he_normal'))
model.add(Activation('softmax'))
```

Fig. 7. Model and Outcome



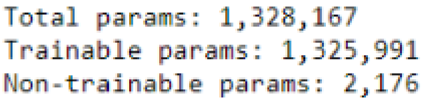
```
Layer (type)                 Output Shape              Param #
===============================================================
conv2d_1 (Conv2D)            (None, 48, 48, 32)        320
_______________________________________________________________
activation_1 (Activation)    (None, 48, 48, 32)        0
_______________________________________________________________
batch_normalization_1 (Batch (None, 48, 48, 32)        128
_______________________________________________________________
conv2d_2 (Conv2D)            (None, 48, 48, 32)        9248
_______________________________________________________________
activation_2 (Activation)    (None, 48, 48, 32)        0
_______________________________________________________________
batch_normalization_2 (Batch (None, 48, 48, 32)        128
_______________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 24, 24, 32)        0
_______________________________________________________________
dropout_1 (Dropout)          (None, 24, 24, 32)        0
_______________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 24, 64)        18496
_______________________________________________________________
activation_3 (Activation)    (None, 24, 24, 64)        0
_______________________________________________________________
batch_normalization_3 (Batch (None, 24, 24, 64)        256
_______________________________________________________________
conv2d_4 (Conv2D)            (None, 24, 24, 64)        36928
_______________________________________________________________
activation_4 (Activation)    (None, 24, 24, 64)        0
_______________________________________________________________
```



```
batch_normalization_3 (Batch (None, 24, 24, 64)        256
conv2d_4 (Conv2D)            (None, 24, 24, 64)        36928
activation_4 (Activation)    (None, 24, 24, 64)        0
batch_normalization_4 (Batch (None, 24, 24, 64)        256
max_pooling2d_2 (MaxPooling2 (None, 12, 12, 64)        0
dropout_2 (Dropout)          (None, 12, 12, 64)        0
conv2d_5 (Conv2D)            (None, 12, 12, 128)       73856
activation_5 (Activation)    (None, 12, 12, 128)       0
batch_normalization_5 (Batch (None, 12, 12, 128)       512
conv2d_6 (Conv2D)            (None, 12, 12, 128)       147584
activation_6 (Activation)    (None, 12, 12, 128)       0
batch_normalization_6 (Batch (None, 12, 12, 128)       512
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 128)         0
dropout_3 (Dropout)          (None, 6, 6, 128)         0
conv2d_7 (Conv2D)            (None, 6, 6, 256)         295168
activation_7 (Activation)    (None, 6, 6, 256)         0
batch_normalization_7 (Batch (None, 6, 6, 256)         1024
conv2d_8 (Conv2D)            (None, 6, 6, 256)         590080
activation_8 (Activation)    (None, 6, 6, 256)         0
batch_normalization_8 (Batch (None, 6, 6, 256)         1024
max_pooling2d_4 (MaxPooling2 (None, 3, 3, 256)         0
dropout_4 (Dropout)          (None, 3, 3, 256)         0
flatten_1 (Flatten)          (None, 2304)              0
dense_1 (Dense)              (None, 64)                147520
activation_9 (Activation)    (None, 64)                0
batch_normalization_9 (Batch (None, 64)                256
dropout_5 (Dropout)          (None, 64)                0
dense_2 (Dense)              (None, 64)                4160
activation_10 (Activation)   (None, 64)                0
batch_normalization_10 (Batc (None, 64)                256
dropout_6 (Dropout)          (None, 64)                0
dense_3 (Dense)              (None, 7)                 455
activation_11 (Activation)   (None, 7)                 0
```

Fig. 8. Model Summary

And therefore, it has over 1 Million parameters which are being trained under 25 epochs with batch size of 897.



```
Total params: 1,328,167
Trainable params: 1,325,991
Non-trainable params: 2,176
```

The model is trained with categorical cross entropy as the loss metric and with Adam regulizer with accuracy metric as its determining parameter. The best model so far had the accuracy of 73.4% and the proposed model gives the accuracy of 74.8% accuracy with the batch size of 128 and an accuracy of 78.63% with the batch size of 8.

5. CONCLUSION & FUTURE WORK

This paper proposes a simple system for recommending music based on face emotion recognition. It suggests music based on different facial expressions: happy, angry, surprised, neutral. This paper deals with the detection and extraction of features from human faces, the detection and prediction of emotions, and the playback of songs based on those emotions. The best model so far had the accuracy of 73.4% and the proposed model gives the accuracy of 74.8% accuracy with the batch size of 128 and an accuracy of 78.63% with the batch size of 8.

6. REFERENCES

[1] Ahmad, F., Najam, A., & Ahmed, Z. (2013). Image-based face detection and recognition:" state of the art". arXiv preprint arXiv:1302.6379.

[2] Hjelmås, E., & Low, B. K. (2001). Face detection: A survey. Computer vision and image understanding, 83(3), 236-274.

[3] Mehtab, S., & Sen, J. Face Detection Using OpenCV and Haar Cascades Classifiers. Mar. 2020. ECC: No Data (logprob:-57.959).

[4] Sharma, S., Shanmugasundaram, K., & Ramasamy, S. K. (2016, May). FAREC—CNN based efficient face recognition technique using Dlib. In 2016 international conference on advanced communication control and computing technologies (ICACCCT) (pp. 192-195). IEEE.

[5] Jain, C., Sawant, K., Rehman, M., & Kumar, R. (2018, November). Emotion detection and characterization using facial features. In 2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE) (pp. 1- 6). IEEE

[6] Fessahaye, F., Perez, L., Zhan, T., Zhang,R., Fossier, C., Markarian, R., ... & Oh, P. (2019, January). T-recsys: A novel music recommendation system using deep learning. In 2019 IEEE international conference on consumer electronics (ICCE) (pp. 1-6). IEEE.

[7] Tarnowski, P., Kołodziej, M., Majkowski, A., & Rak, R. J. (2017). Emotion recognition using facial expressions. Procedia Computer Science, 108, 1175-1184.

[8] Han, Byeong-jun, et al. "Music emotionclassification and context-based music recommendation."

[9] Zhan S., Tao Q.Q., Li X.H. Face detection using representation learning Neuro computing, 187 (C) (2016), pp. 19-26

[10] Y. Li, B. Sun, T. Wu, Y. Wang Face detection with end-to-end integration of a convnet and a 3d model European Conference on Computer Vision, Springer, Cham (2016), pp. 420-436.

[11] S. Ren, He K., R. Girshick, J. Sun Faster R-CNN: towards real-time object detection with region proposal networks Proceedings of the Advances in Neural Information Processing Systems (2015), pp. 91-99.

[12] H. Jiang, E. Learned-Miller Face detection with the faster R-CNN Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on IEEE (2017), pp. 650-657

[13] Qin H., Yan J., Li X., Hu X. Joint training of cascaded CNN for face detection Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016), pp. 3456-3465

[14] S. Wan, Z. Chen, T. Zhang, B. Zhang, K.k. Wong, Bootstrapping face detection with hard negative examples, arXiv:1608.02236.

[15] Sharma, S., Shanmugasundaram, K., & Ramasamy, S. K. (2016, May). FAREC—CNN based efficient face recognition technique using Dlib. In 2016 international conference on advanced communication control and computing technologies (ICACCCT) (pp. 192-195). IEEE.