**IMPLEMENTATION**

Here we propose an Emotion based music player named iEmoDec. It is a music player which plays songs according to the emotion of the user. It aims to provide user preferred music with emotion awareness. Emotion player is based on the idea of automating much of the interaction between the music player and its user. The emotions are recognized using a machine learning method of supervised algorithm. In machine learning, unsupervised learning models are associated with learning algorithms that analyze data used for clustering and regression analysis. It finds an optimal boundary between the possible outputs. So, this is done with the help of CNN. Our model is trained using CNN algorithm. So, CNN classifies the emotion. For the emotion detection, fer2013 dataset is used. This dataset consists of 35,887 gray scale, 48x48 sized face images with 7 emotions in it i.e., happy, sad, surprise, angry, neutral, fear, disgust. The webcam captures the image of the user. It then extracts the facial features of the user from the captured image. Furthermore, with use of the Haar Feature-Based Cascaded Classiﬁer inside the OpenCV framework , all data is preprocessed. For every image, only the square part containing the face is taken, rescaled, and converted to an array with 48x48 grey-scale values. Evaluation allows the testing of the model against data that has never been seen and used for training and is meant to be representative of how the model might perform when in the real world. By adding a Chatbot, we also made it interactive using Google’s Text-to-Speech (gTTS) and Speech-to- Text modules. According to the emotion, the music will be played from the predefined directories.

In machine learning, Convolutional Neural Networks (CNN or ConvNet) are basically complex feed forward neural networks. CNNs are basically used for its high accuracy in image classification and recognition. The CNN follows a hierarchical model that works on building a network, acting like a funnel, and finally gives out a fully-connected layer where all the neurons are connected to each other and the output is processed. The corresponding emotion for the expression captured needs to be detected. So this is done with the help of CNN. Our model is trained using CNN algorithm. So CNN classifies the emotion. For the emotion detection, fer2013 dataset is used. This dataset consists of 35,887 gray scale, 48x48 sized face images with 7 emotions in it i.e., happy, sad, surprise, angry, neutral, fear, disgust.

The networks are programmed with use of the TFLearn library on top of TensorFlow, running on Python. This environment lowers the complexity of the code, since only the neuron layers have to be created, instead of every neuron. The program also provides real-time feedback on training progress and accuracy, and makes it easy to save and reuse the model after training.

The network consists of three convolutional layers and two fully connected layers, combined with maxpooling layers for reducing the image size and a dropout layer to reduce the chance of over ﬁtting. The hyper parameters are chosen such that the number of calculations in each convolutional layer remains roughly the same. This ensures that information is preserved throughout the network. Training is performed using diﬀerent numbers of convolutional ﬁlters to evaluate their eﬀect on the performance. The original network starts with an input layer of 48 by 48, matching the size of the input data. This layer is followed by one convolutional layer, a local contrast normalization layer, and a maxpooling layer respectively. The network is ﬁnished with two more convolutional layers and one fully connected layer, connected to a softmax output layer. Dropout was applied to the fully connected layer and all layer contain ReLu units. *ReLU stands for rectified linear unit, and is a type of activation function. In a neural network, the activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input.*

*The rectified linear activation function is a piecewise linear function that will output the input directly if is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance. The purpose of applying the rectifier function is to increase the non-linearity in our images.*

A second maxpooling layer is applied to reduce the number of parameters. This lowers the computational intensity of the network, while the reduction in performance is claimed to be only 1-2%.

Finally, ChatBot displays the emotion of the user detected through voice and through text. A song is played in accordance with the emotion.

Accuracy and Model Loss

*Loss value implies how poorly or well a model behaves after each iteration of optimization. An accuracy metric is used to measure the algorithm's performance in an interpretable way. The accuracy of a model is usually determined after the model parameters and is calculated in the form of a percentage.* All networks are trained for 60 epochs. The ﬁnal accuracy on the validation data is around 63%. Already after 10 epochs, the accuracy raised above 60%, indicating quick learning capabilities. Furthermore it is noteworthy that adjusting the ﬁlter dimension did not have a big inﬂuence on the accuracy, though it has on the processing time. This means that fast models can be made with very reasonable performance. The accuracy seems to increase in the last epochs. We therefore will train the network for 100 epochs in the ﬁnal run, to make sure the accuracy converges to the optimum. In an attempt to improve the ﬁnal model even more, the network will be trained on a larger set.

A loss is a number indicating how bad the model’s prediction was on a single example. If the model’s prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have low loss, on average, across all examples. The loss is calculated on training and validation and its interpretation is based on how well the model is doing in these two sets. It is the sum of errors made for each example I training or validation sets. Loss value implies how poorly or well a model behaves after each iteration of optimization.