DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY
KUDLU GATE
BANGALORE - 560068

MINOR PROJECT REPORT

ON

"EEG based Emotion Recognition Using Recurrent Neural Networks"

SUBMITTED DURING THE 5th SEMESTER -2020

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

Submitted by
AMOGH G PADUKONE ENG18CS0032
ANIRUDH BM ENG18CS0038
SIVA PRAKASH ANUPAM ENG18CS0277

Under the supervision of
Dr. Reeja SR

DAYANANDA SAGAR UNIVERSITY

School of Engineering, Kudlu Gate, Bangalore-560068

CERTIFICATE

This is to certify that Mr. Amogh G Padukone bearing USN ENG18CS0032 has satisfactorily completed his Minor Project as prescribed by the University for the Fifth semester B.Tech. Programme in Computer Science & Engineering during the year 2020 at the School of Engineering, Dayananda Sagar University, Bangalore.

Date: 25/11/2020

Signature of the faculty in-charge

Max Marks	Marks Obtained

Signature of Chairman
Department of Computer Science & Engineering

DAYANANDA SAGAR UNIVERSITY

School of Engineering, Kudlu Gate, Bangalore-560068

CERTIFICATE

This is to certify that Mr. Anirudh BM bearing USN ENG18CS0038 has satisfactorily completed his Minor Project as prescribed by the University for the Fifth semester B.Tech. Programme in Computer Science & Engineering during the year 2020 at the School of Engineering, Dayananda Sagar University, Bangalore.

Date: 25/11/2020

Signature of the faculty in-charge

Max Marks	Marks Obtained

Signature of Chairman
Department of Computer Science & Engineering

DAYANANDA SAGAR UNIVERSITY

School of Engineering, Kudlu Gate, Bangalore-560068

CERTIFICATE

This is to certify that Mr. Siva Prakash Anupam bearing USN ENG18CS0277 has satisfactorily completed his Minor Project as prescribed by the University for the Fifth semester B.Tech. Programme in Computer Science & Engineering during the year 2020 at the School of Engineering, Dayananda Sagar University, Bangalore.

Date: 25/11/2020

Signature of the faculty in-charge

Max Marks	Marks Obtained

Signature of Chairman
Department of Computer Science & Engineering

DECLARATION

We hereby declare that the work presented in this minor project entitled "**EEG based Emotion Recognition Using Recurrent Neural Networks**", has been carried out by us and it has not been submitted for the award of any degree, diploma or the minor project of any other college or university.

Amogh G Padukone ENG18CS0032
Anirudh BM ENG18CS0038
Siva Prakash Anupam ENG18CS0277

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of a task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We are especially thankful to our **Chairman, Dr. Sanjay Chitnis**, for providing us with necessary departmental facilities, moral support and encouragement.

We are very much thankful to **Dr. Reeja SR**, for providing help and suggestions in completion of this minor project successfully.

We have received a great deal of guidance and co-operation from our friends and we wish to thank all that have directly or indirectly helped us in the successful completion of this project work.

Amogh G Padukone ENG18CS0032

Anirudh BM ENG18CS0038

Siva Prakash Anupam ENG18CS0277

Table of contents

Table of figures

ABSTRACT

Recognition of human emotions plays a vital role in understanding of the human brain when exposed to different situations. In today’s world, we have many methods for recognizing human emotions and one such method would be to monitor the individual’s facial expressions, but the problem with such methods is that they are not completely reliable and not applicable to people who are unable to show their facial expressions because of various reasons such as paralysis. So, to recognize human emotions, a better approach is to use EEG signals which communicate directly to the individual’s electric signals in the brain using electrodes attached to the brain.

Many machine learning and deep learning models exist which recognize the human emotions using the EEG signals, but the main problem lies in the reduction of unwanted noise acquired while EEG signals are recorded and provide accurate results. Deep learning and neural networks provide a promising approach than the rest, since sophisticated mathematical models can be applied to explain how the data is being analyzed in the network.

CHAPTER 1
INTRODUCTION

In this project we develop a deep learning model for recognising human emotions through EEG signals. Recognising human emotions play major role in Brain Computer interface (BCI) or human computer interface (HCI). BCI is used for practical and research purposes and has proved itself to be unique technique that enables the brain's neural activity to communicate with the computer environment. Deep learning has been performing well compared to other conventional methods in many applications. Recurrent Neural Network with Long short-term memory (LSTM) layers prove promising in BCIs as EEG waves are time dependent. In this project we use combination of LSTM layers with different activation functions for dimensionality reduction and classification of emotions.

The electroencephalogram (EEG) is a recording of the electrical activity of the brain from the scalp. The recorded waveforms reflect the cortical electrical activity. The signal intensity of EEG signal is quite small and is measured in Microvolts.

The main frequencies of the human EEG waves are:

FREQUENCY	**BANDWIDTH**
THETA	3-7 Hz
ALPHA	8-13 Hz
BETA	14-29 Hz
GAMMA	30-47 Hz

Table 1.1 Frequency Bandwidth values

The brain contains unique information in many regions at any given time. An EEG signal recorded with electrodes placed on the scalp which consists of many waves with different characteristics. Arrays of electrodes are distributed over the entire scalp and the signals are recorded. The main task would be to interpret the signal, because a large amount of data will be recorded with even a single EEG electrode pair.

This research uses a dataset obtained from an age group ranging from 19 years to 37 years. The product of this research will help in better understanding of human emotions and how to extract them through EEG signals thereby helping other research scholars.

The end results of this research will help other researchers to gain a better understanding of emotions through EEG signals and the application of deep neural network in this field. This research will help in better understanding of the EEG signals, human emotions and how deep learning can provide a better approach than the existing machine learning models.

1.1 Problem Statement

EMOTION is a psycho-physiological process triggered by conscious and/or unconscious perception of an object or situation and is often associated with mood, temperament, personality and disposition, and motivation. Brain Computer Interface systems and Human Computer Interface systems are a group of technologies which aim at brain computer interaction. BCI has proved itself as a unique technique that enables the brain's neural activity to communicate with the computer environment to provide some information without direct physical movements. Most of the contemporary Brain-computer interaction (BCI) systems are deficient in interpreting emotional information and suffer from the lack of emotional intelligence. Brain-computer interface (BCI) systems having the ability to classify emotions with greater accuracy are highly desirable. To this end, several techniques have been proposed aiming at classifying emotions through several cues such as facial expressions, vocabulary, and EEG signals. Our research is aimed at achieving higher accuracy through deep neural networks.

CHAPTER 2
LITERATURE REVIEW

- [1]"**DEAP: A Database for Emotion Analysis using Physiological Signals**", S. Koelstra, C. Muehl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, I. Patras, IEEE Transactions on Affective Computing, Special Issue on Naturalistic Affect Resources for System Building and Evaluation, in press. In this paper, description about the DEAP dataset is given. Stimuli selection procedure, experimental setup (description about the 10-20 system), Self-assessment mannequins-based valence arousal and dominance rating, co-relation between the EEG frequencies and the participants ratings
- **[2] EEG-Based Emotion Recognition Using Deep Learning Network with Principal Component Based Covariate Shift Adaptation** Suwicha Jirayucharoensak, Setha Pan-Ngum and Pasin Israsena Suwicha et al. Used several deep learning techniques to find out correlation between input signals. The DNN used consisted of Stacked Autoencoder (SAE) using hierarchal feature learning approach. To avoid overfitting of the data, Principal Component analysis was used to extract the most import features. PCA based covariate shift adaption boosted the accuracy. Accuracy: 49.52% and 46.03% (valence and arousal)
- **[3] Chrono Net: A Deep Recurrent Neural Network for Abnormal EEG Identification** Subhrajit Roy, Isabell Kiral-Kornek, and Stefan Harrer. Chrono Net is an Inception Convolutional Densely Connected Gated Recurrent Neural Network. Several layers of CNN, RNN and GRU were used for dimensionality reduction and were connected in a feed forward manner. This research has a 86.7% of testing accuracy in predicting EEG abnormalities.
- S. Koelstra et al. [4] presented methods for single trial classification using both EEG and peripheral physiological signals. Power spectrum density (PSD) of EEG signals was used as the features. A Support vector machine (SVM) classifier was used to classify two levels of valence states and two levels of arousal states. For EEG analysis results, average and maximum classification rates of 55.7%and 67.0% were obtained for arousal and 58.8% and 76.0% for valence.

- Soleymani et al. [5] provided a multimodal dataset, called “MAHNOB-HCI,” for an analysis of human affective states. The EEG and peripheral physiological signals were employed to classify emotion states. The system used PSD of EEG signals from 32 channels as input features. A SVM classifier was implemented to classify three levels of valence states and three levels of arousal states. For EEG-based classification, the accuracy rates for valence and arousal are57.0% and 52.4%, respectively
- Y. Huang et al. [6] developed an asymmetry spatial pattern (ASP) technique to extract features for EEG-based emotion recognition algorithm. The system employed K-Nearest Neighbour (K-NN), naive Bayes (NB), and support vector machine (SVM) for emotion classification. The average accuracy rates for valence and arousal are 66.05% and 82.46%, respectively.
- several studies used PSD of EEG data as the input features and performed emotion classification by using SVM. [7,10]
- Soboleva [11] used autoencoder for feature compression and a series of CNNs and RNNs. The autoencoder helped in data dimension reduction feature extraction and reduced input data representation to the minimum level.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1Functional requirements

Our research will use the dataset available from DEAP which has a recording of 32 subjects exposed to multimedia stimuli, the EEG signals are recorded over 40 channels at a frequency of 512Hz. After pre-processing, it is down sampled to 128Hz.

The two main requirements from the research will be

- a deep learning model for recognizing human emotions through EEG signals
- to use minimum number of EEG channels to restrict the data signals that are connected to emotions in the brain

3.2External interface requirements

The initial stage of the research is collecting the dataset, in our case it would be to collect EEG signals from individuals and normalizing them. Since we are using the DEAP (Database for Emotion Analysis Using Physiological Signals) dataset acquired through the net, the external interface requirements are null.

3.3 System features

- An efficient feature extraction method
- A deep learning approach to the problem

3.4Non-functional requirements

There are several variables that are to be considered while doing this research, the authenticity of the dataset, the SNR of the EEG signals and the type of neural networks being applied.

CHAPTER 4
DESIGN METHODS

4.1 Algorithm

1.FFT transformation of signals to get the spectral power in specific bins

- Fast Fourier transform computes the discrete Fourier transform of a sequence. With respect to EEG signals, we get the spectral analysis in bandwidths of theta, alpha, gamma, beta.
  (spectral analysis: it is a frequency domain analysis; it reflects the frequency contents of the signals or distribution of signal power over frequency)

2. Pre-processing the data

- In this step we read the data from all the subjects into one NumPy array (i.e., "data" and "label"). We understand the data using. head (), .tail(), .shape(), .isnull().sum()
- We clean the data by deleting rows having null values and then we normalize it
- We apply standard scaler from sklearn to fit the data into the data frame
- We then split the data into train data, test data and train label, test label

3. Applying deep neural network

- We implement the deep learning network using sequential module from keras
- The recurrent neural network will consist of LSTM layers with relu activations and a dense layer with sigmoid activation (stochastic approach)
- Batch Normalization () and dropout () are applied after every layer to prevent the overfitting of data
- Mean squared error and optimizers from keras will be used to check the accuracy of the model and for the model to learn.

4.2 Architecture diagram

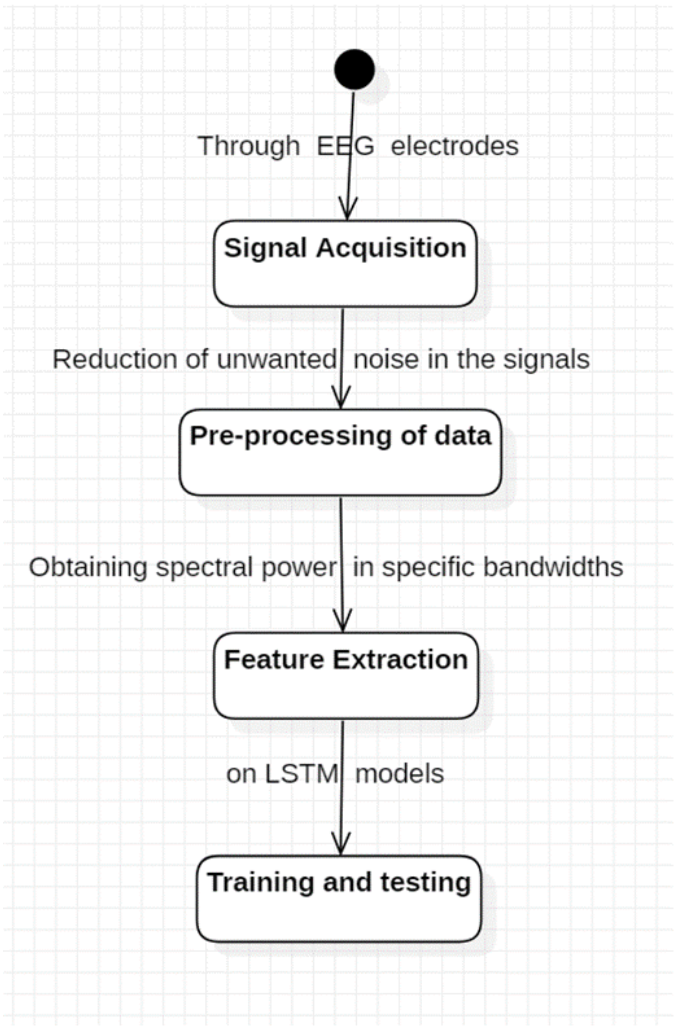


Fig 4.1 Activity diagram of the project

CHAPTER 5

PROJECT BREAKDOWN

Any BCI based model follows the below mentioned steps:

- **Signal Acquisition**: electrodes are placed on the subject's scalp and EEG signals are recorded while the subject is exposed to stimuli.
- **Pre-processing of data:** EEG signals have a high SNR (Signal to noise ratio). These noises can be generated from the artifacts or environmental noise. These noises must be filtered to get higher accuracy. (the above two steps have been done by DEAP dataset)
- **Feature Extraction:** in this step, we are acquiring the spectral power of theta, alpha, beta, and gamma bandwidths using fast Fourier transform.
- **Training and testing:** the dataset that is obtained after feature extraction will be divided into training and testing data.

CHAPTER 6
IMPLEMENTATION

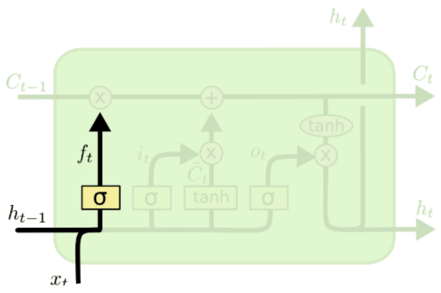
- **Feature extraction using FFT:** We extract the necessary features from raw EEG signals, using FFT (Fast Fourier Transformation). FFT converts time domain signals to frequency domain signals. FFT calculates the discrete wave transform of the signal. We specify the bandwidths and the 14 channels (AF3, F3, F7, FC1, C3, P3, OZ, FP2, F4, CZ, C4, CP6, CP2, P4) to fit the emotiv epoch+ model. We traverse over the 40 trials and separate data and labels, then we apply the FFT transformation to acquire the component frequencies in 14 channels and 5 bandwidths.
- **Scaling the data:** We use min max scaler from 0 to 1, since LSTM gates use sigmoid activation functions which are sensitive to inputs between o and 1. Min-max scaler, performs a linear transformation on the original data. This technique gets all the scaled data in the range [0,1]. The formula to achieve this is the following:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Fig 6.1 Min Max scaler formula

- **Vanishing gradient problem:** The vanishing gradient problem is encountered when training artificial neural networks with gradient-based learning methods and backpropagation. It is a difficulty found in training certain Artificial Neural Networks with gradient based methods (e.g. Back Propagation). This problem makes it hard to learn and tune the parameters of the earlier layers in the network. This problem becomes worse as the number of layers in the architecture increases.
- **Designing the model:** LSTM layers working mechanism is explained below:
  1) **To decide what information must be given to the cell state:** This decision is made by a sigmoid layer called the “forget gate layer.” It looks at ht−1 and xt, and outputs a number between 0 and 1 for each number in the cell state Ct−1.

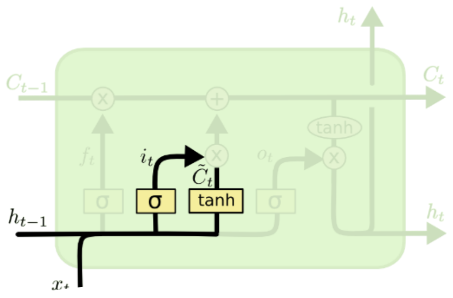
A 1 represents “completely keep this” while a 0 represents “completely get rid of this.”



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Fig 6.2 First step of LSTM layers

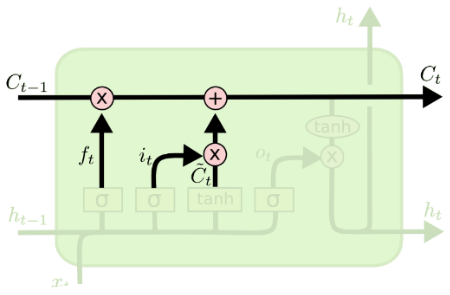
2) **To decide what new information, we are going to store in the cell state:** This has two parts. First, a sigmoid layer called the “input gate layer” decides which values we will update. Next, a tanh layer creates a vector of new candidate values, C~t, that could be added to the state. In the next step, we will combine these two to create an update to the state.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Fig 6.3 Second step of LSTM layers

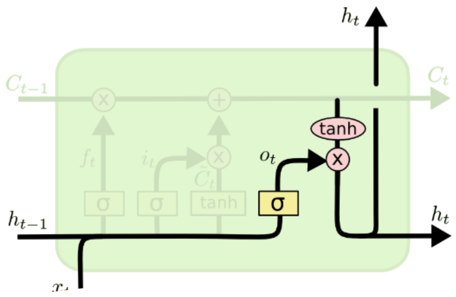
3) **To update the old cell state, Ct−1, into the new cell state Ct:** We multiply the old state by ft, forgetting the things we decided to forget earlier. Then we add it∗C~t. This is the new candidate values, scaled by how much we decided to update each state value.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Fig 6.4 Third step of LSTM layers

4) **To decide what we are going to output:** This output will be based on our cell state but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we are going to output. Then, we put the cell state through tanh (to push the values to be between −1 and 1) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Fig 6.5 Fourth step of LSTM layers

- **Design of the model:** the input layers in LSTM take 3 dimensions

  1)Samples. One sequence is one sample. A batch is comprised of one or more samples.

  2)Time Steps. One-time step is one point of observation in the sample.

  3) Features. One feature is one observation at a time step.

Based on the number of subjects selected, the input dimensions are given as follows (X, 1, 70). Our model is as follows:

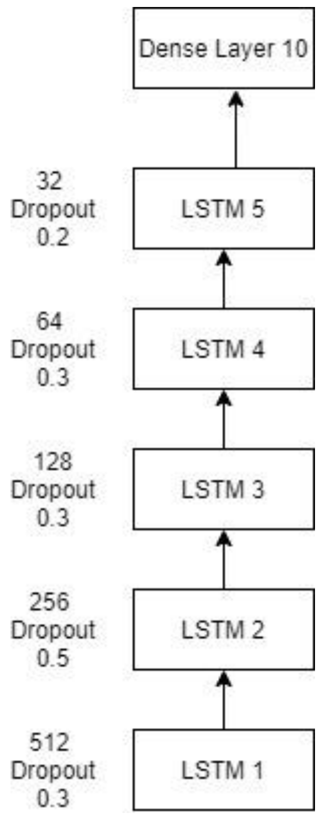


Fig 6.6 LSTM layers

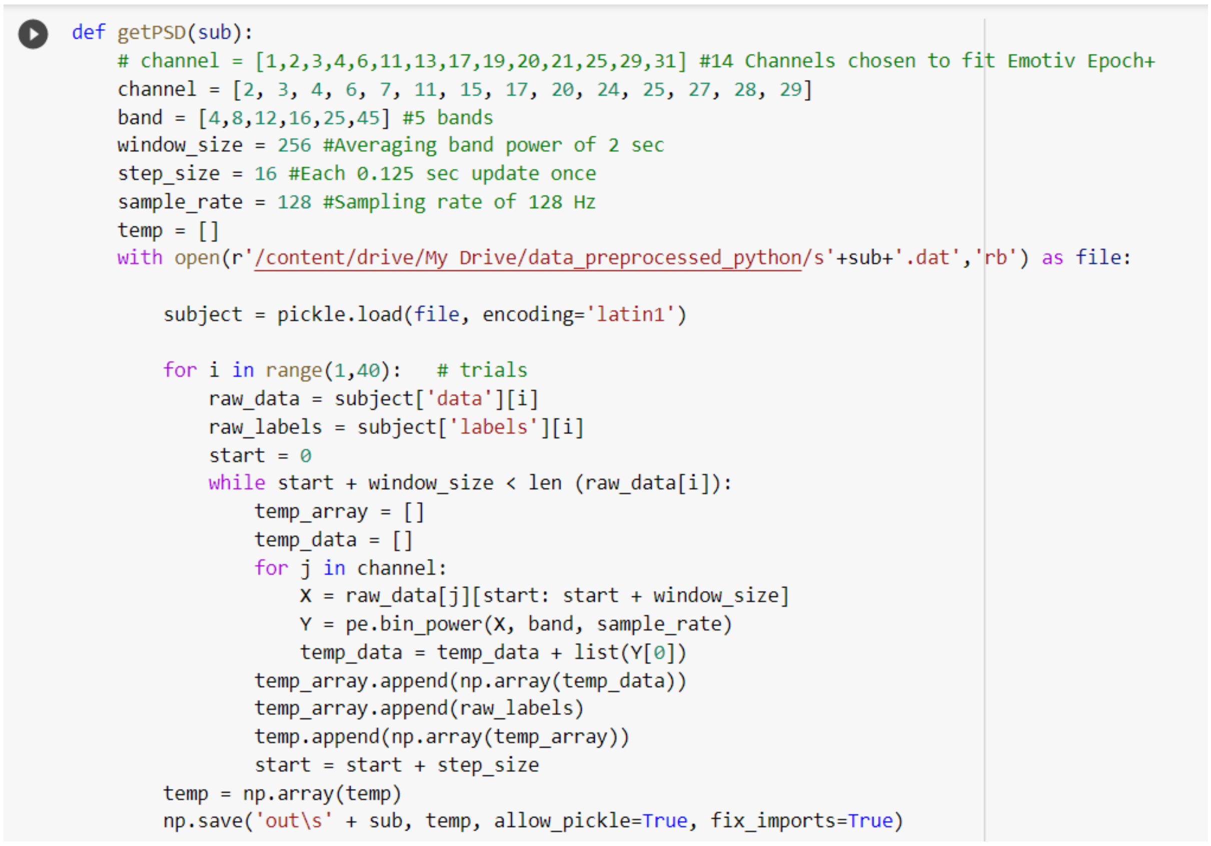
- **RMSprop**: RMSprop is a gradient based optimization technique used in training neural networks. It was proposed by the father of back-propagation, Geoffrey Hinton. Gradients of very complex functions like neural networks tend to either vanish or explode as the data propagates through the function. Rmsprop was developed as a stochastic technique for mini-batch learning. RMSprop deals with this issue by using a moving average of squared gradients to normalize the gradient. This normalization balances the step size (momentum), decreasing the step for large gradients to avoid exploding, and increasing the step for small gradients to avoid vanishing.

$$E[g^2]_t = \beta E[g^2]_{t-1} + (1-\beta)\left(\frac{\delta C}{\delta w}\right)^2$$

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{E[g^2]_t}} \frac{\delta C}{\delta w}$$

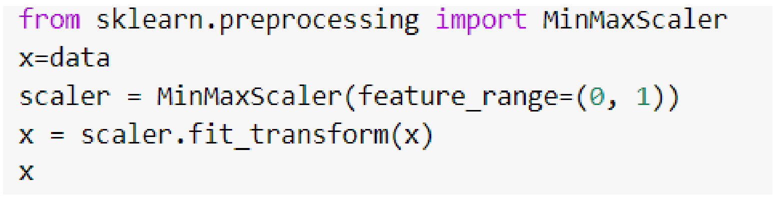
Fig 6.7 RMSprop formula

E[g] — moving average of squared gradients. dC/dw — gradient of the cost function with respect to the weight. n — learning rate. Beta — moving average parameter (good default value — 0.9)



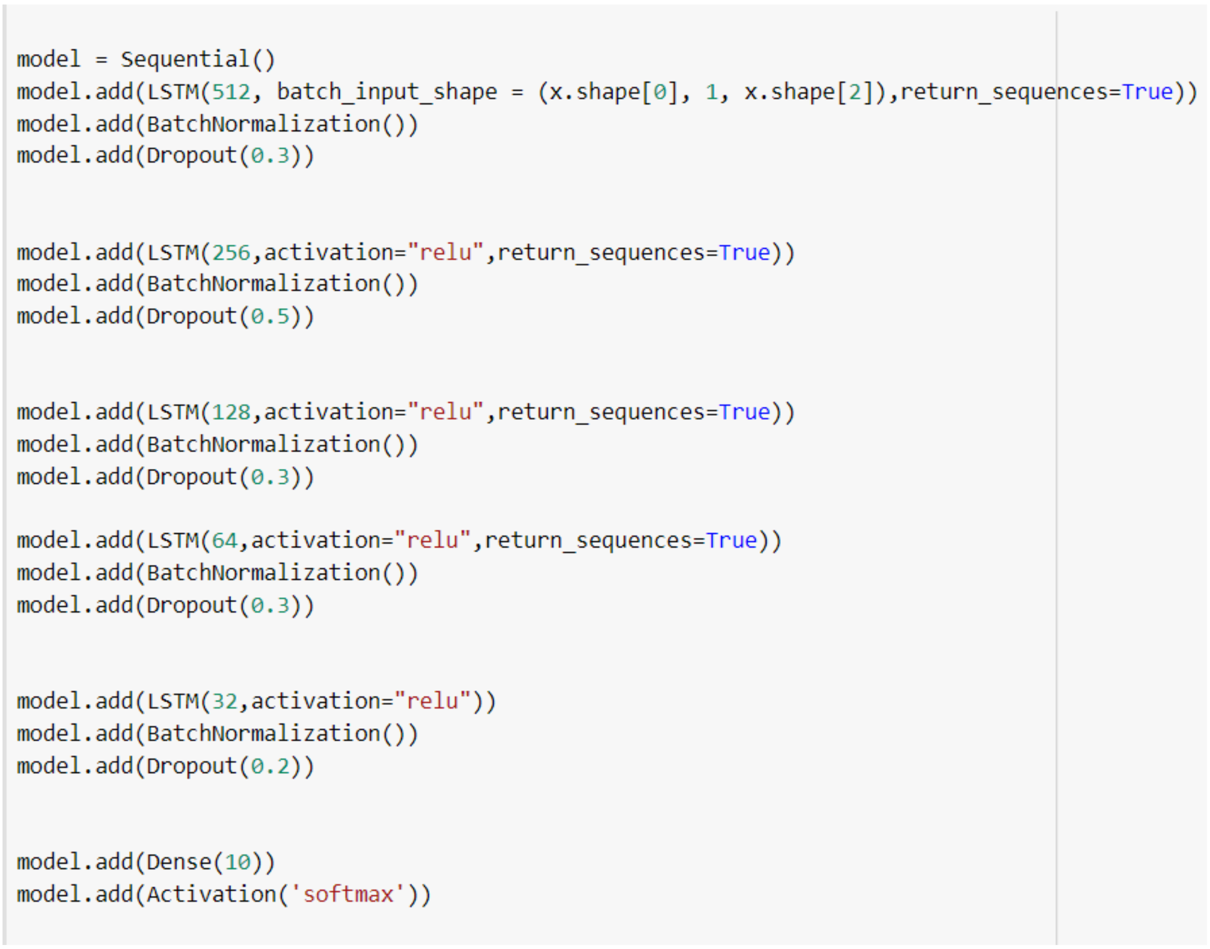
```
def getPSD(sub):
    # channel = [1,2,3,4,6,11,13,17,19,20,21,25,29,31] #14 Channels chosen to fit Emotiv Epoch+
    channel = [2, 3, 4, 6, 7, 11, 15, 17, 20, 24, 25, 27, 28, 29]
    band = [4,8,12,16,25,45] #5 bands
    window_size = 256 #Averaging band power of 2 sec
    step_size = 16 #Each 0.125 sec update once
    sample_rate = 128 #Sampling rate of 128 Hz
    temp = []
    with open(r'/content/drive/My Drive/data_preprocessed_python/s'+sub+'.dat','rb') as file:

        subject = pickle.load(file, encoding='latin1')

        for i in range(1,40):   # trials
            raw_data = subject['data'][i]
            raw_labels = subject['labels'][i]
            start = 0
            while start + window_size < len (raw_data[i]):
                temp_array = []
                temp_data = []
                for j in channel:
                    X = raw_data[j][start: start + window_size]
                    Y = pe.bin_power(X, band, sample_rate)
                    temp_data = temp_data + list(Y[0])
                temp_array.append(np.array(temp_data))
                temp_array.append(raw_labels)
                temp.append(np.array(temp_array))
                start = start + step_size
        temp = np.array(temp)
        np.save('out\s' + sub, temp, allow_pickle=True, fix_imports=True)
```

Fig 6.8 Feature Extraction function



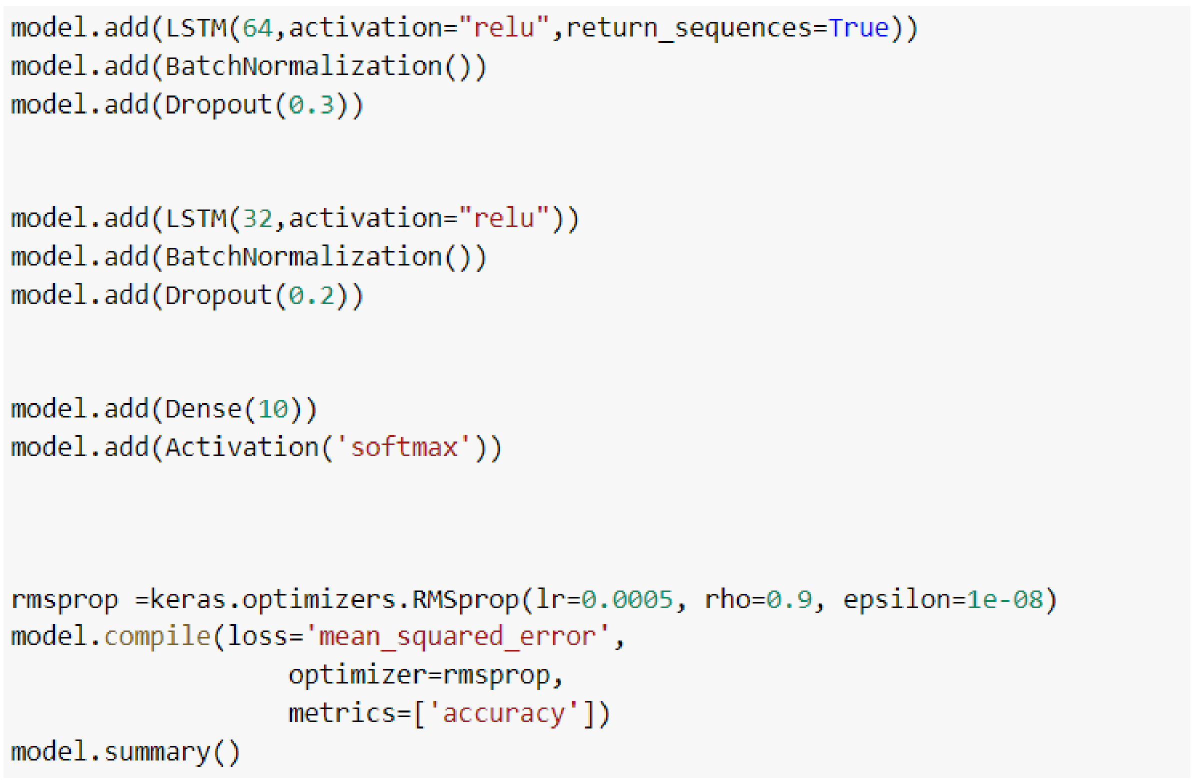
```
from sklearn.preprocessing import MinMaxScaler
x=data
scaler = MinMaxScaler(feature_range=(0, 1))
x = scaler.fit_transform(x)
x
```

Fig 6.9 Min Max Scaler



```
model = Sequential()
model.add(LSTM(512, batch_input_shape = (x.shape[0], 1, x.shape[2]),return_sequences=True))
model.add(BatchNormalization())
model.add(Dropout(0.3))


model.add(LSTM(256,activation="relu",return_sequences=True))
model.add(BatchNormalization())
model.add(Dropout(0.5))


model.add(LSTM(128,activation="relu",return_sequences=True))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(LSTM(64,activation="relu",return_sequences=True))
model.add(BatchNormalization())
model.add(Dropout(0.3))


model.add(LSTM(32,activation="relu"))
model.add(BatchNormalization())
model.add(Dropout(0.2))


model.add(Dense(10))
model.add(Activation('softmax'))
```

Fig 6.10 Model

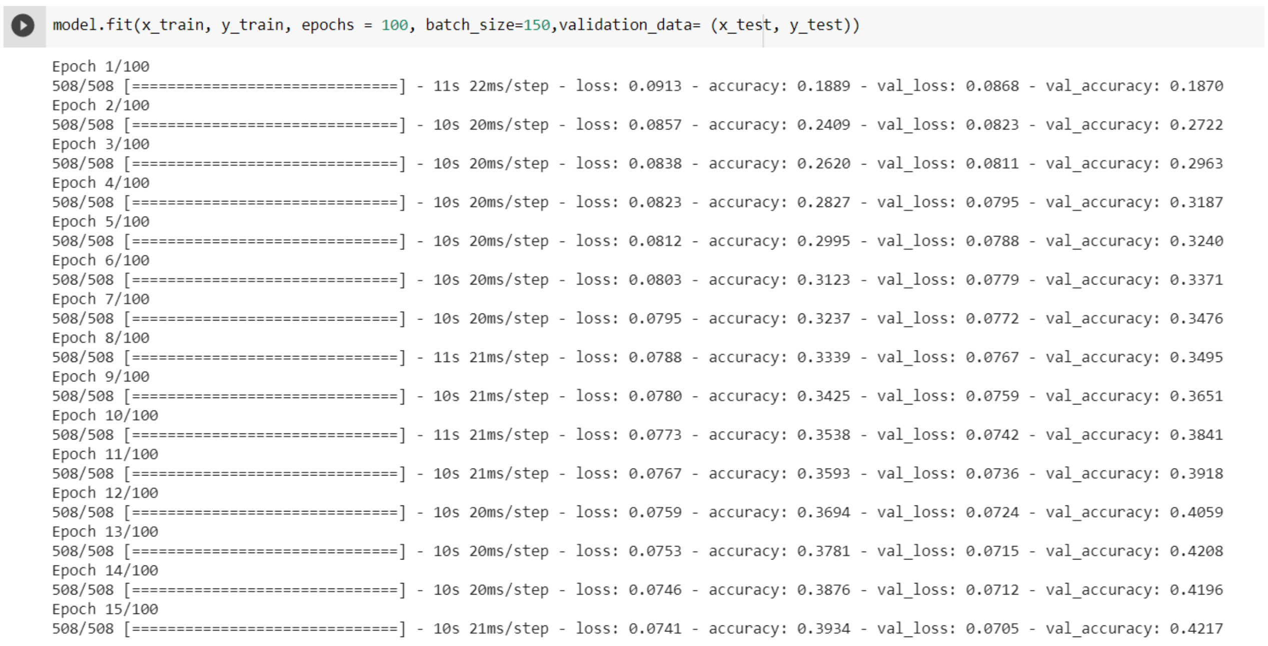


```
model.add(LSTM(64,activation="relu",return_sequences=True))
model.add(BatchNormalization())
model.add(Dropout(0.3))


model.add(LSTM(32,activation="relu"))
model.add(BatchNormalization())
model.add(Dropout(0.2))


model.add(Dense(10))
model.add(Activation('softmax'))



rmsprop =keras.optimizers.RMSprop(lr=0.0005, rho=0.9, epsilon=1e-08)
model.compile(loss='mean_squared_error',
                  optimizer=rmsprop,
                  metrics=['accuracy'])
model.summary()
```

Fig 6.11 Model continued

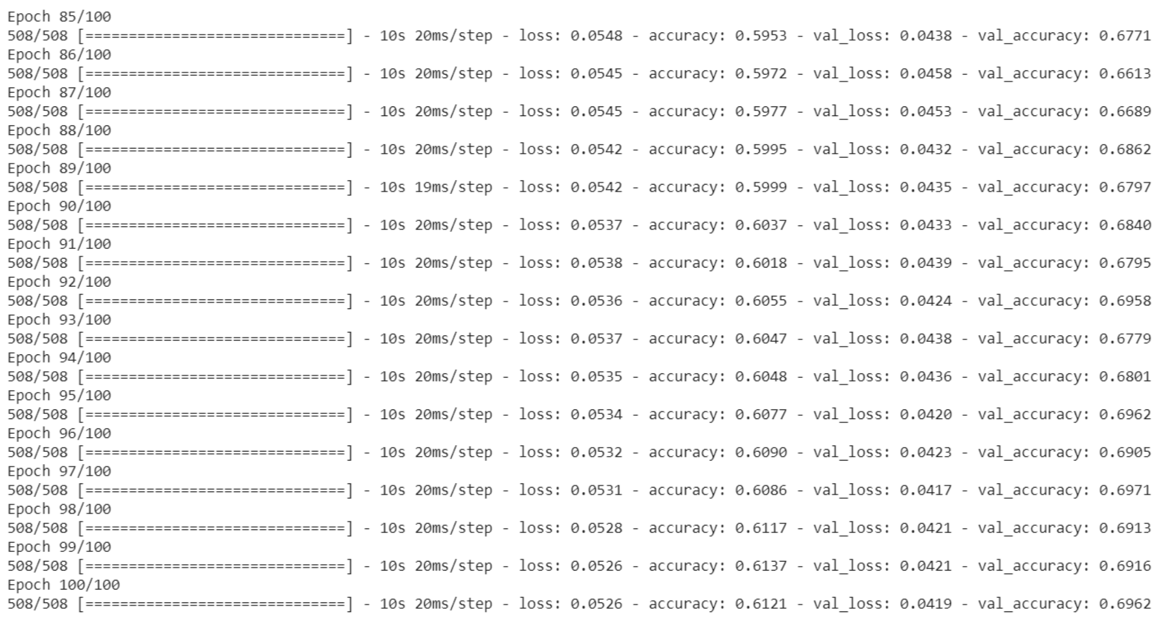
CHAPTER 7

TRAINING AND TESTING



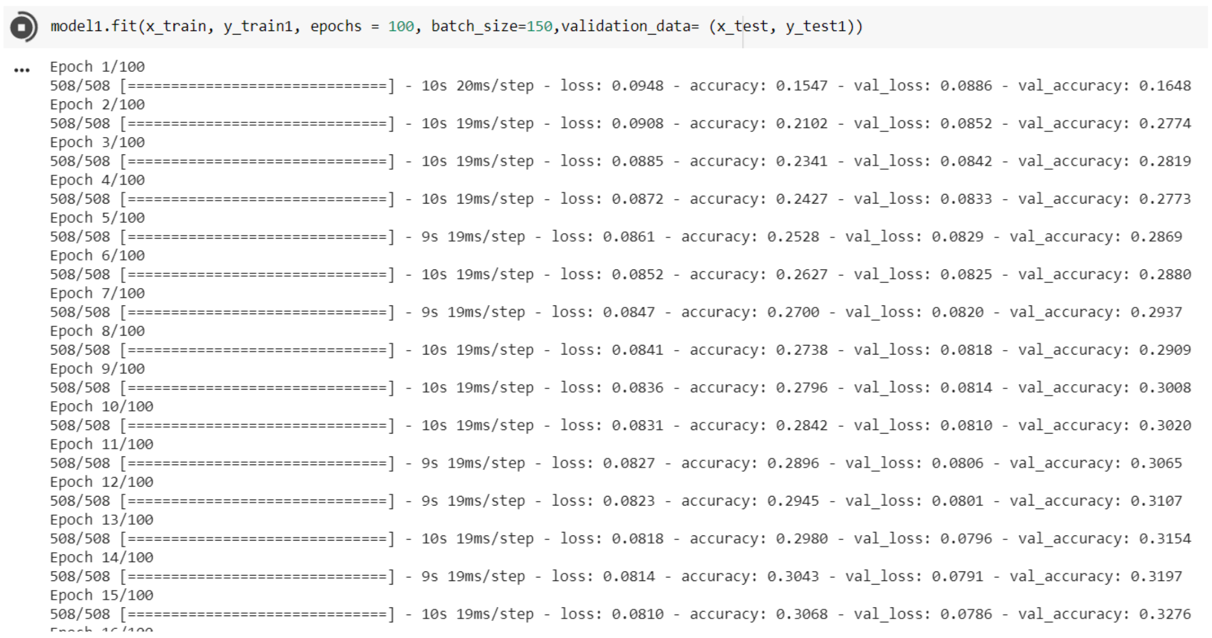
```
model.fit(x_train, y_train, epochs = 100, batch_size=150,validation_data= (x_test, y_test))

Epoch 1/100
508/508 [==============================] - 11s 22ms/step - loss: 0.0913 - accuracy: 0.1889 - val_loss: 0.0868 - val_accuracy: 0.1870
Epoch 2/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0857 - accuracy: 0.2409 - val_loss: 0.0823 - val_accuracy: 0.2722
Epoch 3/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0838 - accuracy: 0.2620 - val_loss: 0.0811 - val_accuracy: 0.2963
Epoch 4/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0823 - accuracy: 0.2827 - val_loss: 0.0795 - val_accuracy: 0.3187
Epoch 5/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0812 - accuracy: 0.2995 - val_loss: 0.0788 - val_accuracy: 0.3240
Epoch 6/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0803 - accuracy: 0.3123 - val_loss: 0.0779 - val_accuracy: 0.3371
Epoch 7/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0795 - accuracy: 0.3237 - val_loss: 0.0772 - val_accuracy: 0.3476
Epoch 8/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0788 - accuracy: 0.3339 - val_loss: 0.0767 - val_accuracy: 0.3495
Epoch 9/100
508/508 [==============================] - 10s 21ms/step - loss: 0.0780 - accuracy: 0.3425 - val_loss: 0.0759 - val_accuracy: 0.3651
Epoch 10/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0773 - accuracy: 0.3538 - val_loss: 0.0742 - val_accuracy: 0.3841
Epoch 11/100
508/508 [==============================] - 10s 21ms/step - loss: 0.0767 - accuracy: 0.3593 - val_loss: 0.0736 - val_accuracy: 0.3918
Epoch 12/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0759 - accuracy: 0.3694 - val_loss: 0.0724 - val_accuracy: 0.4059
Epoch 13/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0753 - accuracy: 0.3781 - val_loss: 0.0715 - val_accuracy: 0.4208
Epoch 14/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0746 - accuracy: 0.3876 - val_loss: 0.0712 - val_accuracy: 0.4196
Epoch 15/100
508/508 [==============================] - 10s 21ms/step - loss: 0.0741 - accuracy: 0.3934 - val_loss: 0.0705 - val_accuracy: 0.4217
```

Fig 7.1 First 15 epochs for valence



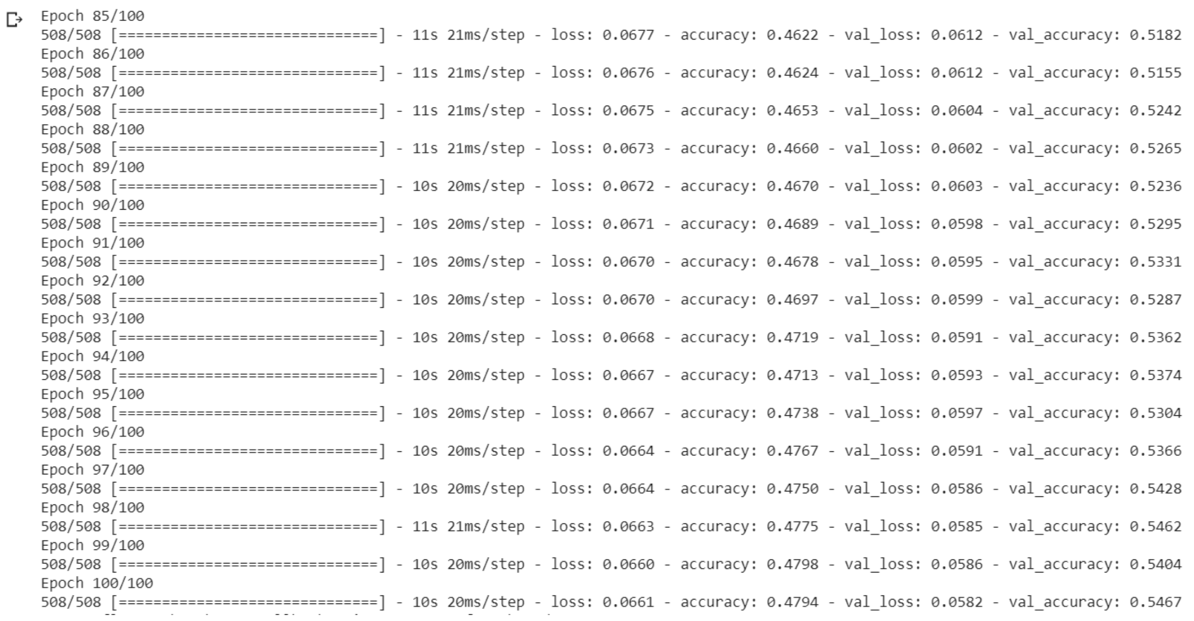
```
Epoch 85/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0548 - accuracy: 0.5953 - val_loss: 0.0438 - val_accuracy: 0.6771
Epoch 86/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0545 - accuracy: 0.5972 - val_loss: 0.0458 - val_accuracy: 0.6613
Epoch 87/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0545 - accuracy: 0.5977 - val_loss: 0.0453 - val_accuracy: 0.6689
Epoch 88/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0542 - accuracy: 0.5995 - val_loss: 0.0432 - val_accuracy: 0.6862
Epoch 89/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0542 - accuracy: 0.5999 - val_loss: 0.0435 - val_accuracy: 0.6797
Epoch 90/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0537 - accuracy: 0.6037 - val_loss: 0.0433 - val_accuracy: 0.6840
Epoch 91/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0538 - accuracy: 0.6018 - val_loss: 0.0439 - val_accuracy: 0.6795
Epoch 92/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0536 - accuracy: 0.6055 - val_loss: 0.0424 - val_accuracy: 0.6958
Epoch 93/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0537 - accuracy: 0.6047 - val_loss: 0.0438 - val_accuracy: 0.6779
Epoch 94/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0535 - accuracy: 0.6048 - val_loss: 0.0436 - val_accuracy: 0.6801
Epoch 95/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0534 - accuracy: 0.6077 - val_loss: 0.0420 - val_accuracy: 0.6962
Epoch 96/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0532 - accuracy: 0.6090 - val_loss: 0.0423 - val_accuracy: 0.6905
Epoch 97/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0531 - accuracy: 0.6086 - val_loss: 0.0417 - val_accuracy: 0.6971
Epoch 98/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0528 - accuracy: 0.6117 - val_loss: 0.0421 - val_accuracy: 0.6913
Epoch 99/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0526 - accuracy: 0.6137 - val_loss: 0.0421 - val_accuracy: 0.6916
Epoch 100/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0526 - accuracy: 0.6121 - val_loss: 0.0419 - val_accuracy: 0.6962
```

Fig 7.2 Last 15 epochs for valence



```
model1.fit(x_train, y_train1, epochs = 100, batch_size=150,validation_data= (x_test, y_test1))

Epoch 1/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0948 - accuracy: 0.1547 - val_loss: 0.0886 - val_accuracy: 0.1648
Epoch 2/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0908 - accuracy: 0.2102 - val_loss: 0.0852 - val_accuracy: 0.2774
Epoch 3/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0885 - accuracy: 0.2341 - val_loss: 0.0842 - val_accuracy: 0.2819
Epoch 4/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0872 - accuracy: 0.2427 - val_loss: 0.0833 - val_accuracy: 0.2773
Epoch 5/100
508/508 [==============================] - 9s 19ms/step - loss: 0.0861 - accuracy: 0.2528 - val_loss: 0.0829 - val_accuracy: 0.2869
Epoch 6/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0852 - accuracy: 0.2627 - val_loss: 0.0825 - val_accuracy: 0.2880
Epoch 7/100
508/508 [==============================] - 9s 19ms/step - loss: 0.0847 - accuracy: 0.2700 - val_loss: 0.0820 - val_accuracy: 0.2937
Epoch 8/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0841 - accuracy: 0.2738 - val_loss: 0.0818 - val_accuracy: 0.2909
Epoch 9/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0836 - accuracy: 0.2796 - val_loss: 0.0814 - val_accuracy: 0.3008
Epoch 10/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0831 - accuracy: 0.2842 - val_loss: 0.0810 - val_accuracy: 0.3020
Epoch 11/100
508/508 [==============================] - 9s 19ms/step - loss: 0.0827 - accuracy: 0.2896 - val_loss: 0.0806 - val_accuracy: 0.3065
Epoch 12/100
508/508 [==============================] - 9s 19ms/step - loss: 0.0823 - accuracy: 0.2945 - val_loss: 0.0801 - val_accuracy: 0.3107
Epoch 13/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0818 - accuracy: 0.2980 - val_loss: 0.0796 - val_accuracy: 0.3154
Epoch 14/100
508/508 [==============================] - 9s 19ms/step - loss: 0.0814 - accuracy: 0.3043 - val_loss: 0.0791 - val_accuracy: 0.3197
Epoch 15/100
508/508 [==============================] - 10s 19ms/step - loss: 0.0810 - accuracy: 0.3068 - val_loss: 0.0786 - val_accuracy: 0.3276
```

Fig 7.3 First 15 epochs for arousal



```
Epoch 85/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0677 - accuracy: 0.4622 - val_loss: 0.0612 - val_accuracy: 0.5182
Epoch 86/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0676 - accuracy: 0.4624 - val_loss: 0.0612 - val_accuracy: 0.5155
Epoch 87/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0675 - accuracy: 0.4653 - val_loss: 0.0604 - val_accuracy: 0.5242
Epoch 88/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0673 - accuracy: 0.4660 - val_loss: 0.0602 - val_accuracy: 0.5265
Epoch 89/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0672 - accuracy: 0.4670 - val_loss: 0.0603 - val_accuracy: 0.5236
Epoch 90/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0671 - accuracy: 0.4689 - val_loss: 0.0598 - val_accuracy: 0.5295
Epoch 91/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0670 - accuracy: 0.4678 - val_loss: 0.0595 - val_accuracy: 0.5331
Epoch 92/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0670 - accuracy: 0.4697 - val_loss: 0.0599 - val_accuracy: 0.5287
Epoch 93/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0668 - accuracy: 0.4719 - val_loss: 0.0591 - val_accuracy: 0.5362
Epoch 94/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0667 - accuracy: 0.4713 - val_loss: 0.0593 - val_accuracy: 0.5374
Epoch 95/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0667 - accuracy: 0.4738 - val_loss: 0.0597 - val_accuracy: 0.5304
Epoch 96/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0664 - accuracy: 0.4767 - val_loss: 0.0591 - val_accuracy: 0.5366
Epoch 97/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0664 - accuracy: 0.4750 - val_loss: 0.0586 - val_accuracy: 0.5428
Epoch 98/100
508/508 [==============================] - 11s 21ms/step - loss: 0.0663 - accuracy: 0.4775 - val_loss: 0.0585 - val_accuracy: 0.5462
Epoch 99/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0660 - accuracy: 0.4798 - val_loss: 0.0586 - val_accuracy: 0.5404
Epoch 100/100
508/508 [==============================] - 10s 20ms/step - loss: 0.0661 - accuracy: 0.4794 - val_loss: 0.0582 - val_accuracy: 0.5467
```

Fig 7.4 Last 15 epochs for valence

CHAPTER 8
RESULTS/OUTPUT SCREENSHOTS

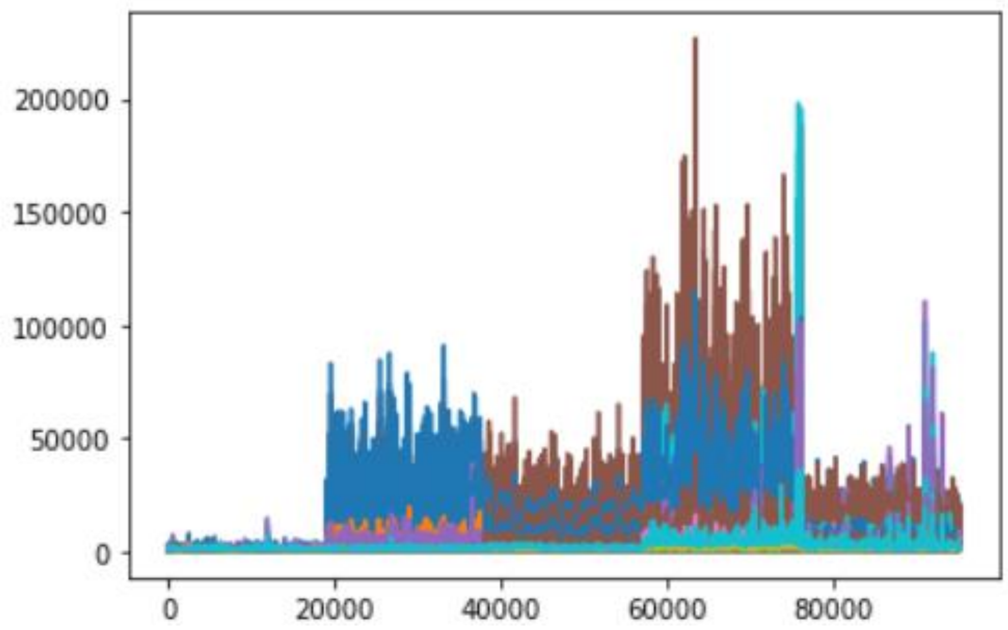


Fig 8.1 Plot showing distribution of frequencies over the trials

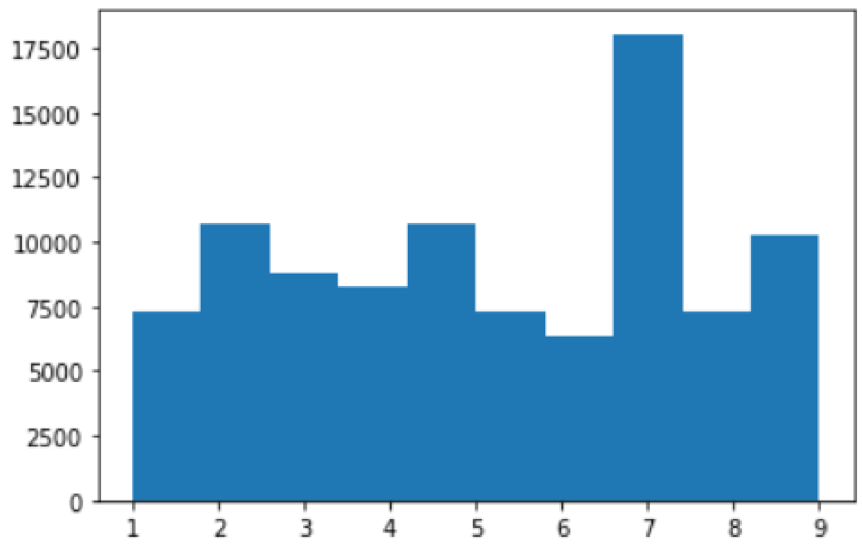


Fig 8.2 Distribution of valence over the trials

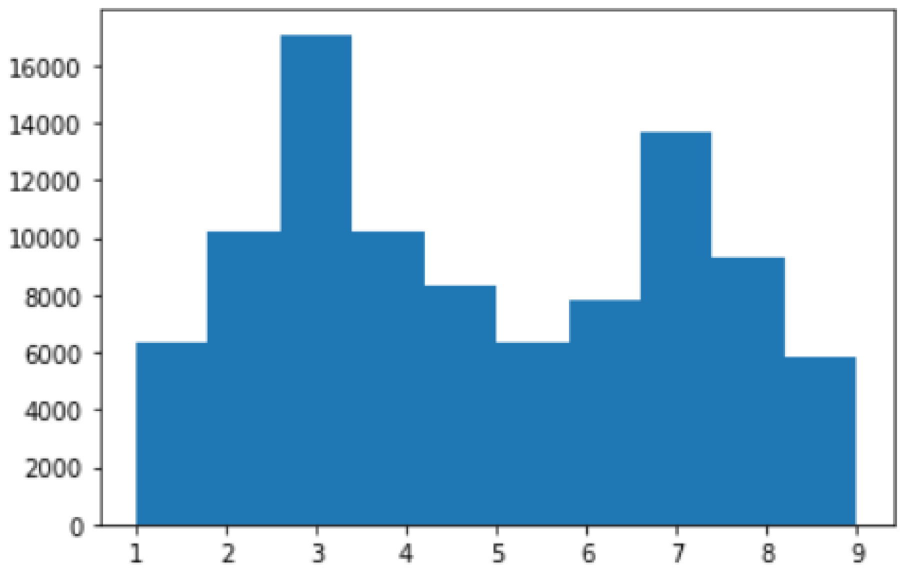


Fig 8.3 Distribution of arousal over the trials

CHAPTER 9

CONCLUSION AND FUTURE WORK

This project helped us to develop a deep learning model for recognizing human emotions through EEG signals. The recognition of human emotions plays a major role in BCI and HCI systems. EEG signals are recorded with electrodes placed on the scalp. In this project we use combination of LSTM layers with different activation functions for dimensionality reduction and classification of emotions.

The accuracy of the current deep learning model is at par with machine learning models. A better way of pre-processing data and a model consisting of different combination techniques might give more accurate results. Recognition of emotions plays a vital role and, in the future, a working prototype which can give results in real-time can be done.

REFERENCES

1. "**DEAP: A Database for Emotion Analysis using Physiological Signals**", S. Koelstra, C. Muehl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, I. Patras, IEEE Transactions on Affective Computing, Special Issue on Naturalistic Affect Resources for System Building and Evaluation, in press

2.**EEG-Based Emotion Recognition Using Deep Learning Network with Principal Component Based Covariate Shift Adaptation** Suwicha Jirayucharoensak, Setha Pan-Ngum and Pasin Israsena

3. **[3] Chrono Net: A Deep Recurrent Neural Network for Abnormal EEG Identification** Subhrajit Roy, Isabell Kiral-Kornek, and Stefan Harrer

4. S. Koelstra, A. Yazdani, M. Soleymani et al., "Single trial classification of EEG and peripheral physiological signals for recognition of emotions induced by music videos

5. M. Soleymani, J. Lichtenauer, T. Pun, and M. Pantic, "A multimodal database for affect recognition and implicit tagging," IEEE Transactions on Affective Computing

6. D. Huang, C. Guan, K. K. Ang, H. Zhang, and Y. Pan, "Asymmetric spatial pattern for EEG-based emotion detection

7. G. Chanel, J. J. M. Kierkels, M. Soleymani, and T. Pun, "Short term emotion assessment in a recall paradigm

8. D. Nie, X.-W. Wang, L.-C. Shi, and B.-L. Lu, "EEG-based emotion recognition during watching movies

9. X.-W. Wang, D. Nie, and B.-L. Lu, "EEG-based emotion recognition using frequency domain features and support vector machines,"

10. N. Jatupaiboon, S. Pan-ngum, and P. Israsena, "Real-time EEG based happiness detection system

11. Glazkova Ekaterina Vasilevna, Soboleva Natalia Alexeevna, Deep Learning for EEG-Based Emotion Analysis

Links referred:

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

https://raphaelvallat.com/bandpower.html

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4045570/

https://www.medicine.mcgill.ca/physio/vlab/biomed_signals/eeg_n.htm#:~:text=The%20electroencephalogram%20(EEG)%20is%20a,measured%20in%20microvolts%20(mV)

https://dl.acm.org/doi/abs/10.5555/3297863.3297883

https://www.bitbrain.com/blog/ai-eeg-data-processing

https://www.kaggle.com/tocodeforsoul/depression-rest-preprocessed

https://medium.com/towards-artificial-intelligence/best-datasets-for-machine-learning-data-science-computer-vision-nlp-ai-c9541058cf4f

https://victorzhou.com/blog/intro-to-neural-networks/