Engagement Recognition Using Video-Based Expression Tracking

Manisha Paliwal

Department of Applied Data Science, San Jose State University

Data 270: Data Analytics Process

Dr. Eduardo Chan

May 14, 2022

4. Model Development

4.1. Model Proposals

During covid -19 pandemic, the lifestyle of people drastically changed. There was a need of less to no human contact while all operations of daily life still needed to be performed for survival. This changed the way industries and education systems work from in person communications to virtual communications. Schools and college lectures, Office meetings and conferences, Court Hearings almost everything was conducted using various online software like zoom, google meet and Webex. Due to lack of in person communications there is an impact in understanding participants engagement levels due to absence of ability to understand non-verbal communications virtually. This project focuses on creating machine learning models to identify and classify engagement levels of individuals participating in virtual communications.

DAiSEE dataset is being used to train the models. The goal is to predict the four classes that are boredom, engagement, confusion, and frustration along with their intensity levels (0, 1, 2, 3) using different facial feature recognition methods in combination with SVM classifier. The dataset is already splits into Training, Validation and Testing folders. As part of this project testing folder is not used as it is not annotated. Training folder data is further split for training and testing in 80:20 ratio. Overall, 112 users-based videos are captured with 9068 video snippets. It is a multi-label classification problem. (Gupta et al., 2018).

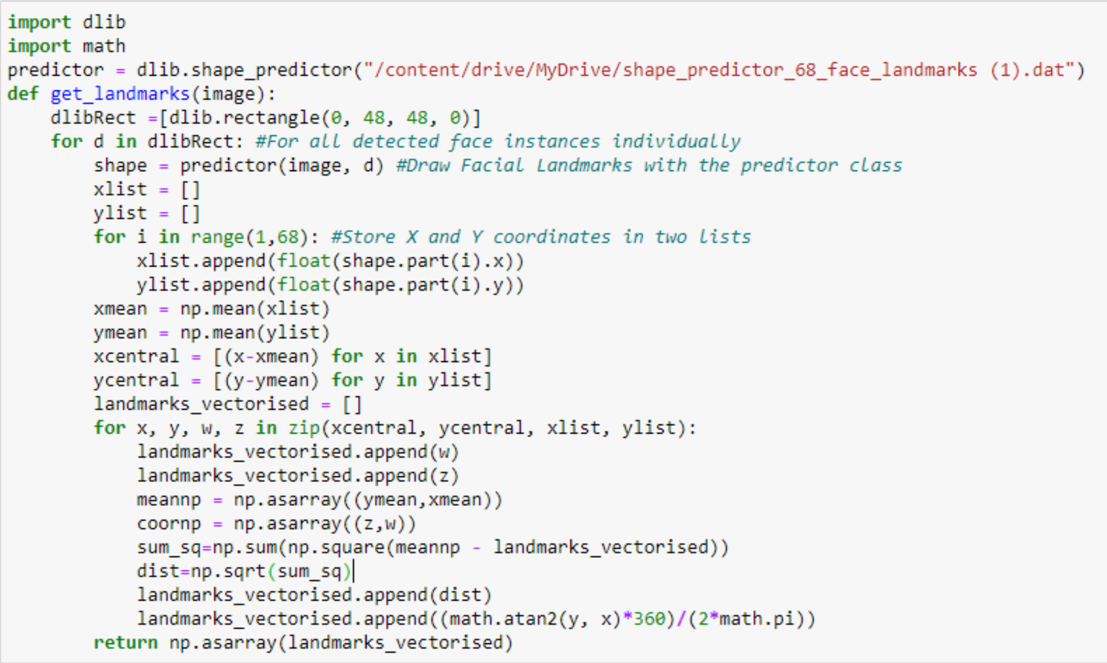
The input data is in the form of videos which is converted into image frames. The model can predict the level of each class. It consists of two techniques; first technique uses hand crafted features to extract the features and make predictions using SVM algorithm. Second, the feature extraction using CNN to extract the features and then pass these features to SVM algorithm to make predictions. The problem was solved using multi class classification concept, where each model is trained for each individual class, each model was able to predict each class with levels. The result of these model is concatenated to predict the overall classes with their levels.

The results are further compared where a hybrid approach to combine features extracted using CNN and 68 Facial Landmarks is used to train SVM classifier by authors in research paper (Rao & Rao, 2021)

The picture frames obtained from movies after data pre-processing are processed through 68 facial landmark identification models utilizing Dlib's approach in the first model methodology. Facial extraction is done in two steps, firstly, face detection by identifying a person’s face which returns values of rectangle vectors in x,y,w,z coordinates. This helps reduce the computational cost. Secondly, the Euclidean distance is used to determine the distances from the center point to each other data point, as well as the related coordinates to map the facial features. The resultant vectors are then fit into SVM- SVC classifier for final classification. Figure 1 shows the code snippet and Figure 2 provides the overview of the model.

Figure 1.

Model Technique 1 Code Snippet

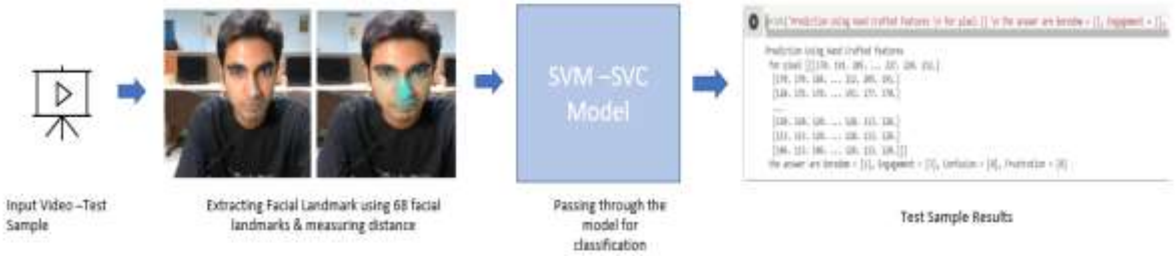


```python
import dlib
import math
predictor = dlib.shape_predictor("/content/drive/MyDrive/shape_predictor_68_face_landmarks (1).dat")
def get_landmarks(image):
    dlibRect =[dlib.rectangle(0, 48, 48, 0)]
    for d in dlibRect: #For all detected face instances individually
        shape = predictor(image, d) #Draw Facial Landmarks with the predictor class
        xlist = []
        ylist = []
        for i in range(1,68): #Store X and Y coordinates in two lists
            xlist.append(float(shape.part(i).x))
            ylist.append(float(shape.part(i).y))
        xmean = np.mean(xlist)
        ymean = np.mean(ylist)
        xcentral = [(x-xmean) for x in xlist]
        ycentral = [(y-ymean) for y in ylist]
        landmarks_vectorised = []
        for x, y, w, z in zip(xcentral, ycentral, xlist, ylist):
            landmarks_vectorised.append(w)
            landmarks_vectorised.append(z)
            meannp = np.asarray((ymean,xmean))
            coornp = np.asarray((z,w))
            sum_sq=np.sum(np.square(meannp - landmarks_vectorised))
            dist=np.sqrt(sum_sq)
            landmarks_vectorised.append(dist)
            landmarks_vectorised.append((math.atan2(y, x)*360)/(2*math.pi))
        return np.asarray(landmarks_vectorised)
```

Note. Code snippet showing method used to implement handcraft method

Figure 2.

Model Technique 1 Process

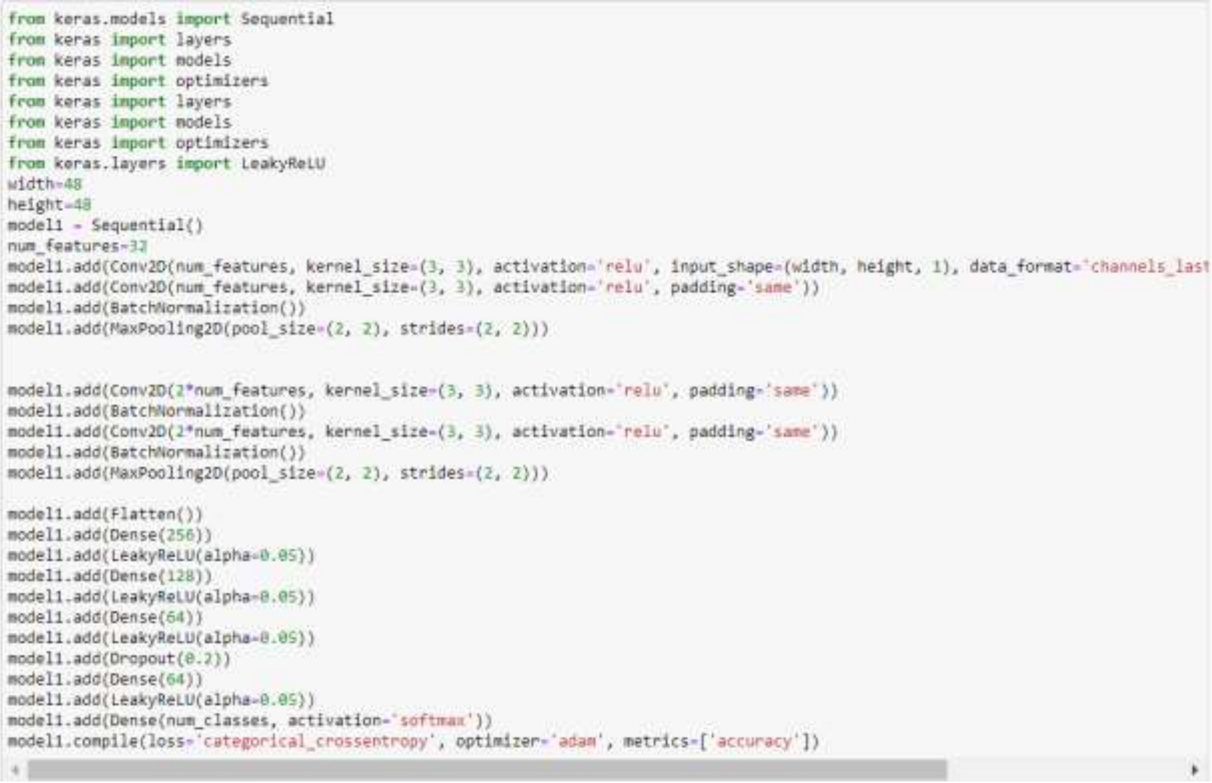

Input Video –Test Sample
Extracting Facial Landmark using 68 facial landmarks & measuring distance
SVM –SVC Model
Passing through the model for classification
Test Sample Results

Note. Overview of proposed model -Technique 1 [Handcraft Feature Extraction +SVM-SVC]

In model technique 2, images are processed using CNN using Kera's sequential model with Leaky ReLu. It consists of convolutional, pooling, and fully connected layers. We have used Conv2D layers with 32 and 64 layers. Leaky ReLu is selected as the activation method upon an input image resized to 48*48. Softmax is used for activation and dense output layer have been set to four corresponding top four intensities of a particular emotion. Most used Adam optimizer is used in this case with cross entropy loss function to evaluate the accuracy. The vector output is fit into SVM-SVC classifier for each emotion and then concatenated to provide overall results. Figure 2 shows the code snippet and Figure 3 shows the overview of the model.

Figure 2.

Model technique 2 Code Snippet

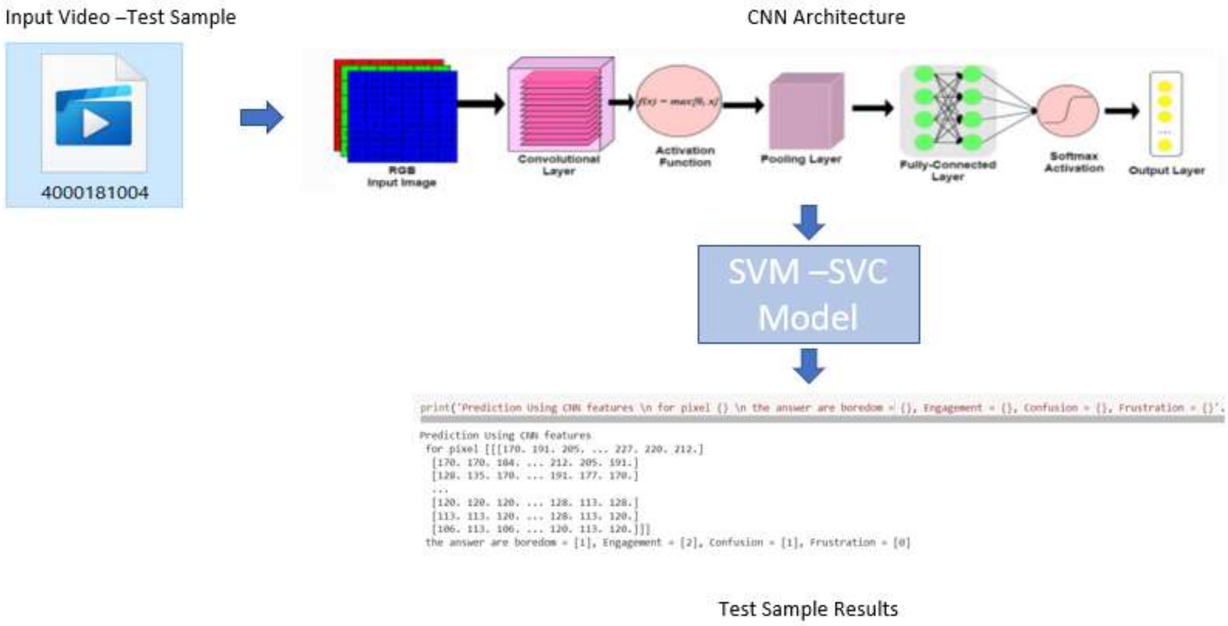


```
from keras.models import Sequential
from keras import layers
from keras import models
from keras import optimizers
from keras import layers
from keras import models
from keras import optimizers
from keras.layers import LeakyReLU
width=48
height=48
model1 = Sequential()
num_features=32
model1.add(Conv2D(num_features, kernel_size=(3, 3), activation='relu', input_shape=(width, height, 1), data_format='channels_last
model1.add(Conv2D(num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model1.add(BatchNormalization())
model1.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))


model1.add(Conv2D(2*num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model1.add(BatchNormalization())
model1.add(Conv2D(2*num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model1.add(BatchNormalization())
model1.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))

model1.add(Flatten())
model1.add(Dense(256))
model1.add(LeakyReLU(alpha=0.05))
model1.add(Dense(128))
model1.add(LeakyReLU(alpha=0.05))
model1.add(Dense(64))
model1.add(LeakyReLU(alpha=0.05))
model1.add(Dropout(0.2))
model1.add(Dense(64))
model1.add(LeakyReLU(alpha=0.05))
model1.add(Dense(num_classes, activation='softmax'))
model1.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Note. Code snippet showing method used to implement custom CNN method

Figure 4.

Model Technique 2 Process



```
print('Prediction Using CNN features \n for pixel {} \n the answer are boredom = {}, Engagement = {}, Confusion = {}, Frustration = {}'.
```

```
Prediction Using CNN features
 for pixel [[[170. 191. 205. ... 227. 220. 212.]
  [170. 170. 184. ... 212. 205. 191.]
  [128. 135. 170. ... 191. 177. 170.]
  ...
  [120. 120. 120. ... 128. 113. 128.]
  [113. 113. 120. ... 128. 113. 120.]
  [106. 113. 106. ... 120. 113. 120.]]]
 the answer are boredom = [1], Engagement = [2], Confusion = [1], Frustration = [0]
```

Note. Overview of proposed model -Technique 2 [CNN +SVM-SVC]

SVM generally doesnot support multi class classification. SVC with linear kernel is used from LIBSVM library. Each emotion class is modelled seperately and intensity levels (0,1,2,3) are broken down into binary classification using one -to -one approach where a hyperplance between two intensity levels in drawn ignoring the other levels. The results are then combined to provide final emotion classification.This method helps us better understand the loss function during training and validation.

4.2 Model Supports

Figure 5 shows the high-level data flow diagram of the model. AWS S3 is a service provided by amazon services for object storage through web services. DAiSEE dataset is downloaded from source and initially stored in local machine. The size of data volume is 15 giga bytes which contains total of 9068 video clips collected form 112 users.

AWS Sage maker is a service provided by amazon to perform end to end machine learning operations from modelling, training, testing, and deployment of models. It also has an inbuilt support for python jyupter notebook. Dataset is uploaded into AWS S3 Bucket to perform modelling using Amazon Sage maker. Data is loaded using python jyupter notebook.

Python modules that are part of the project are TensorFlow, pandas, NumPy, OS, PIL, math, SciPy, matplotlib, sklearn, time is used.

Pandas and NumPy are used for data preparation purposes. Sklearn library is used for machine learning modelling. PIL is an image processing library used to process images.

Matplotlib library is further used for visualization purposes used in this project to plot training and validation loss graph. The dataset post processing has huge volume of images therefore, data is divided into 64 batches before performing modelling.

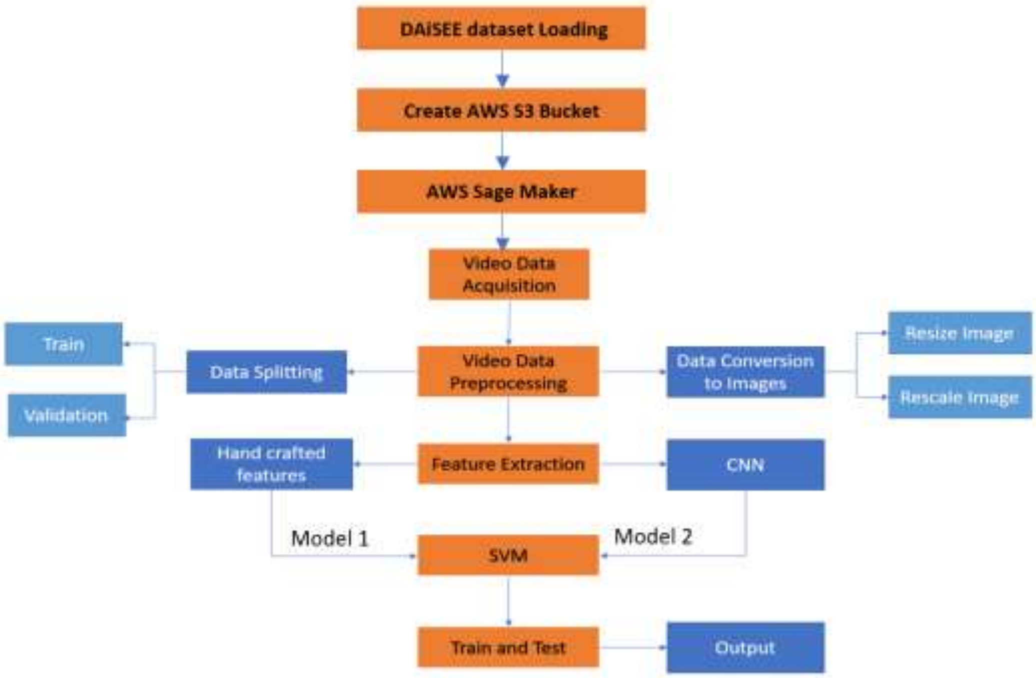
OpenCV is used for image processing like reading, writing, and displaying the image. It is an open-source library which can also be used for complex techniques like facial recognition in combination with NumPy.

Dlib is a library programmed in C++ to provide complex machine learning algorithms and provide classes and functions to aid data analytics.

TensorFlow is a google provided open-source python-based library to implement deep learning techniques. Kera API of TensorFlow is used for performing feature extraction and modelling techniques. Kera is a computational intelligence framework built in Python. It is a high-level TensorFlow API that extend the functionality of it. This is useful for creating a basic categorization model. Sequential model has linear layers. Kera library contains modules like batch normalization, activation functions, optimizers, and loss function. Subsequently, the framework is assembled, fitted, and assessed. (Journal, 2019)

Figure 5

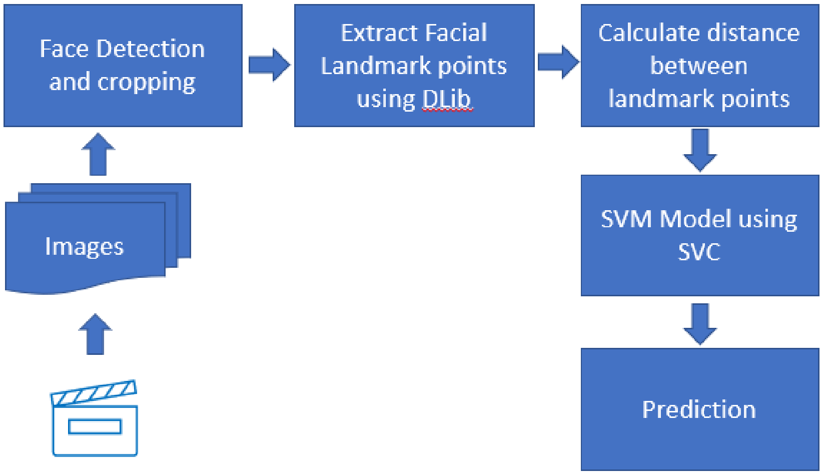
High Level Data Flow Diagram of Model Structure:



Note. The data flow diagram provides a high-level view of end-to-end methodology used in project

Figure 6

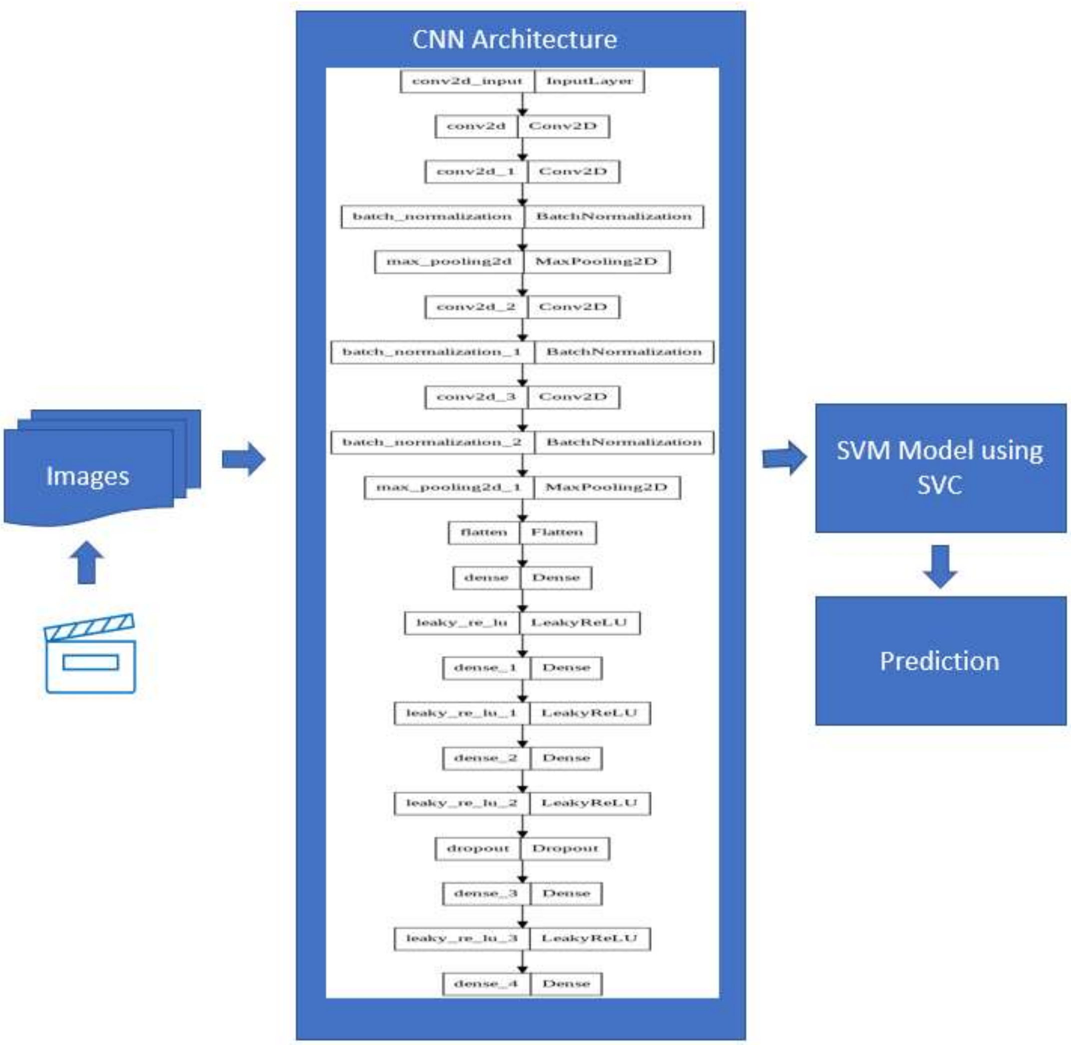
68 Facial Landmark Detection with SVM Model


Face Detection and cropping
Extract Facial Landmark points using DLib
Calculate distance between landmark points
Images
SVM Model using SVC
Prediction

Note. Pictorial representation of model implementation flow

Figure 7

CNN Facial Feature Extraction with SVM Model


CNN Architecture
conv2d_input | InputLayer
conv2d | Conv2D
conv2d_1 | Conv2D
batch_normalization | BatchNormalization
max_pooling2d | MaxPooling2D
conv2d_2 | Conv2D
batch_normalization_1 | BatchNormalization
conv2d_3 | Conv2D
batch_normalization_2 | BatchNormalization
max_pooling2d_1 | MaxPooling2D
flatten | Flatten
dense | Dense
leaky_re_lu | LeakyReLU
dense_1 | Dense
leaky_re_lu_1 | LeakyReLU
dense_2 | Dense
leaky_re_lu_2 | LeakyReLU
dropout | Dropout
dense_3 | Dense
leaky_re_lu_3 | LeakyReLU
dense_4 | Dense
Images
SVM Model using SVC
Prediction

Note. Pictorial representation of model implementation flow along with CNN architecture

4.3 Model Comparison and Justification

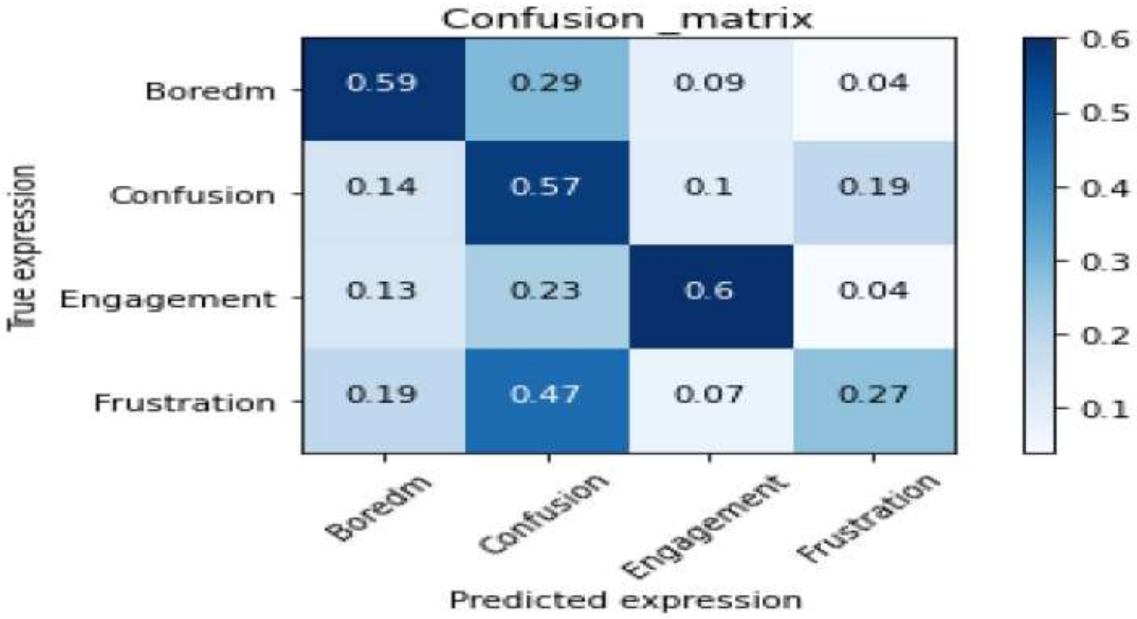
In this project, we are applying two different facial feature extraction techniques separately in machine learning SVM model to evaluate the performance of each model on DAiSEE dataset. SVM classification model trained using 68 facial landmark detection using

Dlib library with calculating distance between data point using Euclidean method provided an accuracy and precision of 45% while when SVM classification model is trained using custom CNN facial feature recognition method with activation function Leaky ReLU and linear kernel the performance was slightly better at 50 % accuracy and precision. The results are shown in figure 11 and Figure 12.

In their research paper, (Rao & Rao, 2021) used a hybrid model. Using the position analyzer, obtain the Handcrafted characteristics (68 important face points). Calculate the distances and slope degrees from the focus point to each other feature point, then save the coordinates. Hand-crafted features were added to CNN features. Using integrated features, the classifier is constructed with a SVM algorithm. The face characteristics are extracted using the last five levels of the CNN architecture. Furthermore, these characteristics, are paired with the geometric characteristics derived with Dlib pose detector. To categorize the attributes, the new feature set is passed into the SVM, which generates hyperplanes with training set. This model has an accuracy of 53.42 percent. Since the dataset was obtained in the environment and not professionally captured, these results are satisfiable and distinguishable in the e-learning sector. The confusion matrix is created to ensure accuracy as shown in Figure 8. According to the data, most of the images having emotion as frustration are forecasted as confusion. The ‘frustration' state has an extremely low detection performance unlike in our project where frustration has higher accuracy compared to other emotions.

Figure 8

Confusion Matrix of hybrid model based SVM classification



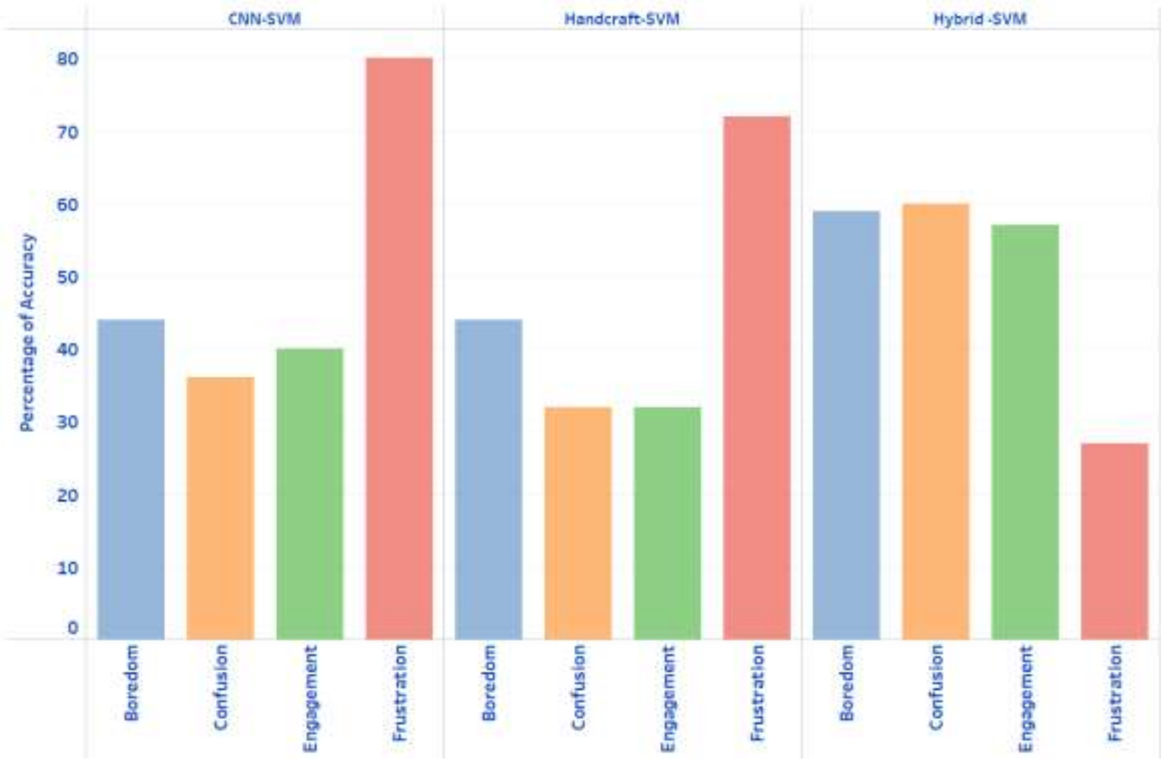
	Boredm	Confusion	Engagement	Frustration
Boredm	0.59	0.29	0.09	0.04
Confusion	0.14	0.57	0.1	0.19
Engagement	0.13	0.23	0.6	0.04
Frustration	0.19	0.47	0.07	0.27

Note. Rao, & Rao. (2021, May). [Confusion matrix for FER using hybrid CNN with DAiSEE dataset]].https://www.researchgate.net/publication/351905322_Recognition_of_Learners'_Cognitive_States_using_Facial_Expressions_in_E-learning_Environments

We can conclude that the hybrid model with SVM performs slightly better than their individual techniques used in this project with consistency in predicting all four emotions. Figure 9 demonstrates the performance of each model compared across emotions.

Figure 9.

Performance Comparison of Models



Note. Bar chart to demonstrate accuracy of each model for a particular emotion

Unlike common deep learning systems where computational techniques are black box, SVMs allow for some intuitive thinking and insight. They cope with unbalanced data and overfitting by permitting certain prediction error on the training images. Even in high dimensional datasets, the model handles data that is linearly inseparable. A hierarchy of binary classifiers integrated together can perform multi-class and multi label classification. SVMs are being used effectively for a lot of classification problems. In a variety of uses, they presently outperform deep neural networks. (Michel & Kaliouby, 2003)

4.4 Model Evaluation Methods

In this project we evaluate the performance of each model using popular evaluation metrics used for classification that is accuracy, precision and recall and confusion matrix. The models are based on supervised learning approach for image classification. The SVM classifier is used to classify emotions along with level of intensity. To evaluate the best approach for emotion recognition models are trained on datasets where facial features are extracted using two different techniques handcraft method and custom CNN.

4.4.1 Accuracy

Accuracy is a statistic used to determine which model is better at spotting patterns and correlations between observable variables based on trained data. The accuracy ratio is calculated by dividing the number of properly classified points by the total number of points. If we have a skewed dataset, we can get accurate results using a naive model. As a result, accuracy cannot be applied to imbalanced datasets. Accuracy is not a great predictor of whether a model is good or bad.

Accuracy = (True Positive + True Negative) /Total

4.4.2 Precision

Precision is a classification model's ability to identify only meaningful data items. It determines the quality of classification. A higher precision values means the model return appropriate results most of the time and lesser inappropriate results and wise versa. It is the count of true positives by sum of true positives and false positives. This refers to all data points that a model identifies as positive.

Precision=True positives/ (True Positive + False Positive)

4.4.3 Recall

Recall determines the quantity of classification and measures all relevant data points. It is classification model’s ability to identify number of correctly predicted classes out of total classification. Statistically, recall is the count of accurate positive predictions by total count of positive predictions possible.

Recall=True Positive/ (True Positive + False Negative)

4.4.4 Confusion Matrix

A Confusion matrix is a N x N grid used to assess the effectiveness of a classification algorithm, where N represents the number of class labels. The matrix evaluates the original label value to the output of algorithm's predicted value. This provides a comprehensive picture of our classification model prediction appropriateness and the kind of inaccuracies it is producing.

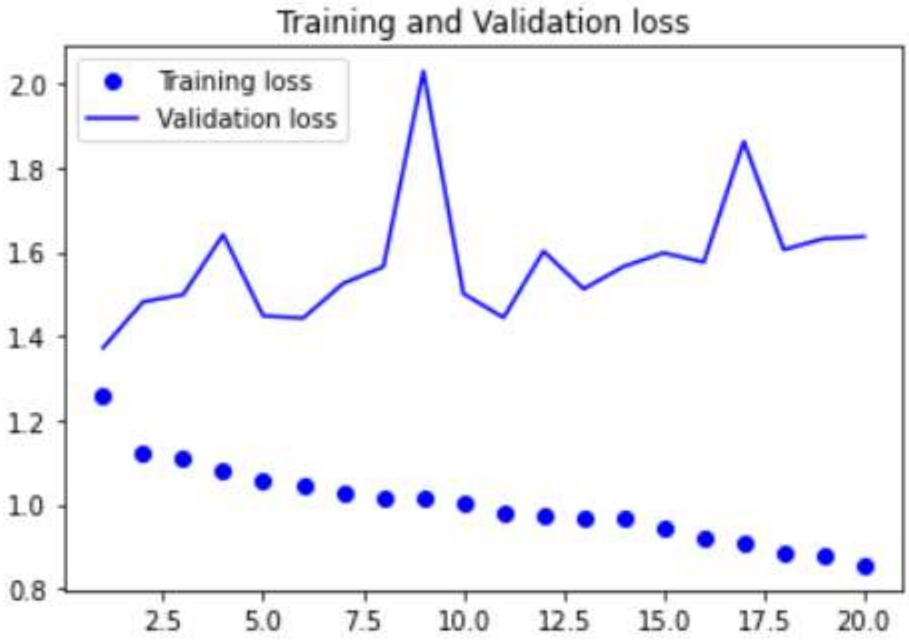
4.5 Model Validation and Evaluation

DAiSEE dataset consist of separate folders for train, validate and test dataset. As part of the project, test folder was not utilized as it was not annotated. After converting videos into image frames and preprocessing, 80% of training images were used for training the model and 20% for testing. Validation folder images were used for validation purposes.

For custom CNN based technique of SVM modelling, the training and validation loss was evaluated. The training loss provides the measure of performance of training data fit in the model while validation loss provides the model's measure to fit testing data. We have used L2 regularize to generalize the dataset. Adam optimizer, gradient descent methodology is used to reduce noise and adjust weights and pace of learning to minimize the loss. Figure 10, show the graph epoch 20 and batch size 64 with validation done for each iteration. Since modelling has been done on sample data due to computational complexities, we can see that validation loss is greater than training loss which shows probability of under fitting scenario.

Figure 10.

Training and Validation Loss Diagram

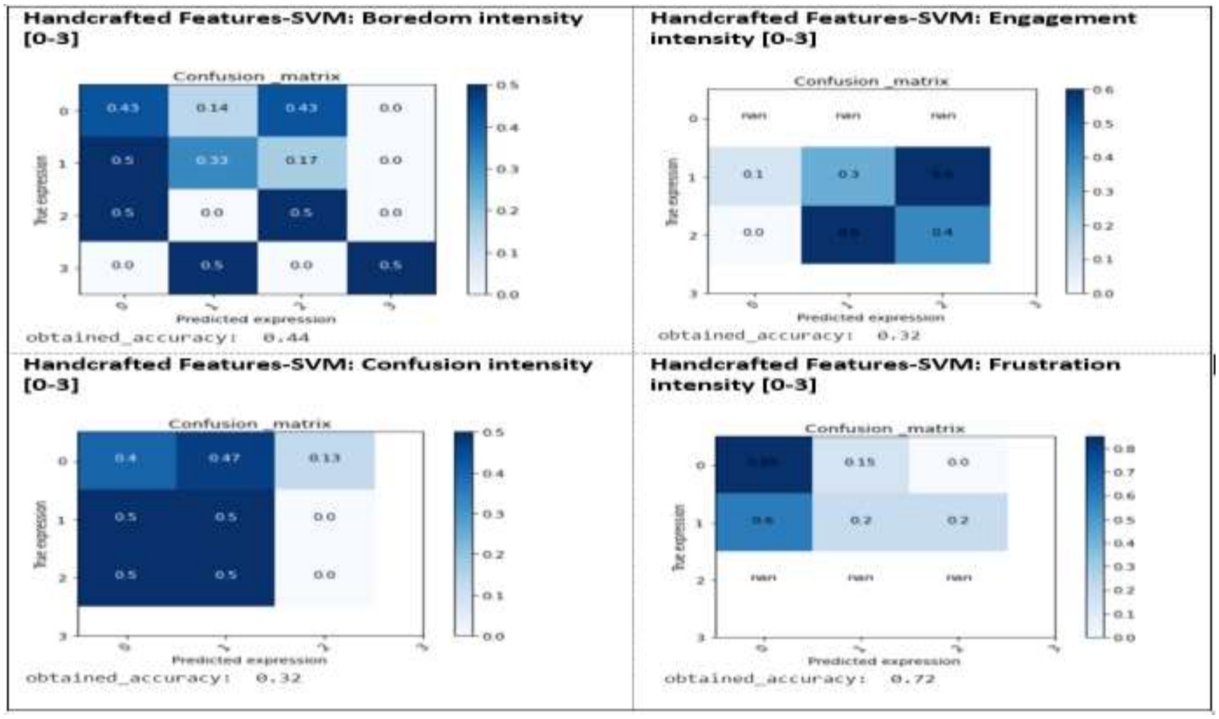


Note. Graphical representation of loss curved while implementing CNN model over multiple iterations

Figure 11 shows confusion matrix of each model created for each emotion using handcraft feature extraction technique. As shown in the figure, accuracy of frustration is higher than other emotions at 72% with higher accuracy for very low and low intensities followed by boredom at an accuracy level of 44% with even accuracies at all intensity levels. Accuracy of engagement and confusion is relatively low at 32%. Therefore, overall accuracy achieved by combining four models to predict final engagement classification using 68 facial landmarks detection with Dlib averages to 45%.

Figure 11

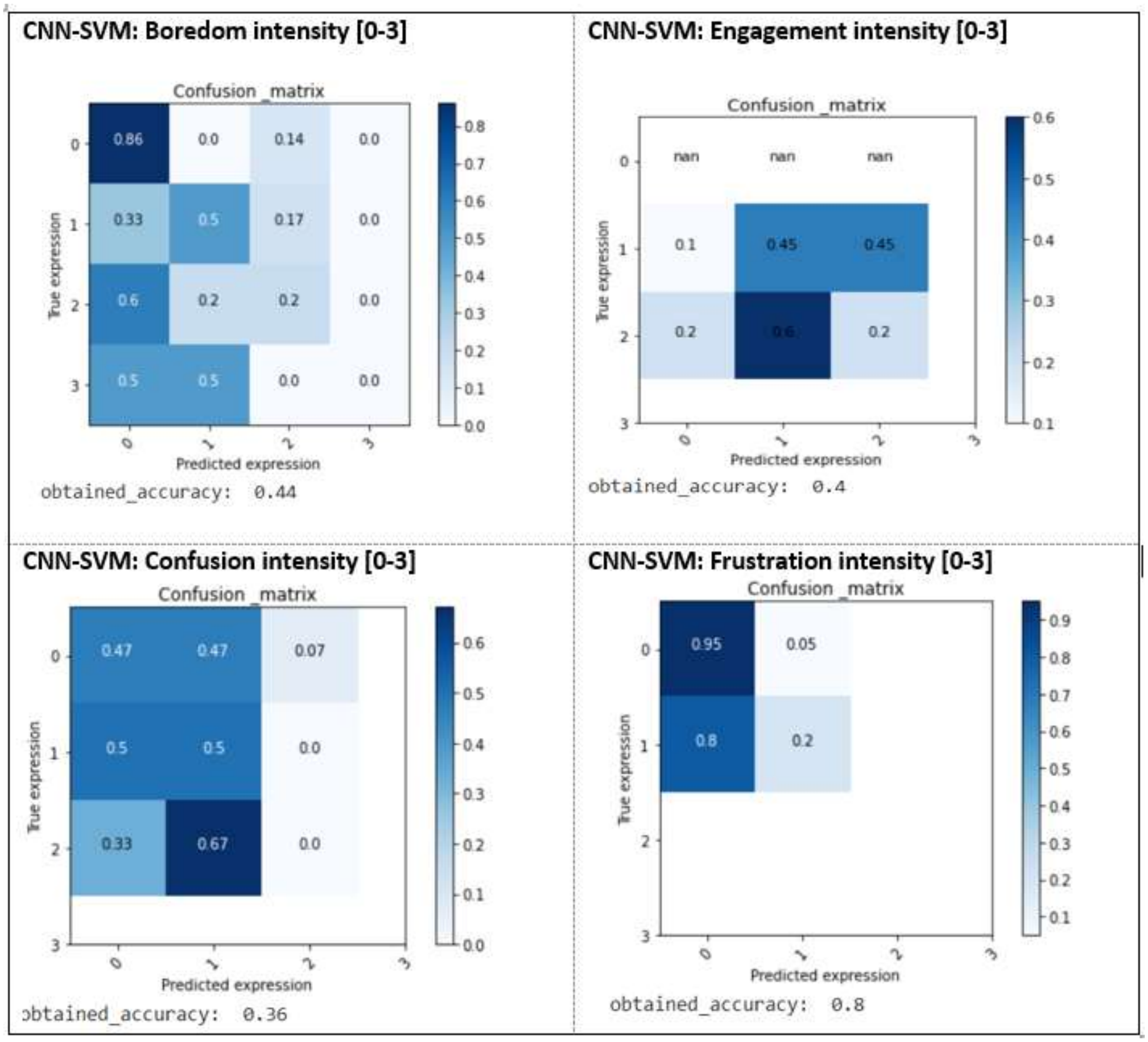
Confusion Matrix of FRE using 68 Facial Landmarks with Dlib and SVM



Note. The graphs shown are for each emotion for which modelling is done separately to classify accuracy across four different intensity levels

Figure 12

Confusion Matrix of FRE using custom CNN and SVM

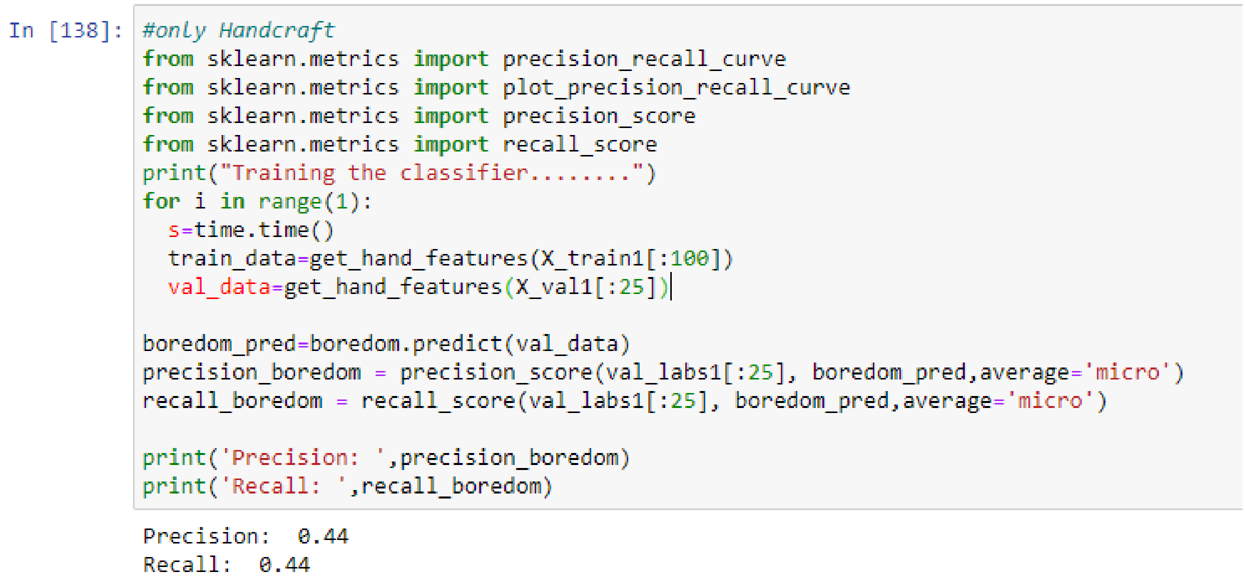


Note. The graphs shown are for each emotion for which modelling is done separately to classify accuracy across four different intensity levels

Figure 12 shows confusion matrix of each model created for each emotion using custom CNN facial feature extraction technique and then SVM algorithm used for classification. As shown in the figure, accuracy for frustration is high at 80% higher accuracy for very low and low emotion intensities followed. Boredom has an accuracy of 44% evenly distributed across all intensity levels closely followed by engagement at 40% and lowest for frustration at 36%. Therefore, overall accuracy achieved by combining four models to predict final engagement classification using custom CNN technique averages to 50%.

Figure 13.

Code Snippet to derive precision and recall



```
In [138]: #only Handcraft
          from sklearn.metrics import precision_recall_curve
          from sklearn.metrics import plot_precision_recall_curve
          from sklearn.metrics import precision_score
          from sklearn.metrics import recall_score
          print("Training the classifier........")
          for i in range(1):
            s=time.time()
            train_data=get_hand_features(X_train1[:100])
            val_data=get_hand_features(X_val1[:25])

          boredom_pred=boredom.predict(val_data)
          precision_boredom = precision_score(val_labs1[:25], boredom_pred,average='micro')
          recall_boredom = recall_score(val_labs1[:25], boredom_pred,average='micro')

          print('Precision: ',precision_boredom)
          print('Recall: ',recall_boredom)

          Precision:  0.44
          Recall:  0.44
```

Note. Code snippet demonstrating implementation of training classifier and determining precision and recall for boredom

Table 1

Overall evaluation metrics of models implemented

Model\Evaluation	Accuracy	Precision	Recall
68 Facial Landmarks Detection with SVM	0.45	0.45	0.45
CNN with SVM	0.5	0.5	0.5

Note. The table demonstrates a comparison of results of accuracy, precision and recall for models

Overall accuracy of custom CNN model is slightly better than handcraft models which is expected. We can also see that CNN technique is better in classifying engagement emotion compared to other technique while confusion classification is least accurate irrespective of technique used. Precision and recall show similar results as accuracy as mentioned in table 1.

Reference:

Gupta, A., D'Cunha, A., Awasthi, K., & Balasubramanian, V. (2018, April 13). *Daisee: Towards user engagement recognition in the wild*. arXiv.org. Retrieved February 19, 2022, from https://arxiv.org/abs/1609.01885

Journal, I. R. J. E. T. (2018, July 28). *IRJET-V5I6191.pdf*. Academia.edu. Retrieved May 14, 2022, from https://www.academia.edu/37138364/IRJET_V5I6191_pdf

Michel, P., & Kaliouby, R. E. (2003, November 1). *Real time facial expression recognition in video using support vector machines: Proceedings of the 5th International Conference on Multimodal Interfaces*. ACM Conferences. Retrieved May 14, 2022, from https://dl.acm.org/doi/abs/10.1145/958432.958479?casa_token=fCs6gH_KryMAAAAA%3Ai9ENUAY8DUKRexd4AwIdUhj0aeb7RLZfvll3UDxpOpjXT70qVCzD-PleYPA9siteGvJrEM_QyyjDHw

Rao, K. P., & Rao, D. M. V. P. C. S. (2021, May). *Recognition of learners' cognitive states using facial expressions in E-learning environments*. Retrieved May 13, 2022, from https://www.researchgate.net/publication/351905322_Recognition_of_Learners'_Cognitive_States_using_Facial_Expressions_in_E-learning_Environments