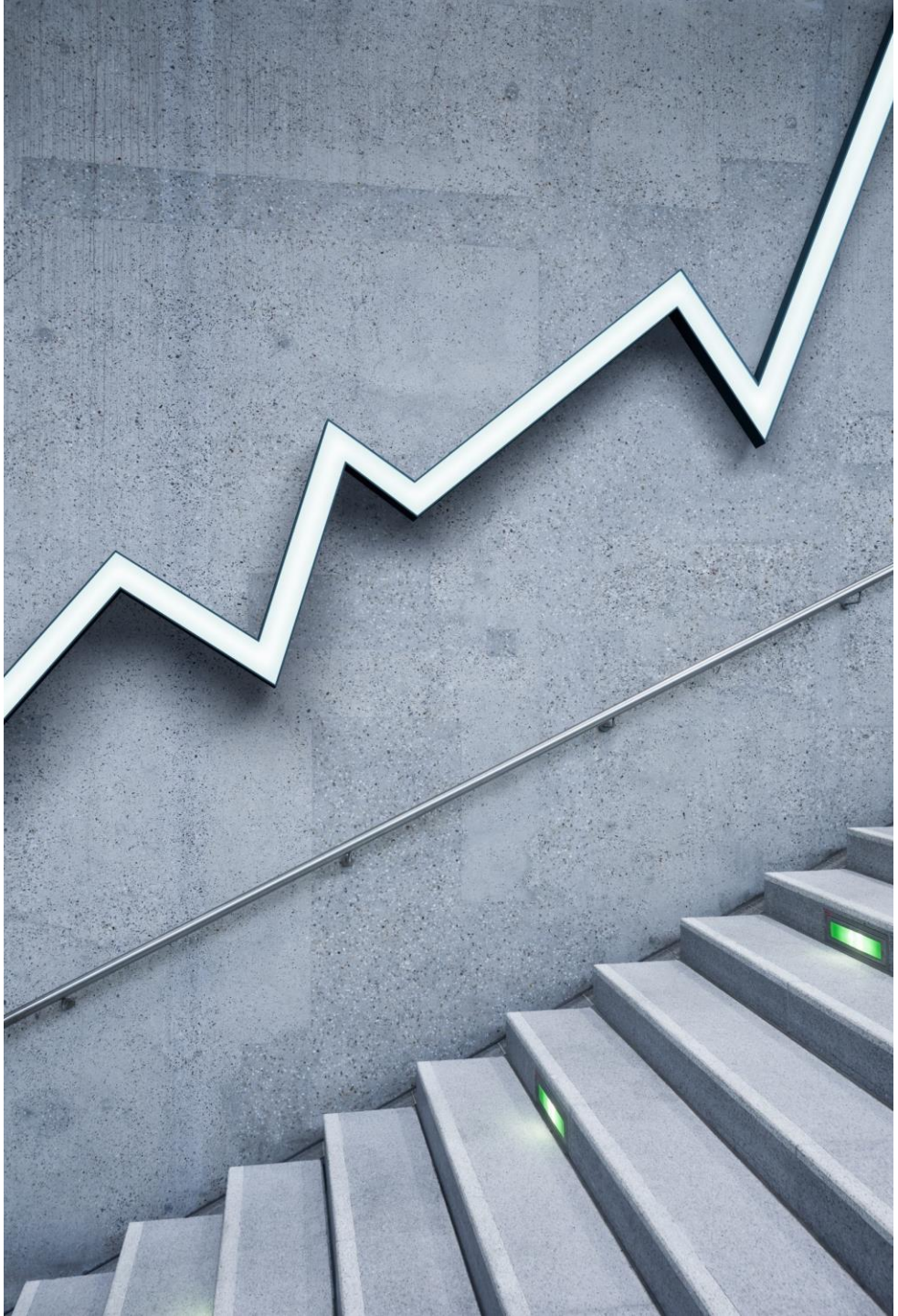

CS360 Mini Project

SETTIPALLI SAI TEJA- 1901180

Face Position Detection

Face detection is the step stone to all facial analysis algorithms, including the face alignment, face modelling, face relighting, face recognition, face verification/authentication, head pose tracking, facial expression tracking/ recognition, gender/age recognition, and many more. Given a digital image, the primary goal of face detection is to determine whether or not there are any faces in the image. This appears as a trivial task for human beings, but it is a very challenging task for computers, and has been one of the top studied research topics in the past few decades. The difficulty associated with face detection can be attributed to many variations in scale, location, orientation (in-plane rotation), pose (out-of-plane rotation), facial expression, lighting conditions, occlusions, etc. Face detection is a computer technology that determines the location and size of a human face in a digital image. Face detection has been a standout amongst topics in the computer vision literature. The main problem with face detection is that a computer/machine cannot be made to a learn a human face as a whole and be expected to recognize another face as they are far to different to compare. The more logical explanation for the machine would be to be able to learn about the facial features of the human face and then based on these feature it can detect whether something is a human face or not since the facial features of humans are comparable to each others and we can get a relative positioning between these features to make a logical prediction of the image. The Algorithm will have to check for the proper count and proportions of these feature like 2 eyes which are apart from each other etc. After detecting these features the coordinates of these features can be used in conjugation with the relative positioning of these features to detect the position of the entire human face in a given image. Hence what appears to be a seemingly simple task can be quite complex to analyze and detect for a machine.

Data Set –

The data set for this Machine Learning model involves a diverse set of images which has people in them. To train this model we need to have a good variation in faces and the expressions of the people represented in the pictures since the model has to detect the face and its position based on the facial features of people represented in the images. The images are frames of a video involving multiple people driving car over multiple days and thus we are assured that background changes thus increasing randomness. Since the images are part of a video and since peoples expressions change over time we are sure to be able to capture a multitude of expressions and thus a variety of facial features in the model. Since the data set involves images involving multiple people driving the car this promotes diversity in the sample space thus preventing underfitting and thus preventing the model from there by becoming biased to a specific person or a group of people. The data set also puts all the different people in similar scenarios so there is an ease in detection and prevents a bias in a way. There is also A second Data Set which provides us with the co-ordinates various features like Eyes, Nose, Mouth, Angle of the face, The position of the entire face. It also contains labels which tell us about the direction in which the face is looking. This data can be obtained by extracting details from the images as well. Using the co-ordinates of these facial features we can find the angle of the face which intern can be used to find the direction of looking of the person.

The field of face detection has made considerable progress in the past decade. Mukherjee et al. (2017) have discussed the formulation for both the methods, i.e., using hand-crafted features followed by training a simple classifier and an entirely modern approach of learning features from data using neural networks. Ren et al. (2017)have presented a method for real time detection and tracking of the human face. The proposed method combines the Convolution Neural Network detection and the Kalman filter tracking. Convolution Neural Network is used to detect the face in the video, which is more accurate than traditional detection method. When the face is largely deflected or severely occluded, Kalman filter tracking is utilized to predict the face position. They try to increase the face detection rate, while meeting the real time requirements. Luo et al. (2018) have suggested deep cascaded detection method that iteratively exploits bounding-box regression, a localization technique, to approach the detection of potential faces in images. Their method leverages cascaded architecture with three stages of carefully designed deep convolutional networks to predict the existence of faces.

Challenges in face detection, are the reasons which reduce the accuracy and detection rate of face detection. These challenges are complex background, too many faces in images, odd expressions, illuminations, less resolution, face occlusion, skin color, distance and orientation etc.

Applications of face detection system can be seen in various day to day scenarios like Gender classification, Document control and access control, Human computer interaction, Biometric attendance, Photography, Face detection is also useful for selecting regions of interest in photo slideshows, Facial feature extraction Facial features like nose, eyes, mouth, skin-color etc. can be extracted from image, Facial recognition, Marketing

Yan, Kriegman, and Ahuja presented a classification for face detection methods. These methods divided into four categories, and the face detection algorithms could belong to two or more groups. These categories are as follows-

1.Knowledge-Based:-

The knowledge-based method depends on the set of rules, and it is based on human knowledge to detect the faces. Ex- A face must have a nose, eyes, and mouth within certain distances and positions with each other. The big problem with these methods is the difficulty in building an appropriate set of rules. There could be many false positive if the rules were too general or too detailed. This approach alone is insufficient and unable to find many faces in multiple images.

2.Feature-Based:-

The feature-based method is to locate faces by extracting structural features of the face. It is first trained as a classifier and then used to differentiate between facial and non-facial regions. The idea is to overcome the limits of our instinctive knowledge of faces. This approach divided into several steps and even photos with many faces they report a success rate of 94%.

3.Template Matching:-

Template Matching method uses pre-defined or parameterized face templates to locate or detect the faces by the correlation between the templates and input images. Ex- a human face can be divided into eyes, face contour, nose, and mouth. Also, a face model can be built by edges just by using edge detection method. This approach is simple to implement, but it is inadequate for face detection. However, deformable templates have been proposed to deal with these problems.

4.Appearance-Based:-

The appearance-based method also called as image based methods depends on a set of delegate training face images to find out face models. The appearance-based approach is better than other ways of performance. In general appearance-based method rely on techniques from statistical analysis and machine learning to find the relevant characteristics of face images. This method also used in feature extraction for face recognition.

The appearance-based model further divided into sub-methods for the use of face detection some of which are as follows-

- ***Eigenface-Based:-***Eigenface based algorithm used for Face Recognition, and it is a method for efficiently representing faces using Principal Component Analysis.
- ***Distribution-Based:-***The algorithms like PCA and Fisher's Discriminant can be used to define the subspace representing facial patterns. There is a trained classifier, which correctly identifies instances of the target pattern class from the background image patterns.
- ***Neural-Networks:-***Many detection problems like object detection, face detection, emotion detection, and face recognition, etc. have been faced successfully by Neural Networks.
- ***Support Vector Machine:-***Support Vector Machines are linear classifiers that maximize the margin between the decision hyperplane and the examples in the training set. Osuna et al. first applied this classifier to face detection.
- ***Sparse Network of Winnows:-***They defined a sparse network of two linear units or target nodes; one represents face patterns and other for the non-face patterns. It is less time consuming and efficient.

The most common algorithms used for face position detection based of image and its points is

- Voila-Jones algorithm to detect faces which is a feature based detection algorithm. Feature-based face detection algorithms are fast and effective and have been used successfully for decades. It is a fast and robust method for face detection which is 15 times quicker than existing techniques at the time of release with 95% accuracy The technique relies on the use of simple Haar-like features that are evaluated quickly through the use of a new image representation. Based on the concept of an integral image it generates a large set of features and uses the boosting algorithm AdaBoost to reduce the over complete set (Zhang et al. 2011). The detector is applied in a scanning fashion and used on gray-scale images, the scanned window that is applied can also be scaled, as well as the features evaluated. This face detection framework is capable of processing images extremely rapidly while achieving high detection rates.

Reference-Face detection techniques: a review
Ashu Kumar, Amandeep Kaur, Munish Kumar

Approach:-

Since image processing is beyond the scope of our course the standard approaches for face detection as mentioned in the previous slides cannot be used. Instead for this application we make use of the locations of various facial features which is obtained from the images before hand. The X and Y coordinates of these facial features can be used to train the ML model. The X and Y coordinates together make up the location of the face. For the training and prediction process the facial features like eyes, nose, mouth are used. The problem is that of classification in our case and we are required to predict where the face is looking. Label 1 being right, Label 2 being forward, Label 3 being towards the left. In other words we predict the angle of the face which we intern use to predict the direction in which the face is looking which we classify into 3 different classes.

To do this we first need to choose the features on which we need to train the model because not all features are use full. We do this using the cor-relation between the head direction and other features. it is clear that head position is weakly cor-related to the y-coordinates of any face feature, which makes sense, and it is also weakly cor-related to the driver's face width and height, hence we can simply drop these columns from our dataset. So we only use 'xF', 'xRE', 'xLE', 'xN', 'xRM', 'xLM’ as the features for our model. This is a multiclass Classification problem involving 3 classes.Using these features we can train multiple types of ML models using algorithms like Logistic Regression, Sigmoid Neuron, Single Layer Perceptron, Multi Layer Perceptron.

Logistic Regression.

Multi-class with One vs. All approach. The dataset is divided as Train: 50% , Validation: 25%, Test: 25%. Hyper-Parameter Tuning with Learning Rates =[0.001, 0.003, 0.005, 0.01, 0.03, 0.05, 0.1, 0.3,0.5] since Alpha value has to be in between 0 and 1. Max Iterations = [500,500,500] and Rho = 1e-6 for our case.

The data set is first shuffled and normalized. Then split the dataset in train, validation and test. Then I randomly initialized the weights which will equal to the number of features (6) +1 the additional being W0. After that we calculate the activation function using sigmoid activation function. Then we calculate the error using log loss function. And then using Stochastic gradient learning algorithm, I update the weights at each pattern. After that completion of each epoch, we checked for the convergence criteria Hyper parameter tuning is done from using a set of Learning rates as mentioned above. The rho value and no of epochs have been taken as constants for simplicity and this being a very small number ensures a good accuracy in the model. This model requires us to create 3 different regressors each would tell whether the given sample is of that class or not. Since the data set is having very low samples from class 1 and class 3 the result Is biased towards class and the model tends to think of all samples as if they belong to class 2.

The final loss obtained for each class after training and validation are [0.205, 0.360, 0.196].

The Overall accuracy for this model is 89.77%

Class-wise Precision = [0%, 89.77% , 0%]

Class-wise Recall = [0%, 100%, 0%]

The main reason behind the 0% for Class 1 and 3 is the lack of needed variety in the dataset. Because of this there is high chance that the randomised training set chosen is made up of all Class 2 elements even if class 1,3 elements are there in the set there are not enough samples for the model to learn from with the learning rate set by us. In cases where the learning rate is greater than 1 like say 3 there is a better class-wise precision as the model has higher scope to learning but then we will be emphasising more on the error of each epoch rather than on the values of the features of the sample which would be wrong.

We also implement the same using K Fold cross validation rather than using just learning rate and rho. In this algorithm we split the entire set in k parts out of which k-1 parts are used for training and 1 part is used for the testing of the model. This ensures that a large number of samples are included in the training stage and even a greater variety is included. But since the total number of samples of the other 2 sets are very few in number the no of each type of samples in each fold is so small making it very hard for the model to learn.

The Performance Statistics for the K-Fold with Logistic Regression are

The Overall accuracy for this model is 89.77%

Class-wise Precision = [0%, 89.77% , 0%]

Class-wise Recall = [0%, 100%, 0%]

Sigmoid Neuron

The Sigmoid Neuron Algorithm uses artificial neurons which get triggered upon meeting a certain threshold value/ function set by us. If a Neuron corresponding to a class is triggered then it is assigned to that class. The samples are trained using these neurons which have weights whose values are updated after each sample. The final weights are used for the purpose of prediction of the values. We use . We use a threshold value of 0.5 for this. We keep checking for convergence of the model while implanting the epochs.

he Performance Statistics for the Sigmoid Neuron are

The Overall accuracy for this model is 91.09%

Class-wise Precision = [100%, 90.97% , 100%]

Class-wise Recall = [25%, 100%, 5.26%]

Since Sigmoid Neuron works on activation rather than on higher chance it tends to give better result in our case as we have very low no of samples from other 2 sets

Single Layer Perceptron

In this we arrange the perceptrons to get a layer of perceptrons. The values of the outputs from these perceptrons are used to generate outputs. We use Learning Rate = 0.1, Max Iterations = 500, Rho = 1e-6. the Loss function used is Mean-Squared Error. The weights are initialized before the training starts and after each sample the weights are updated and sigmoid neurons are used in this so we use its formula. One hot encoding is used to train and even get the output of the prediction.

The performance metrics for this are

The Overall accuracy is 90.43%

Class-wise Precision = [0%, 90.43% , 0%]

Class-wise Recall = [0%, 100%, 0%]

The main reason behind the 0% for Class 1 and 3 is the lack of needed variety in the dataset. Because of this there is high chance that the randomised training set chosen is made up of all Class 2 elements even if class 1,3 elements are there in the set there are not enough samples for the model to learn from with the learning rate set by us.

The Single Layer Perceptron model can also be implemented using the K-Fold validation method instead of using both learning rate and rho. Here we take the value of K as 5 so 4 sets are used for training and 1 set for testing. This is repeated 5 time where the folds keep changing their purpose so that each fold can be used once as testing set and every fold is used up for the training purpose.

The Performance statistics for K Fold validation SLP are

The Overall accuracy is 90.50%

Class-wise Precision = [0%, 90.50% , 0%]

Class-wise Recall = [0%, 100%, 0%]

Since the total number of samples of the other 2 sets are very few in number the no of each type of samples in each fold is so small making it very hard for the model to learn. Even though the model will be trained to the samples of other classes the samples of Class 1 and Class 3 will be dominated by the samples of Class 2 causing the model to be biased towards it and thus believe that samples of class 1 and 3 also belong to class 2.

Since the Data Set being biased towards Class 2 having nearly 90% of the total samples the Machine Learning process is severely hindered. The ML Models which classify the pattern based on its likely hood to belong to a class will have a very high chance to mistake it belong to class 2 because it learns about class 2 the most and not much about class 1 and 3 so even training becomes very hard. So these models will place them in class 2 though they belong to class 1 and 3. The models which classify based on triggering of a specific condition do better since they don't compare with others so it has a slight edge over the rest. SLP has a slightly better performance than Logistic regression And K Fold algos tend to give better results than normal once since they include a lot more variety.