Support Vector Machine model for Offline Signature Classification

Summer Internship Training Report

By

Ananya Singh
Department of Electrical Engineering
Shiv Nadar University, Greater Noida,
Uttar Pradesh, India-201314

P Vishnu Tej
Department of Computer Science Engineering
Shiv Nadar University, Greater Noida,
Uttar Pradesh, India-201314

August 2021

Acknowledgment

We are highly obliged to Dr. Sanjay Singh, Principal Scientist CEERI and Mr. Sumeet Saurav, Senior Scientist CEERI for their valuable guidance, co-operation and encouragement throughout this project work, which we completed under their supervision. Without their supervision and guidance, this project work would not have been possible.

Ananya Singh
&
P Vishnu Tej

Certificate

Certified that this project work “Support vector machine model for offline signature classification”, carried out by Ananya Singh and P Vishnu Tej, in summer 2021, has been completed under our supervision.

Dr. Sanjay Singh
Principal Scientist, CEERI,
Pilani, Rajasthan, India-333031

Mr. Sumeet Saurav
Senior Scientist, CEERI,
Pilani, Rajasthan, India-333031

Index

Abstract

This report details our internship at the Council of Scientific and Industrial Research-Central Electronics and Engineering Research Institute (CSIR-CEERI). CSIR-CEERI is a premier institution in the country and a constituent national laboratory of Council of Scientific and Industrial Research(CSIR), New Delhi, established in 1953 by Govt. of India, for the advancement of Research and Development (R&D) in the field of Electronics. This report details our experiences as a Research Intern at the CSIR-CEERI. We were introduced to numerous Machine Learning Algorithms on a wide range of applications throughout our internship, ranging from Natural Language Processing to Computer Vision. We started with some background study of various machine learning algorithms such as linear regression, logistic regression, neural networks. After all the reading and understanding algorithms, we were asked to implement signature classification where the data set had signatures in two languages- Bengali and Hindi. The image processing algorithms used were Local binary pattern(LBP) and Histogram of oriented gradient(HOG) whereas the classification algorithm used was Support vector machine(SVM). We used three different models, first one being LBP with SVM, second one being HOG with SVM, and the third one being the fusion of LBP and HOG with SVM. The fusion techniques used in the third model were Feature level fusion and Score level fusion. In order to fine-tune each model and produce the best possible results, a lot of effort was invested into finding the proper hyperparameter values using grid search. Finally the accuracies obtained using the classification report of different models were compared to find the best possible model.

Introduction

1.1 Introduction

The summer training work was carried out under the guidance of the scientists from the Intelligent Systems Group at CSIR-CEERI from June 15 to August 15, 2021. The main goal of the brief summer training for Research Interns was to understand the various concepts and algorithms employed by the Intelligent Systems Group, as well as to learn and develop skill sets on the specific projects being carried out at the group.

The fundamental flow of summer training for a Research Intern at CSIR-CEERI is to study the prerequisite concepts in the first weeks of training and then apply those concepts to complete a small project in Python that is linked to the domains that the group concentrates on.

The prerequisite courses allowed us to get a theoretical and practical grasp of numerous Machine Learning Algorithms, as well as the various applications in which they are employed, ranging from Natural Language Processing to Computer Vision. These courses were mainly Coursera based and additional blogs and articles were also provided to us. The project allowed us to put our newly acquired abilities to use in a hands-on project while also learning about the numerous Python packages and libraries that were required to complete the project.

1.2 Scope of Training

This internship develops the relevant skills related to various Machine Learning concepts like Neural Networks, Linear regression, Logistic regression, Local binary pattern, Histogram of oriented gradients, and Support vector machine models. The internship also provided us with a lot of exposure to Python libraries for image processing and computer vision, such as OpenCV, GridSearchCV and SVC. It also dealt with reading massive volumes of image data and processing it with the aforementioned Python libraries. This internship also gave an exposure to building and defining a Support Vector Machine Model in python and training, tuning, and testing it.

1.3 Details of tools and technologies used

1.3.1 Histogram of oriented gradients (HOG)

HOG focuses on the structure of the object. It extracts the information of the edges magnitude as well as the orientation of the edges.It uses a detection window of 64 x 128 pixels, so we converted the image into (64,128) shape.Then the image was further divided into small parts, then the gradient and orientation of every part is calculated. First the image is divided into 8×8 cells. We calculate the gradient for every pixel in the image.

Calculating the gradients

We will extract a patch of pixels and calculate the gradient for each of it :

Change in X direction(x)

Change in Y direction(y)

This process will give us two new matrices – one storing gradients in the x-direction and the other storing gradients in the y direction.The same process is repeated for all the pixels in the image. Then we find the magnitude and orientation using the values obtained.

Total Gradient Magnitude = $\sqrt{[(x)^2+(y)^2]}$

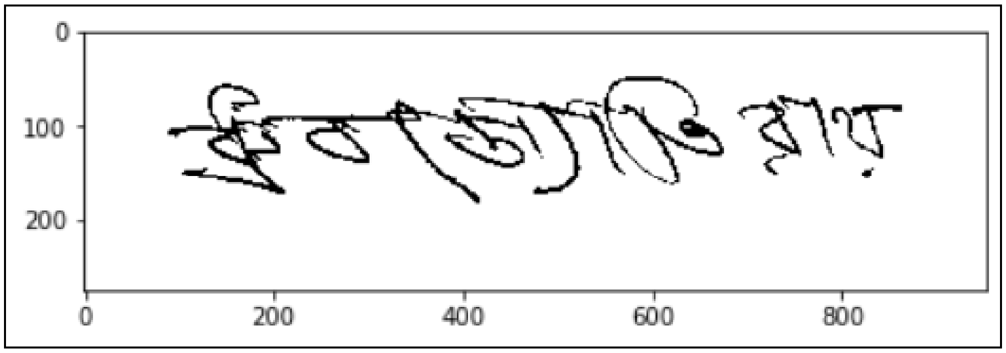
$\tan(\Phi) = (y / x)$

After calculating the gradients, we get the features (or histogram) for the smaller patches which successively represent the entire image. If we divide the image into 8×8 cells and generate the histograms, we'll get a 9 x 1 matrix for every cell. Although when dealing with 8x8 cells, the gradients calculated are very sensitive to light so this would create a problem as some portion of image would be brighter than the other, So, without completely eliminating the light from the image to reduce it we would normalize it by dividing it into blocks of 16x16 cells with 50% overlap. So there are going to be
7x15 = 105 blocks in total, and every block consists of 2x2 cells with 8x8 pixels.We take the 64 gradient vectors of every block (8x8 pixel cell) and put them into a 9-bin histogram. We should note that the histograms created in the HOG feature descriptor are not generated for the whole image.Finally we would have 105 blocks of 16×16 cells. Each of these blocks has a vector of 36×1 as its features. Hence, the entire features for the image would be 105 x 36×1 = 3780 features.

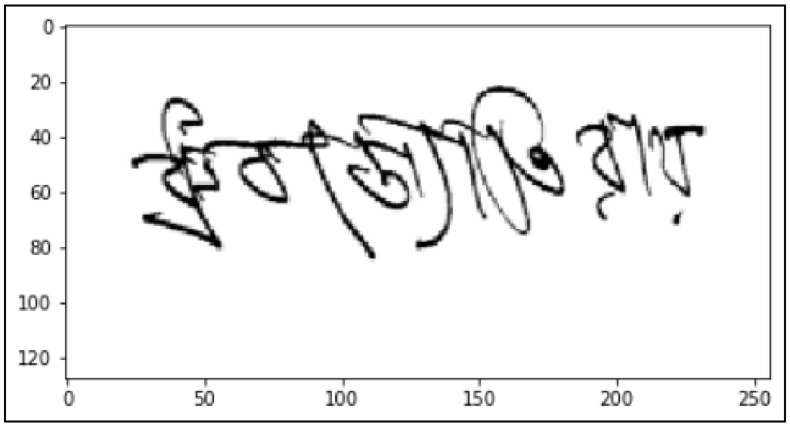
Hog Descriptor

Calculating for each image from a huge dataset will take a long time, to make it simpler we used the hog descriptor from skimage.features directly. Now we don't have to calculate the gradients, magnitude (total gradient) and orientation individually. The hog function would internally calculate it and return the feature matrix.

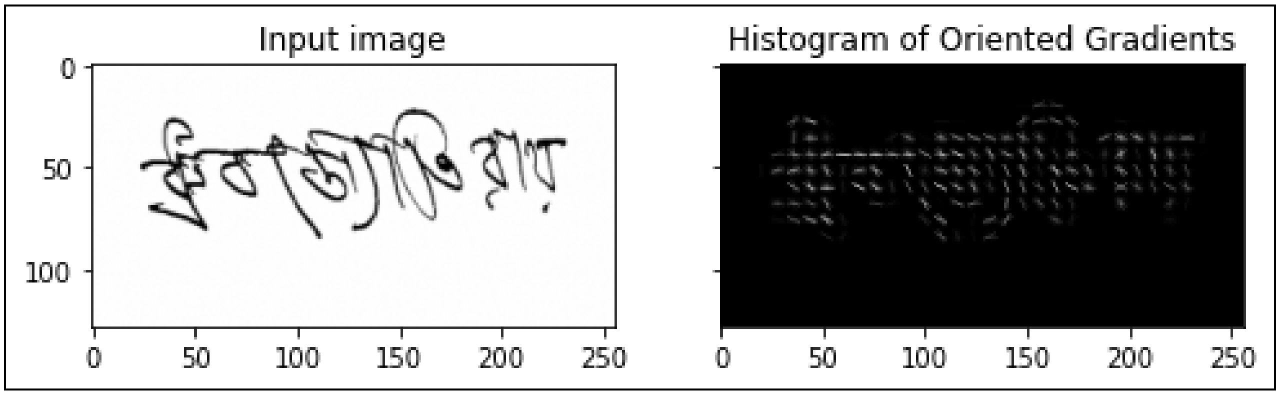
Original image(952 X 275)



Resized image(256 X 128)

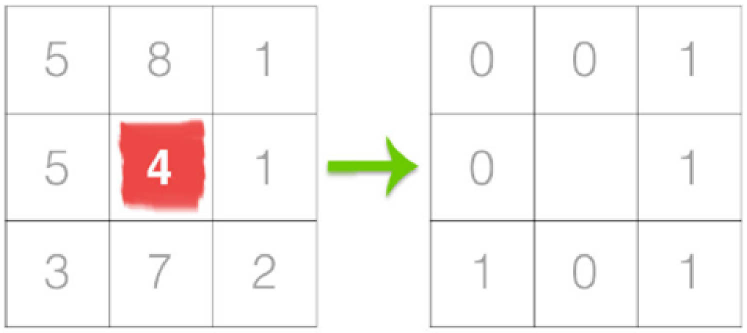


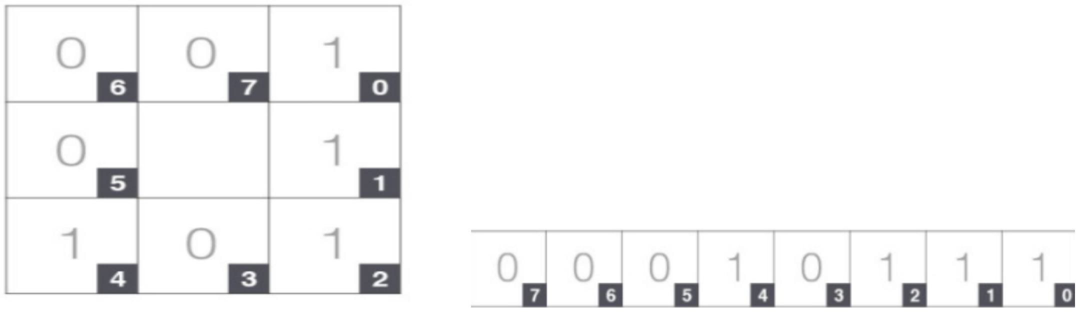
Hog image (256 X 128)



1.3.2 Local Binary Pattern(LBP)

Local binary pattern or LBP is a popular technique used for image/face representation and classification in computer vision. It's a simple yet very efficient texture operator that labels an image's pixels by thresholding each pixel's neighbourhood and converting the result to a binary number. To construct the LBP texture descriptor we convert the image to grayscale. For every pixel within the grayscale image, we select an area of size r surrounding the center pixel, a LBP value is then calculated for this center pixel and stored in the output 2D array with the same width and height as the input image taken. We take the center pixel and compare it against its neighborhood of 8 pixels. If the intensity of the center pixel is greater-than-or-equal to its neighbor, then we set the value to 1; otherwise, we set it to 0. So with 8 surrounding pixels, we have a total of $2^8 = 256$ possible combinations of LBP codes.
We can start from any neighboring pixel and choose clockwise or counter-clockwise direction, but our ordering must be kept consistent for all pixels in our image.
The results of this binary test are stored in an 8-bit array, which we then convert to decimal, like this:

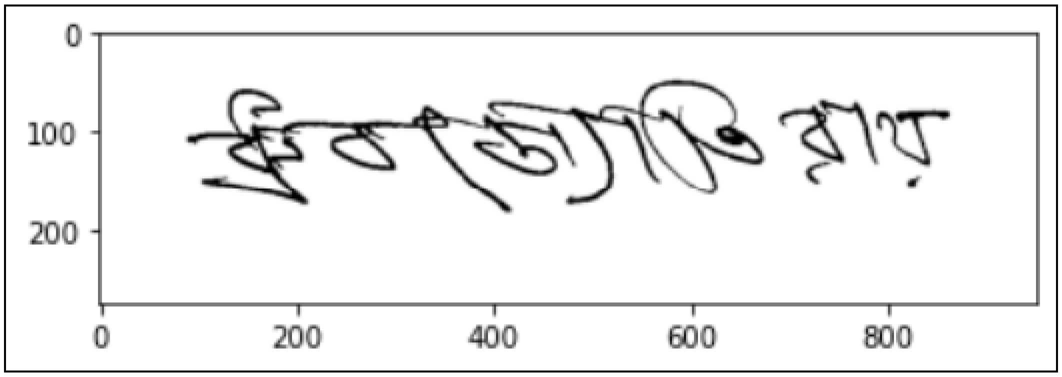




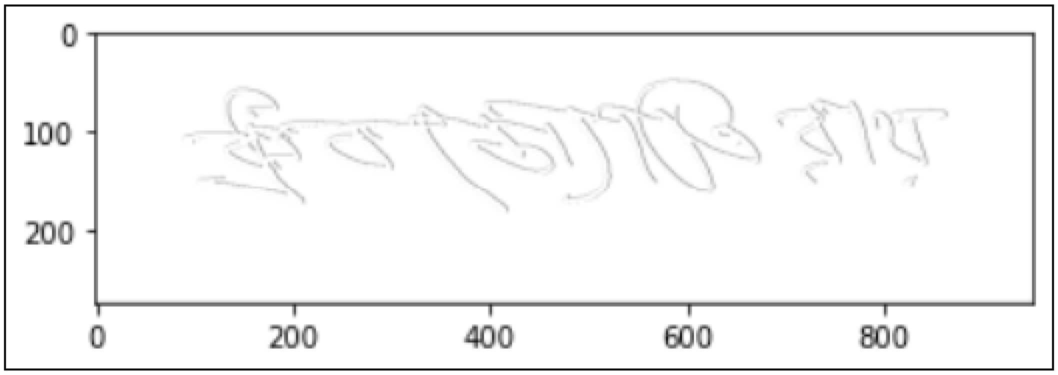
So, the decimal value will be $0*2^7 + 0*2^6 + 0*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 1*2^0 = 16 + 4 + 2 + 1 = 23$

This process of thresholding against the center pixel of 3x3 cell, accumulating binary strings, and storing the output decimal value in the LBP array is then repeated for each pixel in the input image.The last step is to compute a histogram over the output LBP array. Since a 3 x 3 neighborhood has $2^8 = 256$ possible patterns, our LBP 2-D array thus features a minimum value of 0 and a maximum value of 255, allowing us to construct a 256-bin histogram of LBP codes as our final feature vector.

Original image(952 X 275)



LBP image(952 X 275)



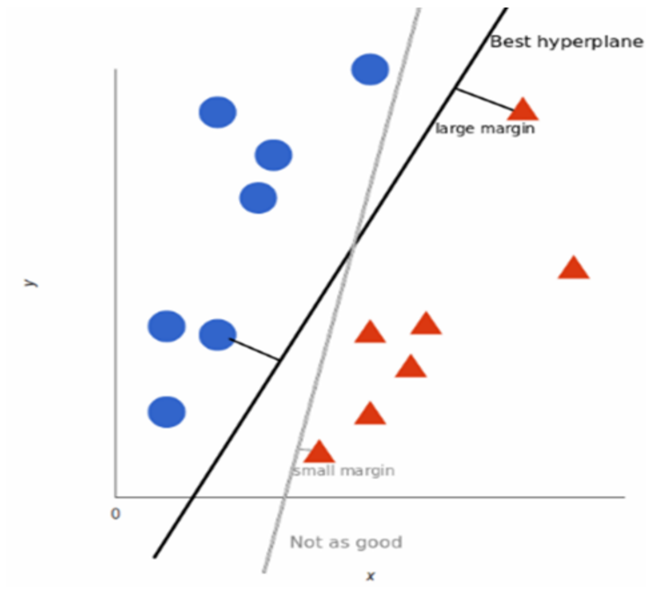
Note: LBP image processing algorithm doesn't require the images to be resized to a certain size. But for the sake of having all the images to be of uniform sizes we have resized them to 128 X 64 since the images in our dataset were all of dissimilar sizes.

1.3.3 Support Vector Machine(SVM)

SVMs are useful in solving both classification and regression problems. It is usually highly preferred as it provides significant accuracy and has less computation power. The main objective as to why we have used svm's is to accurately classify the data-points, in our case the features extracted from images belonging to distinct groups, one belonging to the Bengali signatures and the other to Hindi signatures.To classify we give an SVM model sets of labeled training data for each category, so that they're able to categorize new text.

Working of svm

We plot every data-item in a n-point dimensional space where n is the number of features we have considered and the value of each feature being the value of that particular coordinate. The next step is to perform classification in which we find the best hyperplane in the n-dimensional space that can classify them into their different groups. This hyperplane can be regarded as a decision boundary. This decides that if an item falls on the right it is classified into one group and on the left to another group although there can be some misclassifications.



The type of hyperplane that we choose depends on the data if it's linearly separable or non-linear separable. For example, in the image shown alongside the red and blue tags are linearly separable.

The kernel becomes useful for non linearly separable data items. If the data is not linearly separable then we use the kernel to transform our data items to higher dimensions and then classify them. Some of the popular kernels are: Linear, Polynomial, rbf.

Here for our classification model we have used the kernel polynomial for classifying our dataset.

1.3.4 Grid Search

Grid search is used to finetune appropriate hyperparameters for a model so that we could make the most accurate predictions as a result. A model hyperparameter is a characteristic of a model that is external to the model and whose value cannot be estimated from data. A parameter, on the other hand, is an intrinsic property of the model whose value may be determined using data. The value of the hyperparameter has to be set before the learning process begins. Some of the hyperparameters are: C (regularisation parameter), Kernel, Degree (to support poly kernel), Gamma (coefficient for rbf, poly, sigmoid kernel), Max_iter (maximum number of iterations for the solver) for Support Vector Classifier. In our study we have focussed on C and gamma.

The C parameter balances correct training example categorization with maximising the decision function's margin. A narrower margin will be acceptable for greater values of C if the decision function is better at accurately classifying all training points. A lower C encourages a bigger margin and, as a result, a simpler decision function, albeit at the expense of training accuracy. The gamma parameter determines how far a single training example's influence extends, with low values indicating "far" and high values indicating "near." The inverse of the radius of influence of samples chosen as support vectors by the model can be seen in the gamma parameters.

1.3.5 Fusion techniques

For signature classification we have used two fusion techniques namely Feature level fusion and Score level fusion. The objective of using fusion techniques was to increase the accuracy of the classification model by combining results of both LBP and HOG models.

1.3.5.1 Feature level fusion
Using appropriate feature normalisation, transformation, and reduction strategies, feature sets from diverse biometric sources are combined into a single feature set in feature-level fusion.In our model we had concatenated LBP and HOG feature vectors to create a single feature for all the images in the dataset. The fundamental benefit of feature-level fusion is that it identifies a compact set of prominent characteristics that might increase recognition accuracy by detecting associated feature values provided by distinct biometric methods. Feature-level fusion implies the availability of a large quantity of training data because obtaining this feature set often necessitates the use of dimensionality reduction methods. Template modification and enhancement can also be accomplished with feature-level fusion techniques.

1.3.5.2 Score level fusion
The match scores output by various biometric matchers are aggregated in score-level fusion to make a decision on an individual's identity. This consolidation technique usually culminates in the creation of a single scalar score, which is then employed by the biometric system. In our model we have averaged the prediction probability of LBP and HOG model to obtain the final prediction accuracy. Fusion at this level is the most widely discussed strategy in the biometric literature, owing to the simplicity with which match scores can be accessed and processed (compared with the raw biometric data or the feature set extracted from the data).

1.3.6 Classification report
We have mentioned classification reports obtained from various models in this report. The precision, recall, F1, and support scores for the models are displayed in these classification report visualizers. The report combines numerical scores with a color-coded heatmap to aid in simpler interpretation and problem discovery. To make it easier to compare classification models across multiple classification reports, all heat maps are in the range (0.0, 1.0). The different metrics in these reports are defined as:
(i) Precision: Precision can be regarded as a metric for how accurate a classifier is. It is calculated as the ratio of true positives to the sum of true and false positives for each class.
(ii) Recall: The capability of a classifier to accurately detect all positive cases is measured by recall, which is a measure of its completeness. It is calculated as the ratio of true positives to the sum of true positives and false negatives for each class.
(iii) F1 score: The F1 score is a weighted harmonic mean of precision and recall, with 1.0 being the highest and 0.0 being the lowest. F1 scores, on average, are lower than accuracy measurements since they include precision and recall in their computation. To compare classifier models, the weighted average of F1 rather than global accuracy should be used as a rule of thumb.
(iv) Support: The number of actual occurrences of the class in the provided dataset is known as support. Imbalanced support in the training data could reveal structural weaknesses in the classifier's reported scores, necessitating stratified sampling or rebalancing. Support does not alter depending on the model; instead, it diagnoses the evaluation process.

Experiment details and simulation results

2.1 HOG model

Here we accessed all the images in the dataset one by one and resized them to 128 X 64 and then called the HOG function to append the HOG feature vector and label of the image in an array data. Then this data was randomly shuffled, and features and labels were extracted. The train set and test set were split in the ratio of 7:3. The train set was then fed into the grid search with parameter range {'C': [0.1, 1, 10, 100, 1000], 'gamma': [1, 0.1, 0.01, 0.001, 0.0001], 'kernel': ['rbf']}. The grid search here would find the best fitting parameter and train the model with it. Then by fitting the test set in the obtained model we get the classification report.

	precision	recall	f1-score	support
0	0.93	0.89	0.91	1628
1	0.93	0.96	0.94	2584
accuracy			0.93	4212
macro avg	0.93	0.92	0.93	4212
weighted avg	0.93	0.93	0.93	4212

Here we can see that an accuracy of 93% is obtained for parameter values C=100, gamma=0.01, kernel=rbf.

We also tried changing the kernel to ‘poly’ to see the change in accuracy. Here the parameter range was {'C': [0.1, 1, 10, 100, 1000], 'gamma': [1, 0.1, 0.01, 0.001, 0.0001], 'kernel': ['poly']}.

	precision	recall	f1-score	support
0	0.93	0.92	0.92	1634
1	0.95	0.96	0.95	2578
accuracy			0.93	4212
macro avg	0.94	0.94	0.94	4212
weighted avg	0.94	0.94	0.94	4212

So, the accuracy of 93% was also obtained for the poly kernel. Here the best fitting parameters were C=0.1, gamma=1, kernel=poly.

2.2 LBP model

We started with the definition of LBP function that calculated the LBP value of a specific given pixel. Then here also we accessed all the images in the dataset one by one and resized them to 128 X 64 and then used a nested for loop called the LBP function for all the pixels of the image. Then the obtained 2-D array was flattened to get the LBP feature vector which was appended to the array data with the label of the chosen image. Then this data was randomly shuffled, and features and labels were extracted. The train set and test set were split in the ratio of 7:3. The train set was then fed into the grid search with parameter range {'C': [0.1, 1, 10, 100, 1000], 'gamma': [‘auto’], 'kernel': [‘poly’]}. The grid search here would find the best fitting parameter and train the model with it. Then by fitting the test set in the obtained model we get the following classification report.

	precision	recall	f1-score	support
0	0.90	0.91	0.90	1628
1	0.94	0.93	0.94	2584
accuracy			0.92	4212
macro avg	0.92	0.92	0.92	4212
weighted avg	0.92	0.92	0.92	4212

Hence, we were able to obtain an accuracy of 92% with C=0.1, gamma=’auto’, kernel=’poly’ hyper-parameter values.

2.3 LBP model + HOG model

2.3.1 Feature level fusion

Here also all the images in the dataset were accessed one by one and their respective feature vectors of LBP and HOG were concatenated and appended with the label of that image to the array data. All the images were resized to 128 X 64 before the concatenation to ensure that all the images are of uniform size. Then the data was shuffled and the features (here the features was the concatenation of LBP and HOG feature vector) and labels were extracted. The train set and test set were split in the ratio of 7:3. The train set was then fed into the grid search with parameter range {'C': [0.1, 1, 10, 100, 1000], 'gamma': [‘auto’], 'kernel': [‘poly’]}. The grid search here would find the best fitting parameter and train the model with it. Then by fitting the test set in the obtained model we get the classification report.

	precision	recall	f1-score	support
0	0.90	0.91	0.90	1624
1	0.94	0.94	0.94	2588
accuracy			0.92	4212
macro avg	0.92	0.92	0.92	4212
weighted avg	0.92	0.92	0.92	4212

So we have got an accuracy of 92% at the best fitting hyper-parameters C=1.0, gamma='auto', kernel='poly' reported by grid search for Feature level fusion.

2.3.2 Score level fusion

We tried to increase our accuracy by applying score level fusion which is done by following the sum rule, that is we calculated the probabilities for each data item separately for the HOG and LBP model and combined them by adding them and then dividing them by half. By doing so we are taking into account the results of both the models. We add the new results into a vector called predictionfinal and we test the accuracy against the test set. The classification report gave us the following results:

	precision	recall	f1-score	support
0	0.93	0.89	0.91	1574
1	0.94	0.96	0.95	2638
accuracy			0.94	4212
macro avg	0.94	0.93	0.93	4212
weighted avg	0.94	0.94	0.94	4212

So we achieved an accuracy of 94% for the Score level fusion of LBP and HOG models which had been trained with their respective best fitting hyper-parameters.

Conclusion

We were introduced to various ideas and methods of Machine Learning during our Summer Internship at CSIR-CEERI as Research Interns. Local Binary Pattern and Histogram of Oriented Gradients are some of the most popular image processing algorithms in Computer Vision and Support Vector Machine is one of the most prominent classifiers used in Machine Learning. We applied HOG and LBP to classify the dataset BHsig260 into Hindi and Bengali. When we first trained by SVM using the HOG algorithm on our dataset we obtained an accuracy of 93%. However when we applied the LBP algorithm and used SVM to classify, we got an accuracy of 92%. This would be because HOG deals with finding the edges of the object in the image whereas LBP is a texture descriptor which gives us information regarding the texture of the image. Now for signature classification, the edges of the signature in the image are more important than the texture of the image. Hence, we got better accuracy for the HOG model than the LBP model. Finally to increase the overall accuracy we performed the Feature level fusion that is we concatenated HOG and LBP feature vectors and then classified them using a SVM model and obtained an accuracy of 92%; on performing the Score level fusion that is adding the probabilities from both the HOG and LBP model and dividing them by 2 we obtained an accuracy of 94%. So from these results we can infer that Score level fusion is a better fusion technique for classifying the language dataset correctly. However the execution time for respective HOG and LBP models was better than either of the HOG-LBP combined models. At last, this internship provided us with hands-on experience with all of the procedures involved in working on a Machine Learning task from start to finish, and this competence will be invaluable for future Machine Learning and Deep Learning projects, as well as pursuing a career in this huge application-oriented industry.

References

1. Greeshma K V, Dr. J. Viji Gripsy, 2020, Image Classification using HOG and LBP Feature Descriptors with SVM and CNN, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NSDARM – 2020 (Volume 8 – Issue 04)
2. Machine Learning By Stanford University offered through Coursera.