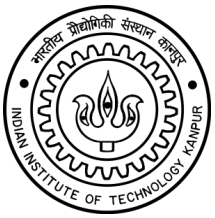
Indian Sign Language Character Recognition



Indian Institute of Technology, Kanpur

Course Project-CS365A

Sanil Jain(12616)
K.V.Sameer Raja(12332)

Mentor - Prof.Amitabha Mukerjee

Abstract

Our project is an effort towards studying the challenges in classification of characters in Indian Sign Language(ISL). A lot of research has been done in the corresponding field of American Sign Language(ASL), but unfortunately the same cannot be said for ISL. Lack of standard datasets, occluded features and variation in the language with locality have been the major barriers which has led to little research being done in ISL. Our project aims at extending a step forward in this field by collecting a dataset from a deaf school, and then use various feature extraction techniques to extract useful information which is then input into various supervised learning techniques. Currently, we have reported four fold cross validated results for the different approaches, and the difference from the previous work done can be attributed to the fact that in our four fold cross validation, the validation set correspond to images of a person different from the persons in the training set.

Contents

1 Introduction

This project aims at identifying alphabets in Indian Sign Language from the corresponding gestures. Gesture recognition and sign language recognition has been a well researched topic for American Sign Language(ASL), but few research works have been published regarding Indian Sign Language(ISL). But instead of using high-end technology like gloves or kinect, we aim to solve this problem using state of the art computer vision and machine learning algorithms.

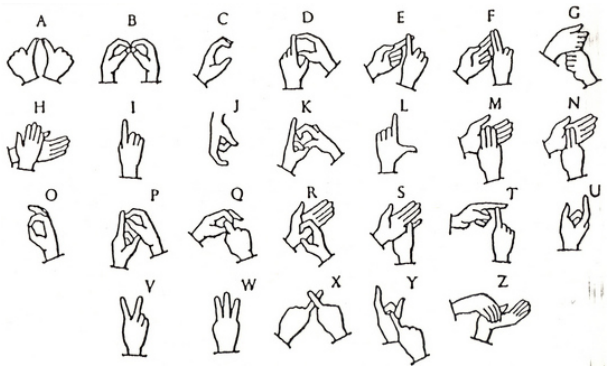


Figure 1: Indian Sign Language

src : http://www.deaftravel.co.uk/signprint.php?id=27

2 Motivation

Communication is one of the basic requirement for survival in society. Deaf and dumb people communicate among themselves using sign language but normal people find it difficult to understand their language. Extensive work has been done on American sign language recognition but Indian sign language differs significantly from American sign language.ISL uses two hands for communicating(20 out of 26) whereas ASL uses single hand for communicating. Using both hands often leads to obscurity of features due to overlapping of hands. In addition to this, lack of datasets along with variance in sign language with locality has resulted in restrained efforts in ISL gesture detection. Our project aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people using Indian sign language. Effective extension of this project to words and common expressions may not only make the deaf and dumb people communicate faster and easier with outer world, but also provide a boost in developing autonomous systems for understanding and aiding them

3 Challenges

The Indian Sign Language lags behind its American Counterpart as the research in this field is hampered by the lack of standard datasets. Unlike American Sign Language, it uses both hands for making gestures which leads to occlusion of features. ISL is also subject to variance in locality and the existence of multiple signs for the same character. Also some character share the same alphabet(E.g V and 2 have the same sign, similarly W and 3 have the same sign) and the resolution of the sign is context dependent.

4 Previous work

Little work has been done previously on ISL. One of the approaches included key point detection of Image using SIFT and then matching the keypoint of a new image with the keypoints of standard images per alphabet in a database to classify the new image with the label of one with the closest match [5]. Another one calculated the eigen vectors of covariance matrix calculated from the vector representation of image and used euclidean distance of new image eigen vector with those in training data set to classify new image [1]. Some of them used Neural networks for training but their dataset comprised of only single handed images and their feature vectors are based on the angle between fingers, number of fingers etc[4]. The major problem with all these works that there was no mention of where the dataset was collected from and from the images it appeared as if it was taken from webcam by the people working themselves. In a thesis for related work we found at [3], the authors had formed the dataset on one of their team members and divided that into training and testing sets before reporting the accuracy. Also they had tested only on a subset of the english alphabets. We didn't find their approach very appealing(especially the dataset collection part from just one person), and so under Prof. Amitabha Mukerjee Sir's advice, decided to collect the dataset ourselves from a deaf school in Kanpur.

5 Dataset

Previous Datasets : We couldn't find any reliable datasets on the internet as most of the videos of the sign language we found on the internet were of people who were demonstrating what sign language looked like and not those who actually spoke the language. Also the dataset found at [3] was taken from one of the team members and so was not reliable.
Approach: So, we went to **Jyothi Badhir Vidhyalaya**, a school for deaf located in a much remote section of Bithoor, Kanpur .We aimed at around 15, but due to time constraints could chose only 8 students of different skin tones and made around 6-11 seconds of video for every alphabet per person using a 30fps camera summing upto 1 minute video for every alphabet. While making the video we tilted and moved the camera to obtain images in different positions within the frame. We asked the students to show the most commonly used static representation for every alphabet(in case of multiple representations). We tried to make background as light as possible to make image segmentation easier. Later these videos were converted into frames evaluating to around 1800 images per alphabet.

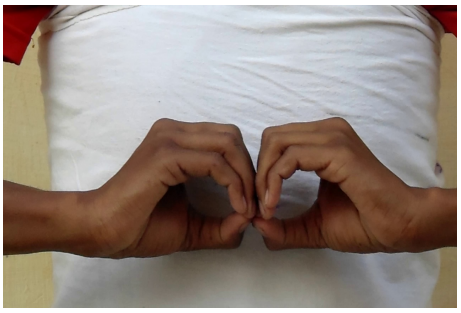


Figure 2: Original Image

Figure 3: Skin Dataset Image

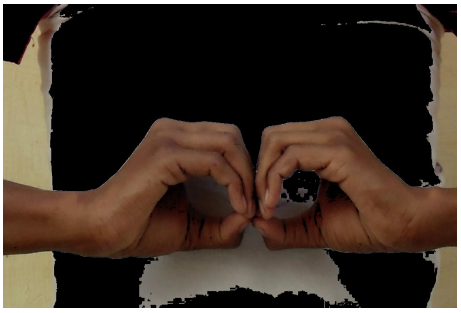


Figure 4: HSV Segmented Image

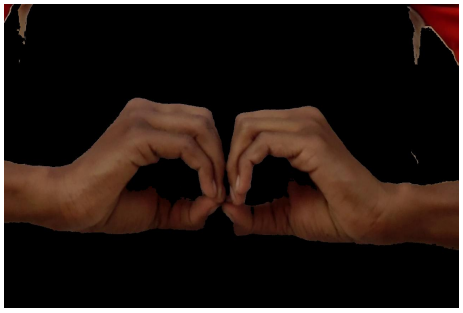


Figure 5: YUV-YIQ Segmented Image

6 Methodology

Having collected the dataset, we divided our approach to tackle the classification problem into three stages.

- The first stage is to segment the skin part from the image, as the remaining part can be regarded as noise w.r.t the character classification problem.
- The second stage is to extract relevant features from the skin segmented images which can prove significant for the next stage i.e learning and classification.
- The third stage as mentioned above is to use the extracted features as input into various supervised learning models for training and then finally use the trained models for classification.

6.1 Image Segmentation

6.1.1 Training on skin segmentation dataset

We used the skin segmentation dataset from UCI containing about 2,00,000 points for training using learning algorithms like SVM and RandomForest. The trained models are then used to segment out the non-skin classified pixels. But as shown from the images 2 and 3, this model performed abysmally, which as we observed later was due to the poor quality of the dataset.

6.1.2 HSV model

Unlike RGB model, HSV model separates the color and intensity components which makes it more robust to lighting and illumination changes.So in this approach, we transformed the image from RGB space to HSV space. We then retain the pixels having H and S values

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Figure 6: YIQ model

src :http://en.wikipedia.org/wiki/YIQ

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Figure 7: YUV model

src :http://en.wikipedia.org/wiki/YUV

in the range 25$<H<$230 and 25$<S<$230. to retain skin pixels and discard background. But the HSV model captures a lot of noise as shown in the images 2 and 4 and requires a lot of tuning. The authors [3] had used motion segmentation along with their HSV model to remove the background noise being captured. Since our images were of static signs, we couldn't do motion segmentation and hence had to modify our approach.

6.1.3 YIQ and YUV model(Final Approach)

So in our final approach, we transformed the image from RGB space to YIQ and YUV space (transformation calculation shown in figure 6 and figure 7) as done by the authors from [7]. From these models, we obtain I and $\theta = tan^{-1}(U/V)$ and we retain pixels with $30 < I <$100 and $105^\circ < \theta <150^\circ$. This approach works quite well as can been seen in images 2 and 5.

6.2 Feature Extraction

As describing our own features may not result in higher efficieny, we started with SIFT(Scale Inverse Feature Transform) features as it computes the key points in the image which is more apt than describing feauters manually. So, after the skin segmented images were obtained using the YUV-YIQ model, we used the following approaches for extracting feature vectors.

6.2.1 Bag of Visual words

Every image is treated as a document. To define the words for the image, we use SIFT feature detectors (shown in figure 8) to detect patches(or interest points) from the image and the SIFT feature descriptors return 128 dimensional vectors for each of the patches. Thus every image is a collection of 128 dimensional vectors.Now we convert these vector represented patches to codewords which produces a codebook(analogous to dictionary of words in text). The approach we use now is Kmeans clustering over all the obtained vectors and get K codewords(clusters) as shown in figure 9. Each patch(vector) in image will be mapped to the nearest cluster. Thus similar patches are represented as the same codeword and the whole image is now represented as a histogram of the codewords.In this histogram, the bins are the codewords and each bin counts the number of words assigned to the codeword. But the best results obtained using the feature vectors from this approach combined with linear kernel SVM was 32.74% and the confusion matrices suggested that similar looking alphabets were being misclassified entirely which suggested that many features were not being captured properly (The code for trying this approach was taken from :[6]).

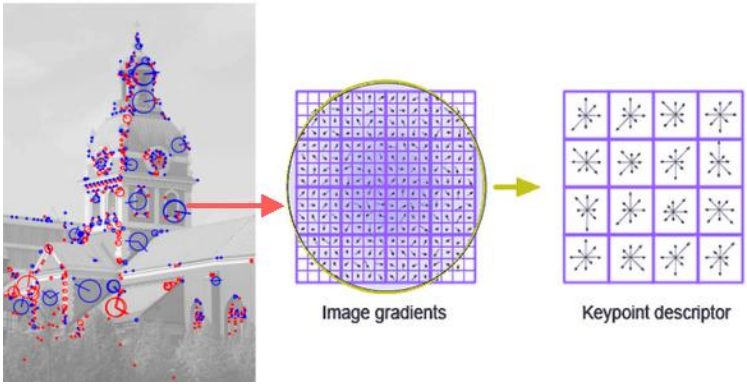


Figure 8: SIFT Features

src : http://www.codeproject.com/KB/recipes/619039/SIFT.JPG

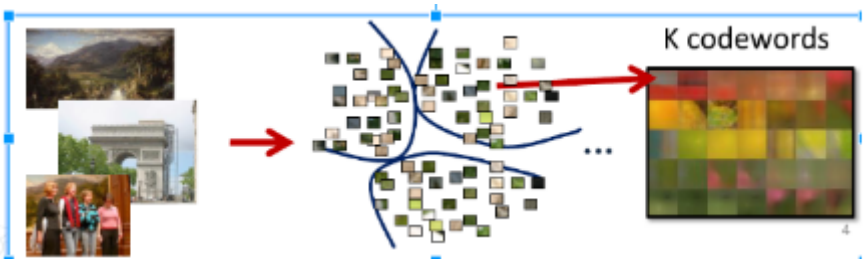


Figure 9: Bag of Visual Words

src : http://mi.eng.cam.ac.uk/ cipolla/lectures/PartIB/old/IB-visualcodebook.pdf

6.2.2 Histogram of Oriented Gradient(HOG) Features with dimension reduction

The previous approach suggested that we were missing features. And since our problem is plagued with the occurence of occluded features, and since SIFT algorithm considers only the keypoints in the image, it ends up losing information if it misses these occluded features in search for keypoints. This is where HOG features (see figure 10) come useful [2]. As per its wikipedia description, HOG features descriptors are similar to SIFT feature descriptors but differ in that they are computed on a dense grid of uniformly spaced cells and use overlapping local contrast normalization for improved accuracy. One problem with HOG features is their high dimension. So before calculating HOG features for the image, we reduced the dimension of the image to 240X135 while preserving the aspect ratio. Then we extracted the hog features of the images which returned us 32000 dimension vectors. Since we considered it too large, we used Gaussian Random Projection library in python to project the hog vectors into a 3000 dimensional subspace. On these vectors, we trained multiclass SVM with linear kernels which shot up the 4 fold CV accuracy to 51.43%.

6.2.3 Histogram of Oriented Gradient Features(without dimensionality reduction)

Encouraged by the previous results, we fed the entire hog feature vectors(of 32000 dimension) into the multiclass SVM with linear kernel. The 4 fold CV increased slightly to 54.63% which suggested that not a lot of information was lost during dimension reduction using Gaussian Random Projection.

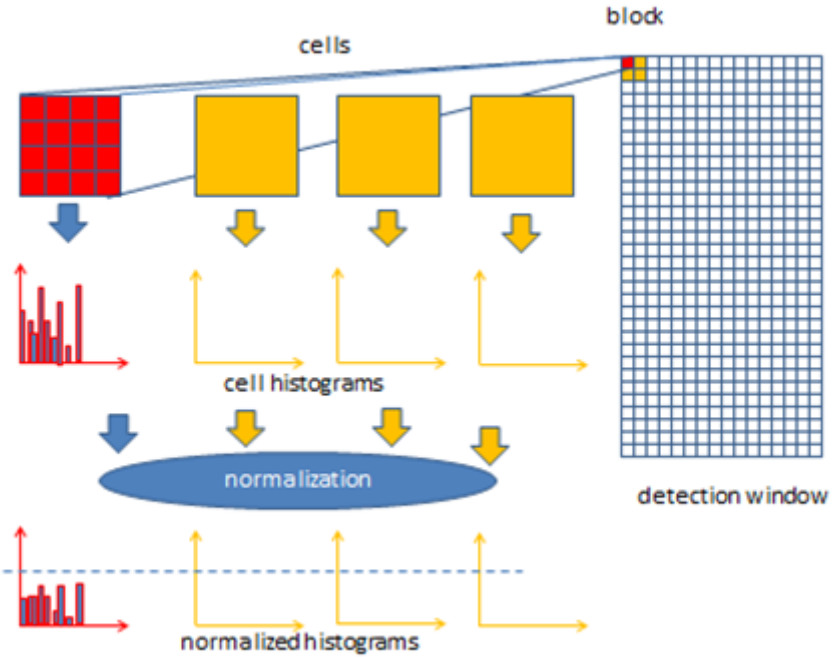


Figure 10: Histogram of Oriented Features

src : https://software.intel.com/en-us/node/529070

6.3 Machine Learning on Feature Vectors

The accuracies we reported in the previous subsection corresponds to the best results we obtained for each of the feature vectors. But before we obtained those best results, we explored the following algorithms on the obtained feature vectors. Multiclass SVM with linear kernel was used with almost every feature vector. Overall the following approaches were tried :-

- **Support Vector Machines** :- Multiclass SVMs were tried on all the feature vectors. Results obtained with linear kernel and four fold Cross Validated accuracies are reported for all feature vectors. The confusion matrices shown in the results sections below correspond to the different techniques tried using linear kernel MultiClass SVMs. The best accuracies were observed for this algorithm. Our try with 'rbf' kernel failed miserably on HOG feature vectors as only 4.76% accuracy was observed.

- **Random Forest** :- Since this was a 26 class problem, we tried our luck at Random Forest with HOG feature vectors on the compressed images. It fell a little short of the Multiclass SVM with 46.45% 4 fold CV accuracy. Our interpretation is that implementation of decision trees in python is monothetic which may have led to formation of rectangular regions whereas the actual boundaries may not have been rectangular.

- **Hierarchical Classification** One of the major reasons for the comparatively lower accuracy is the large number of classes(26). So one approach we tried was breaking the classification problem into multiple levels or hierarchical classification . First we train a linear kernel SVM model to classify alphabets as one handed or two handed.

This model performed with 95% accuracy. Then we trained linear kernel Multiclass SVM models to classify the one handed alphabets(56% accuracy) and two handed alphabets(60% accuracy) and then put the system together. An alphabet is first classified as one handed or two handed, and then depending on the classification, is put into the corresponding model and given a label. Even though the individual models performed better than the direct multiclass SVM on HOG features, overall the performance was nearly the same and four fold CV accuracy of 53.23% was observed.

7 Results

The results reported are four fold cross validated. The four fold cross validation in this instance is implemented as follows, we took about 25 images per alphabet per person from 4 people(the images of the remaining 4 were in very bad lighting and hence not trained upon), and then separated the images of one person as the validation set and trained on the images of the remaining 3 and measured the accuracy on the 4th. Finally the average accuracy is reported. Following are our approaches :

- We used SVM with linear kernel as training model for different features like Bag of Visual words , HOG features, Gaussian random projection of HOG features and their confusion matrices were shown in figure 13, figure 14, figure 15 respectively
- We tried to increase the performance by using Hierarchical clustering to seperate one handed image from two handed images and applied SVM with linear kernel on individual clusters. Their confusion matrix is shown in figure 16
- We also tried to use other training approaches like changing SVM kernel to rbf and using random forests for training. While the former one resulted in poor accuracy, latter one gave better accuracy than Bag of Visual Words model but not higher than multiclass SVM with linear kernel.
- The overall accuracies for the different approaches combined with different feature vectors are shown in the table 1 and a bar chart comparing them is shown in figure 12.

Feature Vector	**Learning Algorithm**	**Accuracy**(in %)
Bag of Visual Words	linear kernel SVM	32.74
Gaussian Random Projection	linear kernel SVM	51.43
Hog Features	Random Forests	46.46
Hog Features	RBF kernel SVM	4.63
Hog Features	linear kernel SVM	54.63
Hog Features	Hierarchical Classifier	53.23

Table 1: Results

8 Observations

- Since there are similar looking gestures for alphabets like (E,F) , (U,V,W) , (M,N) they are misclassified most number of times.

- Hierarchical Classification has done well on individual clusters but the overall accuracy is almost same as svm classifier as under the multiplicative law of probability, two serially connected systems with individually higher probability of accurate classification are likely to have lower overall probability of accurate classification .
- As there was a left handed student in our dataset the overall accuracy dropped by significant amount while that student is used as validation set(Accuracies reported here are 4 fold cross validated).
- While using Gaussian random projection to 3000 dimensional subspace, the accuracy will be decreased a bit which shows that there is slight loss of information.
- A better classification of clusters in hierarchical classifier might yield better results.
- A better dataset(with more variations , so that the model is more robust to the sign variations) would definitely improve the results.

9 Conclusions

Although there was some work done priorly on ISL, they generated the training and test dataset from the same person which lead to higher accuracies. We did four fold cross validation by using the images of 3 students for training and testing the model on 4th student which gave lower but more reasonable accuracies. Some images in the dataset were taken in bad illumination which gave noisy images while image segmentation. A better dataset would have helped in finding more precise features and would have resulted in higher accuracies.

Uploading the Dataset We will be soon uploading the dataset we have collected on a cse server so that it can be used by some other group if they decide to work on the same problem. We hope that in future we or somebody else can add to this dataset and the problem of lacking dataset problem can be solved.

Modularity Our current approach to solve this problem can be divided into four stages as shown in figure 11. The modularity in the approach encourages the fact that we can work individually on each of the modules to improve their individual performance, and thus a module can always be replaced by a better module to improve the overall accuracy of the chain.

10 Future Work

A dataset with more variation and a higher quality can really boost the accuracy of our current models. Also we think that using more complex models like artificial neural networks, or applying deep learning on the HOG vectors should improve the accuracy as they are capable of extracting richer information from these vectors. Increasing the levels of heirarchy with proper hierarchy levels created on the basis of which nodes are being misclassified can result in improvement in accuracy, but we can not surely claim that as suggested by multiplicative probability rule in our experiment with hierarchical classification.

11 Acknowledgement

We thank Prof. Amitabha Mukerjee, Dept. of Computer Science and Engineering for his valuable support throughout the project guiding us from time to time and looking into the project when it was needed. We also thank python community for their user friendly libraries.

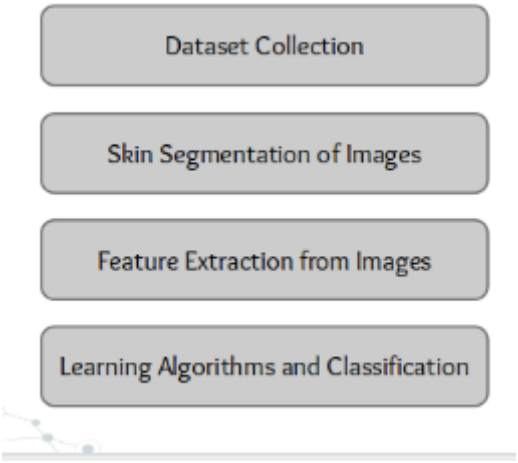


Figure 11: Modular Approach to ISL classification

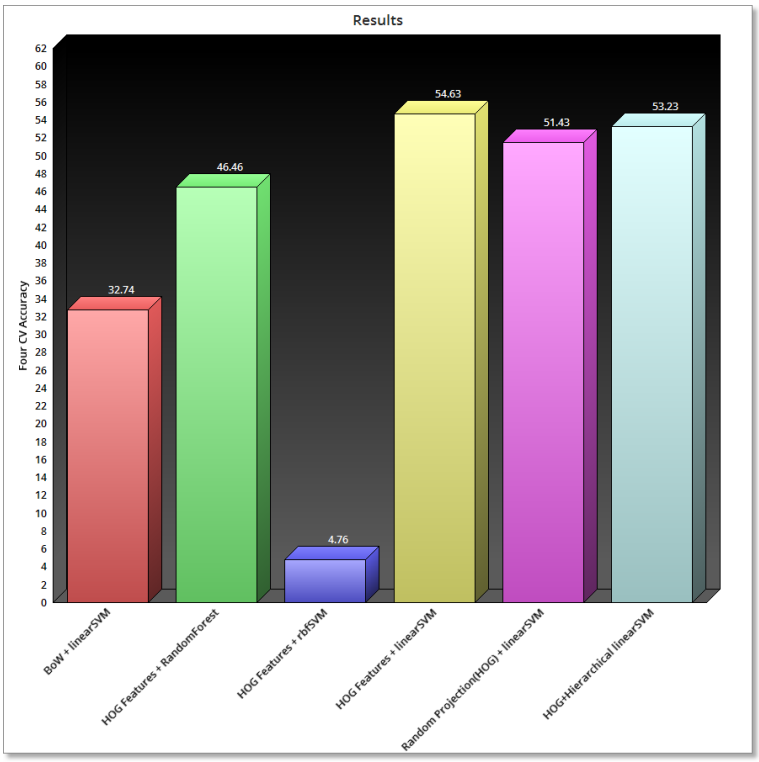


Figure 12: Accuracy plot

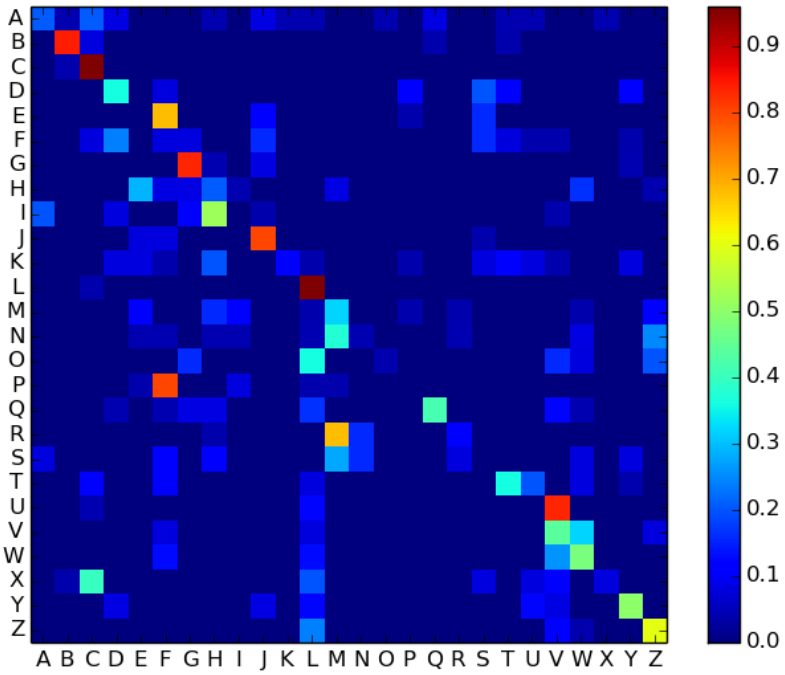


Figure 13: Bag of visual words+linear kernel SVM

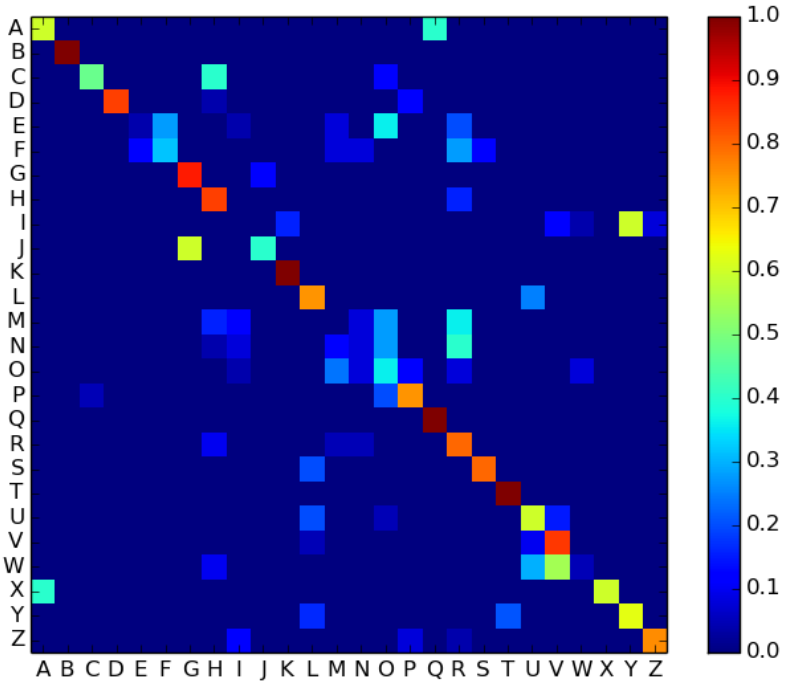


Figure 14: Hog+linear kernel SVM

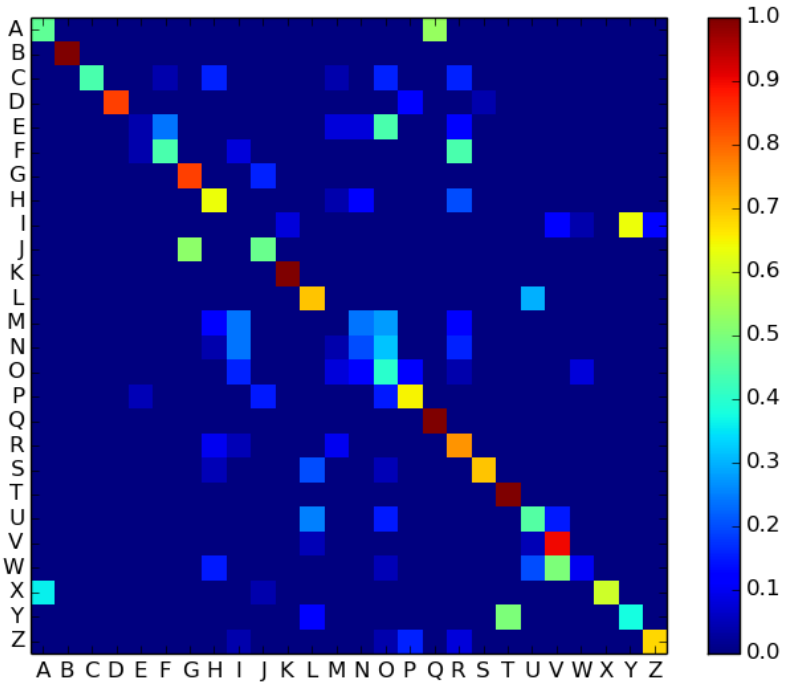


Figure 15: Gaussian Projection+linear kernel SVM

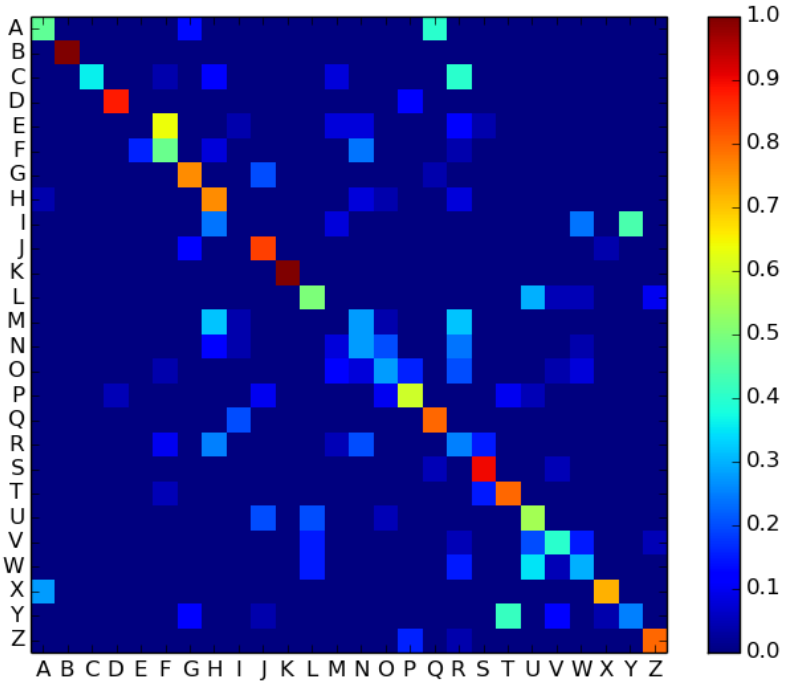


Figure 16: Hierarchical classifier+linear kernel SVM

References

[1] JOYEETA SINGH, K. D. Indian sign language recognition using eigen value weighted euclidean distance based classification technique. *International Journal of Advanced Computer Science and Applications 4*, 2 (2013).

[2] NEHA V. TAVARI, P. A. V. D. Indian sign language recognition based on histograms of oriented gradient. *International Journal of Computer Science and Information Technologies 5*, 3 (2014), 3657–3660.

[3] NMANIVAS. Gesture recognition system. `https://github.com/nmanivas/Gesture-Recognition-System`.

[4] PADMAVATHI . S, SAIPREETHY.M.S, V. Indian sign language character recognition using neural networks. *IJCA Special Issue on Recent Trends in Pattern Recognition and Image Analysis, RTPRIA* (2013).

[5] SAKSHI GOYAL, ISHITA SHARMA, S. S. Sign language recognition system for deaf and dumb people. *International Journal of Engineering Research Technology 2*, 4 (April 2013).

[6] SCHAKENBERG. Schakenberg code for bag of visual words on github. `https://github.com/shackenberg/Minimal-Bag-of-Visual-Words-Image-Classifier`.

[7] XIAOLONG TENG, BIANI WU, W. Y., AND LIU, C. A hand gesture recognition system based on local linear embedding, April 2005.