INM460 Computer Vision - Coursework Report

Daniel Sikar - MSc Data Science PT2

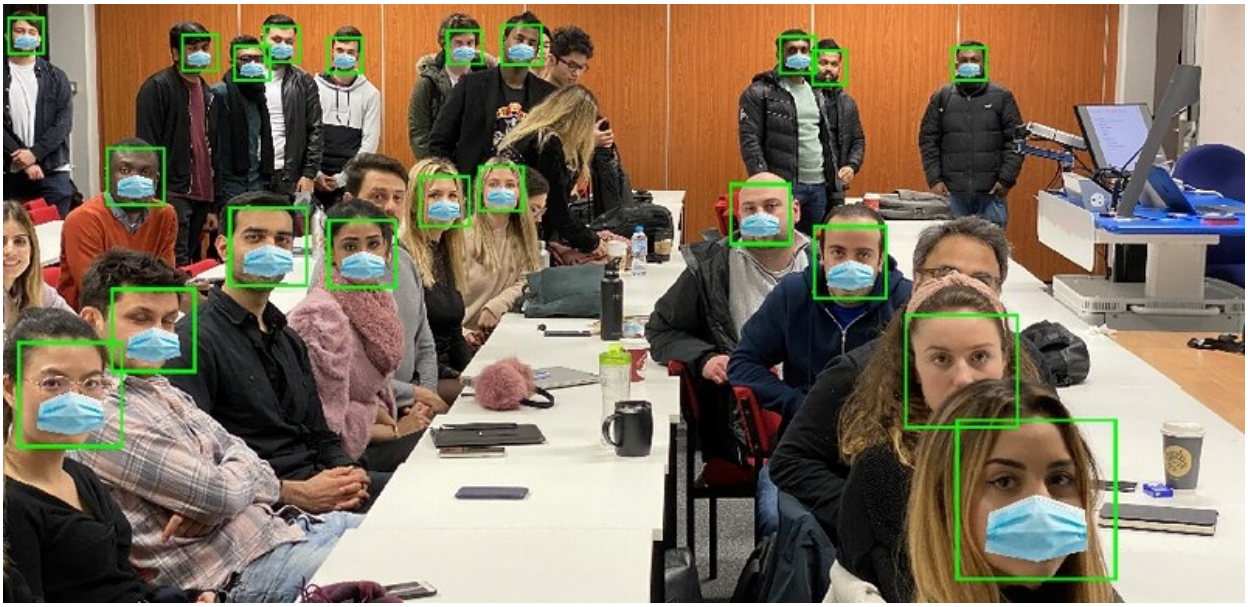

Fig. 1. Creative Mode processing of INM460 coursework classroom image

Keywords—Person Identification, CNN, HOG, SURF, SVM

1 Overview

This INM460 Computer Vision coursework project consists of creating facial classifiers from supplied image stills and videos of two types; individual and group, labelled and unlabelled respectively as shown in Fig. 2.

Fig. 2. Example of individual (labelled) and group (unlabelled) images

Each individual image contains a sign with a two-digit unique identifier, from which labelled data is obtained to train and test classifier models.

1.1 Pre-processing Steps

For our coursework, the chosen programming language was MATLAB [4], version 2018a. All pre-processing steps were sequentially run in script **ImagePreProcessing.m**, using MATALB user-defined functions:

1. fnSaveVideoStills
2. fnLabelIndividualImages
3. fnBlurImages
4. fnRotateImages
5. fnCropFaces

In step 1. all videos were saved as stills, with the exception of beginning and ending frames of each video, which were normally black or excessively blurred. In step 2. all images containing unique identifiers which could be recognised with OCR with our function **fnGetStudentID** were moved to a labelled folder, otherwise set aside for manual labelling. Once all images had been identified and moved to labelled folders, totalling 48 labels, we obtained an average of 402 images per label. In steps 3. and 4. the entire data then was augmented by randomly selecting 50% of each label and applying a random amount of blur to each image, then a further 70% was randomly selected from each label (including recently added blurred images) and randomly rotated from -10 to 10 degrees.

The augmented data after step 4. had an average of 1026 images per label. In Step 5. faces were cropped, using MATLAB's vision.CascadeObjectDetector which implements the Viola-Jones algorithm [7], and moved to another set of labelled folders. An average of 985 faces were obtained from each label, which represents approximately 96% of the augmented data. Each label was then manually checked and any images not correctly cropped were removed, resulting in an average of 815 images per label, doubling the size of face image data with respect to stills obtained in step 2. Table 1 shows label quantities after each processing step, including totals and averages per label. Fig. 3 shows examples of three correct face crops, without and with augmentation, and one shoe incorrectly identified as a face.

1.2 Determining optimal cropped-face size

To determine what image size to use for training our models, we examined individual and group images. Files resulting from pre-processing step 5. were square, with varying side dimensions. We used function

• *Daniel Sikar, City University of London, daniel.sikar@city.ac.uk*

Pre-processing figures					
Step	2	3	4	5	Final
Total images	19323	28984	49273	47302	39135
Label avg.	402	604	1026	985	815

Table 1. Label quantities after each pre-processing step

Fig. 3. Cropped face images including augmented rotated, augmented blurred and misidentified

fnStatsCropSizes.m to calculate the average minimum, maximum, median and mode square side sizes as show in table 2.

Individual face image size size statistics			
Minimum	Maximum	Median	Mode
100	252	171	152

Table 2. Rounded average size in pixels for pre-processed images

We then looked at faces found in group images, both still images, and still images obtained from videos. The latter proved mostly unusable, as the amount of blur in the images did not allow for more than 2 or 3 face images being identified in each image, so group videos were discarded. We used function **fnStatsGroupCropSizes.m** to identify face sizes in still group images and obtained cropped face statistics in table 3.

Since the median side size from faces cropped from group images was 128 and the mode was 107, and group images were the unlabelled data we are looking at classifying, intuitively it seemed to make sense to resized all labelled images closer to the minimum end of the size range, which would require the least amount of resizing to train and test our models, particularly aiming at avoiding to increase the size of cropped faces from group images, introducing some amount of blur. As an example, figure 4 shows a sequence were an image of size 217x217 pixels is decreased to size 115x115, with no loss of sharpness, then resized again to 227x227, with some loss of sharpness.

We used function **fnGetSampleImages.m** to generate sample data for initial models, and based on the results, expanded to full data. We worked with a random sets of 50, 100 and 200 images from 8 labels, in three different sizes 227x227, 128x128 and 70x70 pixels, both RGB and grayscale, and then created a few models to test our hypothesis.

Since a visual inspection showed numerous repetitions in each label, a sample size of 200 images per label was considered sufficient to generate the classifiers.

2 Description of Implemented Methods

Three face recognition feature-classifier types were implemented; CNN, HOG-SVM and SURF-SVM, mostly based on code covered in labs, which some adaptations for HOG-SVM and SURF-SVM, as well as CNN for grayscale images. A gridsearch was run for each method, aiming to find optimal image sizes, colour schemes and some parameter tuning for HOG-SVM and SURF-SVM. Script **TrainNetworks.m** was used for the grid searches and to generate classifiers, confusion matrices and statistics, by looping through all parameters being tested for each classifier. Best performing classifiers for labelled individual image dataset, tested on 980 images, are displayed in table 4 We then tested each classifier on 5 group images. We used function **RecogniseFace** which identified a total of 101 faces, then computed the accuracies manually. Results are shown in table 5.

Group face image size size statistics			
Minimum	Maximum	Median	Mode
100	322	128	107

Table 3. Rounded average size in pixels for pre-processed images

Fig. 4. Sequence showing the effect of decreasing and increasing cropped-face images sizes, from left to right, image is decreased with no apparent loss in sharpness, then increased with some loss in sharpness

2.1 CNN

Our Convolutional Neural Network image classifier was our best model by a narrow margin. We do not include a block diagram of the key algorithmic components as the image processing was quite simple and limited to augmentation consisting of blurring and rotation.

Our network was based on lab code [6]. Some alternative network architectures were tried, with added layers, different learning rates and epochs. We could not improve on the network given in class and in the end used it without modification, that is, two stacks of convolutional, batch normalization, rectified linear unit and max polling layers, followed by the same stack excluding max pooling layer, followed by a fully connected, softmax and classification layer. For grayscale images, we changed the dimensions of the image input layer, to account for the single colour channel.

We varied training epochs from 2 to 32 and found that 4 was an adequate number (see figure 5), as the network had usually converged by then to a stable accuracy value that was unlikely to change with the addition of more epochs.

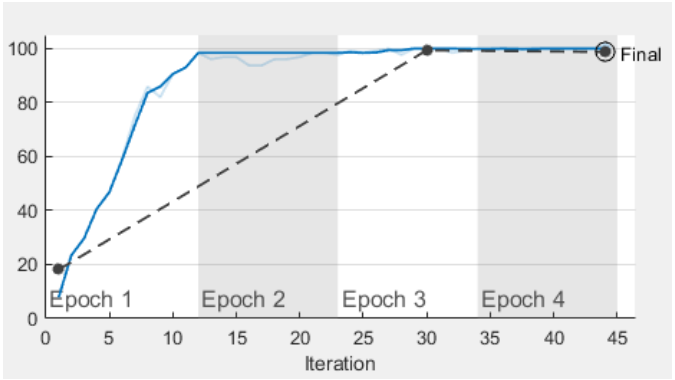

Fig. 5. CNN training progress showing conversion after three epochs, for 200 samples per label, 128x128 pixel RGB images

We used function **fnTrainCNN** to train our networks. This returned the trained network and accuracy as output parameters, used to build table 6 in section A. The outputs showed that our networks worked better with smaller images, consistently generating higher accuracies for grayscale image sizes of 70x70 pixels. Overall, networks trained using grayscale images performed slightly better than the RGB counterparts on individual images, which suggests variability such as introduced by differing light conditions might have been attenuated in grayscale images.

We tried using grayscale images as a single channel input to our

Accuracy of face recognition models		
Feature-Classifier	Name	Accuracy
CNN	dan_net_70_rgb	0.9990
HOG-SVM	HOGSVMMdl_70x70	0.9990
SURF-SVM	SURFSVMMdl_70px_50pt	0.8656

Table 4. Accuracy table for labelled individual images

Accuracy of face recognition models		
Feature-Classifier	Name	Accuracy
CNN	dan_net_70_rgb	0.1386
HOG-SVM	HOGSVMMdl_70x70	0.1287
SURF-SVM	SURFSVMMdl_70px_50pt	0.1188

Table 5. Accuracy table for group images

convolutional neural network as they trained faster than 3 channel RGB images. We tested our grayscale image network on one group image (class2.jpg), using our **RecogniseFace** function. One in twenty images was classified correctly. Compared to 4 in 20 using our best RGB model. The reason for the lower performance of grayscale images on group images may be related to further processing being required in group images, as performance on individual images was better for grayscale than RGB images. We discuss this in section 3.

Based on ad hoc testing and grid search results, 70x70 pixel RGB images were chosen to train the coursework CNN models. Figure 6 shows a detail of image generated by our **RecogniseFace** function with one correct and one incorrect classification

Fig. 6. Correct (left) and incorrect (right) classification labels from CNN trained with 70x70 pixel rgb images

2.2 HOG-SVM

For our HOG-SVM feature type / classifier combination we used locally normalized Histogram of Oriented Gradient (HOG) descriptors [2] as the feature type and a multiclass model for support vector machine as the classifier. The method takes an input image which is normalized for gamma and colour for better invariance to factors such as illumination and shadowing, then a grid is placed on the image and for every cell a Histogram of Oriented gradient features is extracted, the concept being local object appearance and shape are characterized by a distribution of intensity gradients or edge positions. Each cell holds a 1-D histogram of gradients and edge orientations over the pixel of the cell, noting that the referenced work applied to human detection in general.

We defined function **fnCreateHOGSVMClassifier** to create a multiclass model for support vector machine using matlab function **fitcecoc**, and script **TrainNetworks.m** to test different classifiers, using code supplied in lab 6 [5] with some modifications. Data was split at a ratio of 90%/10% for training and testing. Training data was then run through the feature extraction function **extractHOGFeatures**, which generated our encoded shape information. Since we had to somehow turn the features into a vector to be used by our Support Vector Machine classifier, we chose to use grayscale images as the intuition was this approach would constrain the task and make it more manageable - see section 3. We tested different cell sizes; [50 50], [25 25] and [10 10], finding that smaller cell sizes generated a larger number of HOG features, as well as higher accuracies. The data generated from our grid search is shown in table 7, section A. Given the feature length was not necessarily the same for each image and smaller cell sizes generated more features, we defined a value of 2500 features, which is most cases exceeded the number of features extracted for each image. We then used this value to adjust the size of the vector that would be used to train our SVM. If the feature vector returned by **extractHOGFeatures** was greater, we shortened the vector, and if it was less, we padded the feature vector with zeros, to ensure the final feature vector was always equal to our defined feature size.

The same process, as implemented in function **fnHOGSMVPredict**, was used when classifying unseen group images. HOG features were extracted, vector size adjusted before being passed to the SMV classifier.

Our best HOG-SVM model accuracy was 0.9990, as shown in table 4, section 2. Since we had a total of 48 labels, the resulting confusion matrix was not readable, so we created a subset with 8 labels with the same 200 images for each label, generating the confusion matrix shown in figure 7 with approximately the same accuracy.

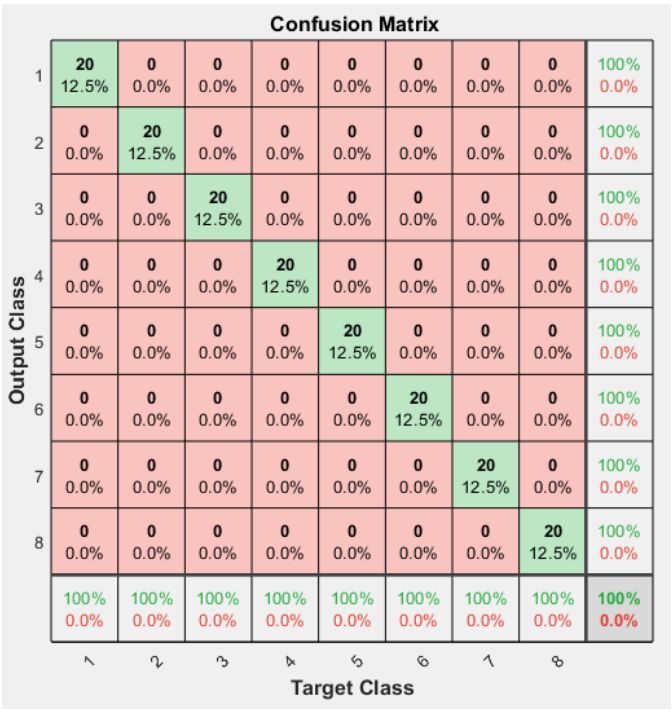

Fig. 7. HOG SVM Confusion Matrix, 8 labels, 200 images each, 90%/10% train/test split

The widely different accuracy obtained from testing individual labelled images and testing group images suggests the model is overfitting. We discuss this in section 3.

2.3 SURF-SVM

SURF uses the concept of interest points (as covered in lecture 5) as a means to build a set of features that will later help establish a correspondence between two images. SURF uses windows (somewhat analogous to HOG cells), which are divided into subregions, and further subdivided into grids. By using Haar wavelets pixel intensity sums, a 4-number descriptor is obtained to define a subregion [1].

We used MATLAB functions **detectSURFFeatures** to obtain SURF features and **extractFeatures** to obtain descriptors. Once again we

observed that different images generated different feature numbers, so applied the same principle used to generate our HOG-SVM classifier. We used script **TrainNetworks.m** and function **fnCreateSURFSVMClassifier** to try out different parameters, results shown in table 8, section A. We opted to use 50 SURF features with 200 70x70 grayscale images, as RGB images could not be used. This was not our best performing model but for the sake of having a single data for the HOG-SVM and SURF-SVM classifiers we sacrificed SURF-SVM accuracy. See section 3 for discussion.

To generate the matrix to fit our multiclass models for our support vector machine, we used the same approach as with our HOG-SVM, except the final vector adjustment had a slightly different implementation, to account for the 64 dimensions used in SURF.

To run our SURF-SVM classifier we used function **fnSURFSMVPredict**, again using the same principle as with the HOG-SVM classifier, where the same SURF feature number is used to make predictions, as was used to fit the classifier.

The final SURF-SVM classifier accuracy obtained from fitting a model with 50 SURF features, 48 labels and 200 70x70 pixel grayscale images was 0.90625 as shown in table 8, section A. The confusion matrix for our best model was unreadable due to the amount of rows and columns, so we generated a second confusion matrix with 8 labels only, as shown in figure 8. The accuracy obtained for the 8-label model 96.3%. This is assumed to be an artifact, due to individual labels presenting different classification accuracies,

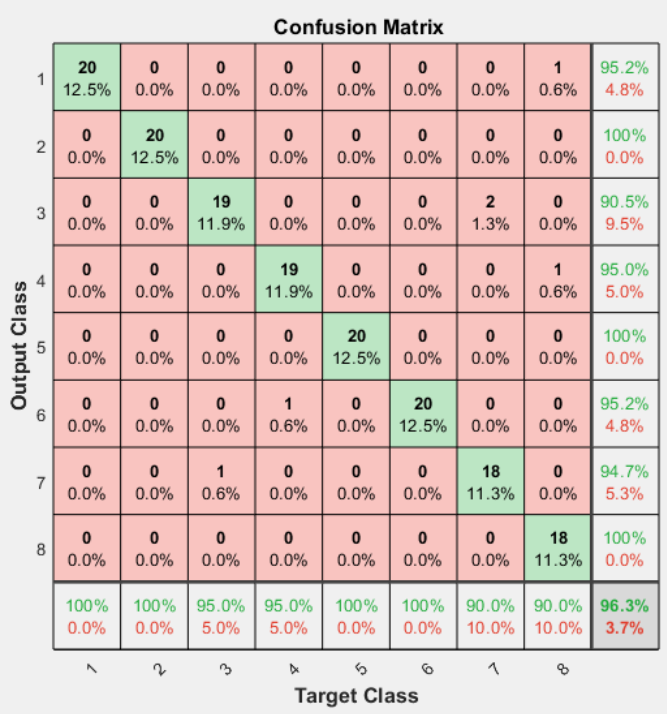

Fig. 8. SURF SVM Confusion Matrix, 8 labels, 200 images each, 90%/10% train/test split

2.4 Creative Mode

For the creative mode task we chose to place a face mask on every face found in an image. This was implemented in function **fnPandemicMode**.

We used the **roipoly** method to define a polygonal region of interest in the original face mask image (facemask-cropped.jpg). we then scaled the face mask image and the polygonal region of interest with respect to cropped face image size. We then created "positive" and "negative" segmentation mask matrices, to add or remove the face mask image or region as required. The dot product between the cropped face image and the negative mask matrix removed the face mask region, all black colour values in that region being equal to zero. This resulting matrix was then added to the positive mask, containing the face mask image surrounded by a blacked out region.

To define where the mask would be placed, our initial approach was to use a mouth locating feature from MATLAB object **vision.CascadeObjectDetector**. We did not find this approach robust, as returned mouth locations varied between nose and chin. In the end we found it simpler to display the mask at a fixed ratio from the bottom edge and mid-point between side edges of cropped face.

On the top row of figure 9, the middle image represents the negative segmentation mask. A dot product with the cropped face on left-hand size image creates a black void in the face mask area on right-hand side image. On the bottom row, the middle image represents the positive segmentation mask, which is added to "voided" left-hand side image, resulting in void being filled by face mask on right-hand side image. These were the concepts used for prototyping, in practice we removed a section from cropped face, performed the matrix operations and inserted the resulting section back into the original image.

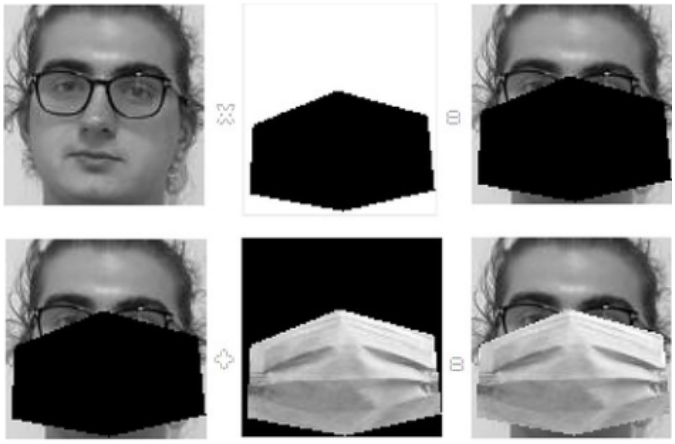

Fig. 9. Creative Mode superimposing facemask on face image

3 Discussion

This coursework project had a variety of tasks which required a good level of organisation, from data pre-processing onwards. Therefore it also turned out an exercise in time management and resource allocation.

Organising the image training sets, which seemed to be a minor task, turned out to be major, and provided an appreciation for what projects like ImageNet [3] must have been like.

The original SURF paper [1] does account for RGB images being used though the MATLAB implementation of detectSURFFeatures only accepts grayscale image. Future work with the HOG-SVM and SURF-SVM classifiers could include using RGB images to generate HOG and SURF features, by encoding each channel separately and to a fixed length channel vector as per single channel grayscale images, then stacking channels sequentially, such that the vector would always have the channel features in the same vector indices.

One of SURF's strengths is finding descriptors invariant to orientation change, which we could have tested by not rotating images in our training set, then using rotated images in our test set. Under such conditions, we would expect that SURF-SVM might gain on the CNN and HOG-SVM classifiers.

We did not use the best possible SURF-SVM model for the sake of using a common set of images for both HOG-SVM and SURF-SVM.

The different accuracy obtained by testing on unseen individual images and unseen group images suggests overfitting may have occurred. We tested different splits for SURF-SVM models (which presented the lowest accuracy) of 80/20 and 70/30 without substantial change in accuracy observed for testing data. We assumed that, if all three classifiers, CNN, HOG-SVM and SURF-SVM, were indeed overfitting, one cause may be there is a limited amount of variability that can be

introduced by augmentation, and ending up with 200 sample images for every label may be the cause. Further splits would be required to determine if this was indeed the case. The expectation being, if an unorthodox split of 30/70 was tried and the accuracies for individual image classification were still high, this would confirm the hypothesis.

Also, in terms of training/testing splits, K-fold cross-validation might help with establishing if the data was somehow impoverished by our augmentation.

Another cause of the model accuracy discrepancy between test individual and group images, may be due to group images not being representative enough. This may be a shortcoming of our processing pipeline, that might have benefited from trying to approximate individual and group image characteristics such as contrast.

We were not able to implement a confidence measure for HOG-SVM and SURF-SVM classifiers, so all labels generated by both classifiers have an ID, unlike our CNN classifier which will show ID according to a confidence measure.

Adding a confidence measure to HOG-SVM and SURF-SVM classifiers could be generated by a using measure of distance to the classification boundary, and further refinements, such as the use of multiple models and "voting" schemes could be one option, whereby the output labels from each model would be grouped and summed and the highest number chosen as the output label.

Producing the HOG-SVM and SURF-SVM classifiers was the highlight of this project and one of the highlights of my time at City so far, as it is the culmination of practical and theoretical knowledge obtained since joining the programme. The fact that HOG-SVM matched the CNN classifier in performance and the SURF-SVM did quite well was a surprise in the true spirit of research, where the outcome is never certain.

ACKNOWLEDGMENTS

The author is grateful for the support provided by Giacomo, Alex and Olga.

REFERENCES

[1] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. Similarity Matching in Computer Vision and Multimedia. doi: 10.1016/j.cviu.2007.09.014

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1:886–893, 2005.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[4] MATLAB. *version 9.5.0 (R2018b)*. The MathWorks Inc., Natick, Massachusetts, 2018.

[5] Unknown. *INM460 / IN3060 Computer Vision Lab 6 solutions*, 2020 (May, 2, 2020).

[6] Unknown. *INM460 / IN3060 Computer Vision Lab 8*, 2020 (May, 2, 2020).

[7] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, pp. I–511–I–518 vol.1, 2001. doi: 10.1109/CVPR.2001.990517

A APPENDIX

We include tables generated from data obtained during the grid search training and testing phases, output by script **TrainNetworks.m**, as well as data obtained manually by running unseen group images (included in .zip project file) though our RecogniseFace function.

Table 6 shows accuracies obtained for each combination of images type and sample size for CNN classifiers.

Table 7 shows accuracies obtained for combination of grayscale image and sample size, HOG feature size and cell size for HOG-SVM classifiers.

Table 8 shows accuracies obtained for combination of grayscale image and sample size and SURF feature size for SURF-SVM classifiers.

Table 9 shows the classification results obtained manually and used to compute model accuracy for group images.

CNN accuracy for different image parameters			
Images	Size	Colour Scheme	Accuracy
50	227x227	rgb	0.875
50	128x128	rgb	0.75
50	70x70	rgb	1
50	227x227	grayscale	0.8
50	128x128	grayscale	1
50	70x70	grayscale	1
100	227x227	rgb	0.1375
100	128x128	rgb	0.375
100	70x70	rgb	0.9
100	227x227	grayscale	0.275
100	128x128	grayscale	0.7375
100	70x70	grayscale	0.925
200	227x227	rgb	0.71875
200	128x128	rgb	0.99375
200	70x70	rgb	1
200	227x227	grayscale	0.14375
200	128x128	grayscale	0.95
200	70x70	grayscale	1

Table 6. Optimal image-size search for face classification CNN

HOG-SVM classifier accuracies				
Images	Size	HOG features	Cell Size	Accuracy
50	227x227	1500	[50 50]	1
50	128x128	1500	[50 50]	0.675
50	70x70	1500	[50 50]	0.125
50	227x227	2000	[25 25]	1
50	128x128	2000	[25 25]	1
50	70x70	2000	[25 25]	0.775
50	227x227	2500	[10 10]	0.825
50	128x128	2500	[10 10]	1
50	70x70	2500	[10 10]	1
100	227x227	1500	[50 50]	0.9875
100	128x128	1500	[50 50]	0.65
100	70x70	1500	[50 50]	0.125
100	227x227	2000	[25 25]	0.9875
100	128x128	2000	[25 25]	0.9875
100	70x70	2000	[25 25]	0.6125
100	227x227	2500	[10 10]	0.7375
100	128x128	2500	[10 10]	0.9625
100	70x70	2500	[10 10]	1
200	227x227	1500	[50 50]	1
200	128x128	1500	[50 50]	0.65625
200	70x70	1500	[50 50]	0.125
200	227x227	2000	[25 25]	0.99375
200	128x128	2000	[25 25]	0.99375
200	70x70	2000	[25 25]	0.775
200	227x227	2500	[10 10]	0.8625
200	128x128	2500	[10 10]	0.9875
200	70x70	2500	[10 10]	1

Table 7. Optimal parameter search HOG-SVM classifier

SURF-SVM Classifier Accuracy Table			
Images	Size	SURF features	Accuracy
50	227x227	40	0.525
50	227x227	50	0.375
50	227x227	60	0.325
50	128x128	40	0.575
50	128x128	50	0.675
50	128x128	60	0.6
50	70x70	40	0.875
50	70x70	50	0.95
50	70x70	60	0.85
100	227x227	40	0.925
100	227x227	50	0.9
100	227x227	60	0.9
100	128x128	40	0.925
100	128x128	50	0.9375
100	128x128	60	0.825
100	70x70	40	0.8625
100	70x70	50	0.825
100	70x70	60	0.8625
200	227x227	40	0.9
200	227x227	50	0.9125
200	227x227	60	0.9
200	128x128	40	0.86875
200	128x128	50	0.8875
200	128x128	60	0.89375
200	70x70	40	0.89375
200	70x70	50	0.90625
200	70x70	60	0.85

Table 8. Optimal parameter search for SURF-SVM classifier

Classifier accuracy for group images					
Model	Faces	Correct	Incorrect	No ID	Image
CNN	19	2	9	7	class1.jpg
CNN	20	4	13	3	class2.jpg
CNN	17	4	9	4	class3.jpg
CNN	24	1	14	9	class4.jpg
CNN	21	3	13	5	class5.jpg
HOG-SVM	19	4	15	N/A	class1.jpg
HOG-SVM	20	4	16	N/A	class2.jpg
HOG-SVM	17	2	15	N/A	class3.jpg
HOG-SVM	24	1	23	N/A	class4.jpg
HOG-SVM	21	2	19	N/A	class5.jpg
SURF-SVM	19	1	18	N/A	class1.jpg
SURF-SVM	20	3	17	N/A	class2.jpg
SURF-SVM	17	3	14	N/A	class3.jpg
SURF-SVM	24	2	22	N/A	class4.jpg
SURF-SVM	21	3	18	N/A	class5.jpg

Table 9. Values used to compute best-model classifier accuracy