CSC3061: Group Assignment

Face Recognition and Detection

Group 15

Lee Husain	40172738	lhusain01@qub.ac.uk
Conchur McCullough	40178986	cmccullough29@qub.ac.uk
Thomas Pickup	40145342	tpickup01@qub.ac.uk

Table Of Contents

Training

Image pre-processing

Because the provided images have faces of different complexions and under different lighting conditions, we decided to apply contrast enhancement to each image. There are several methods to enhance the contrast of an image. The simplest way is linear stretching, which stretches the entire histogram of the image to use the full range of pixel values. This uniformly increases contrast for the whole image and does not target low contrast regions.

The power law function is a contrast enhancement method which raises the input pixel values to the power specified by the parameter *gamma.* A gamma value greater than 1 enhances contrast in the brighter regions of the image, while a gamma of less than 1 increases the contrast in dark regions. While this can be useful for an individual image where the contrast can be compared visually, it may be impractical to use for a large number of images where the optimal gamma value may be different for each one.

Histogram equalisation aims to make the image histogram as flat as possible by stretching only the peaks. Figure 1 shows an example of a cropped face image before and after histogram equalisation. We decided to use this method as it is more appropriate for a large set of images. While the power law function may actually decrease the contrast of some images if the wrong gamma is used, histogram equalisation is guaranteed to improve the overall contrast of all images, making features more easily detectable.

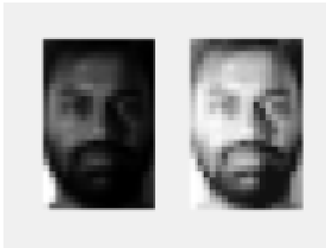

Figure 1: Face image with contrast enhanced by histogram equalisation

Feature description methods

The easiest way to train our model is to directly use the full image's pixel values as its training data. However there are many methods to describe features in an image region, and a more accurate model can be achieved by training the model using a feature vector instead of raw pixel values.

Histogram of oriented gradients (HOG) is one such feature description method. It involves breaking the image down into separate cells and computing the orientation of the gradients in each cell. HOG vectors are useful for identifying objects by characterizing the edges present in the image.

Gabor features are another feature description method. A Gabor filter is a sinusoidal signal that is used with different orientations and frequencies in each image region, to describe the textures in that region. Many researchers believe Gabor filtering is similar to the way humans process visual stimuli. Although this method is an effective way to extract features from an image, Gabor feature vectors are usually very large. Each 18x27 pixel cropped image produces a Gabor feature vector of over 19,000 values, making this the most computational and time intensive method.

Dimensionality reduction is the process of reducing the number of dimensions describing an image, in order to make the data more easily separable into groups, i.e. face or non-face. This is often needed as Machine Learning doesn't scale well to data with a high amount of dimensions. In this case, we use Principal Component Analysis (PCA) to reduce the number of dimensions. This feature descriptor takes the same 18x27 pixel cropped image and reduces each image from a 486 value dataset to a 15 value dataset. This has been the least computationally intensive method and for larger datasets, the performance would definitely be noticeable.

Machine learning methods

Nearest Neighbour is one of the simplest machine learning models. If a test image is described by feature *f* it will be classified with the same classification as the image with the next closest value of *f.* KNN, or K nearest neighbours, uses the same principle as nearest neighbours but uses whichever class the nearest *k* neighbours belong to in order to determine classification. For example, if k=7 and the 7 nearest neighbours are 4 face images and 3 non-face images, the test image will be classified as a face. Nearest Neighbour does not need its own specific implementation, as it can train with the same model as KNN by simply using k=1.

Although KNN is a powerful machine learning method, it is better suited to large data sets with multiple classes. Support Vector Machines (SVM) is a binary classification method which separates the two classes with a line, plane, or hyperplane (depending on dimensionality). The classification of the test image is based on which side of the hyperplane it falls. For data which cannot be linearly separated, a kernel function must be implemented. The kernel function maps the data to a higher-dimensional space allowing a non-linear decision boundary to be defined. Although SVM can be very computationally complex, it works well for small datasets.

Considering that our objective is binary classification using a small dataset, SVM is theoretically the most appropriate learning method, but due to the somewhat elastic nature of machine learning, we decided to test this empirically. To determine the best combination of feature extraction methods and machine learning models, we chose to test each feature extraction method with both SVM and KNN (with varying k values) and cross-reference the accuracies of each combination.

Testing

Testing The Model

In order to evaluate the accuracy of each model, we collected the recognition, recall, precision, specificity, F1, and false alarm rates for each combination of feature descriptors, learning methods, and training-testing data splits that were used to create the binary classifier. This was achieved by calculating the rates using the TP, TN, FP, and FN values obtained from the confusion matrix of the classifier.

Using training-test split

	Recognition rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	78.33	79.17	71.67	84.17	70.83
Dimensionality reduction	57.92	57.92	49.17	54.58	31.67
Gabor vector	24.58	15.00	83.33	83.33	79.58
HOG vector	52.08	67.50	70.42	75.42	85.42

	Recall rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	90.00	85.00	80.00	95.00	69.00
Dimensionality reduction	65.50	65.50	55.50	62.00	31.00
Gabor vector	11.00	0.50	100.00	100.00	77.50
HOG vector	59.50	77.50	83.00	89.50	86.50

	Precision rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	84.91	89.47	85.11	87.16	94.52
Dimensionality reduction	80.37	80.37	77.08	78.98	70.45
Gabor vector	88.00	16.67	83.33	83.33	97.48
HOG vector	77.78	82.45	81.77	82.49	95.58

Feature descriptor	Specificity rate of model (%) KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	20.00	50.00	30.00	30.00	80.00
Dimensionality reduction	20.00	20.00	17.50	17.50	35.00
Gabor vector	92.50	87.50	0.00	0.00	90.00
HOG vector	15.00	17.50	7.50	5.00	80.00

Feature descriptor	F1 rate of model (%) KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	87.38	87.18	82.47	90.91	79.77
Dimensionality reduction	72.18	72.18	64.53	69.47	43.06
Gabor vector	19.56	97.09	90.91	90.91	86.35
HOG vector	67.42	79.90	82.38	85.85	90.81

Feature descriptor	False alarm rate of model (%) KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	80.00	50.00	70.00	70.00	20.00
Dimensionality reduction	80.00	80.00	82.50	82.50	65.00
Gabor vector	7.50	12.50	100.00	100.00	10.00
HOG vector	85.00	82.50	92.50	95.00	20.00

Using 10-fold cross-validation

Feature descriptor	Recognition rate of model (%) KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	98.68	96.59	95.60	92.09	84.18
Dimensionality reduction	96.48	94.95	93.96	90.88	70.88
Gabor vector	94.51	93.30	91.87	88.68	97.03
HOG vector	88.57	85.05	83.08	80.55	88.57

	Recall rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	99.86	100.00	100.00	99.86	90.43
Dimensionality reduction	100.00	100.00	100.00	100.00	75.94
Gabor vector	100.00	100.00	100.00	100.00	99.13
HOG vector	100.00	100.00	100.00	100.00	100.00

	Precision rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	98.00	95.30	94.12	91.49	88.89
Dimensionality reduction	95.57	93.75	92.62	89.26	84.11
Gabor vector	93.24	91.88	90.31	87.01	97.02
HOG vector	86.90	83.54	81.75	79.58	86.90

	Specificity rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	93.63	84.54	80.45	70.91	64.54
Dimensionality reduction	85.45	79.09	75.00	62.27	55.00
Gabor vector	77.27	72.27	66.36	53.18	90.45
HOG vector	52.73	38.18	30.00	19.55	52.73

	F1 rate of model (%)				
Feature descriptor	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	98.78	97.60	96.83	95.42	89.66
Dimensionality reduction	97.73	96.77	96.17	94.33	79.82
Gabor vector	96.50	95.77	94.91	93.05	98.06
HOG vector	92.99	91.03	89.96	88.63	92.99

Feature descriptor	False alarm rate of model (%)				
	KNN (k=1)	KNN (k=3)	KNN (k=5)	KNN (k=10)	SVM
Full image	6.36	15.45	19.54	29.09	35.45
Dimensionality reduction	14.54	20.91	25.00	37.73	45.00
Gabor vector	22.73	27.73	33.64	46.82	9.55
HOG vector	47.27	61.82	70.00	80.45	47.27

Selected Method

Given that we are considering the classifiers created using the provided training-testing data split, there are 3 classifiers of interest based off of their recognition rates: A classifier created using the full image as the feature descriptor & 10-NN as the learning method, another using a Gabor feature vector as the feature descriptor & SVM as the learning method, and another using a HOG feature vector as the feature descriptor & SVM as the learning method.

From the testing and training stages, the initial thought was that we would select the HOG Feature Descriptor with SVM Learning Method at a basic recognition rate of 85.42%, however after looking at the false alarm rate, it appears that the HOG-SVM approach had a very high false alarm rate at 20.00%.

Off of the basic recognition rates it would appear that the Full Image Feature Descriptor with the KNN Learning method (at k = 10) would be the next most accurate method, however looking at the false alarm rate we get a rate of 70.00%, this would strike the Full Image-KNN method out.

The Gabor Feature descriptor with SVM Learning method was the next most accurate with a basic accuracy rate of 79.58% and a false alarm rate of only 10.00%.

We decided to produce detectors for both of these methods as seen in the Detection section.

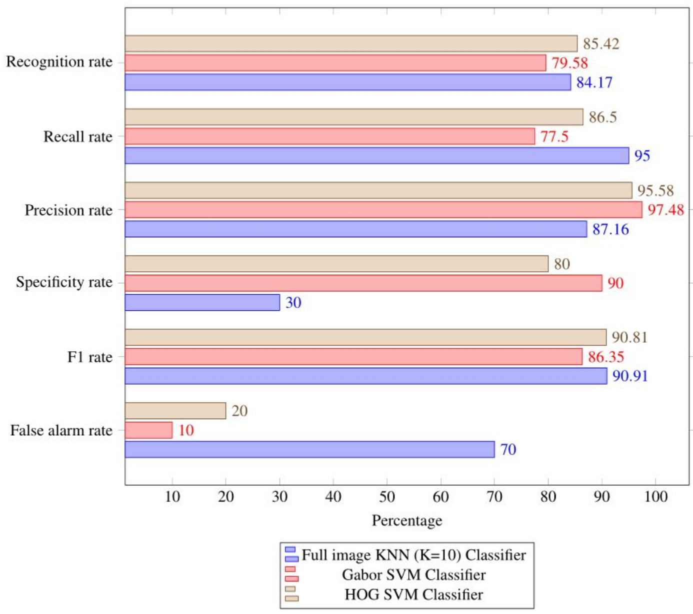

Figure 2. Comparison of Various Rates for test-training splits.

Detection

Detection Method

The detection method uses a sliding window, that moves across the image one pixel at a time. Each time it moves the program takes a crop of the image at the window size of [27, 18]. This is the same image size as the training dataset images.
Each of these images is then fed into the machine learning method that we selected. As with the test and training of the models, the algorithm gives us a prediction of whether or not it thinks there is a face present in the cropped image.
If the machine learning algorithm does detect a face, it stores the coordinates for the bounding box in an array. This gives us the example below (Fig. 3).

Figure 3. HOG-SVM Detection Results (No NMS)

This leaves an image where each and every iteration of a face is detected, this leaves multiple detections for often the same face, as each bounding box could contain a partial part of the face.
One algorithm that could be used to cut down on the number of detections would be the Non-Maximum Suppression (NMS) Algorithm. This algorithm works to cut down the number of bounding boxes by choosing the boxes that have the least amount of overlapping space to represent the detections.
It starts by looping through each of the bounding boxes on the image, for each bounding box (IB) it loops through all the other boxes (OB) apart from itself and calculates the overlapping area. If the overlap area between the IB and OB is greater than a predetermined threshold, then the other box (OB) is rejected as a result. This is repeated until there are no more bounding boxes to test.
After this algorithm has been run, the algorithm returns the accepted bounding boxes as it did before and these are drawn onto the image for verification and example of the output can be seen in Figure 4 and Figure 5.

When the HOG-SVM detector was run, the detections were more sporadic and as the results above suggested, there was a high amount of false alarm rates. However, the speed of detection was short at only 3 minutes and 12 seconds for the example in Figure 4.

Figure 4. HOG-SVM Detection Results

The biggest problem when running the Gabor-SVM Sliding Window method was its speed. For im4.jpg it took on average 47 minutes and 32 seconds to run. It was, however, more accurate than the HOG-SVM Detector giving fewer false positives as seen in Figure 5.

Figure 5. Gabor-SVM Detection Results

It was because of this that we decided to pair the HOG and Gabor Feature extractions together. The script, hog_gab_SVM_dector.m starts by running the HOG-SVM detector, this gives us an accurate but more sporadic detection, with a high percentage of false detections. These detections are then fed into the Gabor-SVM Detector that refines these predictions.
This gives us a quicker detection speed with the im4.jpg now only taking 10 minutes and 43 seconds to run. The dramatic decrease in speed is not all that has changed, the images detected faces are more accurately aligned and by comparing im4.jpg ran through each individual scanner, we can see that there are fewer false positives, this is shown in Figure 9.

Basic Analysis of Results

Image 1 (im1.jpg)

In image one we have identified all seven faces, however, there is a question of accuracy of alignment with the third and fourth faces from the left.

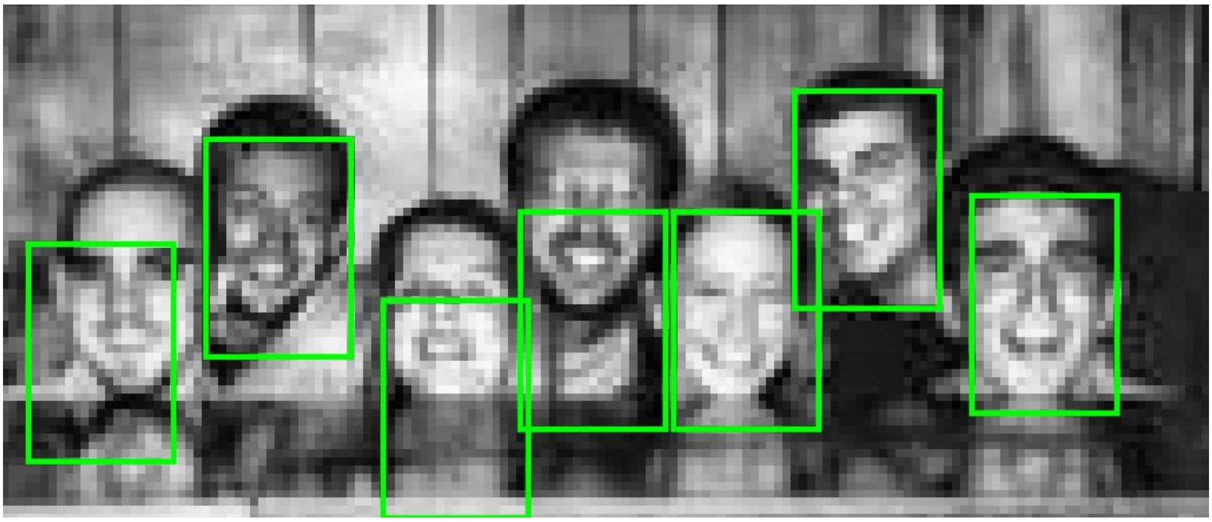

Figure 6. IM1.jpg Detection Result

Image 2 (im2.jpg)

As with Image one, the HOG-GAB-SVM method has recognized all the faces in the test image. One thing to note is that the bounding boxes seem to be more accurately aligned in this test image. As with all the test images, contrast enhancement has been applied before passing to the machine learning algorithm, this seems to have helped with the last face on the second row.

Figure 7. IM2.jpg Detection Result

Image 3 (im3.jpg)

In image three, we can see that all apart from one face has been identified, from looking at the test dataset it appears that the vast majority of the images are straight on and at most only have a minor bit of the face obscured. The face that wasn't picked up is looking down and has half of the face obscured, this is most likely the reason why this face wasn't picked up.

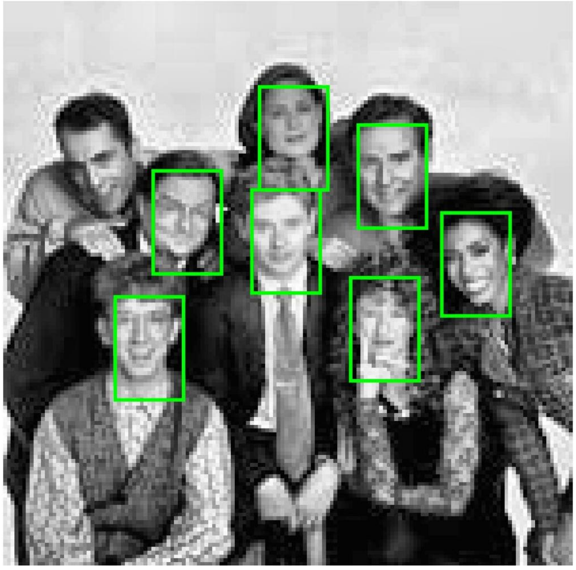

Figure 8. IM3.jpg Detection Result

Image 4 (im4.jpg)

This image was the most computationally intensive, even with the reduction in time from the HOG-GAB-SVM method, it still took 10 minutes to compute this photograph. As we can see below, there are three false positives, and two faces not picked up by the algorithm. In the case of the correctly identified faces, the bounding boxes are correctly aligned.

Figure 9. IM4.jpg Detection Result

Conclusion

In Conclusion, we have created and tested various feature descriptors and learning methods. The best of these methods were then selected, in this case, the Hog and Gabor feature descriptors with the SVM Learning method. This then allowed us to combine the two methods in tandem to create a more accurate and less resource intensive algorithm that gave us accurate results in a timely method.

Judging by the test images above, we have created a successful detector and model.

However, we could improve on the model by including more training images in different lighting as well as different positions where part of the face is obscured.

Another technique that could be used to improve the effectiveness of the detector, is using a dynamic window, as currently if a face is bigger than the 27x18 pixel window it may not be picked up by the detector in full.