INM460 Computer Vision
Coursework report: Face Recognition and OCR

Shengqiang Fan (Student No. 180045953)
MSc Data Science – City, University of London

1. Overview

Two tasks were accomplished in this coursework, which are face recognition and OCR detection. This report will demonstrate the approach used to accomplish the tasks as well as the theory of the approach. The whole process will be discussed in this report, section 2 will show the dataset and preprocess of the dataset. Approach and theory used for face recognition will be demonstrate in section 3 as well as result of face recognition. The output of face recognition is a Matlab function named 'RecogniseFace'(I, featureType, classifierName), where 'I' will be the input image, 'HOG' and 'SURF' is used for feature extraction as 'featureType' input in this function and three classification method are used, which are support vector machine (SVM), multi-layer perceptron (MLP) and convolutional neural network (CNN). In section 4, approach for OCR detection will be discussed and a Matlab function named 'detectNum' (filename) is developed for this task. Section 5 is the conclusion and future works.

2. Data set

2.1 Description of data set

Data set is divided into group and individual images and videos for the whole class. Figure 1 is an example of group photo from one angle. Different angle was used in order to capture all classmates sitting in the different areas.

Figure 1: Group photo example

Figure 2 is an example of individual photos, each classmate was asked to hold a paper with different numbers, which would be used for OCR detection task. The number is unique for different persons. In other words, each number represents for one person. Therefore, there is no need to remember or match people's name in this task, only the number is enough.

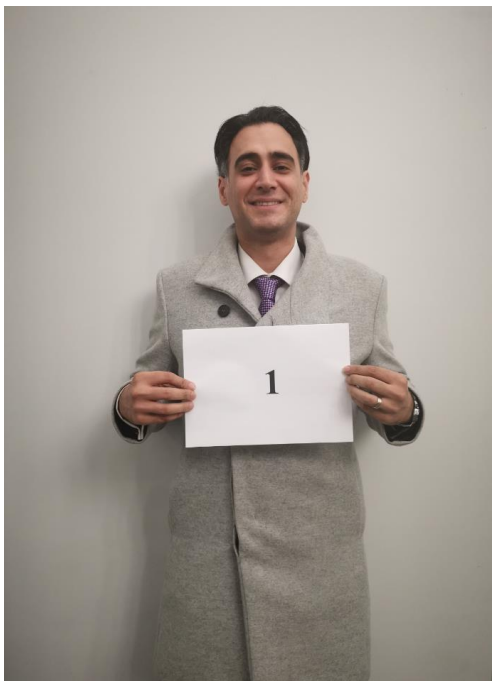

Figure 2: Individual photo example

2.2 Preprocess of data set for face recognition

2.2.1 Extract face from photos and videos

Firstly, all images and videos were converted into same format since photos and videos. The original photos are in HEIC and jpg format because different phones were used to take pictures, so all images were converted to jpg format for convenience. Similarly, videos are all converted to mp4 format due to original format of videos are in both mp4 and mov format. For photos in HEIC format, iMazing HEIC Converter software was used to accomplish the transformation task. In terms of videos, software named Format Factory was used to convert mov file into mp4 format.

Secondly, faces were detected, cropped and resize from individual photos by using built-in function 'CascadeObjectDetector' in Matlab. This function uses the Viola-Jones algorithm to detect people's faces, noses, eyes, mouth, or upper body, where Viola-Jones algorithm uses Haar-like features, a scalar product between the image and some Haar-like templates (Viola and Jones, 2004). A common Haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region as shown in figure 3 (Viola and Jones, 2001). Face area is the target area because the background or clothes will introduce extra noise when face recognition. Once faces were cropped, these new faces were saved in jpg format and classified in different folders correspond to the unique numbers as our face gallery for training a classifier. Videos will be read frame by frame and then use the same procedure to extract faces. Only individual photos and videos were used to create face gallery because there will be time consuming if crop all face from group photos and then divided the face into different groups manually. This could be one of future work to improve the accuracy of face recognition. Manual check was necessary during this preprocess because 'CascadeObjectDetector' is not 100% to extract a face, some error images are also cropped as shown in figure 5. This is mainly due to the Haar-like features was also matched with this white shirt and black suit. In addition, some blurred faces extracted from videos also need to be deleted otherwise it would become noise as shown in figure 6.

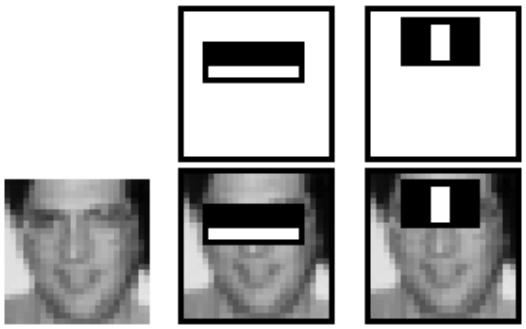

Figure 3: Example of Haar-like feature (Viola and Jones, 2004)

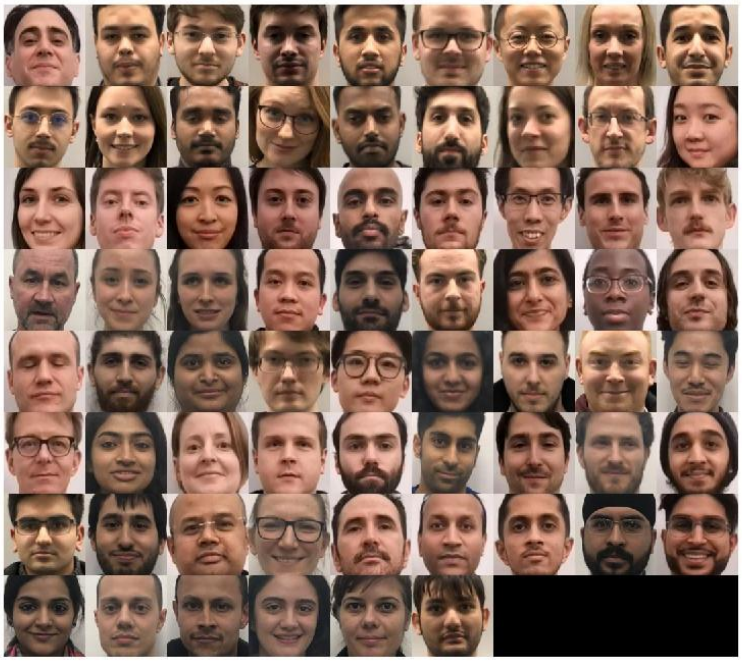

Figure 4: Overview of extracted faces gallery

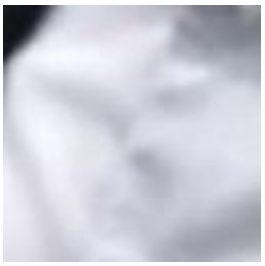

Figure 5: Incorrect cropped image

Figure 6: Blurred image extract from video

2.2.2 Split data

There are 69 persons in total in our face gallery, but the number is not continuous from 1 to 69, it is 1-17, 20-24, 33,34,36-81. The distribution of cropped faces is shown in figure 7. Some faces such as No.45, No. 55 and No. 77 has more than 350 images however, number of images of No. 14 and No. 15 is lower than 100. Therefore, partition method was used to trim the cropped data set as so that each set has exactly the same number of images (80 images). And the data is split into 80% of training set and 20% of validation set after randomize operation. Therefore, each person has 64 images for training and 16 images for validation.

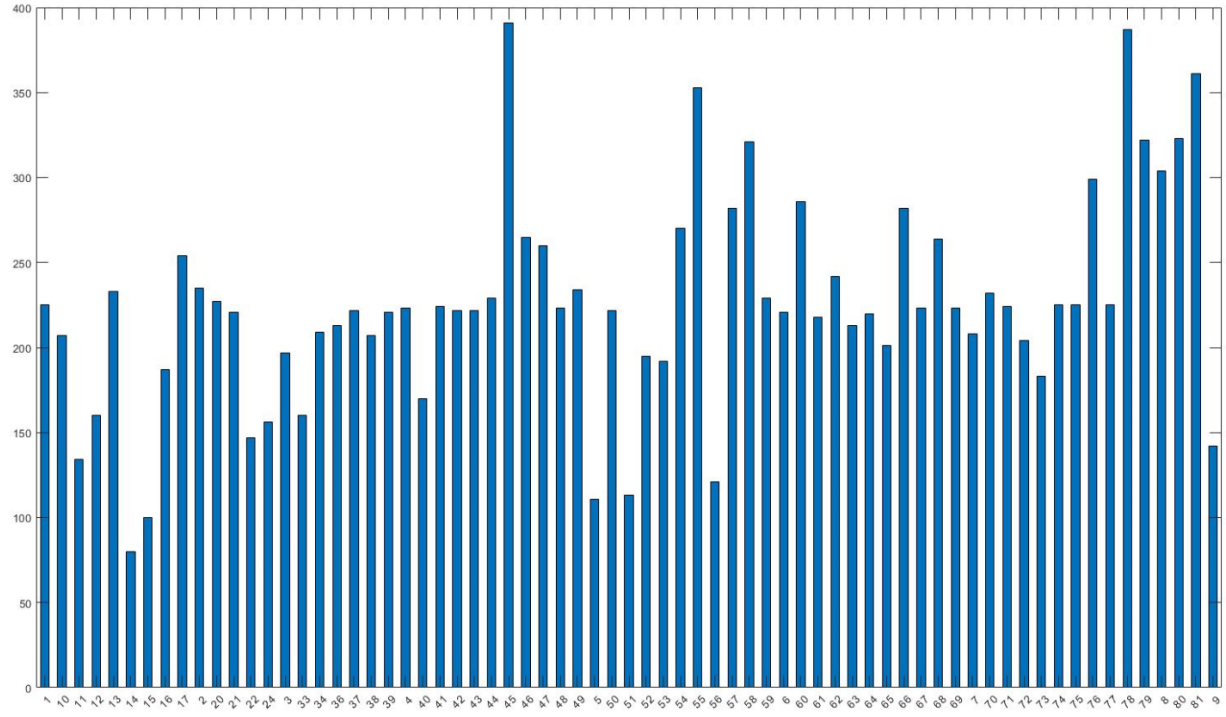

Figure 7: Distribution of cropped faces

3. Approach, methodology and result for face recognition

This section will talk about the approach, methodology and final result for face recognition task. The general pipeline for face recognition task is shown in figure 8 provided by Mathworks (2019). Firstly, features should be extracted from face database/gallery. Then, these extracted features will become input for training a model for classification task. Thirdly, face recognition task can be conducted. When a new query image is given, face detection is needed on the image followed by a face registration step. In face registration step, face is cropped and resized to match the size of images used in the training face gallery. Although it is not necessary for resize the image, it mainly depends on method used for feature extraction and classification. But in this report, all cropped face will be resized to same size as used in the training face gallery for convenience. Finally, use the extracted features to make classification from the trained model to assign input face to the specific classes of faces in the dataset. There are multiple approaches for both feature extraction and classification process. In this report, two feature extraction methods (HOG and SURF) are used combined with 3 classification methods (SVM/MLP/CNN). One note is that feature is automatically extracted during convolution neural network training so that CNN can use the faces images as input directly. Therefore, there are 5 approaches in total, HOG+SVM, HOS+MLP, SURF+SVM, SURF+SVM and CNN. It is expected to see CNN can achieve the highest accuracy among all approaches. The methodology will be discussed in the following part and comparison result will be discussed.

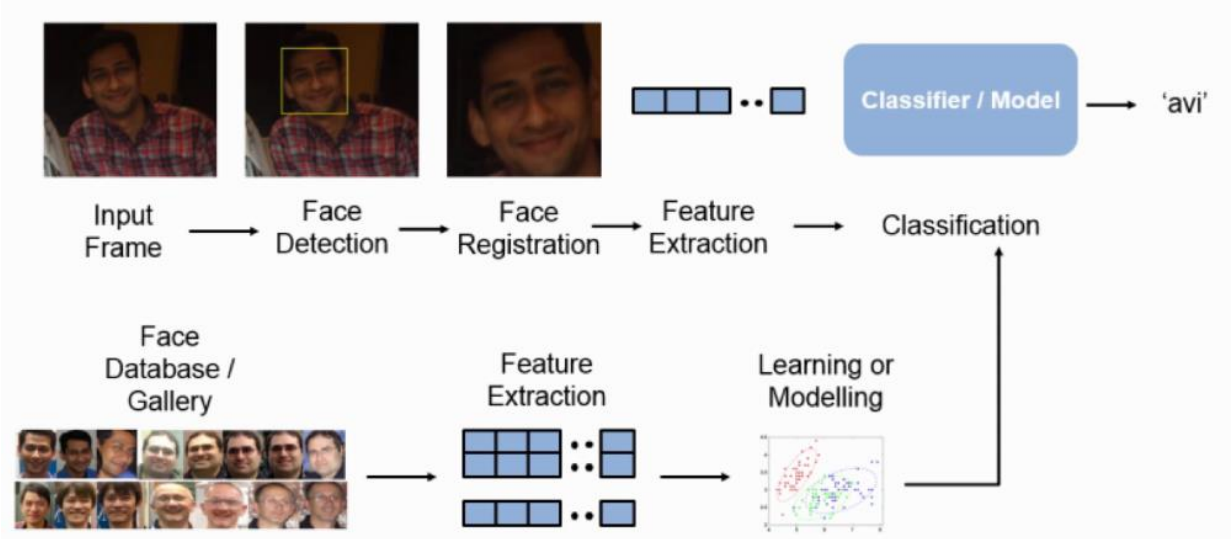

Figure 8: pipeline for face recognition task (Mathworks, 2019)

3.1 Feature extraction method

3.1.1 Histogram of Oriented Gradients (HOG)

Histogram of Oriented Gradients was put forward by Dalal and Triggs (2005), which is widely used in computer vision and image processing for the purpose of object detection. It computes horizontal and vertical gradients as well as gradient orientation and magnitude from input images. Figure 9 demonstrates an example of HOG features extracted from a single face. Matlab code 'extractHOGFeatures' is used to conduct HOG feature extraction task.

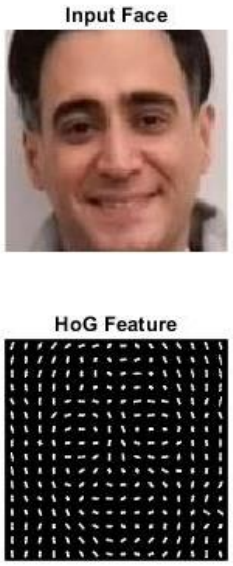

Figure 9: Example of HOG features extracted from a single face

3.1.2 Speeded-Up Robust Features (SURF)

Speeded-Up Robust Features (SURF) was proposed by Bay *et al* (2006), which is partly inspired by the scale-invariant feature transform (SIFT) descriptor. It is proved that SURF describes image faster than SIFT by 3 times. SURF algorithm has three main parts: interest point detection, local neighborhood description and matching. It uses square-shaped filters as an approximation of Gaussian smoothing and a blob detector based on the Hessian matrix to find points of interest. Matlab code 'bagOfFeatures' can use SURF features to create a dictionary of features. It extracts

SURF features from all training images in all image categories. Then, it constructs the visual vocabulary by reducing the number of features through quantization of feature space using K-means clustering. In this coursework, vocabulary is composed of 500 clusters by setting ('VocabularySize', 500) and 'StrongestFeatures' is set as 0.80. Figure 10 shows the occurrence frequency of visual word for a single face image from training set.

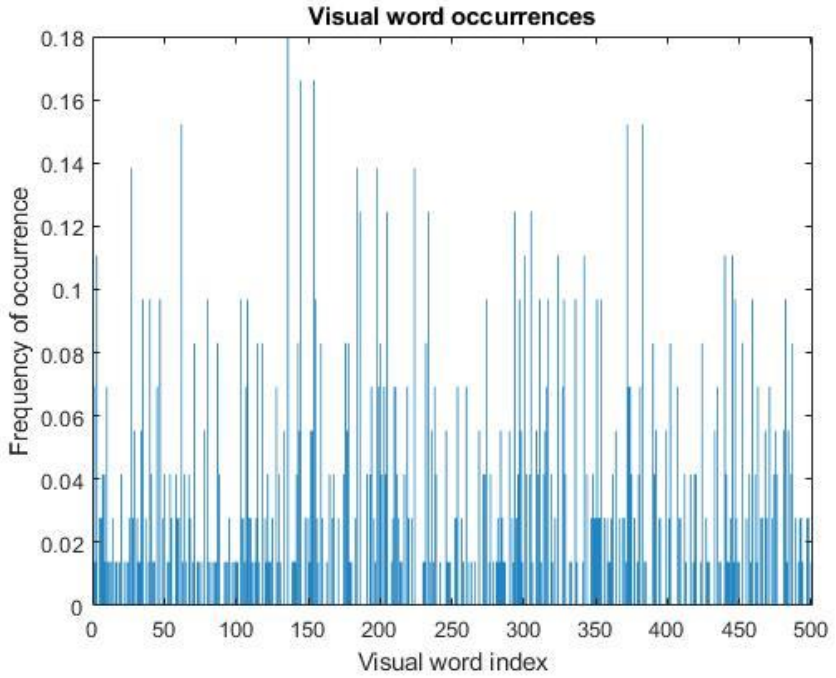

Figure 10: Example of occurrence frequency of visual word for single face from training set.

3.2 Classification method

3.2.1 Support Vector Machine (SVM)

Support vector machine (SVM) is a one of the supervised learning algorithms that can be used for both classification and regression tasks (Wang, 2005). It is robust to outliers and effective in high dimensional spaces. SVM classifies by defining an optimal hyperplane. When there are serval hyperplanes that classifies the data, it then finds the hyperplane that maximizes the margin between nearest data points from either class as shown in figure 11. SVM has a unique technique called the kernel trick. It is used for non-linear separable problems. We transform the dataset into a higher dimension, and then the problem become separable. Initially, SVM is used for binary classification. However, we can extend the algorithm to multi-class task by converting it to several binary problems, which is demonstrated in this project.

In this coursework, Matlab built-in function 'trainImageCategoryClassifier' is used which contains a linear support vector machine (SVM) classifier trained to recognize an image category.

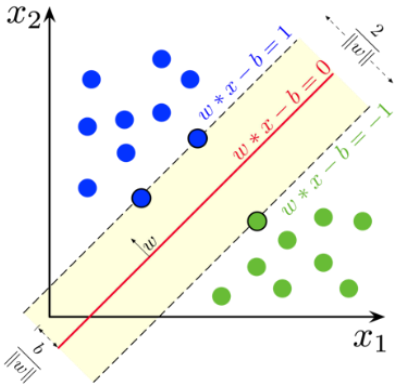

Figure 11: Maximum-margin hyperplane and margins for an SVM model

3.2.2 Multi-layer perceptron (MLP)

MLP has the ability to learn and model non-linear and complex relationships and is a benchmark classifier in many pattern recognition tasks. Therefore, it is chosen as one of classification method for comparison. A multilayer perceptron is a class of feedforward artificial neural network, which has three essentials elements named input layer, hidden layer and output layer as shown in figure 12. All nodes/neurons are fully connected associated with weights and bias. Each neuron uses a nonlinear activation function except for input nodes, common activation functions are sigmoid/tanh/ReLu/softmax etc. Input data pass though model by multiply with weights and add bias at every layer and find the calculated output of the model, which is called forward pass. Then, backpropagation is introduced for training, which is a supervised learning technique proposed by Rumelhart et al (1986). In backpropagation, error/loss calculated from forward pass and then propagates to update weights and bias by using gradient. In a conclusion, there are three basic steps for training an MLP model, which are forward pass, error/loss calculation and backpropagation.

In this coursework, overall structure of MLP is one hidden layer of with 128 neurons with 69 neurons in output layer. 'Trainscg' is used which is a network training function that updates weight and bias values according to the scaled conjugate gradient method. Matlab built-in function is used as follows:

```
MLP_SURF_net = feedforwardnet([128, 69], 'trainscg');

MLP_SURF_net.trainParam.epochs=2000;

MLP_SURF_net = configure(MLP_SURF_net, trainingx, trainingy);

MLP_SURF_net = train(MLP_SURF_net, trainingx, trainingy);
```

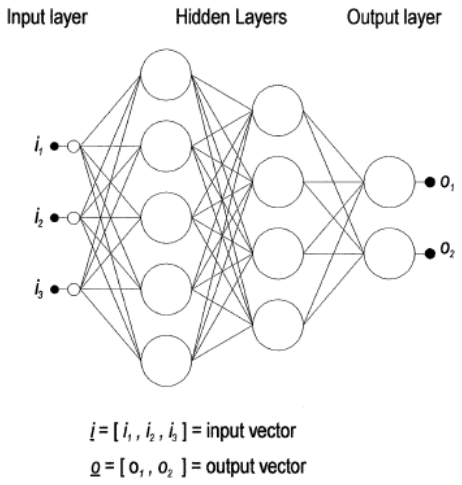

Figure 12: An illustration of multilayer perceptron (two hidden layers)

3.2.3 Convolutional neural network (CNN/alexnet)

Convolutional neural networks are special cases of Multilayer Networks inspired by the structure of the visual cortex (Lawrence *et al*, 1997). Krizhevsky *et al* (2012) achieved break through in accuracy by using deep convolutional neural network to classify the 1.3 million high-resolution images in the LSVRC-2010 ImageNet training set into the 1000 different classes. Therefore, in deep learning, it is widely applied to analyzing visual imagery recent years. A convolutional neural network consists of an input layer, output layer and multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, RELU layer (activation function), pooling layers, fully connected layers and normalization layers. The existing convolutional layers makes CNN differ from other classification method (e.g. SVM and MLP) because convolutional layers play a role in generating features by combining data. Therefore, there is no need to do

feature extraction individually when using CNN as the classifier to accomplish face recognition task.

In this coursework, major architecture of AlexNet is used, which is proposed by Krizhevsky *et al* (2012). Figure 13 demonstrates the architecture of AlexNet, it contains 5 convolutional layers and 3 fully connected layers. Relu is applied after very convolutional and fully connected layer. Dropout is applied before the first and the second fully connected year. The image size in the following architecutre chart should be 227 * 227, which is very important. AlexNet can be loaded by calling 'net = alexnet' in Matlab. Final layers are replaced to fit our face recognition purpose and parameter setting are shown in figure 14. Minibatch size is set to be 10 and 4 MaxEpoches is used because it is already converged also shown in figure 15 as training pregress. In addition, Data augmentation is conducted in order to make our CNN classifier more robust by calling 'imageDataAugmenter' function in matlab.

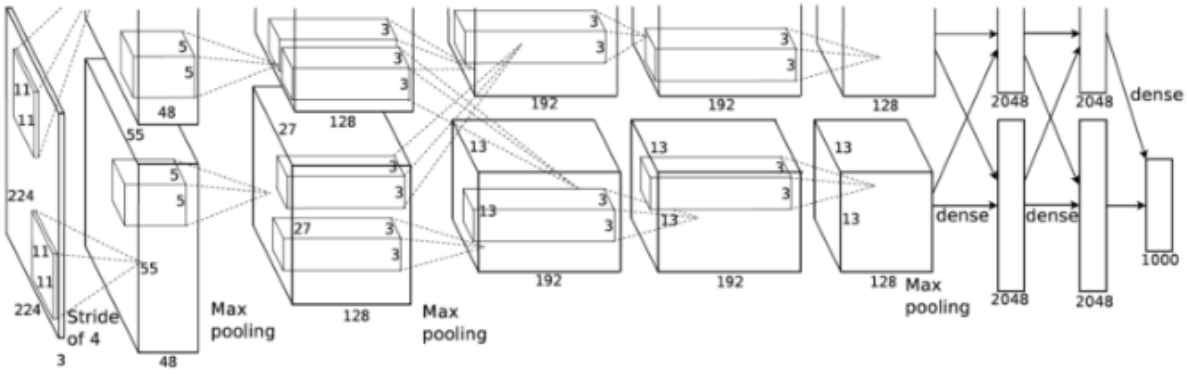

Figure 13: Architecture of AlexNet (Krizhevsky et al, 2012)

```
% Load the network
net = alexnet;
inputSize = net.Layers(1).InputSize; % Check the alexnet inputsize: 227*227*3
img = readimage(images,1);
size_i = size(img); % Check our data image size: 227*227*3
% Data augmentation
pixelRange = [-30 30];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange);
augimdsTrain = augmentedImageDatastore(inputSize(1:2),trainingImages, ...
    'DataAugmentation',imageAugmenter);
augimdsValidation = augmentedImageDatastore(inputSize(1:2),validationImages);
% Replace final layers to fit our face recognition task
layersTransfer = net.Layers(1:end-3);
numClasses = numel(categories(trainingImages.Labels));
layers = [
    layersTransfer
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
miniBatchSize = 10;
numIterationsPerEpoch = floor(numel(trainingImages.Labels)/miniBatchSize);
options = trainingOptions('sgdm',...
'MiniBatchSize',miniBatchSize,...
'MaxEpochs',4,...
'InitialLearnRate',1e-4,...
'Verbose',false,...
'Plots','training-progress',...
'ValidationData',validationImages,...
'ValidationFrequency',numIterationsPerEpoch);
%% Train network using Traing Data
CNN_net = trainNetwork(augimdsTrain ,layers, options);
```

Figure 14: Code used for training alexnet

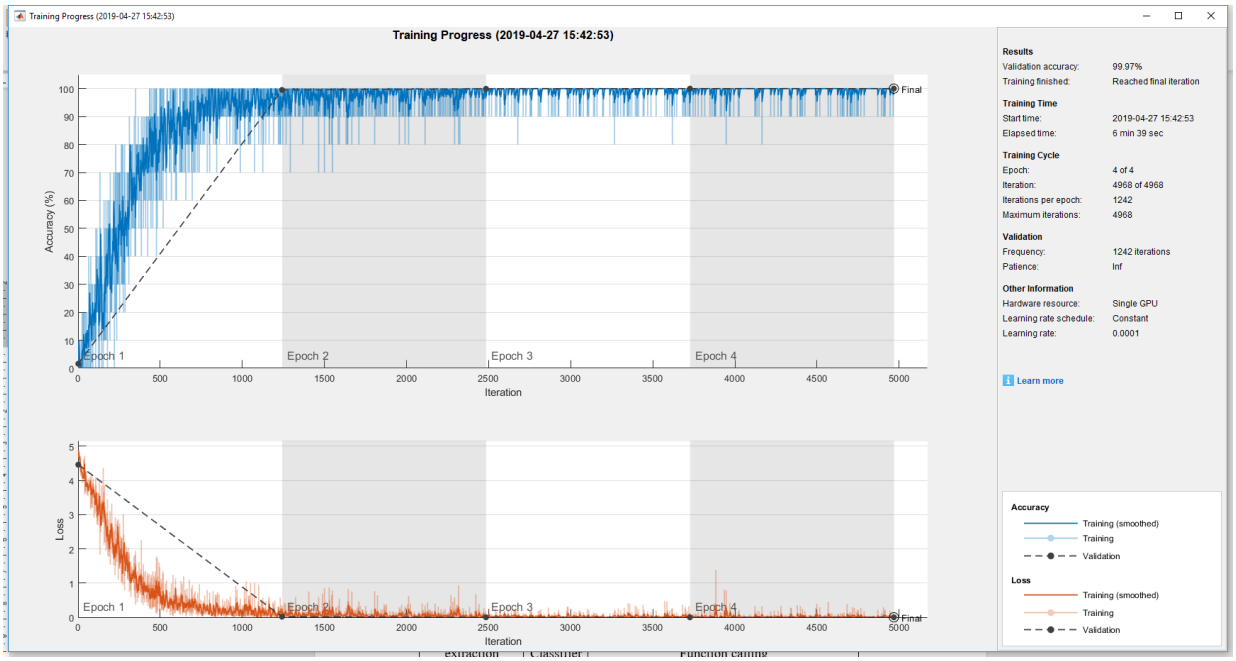

Figure 15: CNN training progress

3.3 Result of face recognition and discussion on individual photos

There are 5 combinations in total as discussed before, HOG+SVM, HOS+MLP, SURF+SVM, SURF+SVM and CNN. With 80% training set and 20% validation set, training accuracy and validation accuracy is summarized for all 5 models as shown in table 1. It is obvious that all approaches can achieve relatively high accuracy on both training and validation set while CNN has the best performance as expected. The confusion matrix of CNN is shown in figure 16. Only one wrong prediction on validation set as highlighted in figure 16, which should be person 4 but it is predicted as person 38. The wrong prediction face is shown in figure 17 (left) with number 4 but it is predicted to be 38, face 38 are also shown in figure 17 (right). One possible reason is that they are looking to the similar direction and has similar position in the image. In other word, the 1/5 left portion of image are all white wall.

Based on that, A Matlab function named ‘RecogniseFace’(I, featureType, classifierName) was create, where ‘I’ will be the input image, ‘HOG’ and ‘SURF’ is used for feature extraction as ‘featureType’ input in this function and three classification method are used, which are support vector machine (SVM), multi-layer perceptron (MLP) and convolutional neural network (CNN). ‘RecogniseFace’ function will detect all faces in testing images and return the prediction number corresponding to the face with x,y position of face. Table 2 summarized how to call this function with different combinations.

Table 1: Training and validation accuracy of different approaches

Feature extraction	Classifier	Training Accuracy	Validation Accuracy
HOG	SVM	1.000	0.9991
SURF	SVM	0.9991	0.9964
HOG	MLP	0.9878	0.9801
SURF	MLP	0.8845	0.8687
	CNN	1.000	0.9997

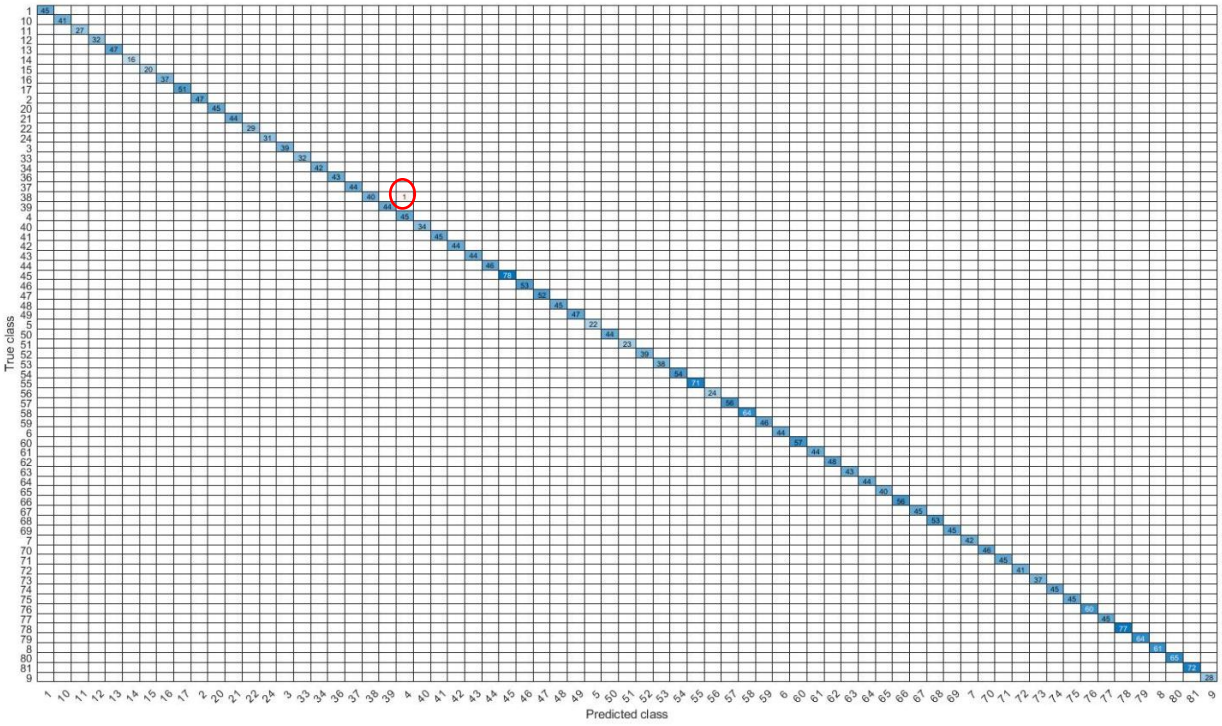

Figure 16: Confusion matrix of CNN

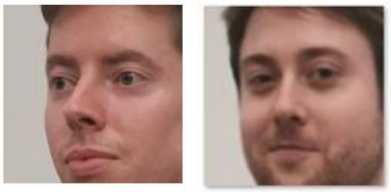

Figure 17: Wrong prediction face (left 1) and predicted number face (right)

Table 2: Example of calling 'RecogniseFace' function

Feature extraction	Classifier	Function calling
HOG	SVM	RecogniseFace('***.jpg' , 'HOG' , 'SVM')
SURF	SVM	RecogniseFace('***.jpg' , 'SURF' , 'SVM')
HOG	MLP	RecogniseFace('***.jpg' , 'HOG' , 'MLP')
SURF	MLP	RecogniseFace('***.jpg' , 'SURF' , 'MLP')
	CNN	RecogniseFace('***.jpg' , '' , 'CNN')

3.4 Result of face recognition and discussion on group photos

So far, face gallery is created from individual photos and videos, no group photos included. Since 'RecogniseFace' is already create, it also can be used to recognize face from group photo. Two group photos (IMG_8225 and IMG_8232) from different angle were used to test the accuracy of all approach. The main process is manually because the person in the group photo is not in order so manually check is needed to see if it is wrong or right prediction. Figure 18 shows the face recognition result from group photo IMG_8225 as an example. From figure 18, it can be seen that one lady's face at left bottom was not detected due to there is only half face in this picture. One face was not detected due to overlap issues. Table 3 summarized the performance of approaches when dealing with group photo. It is obvious that accuracy on group photos is much lower than individual photos for all approaches. HOG+SVM approach has higher accuracy on

group photo however, it still lower than 50%. The major reason for accuracy has a big drop for all approach on group photo is that faces in group photo has lower resolution compared to individual photo, especially for person who sitting behind. Secondly, it is found that the box of face area was cropped a bit too large which includes some extra information (e.g. background) as noise. So smaller box should be considered to improve the accuracy. Thirdly, there are some face overlap occurs in group photos which reduce the accuracy. Region of interest procedure can be considered to reduced overlap issues. In addition, lighting condition is different in group photo and individual photo, which may have more impact on CNN approach. Moreover, it is convinced that accuracy will increase a bit if also includes faces from group for training the model.

Figure 18: Face recognition example of group photo IMG_8225

Table 3: Accuracy on group photo

Feature extraction	Classifier	Accuracy on group photo 1	Accuracy on group photo 2	Average
HOG	SVM	0.5517	0.4444	0.4981
SURF	SVM	0.5862	0.2963	0.4413
HOG	MLP	0.4483	0.2963	0.3723
SURF	MLP	0.1379	0.1481	0.1430
	CNN	0.5172	0.2963	0.4068

4. Approach for OCR detection

Other task for this coursework is to develop a function that accepts an image/video file (filename) of a person holding a number in hand (as shown in individual phtos) and 'detectNum' function will return the number seen in that image/video. There are different options to conduct this task. One possible solution could be training a CNN model to detect the numbers. Other possible solution could be using rectangle detection to crop A4 white paper as interest area and then detect numbers in the region of interest. In this report, MSER approach is proposed to detect candidate text regions and then use built-in OCR function in Matlab. MSER is maximally stable extremal regions, which is proposed by Matas *et al* (2002). MSER is based on blob detection, which detect regions difference in properties such as brightness compared to surrounding area. It is based on the idea of taking regions which stay nearly the same through a wide range of thresholds. All the pixels below a given threshold are white and all those above or equal are black. Therefore, find a proper threshold is important. MSER extraction implements the following steps: 1. Sweep threshold of intensity from black to white, performing a simple luminance thresholding of the image. 2. Extract connected components. 3. Find a threshold when an extremal region is

“Maximally Stable” and 4. Keep those regions descriptors as features. Built-in function ‘detectMSERFeatures’ can be used to implement MSER in Matlab. Parameter was test to be ('RegionAreaRange',[2500 7000],'ThresholdDelta',5) after some trails under image resolution in 3648*2736. Once regions of MSER is detected, boxes are created and expand. And merging the overlapped region together. This process is important so that two numbers will be detected together rather than individually. Once, these regions are created, OCR detection is computed by calling built-in function ‘ocr’ in Matlab. Loop all regions to make OCR detection and output the ocr detection result with highest confidence level. Example of OCR detection is shown in figure 19. If input is video, similar process will be conducted from frame by frame. Frames of videos are extracted by order until number can be detected in one frame. The example to use ‘detectNum’ function is that

detectNum(‘***.jpg’) or detectNum(‘***.mp4’)

Around forty individual photos and videos are tested, and it can correctly detect number by 90% accuracy. Error occurs when numbers of characters appears on clothes which introduce noise to or target number. Possible way to improve accuracy is that narrow interest region first by some reference object (e.g. A4 white paper four corners).

Figure 19: Example of OCR detection

5. Conclusion and future works

5.1 Face recognition

‘RecogniseFace’ function is created for the face recognition task. Example of calling this function is already shown in table 2. It returns the prediction number corresponding to the face with x,y position of face. Only individual images and videos were used when training the classifier so that all approaches do not achieve high accuracy when dealing with group photos although has relative high accuracy in individual photos. Possible extension to improve accuracy is that increasing faces from group photo for training the model. In addition, it is found that the box of face area was cropped a bit too large which includes some extra information (e.g. background) as noise. So smaller box should be considered to improve the accuracy. Possible approach is detecting two eyes and cropping box based on a common proportion of face. Moreover, selection of region of interest procedure can be considered to reduced overlap issues. Face detection and prediction in real-time is another possible extension.

5.2 OCR detection

‘detectNum’ function is created for OCR detection task. It will return the detected number in images or videos. MSER algorithm was used to detect candidate text regions and then built-in OCR function in Matlab was followed to detected number with highest confidence level. For

videos, similar procedure was repeated frame by frame until number is detected. Overall accuracy of this approach is around 90%. Error occurs when numbers of characters appears on clothes which introduce noise to or target number. Possible way to improve accuracy is that narrow interest region first by some reference object (e.g. A4 white paper four corners). OCR detection in real-time could be an extension of this task since both accuracy and processing time should be considered in real-time case.

Reference:

Bay, H., Tuytelaars, T. and Van Gool, L., 2006, May. Surf: Speeded up robust features. In European conference on computer vision (pp. 404-417). Springer, Berlin, Heidelberg.

Dalal, N. and Triggs, B., 2005, June. Histograms of oriented gradients for human detection. In international Conference on computer vision & Pattern Recognition (CVPR'05) (Vol. 1, pp. 886-893). IEEE Computer Society.

Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

Lawrence, S., Giles, C.L., Tsoi, A.C. and Back, A.D., 1997. Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks, 8(1), pp.98-113.

Matas, J., Chum, O., Urban, M. and Pajdla, T., 2004. Robust wide-baseline stereo from maximally stable extremal regions. Image and vision computing, 22(10), pp.761-767.

Mathworks, 2019. Face recognition with computer vision. Available at: https://uk.mathworks.com/discovery/face-recognition.html [Accessed 27 Apr. 2019].

Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. CVPR (1), 1, pp.511-518.

Viola, P. and Jones, M.J., 2004. Robust real-time face detection. International journal of computer vision, 57(2), pp.137-154.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1985. Learning internal representations by error propagation (No. ICS-8506). California Univ San Diego La Jolla Inst for Cognitive Science.

Wang, L. ed., 2005. Support vector machines: theory and applications (Vol. 177). Springer Science & Business Media.