IN3060/INM460 Computer Vision Coursework report

- **Student name, ID and cohort:** Edward Costache (180012534) - UG
- **Google Drive folder:** https://drive.google.com/drive/folders/13NEaXpB-spLehYLp3j0mLN9OfcJ4Zc8a?usp=sharing

Introduction

Emotions such as happiness, sadness and fear are one of the most important factors in human communication. We use emotions to react to certain events or other people's emotions as a way of expressing ourselves without necessarily involving communication, after all a picture may speak a thousand words, while in this case our emotions convey a thousand words. While reading emotions and interpreting them is very easy for us humans, it can't be said the same for computers. However, it is possible to train the computer to read emotion by using feature descriptors and classifiers on a dataset of images portraying different emotions. For such a task, a lot of data of faces labelled with the corresponding emotion will be required to get accurate results.

The task is based around using different types of feature descriptors such as SIFT and HOG and pairing them with a combination of different classifiers such as SVM or MLP. Feature descriptor and classifier combinations will be compared qualitatively and quantitatively to access the performance of the models (accuracy, speed, and model size). The best model will be assessed on a personal dataset.

Data

The dataset was provided to me by City University of London for this project. The dataset consists of 12,271 training images and 3,068 testing images of cropped human faces. The images come with labels which are kept in a .txt file. There are a total of 7 emotions (namely: surprise, fear, disgust, happiness, sadness, anger and neutral) and they are numbered 1 to 7. The data was also not sampled equally, this meant that come labels would contain more samples than others. For example, label '4' which stands for 'happiness' contained 4772 samples, while label '2' (fear) contained 281.

Using python, the data was reformatted such that image samples were contained within folders representing their labels. For example, within 'CW_Dataset/test' seven folders representing all emotions were created, and images were moved into their respective folders using the label files. Because of the transition the labels files were now removed as they were no longer necessary. Also using python, I was able to create a function that would take a 'target' argument which would represent the target of samples I want in every emotion folder. If the current folder contained more than the target, samples would be randomly removed to meet the target, this is known as Random Under sampling. If the samples were less than the target the function will attempt Random Oversampling where random images in that folder would be duplicated until the target was reached.

The Personal_Dataset replicated this same format without sampling the images because it is unnecessary. The dataset contains cropped images provided by me. This dataset would later be used to assess the best model in the wild. The data contains selfies and images from google under Creative Commons Licencs.

Implemented methods

The implementation and the training of the models were created on Jupyter Notebooks. This decision was made by considering hyperparameters. It would be easier to test for hyperparameters on a notebook than on a sequential IDE, this is because we can save the descriptors from the feature descriptors in a variable and re-train the model with different parameters and compare the accuracy.

SFIT-SVM Model

The first model was based on Bag of Visual Words approach and used SIFT and SVM as its feature descriptor and classifier. SIFT was chosen here because it is proven to have good recall rates and accuracy and is included within the OpenCV library after the expiration of its patent. The algorithm is also good at gathering descriptors from images with a lot of occlusion and clutter which is shown in the dataset. While SIFT can be efficient it is reported that the algorithm is sometimes very slow compared to SURF. However, SUFT was not used because it is not yet included in the OpenCV library due to its patent. It is also reported that the algorithm struggles with lighting and blur which is also present in the dataset and will be analysed in the 'Results' section.

On the other hand, SVM was chosen as the classifier because of my previous experience with the model and machine learning in general, thus being easy to implement and train. A paper on using SVMs as a technique for solvency analysis claims that 'SVMs are a new technique suitable for binary classification tasks' Auria, Laura; Moro, Rouslan A. (2008). Does this mean that SVMs are not suitable for anything more than binary? Not at all, while 'SVMs do not support native multiclass classification, the sample principle of separating datapoints within 2 classes is utilized after breaking down the multiclassification into multiple binary classification problems.' Baeldung, Multiclass Classification using support vector machines.

With the descriptors generated from SIFT, kmeans clustering was used to cluster points together and form codewords. A vocabulary of words would thus be constructed from all the images and each image will have its own frequency histogram of every codewords appearing in the image. This will then be provided to the classifier to train.

HOG-MLP Model

Histogram of oriented gradients is a feature descriptor that counts occurrences of gradient orientation in a portions of an image. The reason for implementing this technique as the feature descriptor for this model is that it is often used for object detection. The feature descriptor could be used to find 'objects' in an image such as a smile, or the direction of the eyebrows and many more that could be used to detect emotions. Some research also suggests that HOG is good for human detection, that is detecting humans in an image. However, HOG will be tested for its accuracy in being able to detect emotions in human faces.

To pair up the feature descriptor, MLP (Multilayer Perceptron) will be used as the classifier for the model. Since MLP is highly dependent on the quality of its data it was a good choice to sample it beforehand and not within the training process of the model. For this application, I am going to use the default call for a MLP from sklearn. This initiated the MLP with 100 hidden neurons within the hidden layers, an alpha value which was set to 0.01 and max_iterations to 500.

HOG-SVM Model

For the last model, it was decided to combine the classifier from the first model and the feature descriptor from the second and see the results. Bag of Visual words however cannot be used just like in the first model. This is simply because the HOG feature descriptor already outputs a histogram, that of gradients. We cannot use this histogram to build a vocabulary as we are missing the codewords from the images.

The feature descriptors outputted in the form of a histogram by HOG will be sent directly to train under a SVM classifier.

Results

SIFT-SVM Model

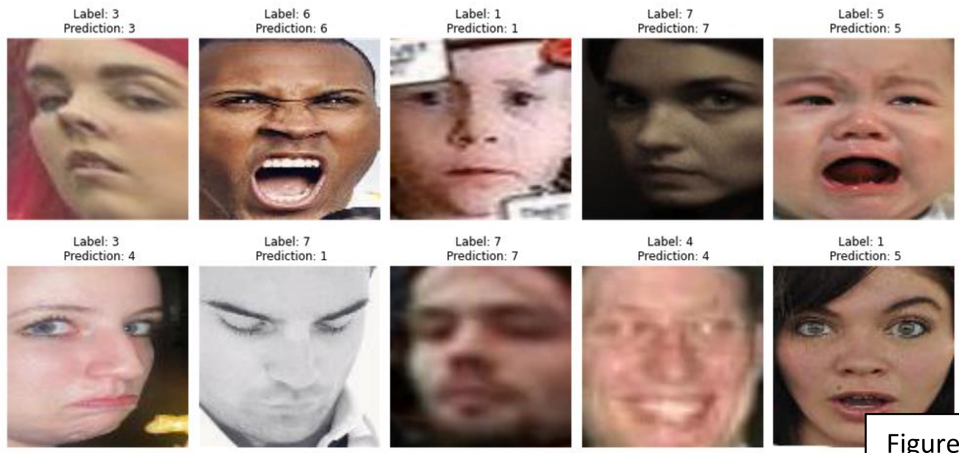


Figure 1

Feature Descriptor time	Histogram of codewords time	Training time	Testing time	Total time	Accuracy	Model size
19 (seconds)	88 (seconds)	6 (seconds)	8 (seconds)	121 (seconds)	50%	7,543 KB

HOG-SVM Model

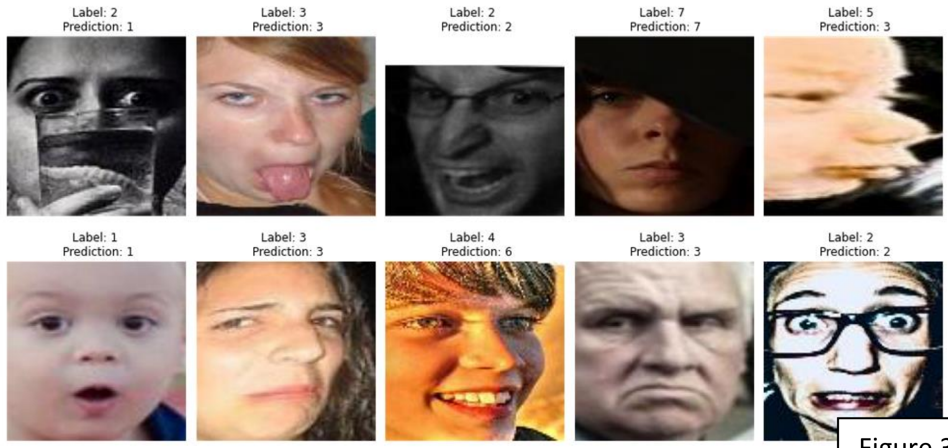


Figure 3

Feature Descriptor time	Training time	Testing time	Total time	Accuracy	Model size
56 (seconds)	13 (seconds)	18 (seconds)	87 (seconds)	67%	17,293 KB

HOG-MLP Model

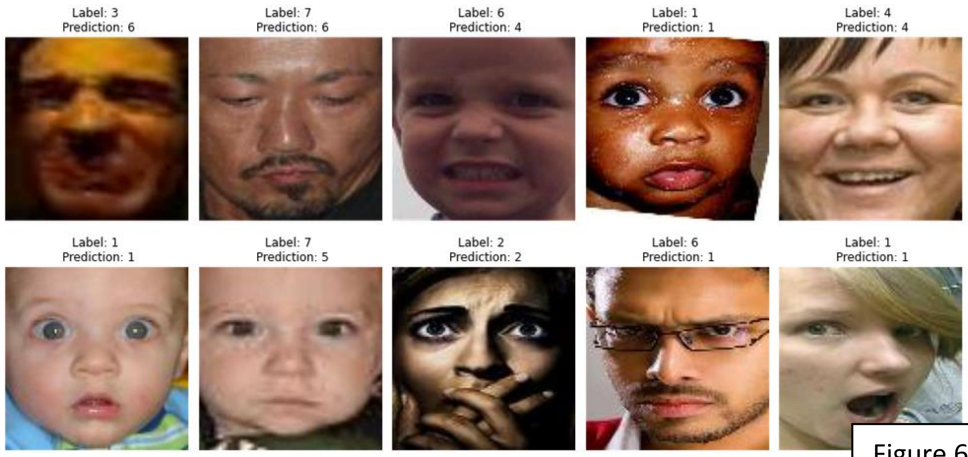
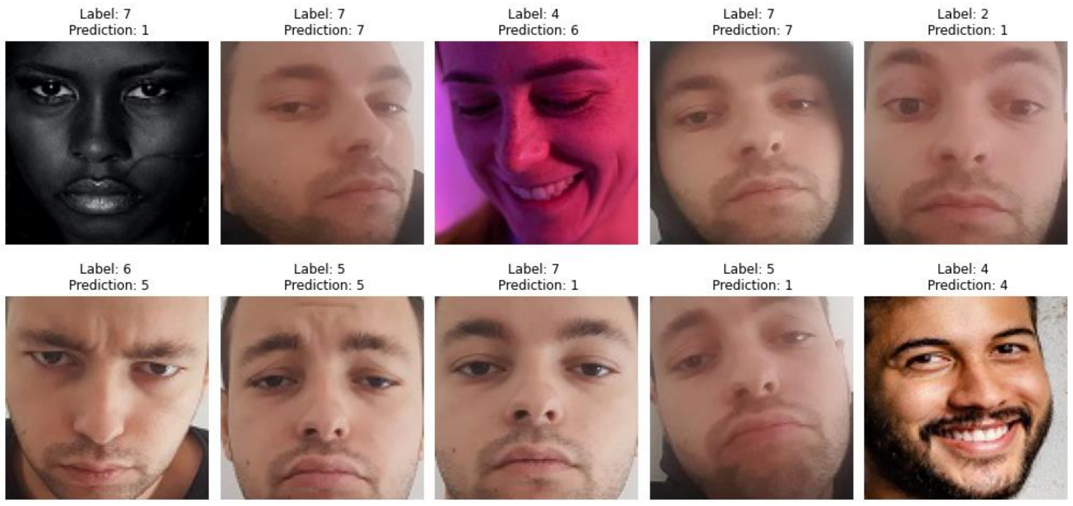


Figure 6

Feature Descriptor time	Training time	Testing time	Total time	Accuracy	Model size
57 (seconds)	50 (seconds)	14 (seconds)	121 (seconds)	59%	940 KB

Testing the best model (HOG-SVM) on the personal dataset



Please refer to the confusion matrix for all the models which can be found in the Appendix section. Namely Figures 2, 5 and 8. From these matrices we can see that for all models the easiest label to predict was 2 due to its high number of true positives.

Discussion

The most accurate model was HOG-SVM at 67% accuracy. This level of accuracy is not suitable for standard applications, however based on the dataset and the lack of hyperparameter tuning it is not bad. Figure 3 shows us that the model was able to mostly predict the emotions even when the images are distorted such as the 4th image on the 1st row. The fastest model to train and test was again the HOG-SVM model by more than 30 seconds than the other two. However, this was the biggest model in size by far when compared to the rest of the model at 17,293 KB. HOG-MLP also performed average of accuracy but its model size if the lowest of the three and has the highest total time. The time can be explained because we were using a Multilayer Perceptron with 500 iterations, perhaps better hyperparameter tuning would have increased the accuracy as well.

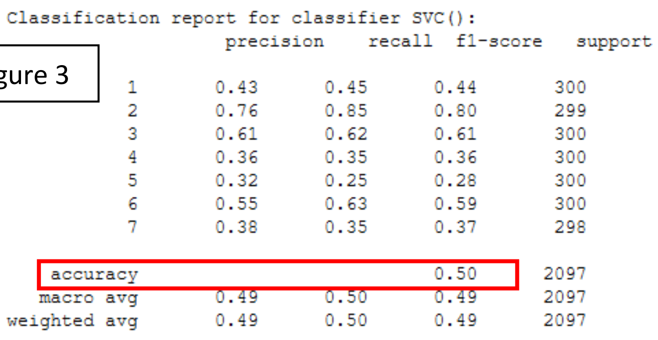
To build upon this project I would introduce a function where it would contain an algorithm that could find the hyperparameters for each classifier that produces the highest accuracy. Furthermore, a CNN would be added to assess whether the accuracy would improve if training was done with a Neural Network. Lastly, I would build upon the dataset so that the model would have sufficient samples to work with. This would potentially require GPU processing as the computational time would increase tremendously, especially with a CNN.

References

Auria, Laura; Moro, Rouslan A. (2008) : Support Vector Machines (SVM)
as a technique for solvency analysis, DIW Discussion Papers, No. 811, Deutsches Institut für Wirtschaftsforschung (DIW), Berlin

Baeldung (2020) : Multiclass Classification Using Support Vector Machines [Available Here : https://www.baeldung.com/cs/svm-multiclass-classification]

Appendix



```
Classification report for classifier SVC():
                precision    recall  f1-score   support

           1       0.43      0.45      0.44       300
           2       0.76      0.85      0.80       299
           3       0.61      0.62      0.61       300
           4       0.36      0.35      0.36       300
           5       0.32      0.25      0.28       300
           6       0.55      0.63      0.59       300
           7       0.38      0.35      0.37       298

    accuracy                           0.50      2097
   macro avg       0.49      0.50      0.49      2097
weighted avg       0.49      0.50      0.49      2097
```

Figure 3

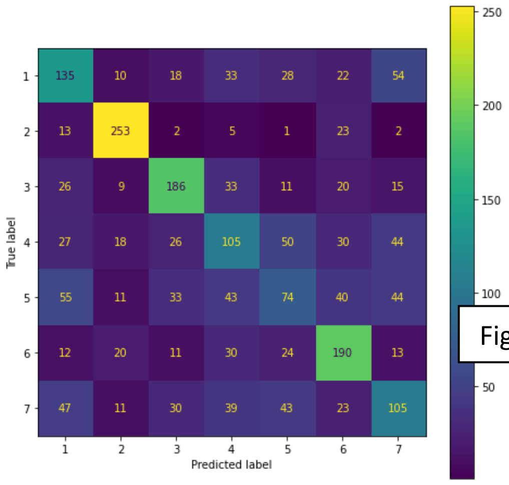
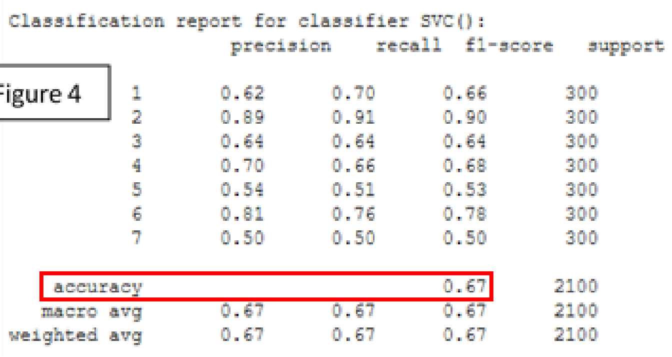


Figure 2



```
Classification report for classifier SVC():
                precision    recall  f1-score   support

           1       0.62      0.70      0.66       300
           2       0.89      0.91      0.90       300
           3       0.64      0.64      0.64       300
           4       0.70      0.66      0.68       300
           5       0.54      0.51      0.53       300
           6       0.81      0.76      0.78       300
           7       0.50      0.50      0.50       300

    accuracy                           0.67      2100
   macro avg       0.67      0.67      0.67      2100
weighted avg       0.67      0.67      0.67      2100
```

Figure 4

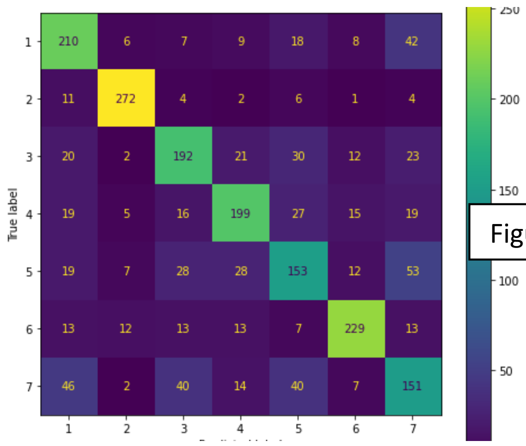
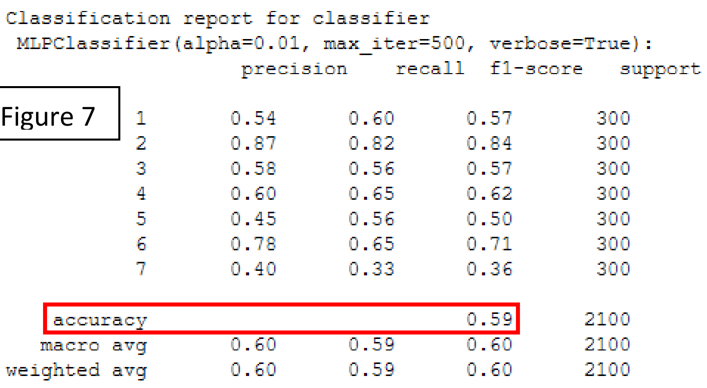


Figure 5



```
Classification report for classifier
 MLPClassifier(alpha=0.01, max_iter=500, verbose=True):
                precision    recall  f1-score   support

           1       0.54      0.60      0.57       300
           2       0.87      0.82      0.84       300
           3       0.58      0.56      0.57       300
           4       0.60      0.65      0.62       300
           5       0.45      0.56      0.50       300
           6       0.78      0.65      0.71       300
           7       0.40      0.33      0.36       300

    accuracy                           0.59      2100
   macro avg       0.60      0.59      0.60      2100
weighted avg       0.60      0.59      0.60      2100
```

Figure 7

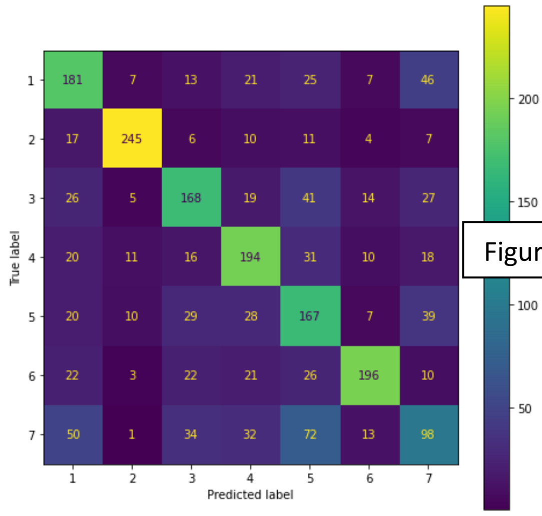


Figure 8