APPLIED MACHINE LEARNING SYSTEM ELEC0134 19/20 REPORT

SN: 14002056

ABSTRACT

In this study, a wide number of shallow learning classifiers are considered to solve such tasks as gender classification, emotion detection, eye-colour detection and face shape detection. Prior to that, literature review has been conducted to explore conventional methods that have been used for binary classification (Task A) and multi-class classification tasks (Task B). Alongside conventional methods, newer approaches involving deep learning specifically the use of Convolutional Neural Networks are also explored. [1] Overall, after having applied the models to the unseen test dataset provided for the tasks I found varying results using Linear SVM regression, with and without using a Stochastic Gradient Descent (SGD) as an optimiser on Task A1, where without using the SGD, logistic regression gave me the best test accuracy of 92.6%, but for task A2, using the SGD optimiser on the logistic regression gave the best test accuracy of 88.2%. For task B1, I found that Random Forest was the best model giving 99.9% accuracy on the test set. For task B2, again Random Forest was the best classifier but this time giving me a best accuracy of 85.0% All results have also been represented using confusion matrices. An appendix at the end has a collection of plots and results while going through the hyper parameter tuning phase.

***Index Terms*—** Logistic regression, Support Vector Machines (SVM), Stoachastic Gradient Descent (SGD), Tree classifiers, Confusion matrix

1. INTRODUCTION

A quick Google search on image classification will yield in the majority of results being deep learning solutions. Convolutional neural networks (CNNs), ever since the advent of the GPU, have taken the fore in terms of giving the best speed to performance ratio of identifying not only binary classes, but also multi-classes from images. The usage of neural networks have become even more prevalent due to the emergence of high-level popular deep learning frameworks such as Tensorflow, Keras and PyTorch. However due to the purposes of this assignment and the lack of an efficient GPU/hardware to run complex neural network layers, I will investigate "shallow" learning approaches, and provide a comparison between the different techniques. Research shown from state-of-the-art (SOTA) techniques will also be provided in the literature survey, which unsurprisingly consists mainly of neural networks. However the task of image classification is not limited to the choice of the training model. It is equally imperative to have a robust feature selection mechanism in place which is also explored in this paper.

Further, machine learning problems can themselves be grouped into supervised, unsupervised and reinforcement learning tasks. However due to the nature of this assignment having fully labelled data and the need to only predict what class (gender, emotion, face shape, eye shape) a particular data-point would fall into, we can narrow it down to a supervised learning classification problem. Therefore by allowing us to take a supervised learning approach, I explore different classifiers as well as feature extraction techniques. The primary methodology of our tasks whether binary or multiclass is represented by a flow chart in Figure 1.

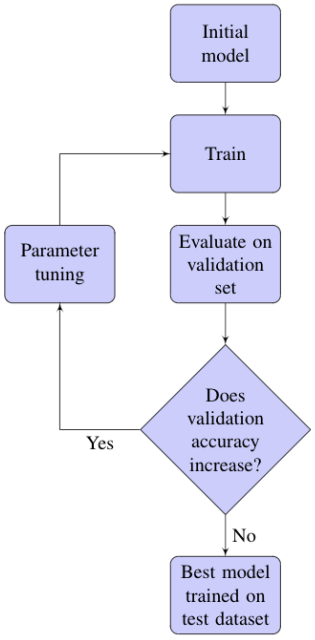


Fig. 1. Standard machine learning methodology followed

Section 2 concerns with literature survey around this topic ranging from shallow learning techniques such as support vector machine classifiers and optimisers such as stochastic gradient descent, as well as deep learning techniques involving neural networks. In the literature review, an effort has also

[1] The code for this assignment is provided here

been made to research feature extraction techniques from images such as the famous Histogram of Oriented Gradients (HOG), as well as Principle Component Analysis (PCA).

Data preprocessing techniques are discussed in Sections 3 and 4 alongside discussion of model choices and further detail into the training pipeline/methodology undertaken. The code in the Github repository provided follows the methodology alongside instructions on how to run the code and the program can be ran to verify the results achieved. In section 4 alongside the implementation, there is an in depth analysis and discussion of results that were obtained and the final section is for concluding remarks. An appendix is followed at the end to display further plots/confusion matrices of the results that were achieved while hyper parameters were being tuned.

2. LITERATURE SURVEY

2.1. Feature extraction

There are specifically two kinds of feature descriptors within an image, as advised by Shyu et al[1]: local and global. Global features are used to describe the entirety of the image, generalising the entire image as a model, whereas a local descriptor describe specific sections, corners or textures.

Dalal and Triggs[2] proposed a pioneering human detection algorithm - Histogram of Oriented Gradients (HOG) in 2005, which became the de facto standard for extracting features from images concerning humans. The innovation was one of the last 'hand-crafted' processes to extract features from an image. It does not involve any neural networks, but it simply relies on changes of the pixel values in the x or y directions as shown by equation 1 below.

$$\nabla f = \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \tag{1}$$

This gradient change is summed up linearly to calculate the derivatives of the image. It can be simplified to say that the image 'gradient' denotes change in the image in a specific direction, since the gradient is a vector function. Therefore HOG transformers have been quite useful for edge detection use cases. The original use case for the HOG descriptor was to detect human pedestrians. Since the entirety of the image is taken as part of the feature selector, HOG can be described as a global feature descriptor.

Figure 2 depicts the HOG transform. As it can be seen the edges are clearly noticed after the transform has been applied, which is where the usefulness of HOG comes in. At a deeper level, HOG implements a sliding window of arbitrary size which is narrowed down into 8x8 cells (which can be treated as a hyperparameter) where an unsigned gradient vector is generated of size 64. The histogram part of the algorithm then uses 9 bins of 20 degrees to determine at which weights should the gradient vector distribute or split its 'image' content. As the sliding window shifts across the image,

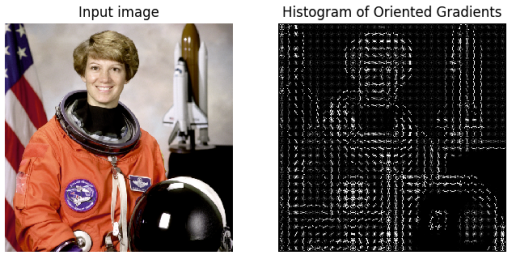


Fig. 2. Representation of the HOG transform applied to a sample image

the histogram is updated and is affected more strongly by stronger gradient vectors. While on the cartoon dataset there isn't that much of a necessity to use the HOG transform, it is necessary to use it on the celebrity dataset.

Moving to more novel approaches on feature selection, deep learning has also been used. Dara and Tumma[3] demonstrated the use of convolutional neural networks (CNN) demonstrated first by LeCun et al[4]. A CNN uses filters at the input to automatically extract features. No other data preprocessing is needed, as the hidden layers in the CNN use the concept of 'learnable' filters which then feed the data into the Rectified Linear Units (RELU), after which the processed matrices are pooled (sampled). Following this, classification is performed on the dataset.

Throughout the literature survey of related tasks, it can be seen that more solutions are adopting deep learning frameworks to preprocess, filter and extract useful features out of the raw dataset. This makes sense as usually feature engineering is the most complex part of a machine learning task.

2.2. Classification models

For classification there are also a variety of models that have been researched. Arriaga et al.[5] proposed a State of the Art (SOTA) method using CNNs in real time to classify emotion and gender. They reported accuracies upwards of 96% for gender in the IMDB dataset and 66% for emotion in the FER-2013 dataset. Figure 3 below, demonstrates the pipeline of the SOTA proposal. As it can be seen there are a total of 9 convolutional neural layers that are used to firstly extract key features from the image but then also uses Batch Normaalisation and Pooling. It can be seen similar to a sequence to sequence (CNN) modelling scheme.

The key part to this SOTA method though however is the fact that they have prioritised using as few parameters as possible while maintaining a high accuracy. This high accuracy/-parameters ratio as mentioned has a number of advantages, the main one being that it allows the use of smaller CNNs which would work well in real time on low specification hard-

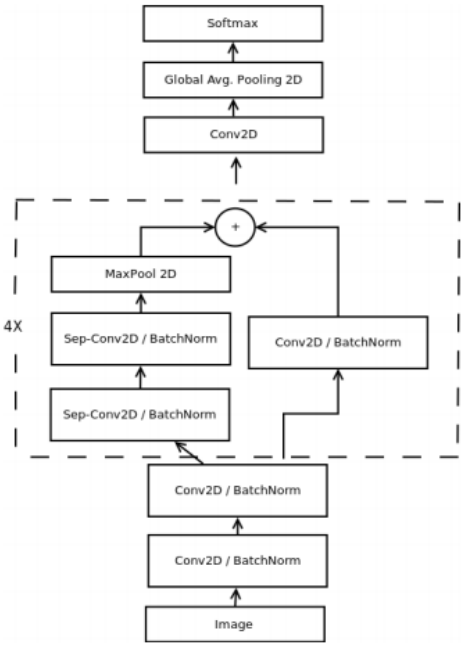


Fig. 3. Real time classification model proposal by Arriaga et al.

ware. Another philosophical approach that was taken was that using a few number of parameters allows a better generalisation of the Occam's razor framework, which basically means that the 'simplest solution is most likely the best one'.

Lemley et al.[6] in their study of the comparison between different machine learning techniques for gender recognition deduced that while there are many algorithms that allow for gender detection, they are all state of the art based on different metrics and benchmarks. They found that ultimately each classifier returns the best accuracy depending on the benchmarks used. They compared 9 different machine learning methodologies, primarily based on support vector machines (SVMs) and convolutional neural networks (CNNs), part of the deep learning paradigm, with different feature extraction methods including dual tree complex wavelet transform (DTCWT), principal component analysis (PCA), HOG and using SVM with radial basis function (RBF). Unsurprisingly CNNs outperformed the rest purely due to the high performance and nature of neural networks.

Agarwal and Sharma[7] demonstrated the use of decision trees whereby each labelled object that is described by the feature/attribute is collected into a group of properties summing up to a node. An overall leaf node represents the attribute that classifies the object. Therefore a decision tree classifies by traversing through the leaf nodes from top to bottom and Agarwal and Sharma in this paper demonstrated the use of Decision Trees for image understanding.

3. DESCRIPTION OF MODELS

3.1. Task A1: Gender detection

For the sake of simplicity, and limiting the number of model-testing combinations of the various machine learning techniques, I will opt for a stochastic gradient descent (SGD) optimiser across the shallow and deep learning approaches. This is due to the fact that mainly SGD has found renewed success around minimising the error in neural nets[4] and historically has been a predominant optimiser for linear/logistic SVMs too. Alongside SGDClassifier on linear regression, logistic regression is also used and as a comparison to the support vector machine classifier using a radial basis function (RBF) kernel, which is explored in the next section.

3.2. Task A2: Emotion Detection

As mentioned earlier Support Vector Machines (SVM) classifier is used. I experimented with different kernels for the SVM. The first, linear kernel was used and secondly RBF.

$$\mathrm{sgn}(\sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + \rho)$$

Fig. 4. SVM Decision Function

The reason why I chose to use SVMs for task A, is due to the inherent binary classifier nature of SVMs. The idea that SVMs nonlinearly project the input training data to a higher dimension feature space using the respective kernel makes sense for our task set, since we only have binary classes and we can then use a kernel to separate the data - either linearly or RBF.

Figure 4 demonstrates the decision function for the SVM, whereby we are looking to optimise the model by tuning the parameter α , or in the case of scikit-learn library, they define $\alpha = 1/C$, therefore our hyperparameter tuning focuses on the value of C. The actual library used was sklearn.svm.SVC. The Support Vector Classifier scales quadratically, so with a large number of datasets and our datapoints consisting above 10k after extracting features, I decided to perform PCA too before feeding the input vector into the SVC.

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right)$$

Fig. 5. Radial Basis Function kernel used in SVC

The RBF kernel is demonstrated by figure 5. The euclidean distance can be seen by the numerator in the exponential, but the key parameters are gamma and C. Since C is also a parameter of the linear kernel of SVC, I chose C to be the common hyperparameter to be tuned.

3.3. Task B1: Face shape recognition

For Task B I adopted the use of tree classifiers due to the multiclass nature of the tasks. Multiclass classification using random forests was proposed in the paper by Prinzie and Van den Poel[8] and the main advantage was that there was no need to standardize data which meant that more of the original input dataset was retained during the training phase.

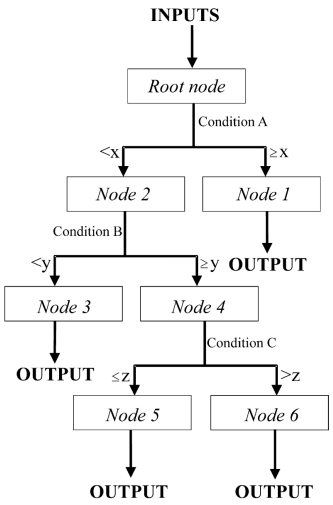


Fig. 6. Generalised Decision Tree flowchart

Figure 6 shows the general structure of how decisions are made in a decision tree in the form of a flowchart. The forms the basis of a Random Forest Classifier which is also compared within Task B.

3.4. Task B2: Eye colour recognition

A Random Forest is essentially just a collection of decision trees. However instead of an average prediction of single trees, there is random sampling of the training dataset that takes place as well as the random sampling of the features that are considered when node splitting occurs.

Furthermore, a feature extractor technique also needs to be analysed. Savakis et al.[9] in fact do an in depth study on the direct comparison between HOG and PCA and using both feature extractors, first being HOG and then add PCA for its dimensionality reduction and they found great success. Therefore I have followed the same approach as it has been proven to work on a very similar scale of the dataset.

4. IMPLEMENTATION & ANALYSIS OF RESULTS

It is important to realise key differences between the two datasets: the celebrity and the cartoon dataset. While for the celebrity dataset we are only required to detect the gender and the emotion (whether the person is smiling or not) - a binary task, for the cartoon dataset we are required to detect a specific face shape or eye colour from a variety of classes. Task A is concerned with real-life images with varying lighting in the background, whereas the task B dataset is only of non-living caricature facial shapes. This requires the manipulation of data in different ways as we can assume for gender/emotion detection there are a lot more variances that would be picked up by the pixels, whereas for cartoons we should be limited to a distinct/discrete type of information. This also makes the case for the use of a HOG transform in Task A dataset but not necessary in Task B.

4.1. Data pre-processing

Image data must be standardised/normalised as part of the preprocessing phase. This is to ensure that as each pixel is traversed by the model, convergence can be reached quicker as all the pixels would roughly have the same data distribution. For example from an array of $[0, 255]$ which is a standard colour channel to a $[0, 1]$ would drastically improve the convergence rate. Dalal and Triggs[2] demonstrated that Histogram of Oriented Gradients (HOG) based approach outperformed significantly all the existing feature selecting techniques from images to detect humans. However HOG was used on a dataset in a more recent study from Lemley et al.[6] which demonstrated that in fact principle component analysis (PCA) was a better feature extractor in a 1 to 1 comparison.

4.2. Task A: Binary classification

4.2.1. A1: Gender detection

In the interest of computational time, different hyper parameters were tuned according to the different task until a satisfactory level of accuracy was achieved. For A1, since I was using the SGDClassifier using a linear and a logistic regression as an estimator, I tuned the tolerance level hyperparameter. Firstly however I made no change to hyper parameters to get raw results by the default values. I found that SGDClassifier on a linear regression was the 2nd best in terms of performance but much quicker than the logistic regression model which was the most accurate. I constructed a training pipeline, cross validation and different models were compared against each other. This automated training pipeline was extremely efficient for me to plot different results from models. Figure 7 shows a grid search block of code which allowed me to have an N fold cross validation task set up and each iteration (the model that was being trained) would return with an accuracy score that would determine the best model out of the grid search.

```
A1Pipeline = Pipeline([
        ('toGray', Rgb2Grayscale()),
        ('toHog', HogTransform()),
        ('toScale', StandardScaler()),
        ('toPCA', PCA(0.95)),
```

```
        ('classify', SGDClassifier())
    ])


grid_search = GridSearchCV(A1Pipeline,
                           param_grid,
                           cv=2, # K-fold cross validation, in this case, K = 2
                           n_jobs=-1,
                           scoring='accuracy',
                           verbose=50,
                           return_train_score=True)
```

Fig. 7. Gridsearch for training pipeline

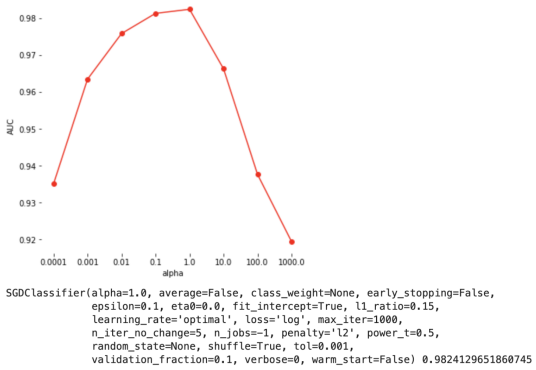


Fig. 8. Linear SVM: α tuned

Each part of the Pipeline corresponds to the class that will be called in order to perform that specific function. First we convert the input array of pixels from RGB to Grayscale, then perform the HOG Transform, followed by a standardisation. Finally a PCA transform of 0.95 is performed, as the dataset isn't that large that the computational time is too large, while still maintaining good set of features from the data. After having tried different levels of PCA (worse than 0.95), the approximation was getting a lot worse with no added benefit in computation time. The input training dataset was converted from $4000 * (218, 178, 3)$ features to $4000 * 19656$ after the PCA transform. This reduced the dimensions while maintaining the feature set.

The first model that was used was a linear SVM estimator using the SGD cost function. As mentioned, the classifier had already been set up in the pipeline as shown above and the different α values were tuned as shown by Figure 8. After this, the logistic regression classifier was also used using the same SGD optimiser.

The Receiver Operating Characteristic (ROC) curve is a very useful plot to describe how well the binary classifier has performed. It is essentially the ratio of the true positive rate (TPR) and the false positive rate (FPR). The Area under Curve (AUC) plot is also shown which is essentially the cumulative function of the ROC.

The regularisation hyperparameter C in logistic regression was also tuned and the results of the AUC curve are shown in the appendix. However the performance of the logistic regression was not up to par as compared to the linear SVM. The detailed plots are shown in the appendix. In the interest of keeping the report within the page limits, only the best results have been plotted as part of the implementation stage.

This graph (Figure 8) shows that as the alpha parameter, which is both a regularisation and a learning rate hyperparameter, is changed from 0.0001 to 1000, the AUC score reaches its maximum and then dips again. The AUC curve shown in Figure 9 correlates that there is very high accuracy, as portrayed by figure 8 using $\alpha = 1$ and epochs = 100.

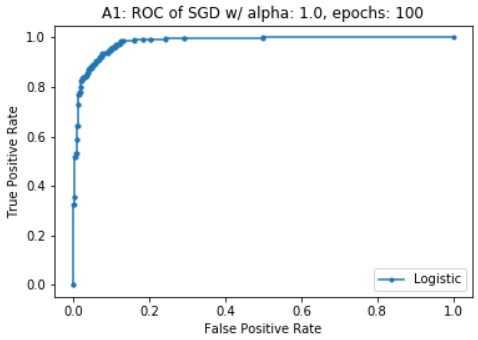


Fig. 9. A1: ROC curve for best hyperparameters

4.2.2. A2: Emotion detection

Because of the fact that the emotion detection and gender classification tasks are both binary, we can still approach a shallow learning task using libraries from Scikit-learn. A similar approach was taken for task A2. This time I tuned the regularisation hyperparameter C in logistic regression, as already discussed, it's often the most important parameter to tune in terms of the strength of regularization. Predicting on the validation set without any tuning and using default values, we saw an 87.6% accuracy. Using the 'Parfit' library which is a fork of the GridSearchCV class, I was able to extract results and plot graphs as each model/hyperparameter was tuned. This is shown in Figure 11 for the logistic regression estimator being tuned under the Stochastic Gradient Descent optimiser whilst its α and 'max-iter' values were tuned. α as already mentioned is the regularization parameter, and can be inferred as the inverse of C. The 'max-iter' value is simply the number of epochs (the number of times that the model traverses through the entire dataset) before computing an average.

After a linear kernel on the SVM, I tried to use a different SVM solver: the RBF. However since it's the most complex and has complexity is $O(n^3)$, it becomes computationally impossible to compute with the large number of features that were obtained following the HOG transform. Even after re-

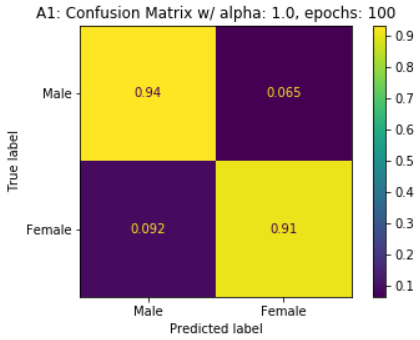


Fig. 10. A1: Best Confusion Matrix

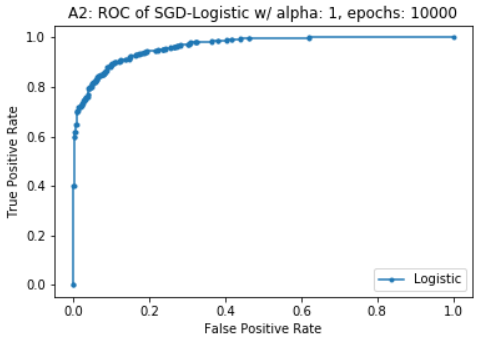
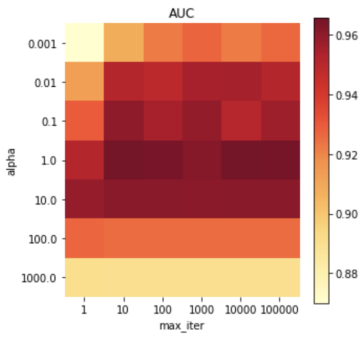


Fig. 12. A2: ROC curve for best hyper parameters

ducing the number of features using a low PCA (0.65) value, I was obtained 19656 features per input datapoint - still way too much to solve using the RBF kernel.

For a Support Vector Classification (SVC) model using the RBF kernel(SVM using RBF), it is extremely hard to get an ROC curve since it uses Platt Scaling to compute the probabilities and its computationally very expensive. I noted that it was taking longer than 30 minutes on a 64GB, 8 core machine for one computation therefore only the linear kernel for SVM was tested.

Between linear SVM and logistic regression, we can see logistic did much better. The comparison plots are shown in the appendix, but using a larger epoch of 10000, I was able to increase the accuracy of the model while maintaining the $\alpha = 1$ which was the best value. Linear SVM showed very mixed results across the board using grid search by tuning the hyperparameters. A comparative plot of Figure 11 is provided in the appendix. Figure 12 shows the best ROC curve obtained for the hyper parameters that were tuned which led us to the confusion matrix produced in Figure 13.




```
SGDClassifier(alpha=1.0, average=False, class_weight=None, early_stopping=False,
              epsilon=0.1, eta0=0.0, fit_intercept=True, l1_ratio=0.15,
              learning_rate='optimal', loss='log', max_iter=10000,
              n_iter_no_change=5, n_jobs=-1, penalty='l2', power_t=0.5,
              random_state=None, shuffle=True, tol=0.001,
              validation_fraction=0.1, verbose=0, warm_start=False) 0.9657710710342289
Logistic: ROC AUC=0.964
Percentage correct:  89.6
```

Fig. 11. Logistic regression tuned parameters comparison

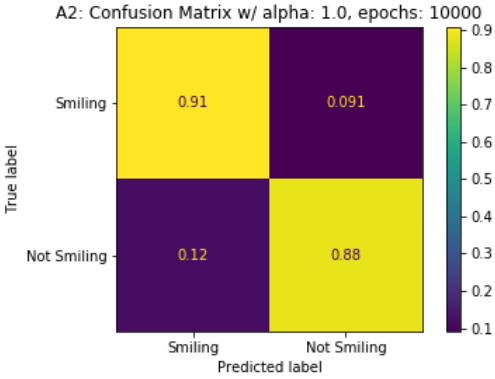


Fig. 13. A2: Best Confusion Matrix

4.3. Task B: Multiclass classification

Due to the fact that the cartoon dataset images each were much larger than the celebrity dataset, I could not reuse the same data processing pipeline for Task B as I did for Task A.

Therefore additional transformation needed to take place at the image input layer. I resized the 500px square images to 175px square images which allowed me to compute and store the image vectors into memory much quicker. I minimised a 1m+ length feature set tenfold. As seen by figures 14 and 15, even though to the naked eye, the difference is negligible, it makes a huge computational difference.

4.3.1. B1: Face shape recognition

I obtained 133,956 features per image input after the HOG transformation. I applied the same logistic regression model using the SGD optimiser on the B1 dataset at the specific hyperparameters. This came out to be unusually high, which meant that the model had been overfitted I felt, and therefore would not perform well on the test dataset.

For that reason, I decided to use tree classifiers for better reliability and to eliminate potential for overfitting. After resizing (minimising the features) to 35% of the original scale,

Fig. 14. 500x500 image

Fig. 15. 175x175 image

accuracy went down to 58.9% which was as expected at a low depth, $d = 2$. whereas at SGD, with the original 500*500 image, we saw 99.95% accuracy, but computationally it was extremely slow. It was taking roughly 10 minutes per iteration of one hyperparameter. With decision tree, it took roughly 30 seconds per depth.

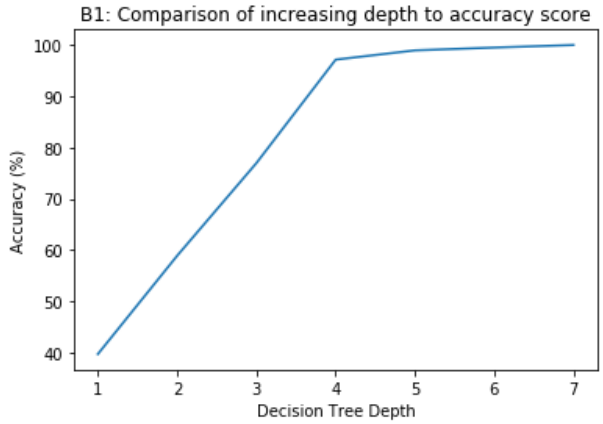


Fig. 16. Task B1: Effect of increasing depth on accuracy score on test dataset.

Figure 16 shows the plot of the increasing decision tree depth and how it affects the accuracy of the model on the validation set. Best confusion matrix was found at tree depth 7 when it started to plateau in terms of accuracy score as shown by Figure 17. Further plots are also attached in the appendix that demonstrate the increasing accuracy as more depths are calculated.

4.3.2. B2: Eye colour recognition

Since Decision Trees classifier seemed to provide a strong shallow learning fit with its accuracy and ability to tune the hyperparameter of 'depth', I decided to use that again as the main shallow learning task for B2. As expected after a certain depth of trees, the accuracy score remains unchanged, as portrayed by Figure 18. Fig. 19 illustrates the best confusion matrix that was found using $d = 8$ as its parameter. The tree

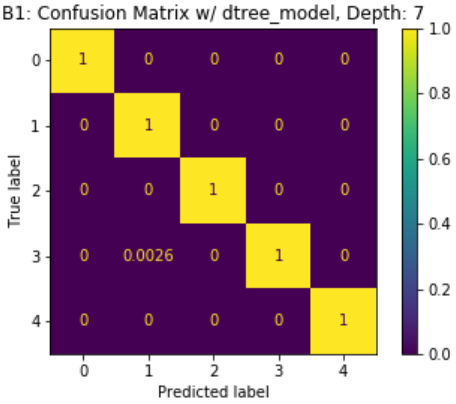


Fig. 17. Task B1: Using Decision Tree, best accuracy of 99.95% w/ depth = 7.

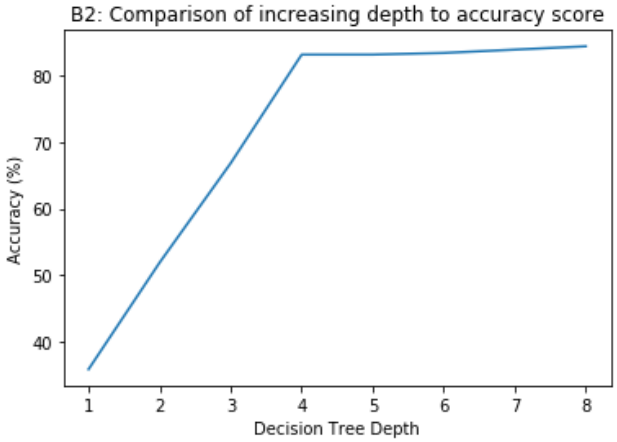


Fig. 18. Task B2: Effect of increasing depth on accuracy score on test dataset.

classifiers, I found have become the obvious choice for multiclass classification since they trained on the datasets very quickly; I was noticing around 15-20 seconds per depth scan.

However, still there's quite a difference between the highest accuracy scores of task B1 and B2, even though the datasets are the same. The difference however is that in B2 we're analysing eye shape. Upon a quick glance through the dataset I noticed that the eye colour not easily being seen from the pixel information as there were cases where glasses were being worn. Upon a future improvement might be the case where we take an unsupervised approach and try and see if a k-means approach can distinguish a person's face wearing glasses with a particular eye colour or not.

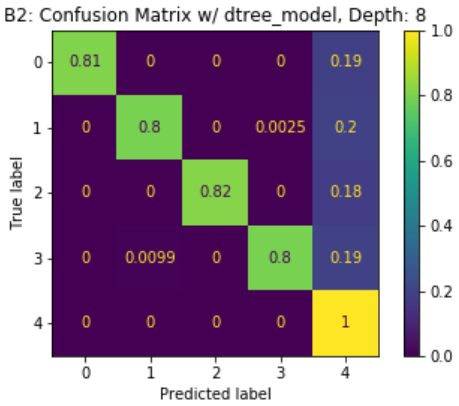


Fig. 19. Task B2: Using Decision Tree, best accuracy of 84.5% w/ depth = 8.

Task	Model	Train Acc	Val Acc	Test Acc
A1	Logistic	90.2	91.5	92.6
A2	SGD: Logistic	86.4	87.0	88.2
B1	Random Forest	96.0	99.7	99.9
B2	Random Forest	80.0	84.2	84.0

Table 1. Final results

5. CONCLUSION

Table 1 shows the final results achieved on the test dataset that was completely unseen to the model. It shows an impressive accuracy with random forest on the B1 task, but as expected not as strong on B2 relatively to the other tasks. In order to reproduce the above results please run the code as provided in the Github repository. It also details each model training time as I have set the GridSearch verbose level to 50 to better display as much information while the training task is taking place. Unexpectedly the test dataset accuracy is higher than the validation set accuracy. However, considering this was a shallow learning approach all throughout, accuracies of 90%+ are commendable which leads us to believe that traditional optimisers and estimators still hold their value even in the advent of deep learning techniques.

It was mentioned earlier in the report that taking these tasks as anything other than a supervised learning task might be pointless. However having conducted the investigation and looking deeper into the datasets, one can take an unsupervised learning approach to a task such as emotion detection. A 'smile' or any other emotion can be seen as something that can be subjective depending on who's looking at the smile or 'analysing' the emotion. Of course, there will be cases where if a person is smiling / laughing, that can be categorised as a blatant smile, however there were cases in the dataset where two different humans may judge differently as to whether a person is smiling or not (e.g. famously the Mona Lisa). In such a case I propose an unsupervised learning task whereby such images which are not obvious are collected into a dataset and then an unlabelled clustering approach is taken. It would be interesting to see if any clusters have been formed. Of course this is something of the deep learning magnitude where a large number of hidden layers might need to be constructed. Picking up subtle emotions would be hugely beneficial for business applications in the world of marketing or advertising.

6. REFERENCES

[1] Chi-Ren Shyu, CE Brodley, AC Kak, Akio Kosaka, A Aisen, and L Broderick, "Local versus global features for content-based image retrieval," in *Proceedings. IEEE Workshop on Content-Based Access of Image and Video Libraries (Cat. No. 98EX173)*. IEEE, 1998, pp. 30–34.

[2] Bill Triggs Navneet Dalal, "Histograms of oriented gradients for human detection," *International Conference on Computer Vision Pattern Recognition*, p. pp.886â893, 2005.

[3] S. Dara and P. Tumma, "Feature extraction by using deep learning: A survey," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, March 2018, pp. 1795–1801.

[4] et al. LeCun, Yann A., "Efficient backprop.," *Neural networks: Tricks of the trade*, p. pp.9â48, 2012.

[5] Octavio Arriaga, Matias Valdenegro-Toro, and Paul Plöger, "Real-time Convolutional Neural Networks for Emotion and Gender Classification," *arXiv e-prints*, p. arXiv:1710.07557, Oct 2017.

[6] Joseph Lemley, Sami Abdul-Wahid, Dipayan Banik, and Razvan Andonie, "Comparison of recent machine learning techniques for gender recognition from facial images," 2016.

[7] C. Agarwal and A. Sharma, "Image understanding using decision tree based machine learning," IEEE, 2011, pp. 1–8.

[8] Anita Prinzie and Dirk Van den Poel, "Random Forests for multiclass classification: Random MultiNomial Logit," *Expert Systems with Applications*, vol. 34, no. 3, pp. 1721–1732, 4 2008.

[9] Andreas Savakis, Riti Sharma, and Mrityunjay Kumar, "Efficient eye detection using hog-pca descriptor," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 9027, 02 2014.

A Appendix

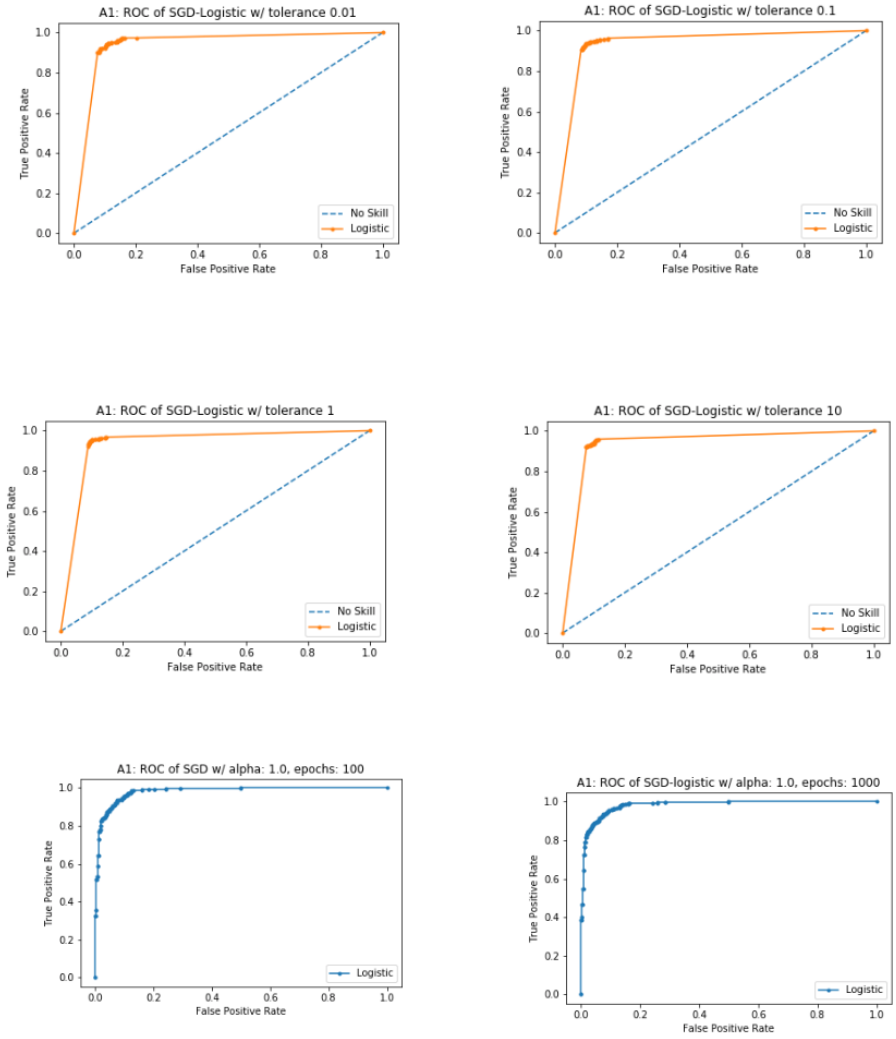


Figure 1: A1 extra results

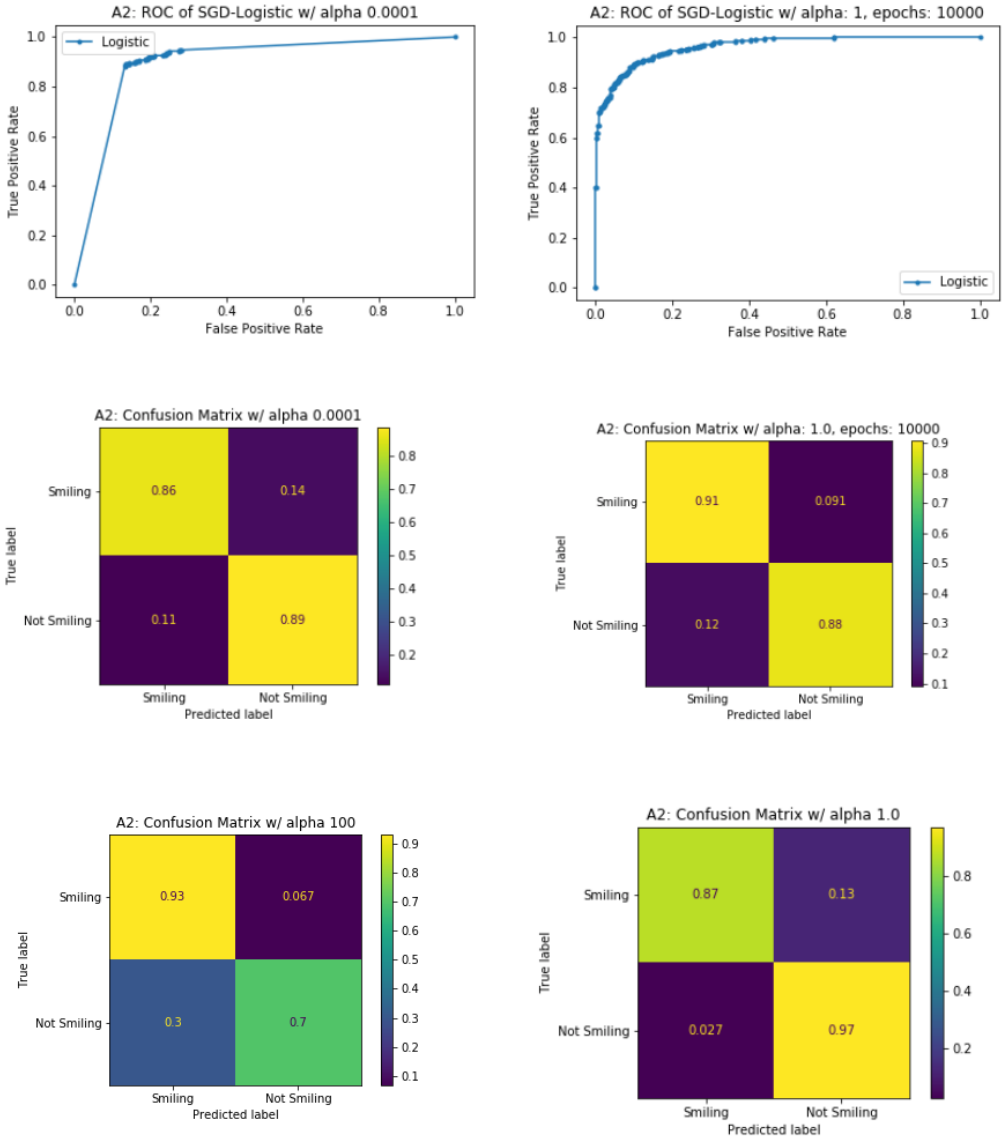


Figure 2: A2 extra results

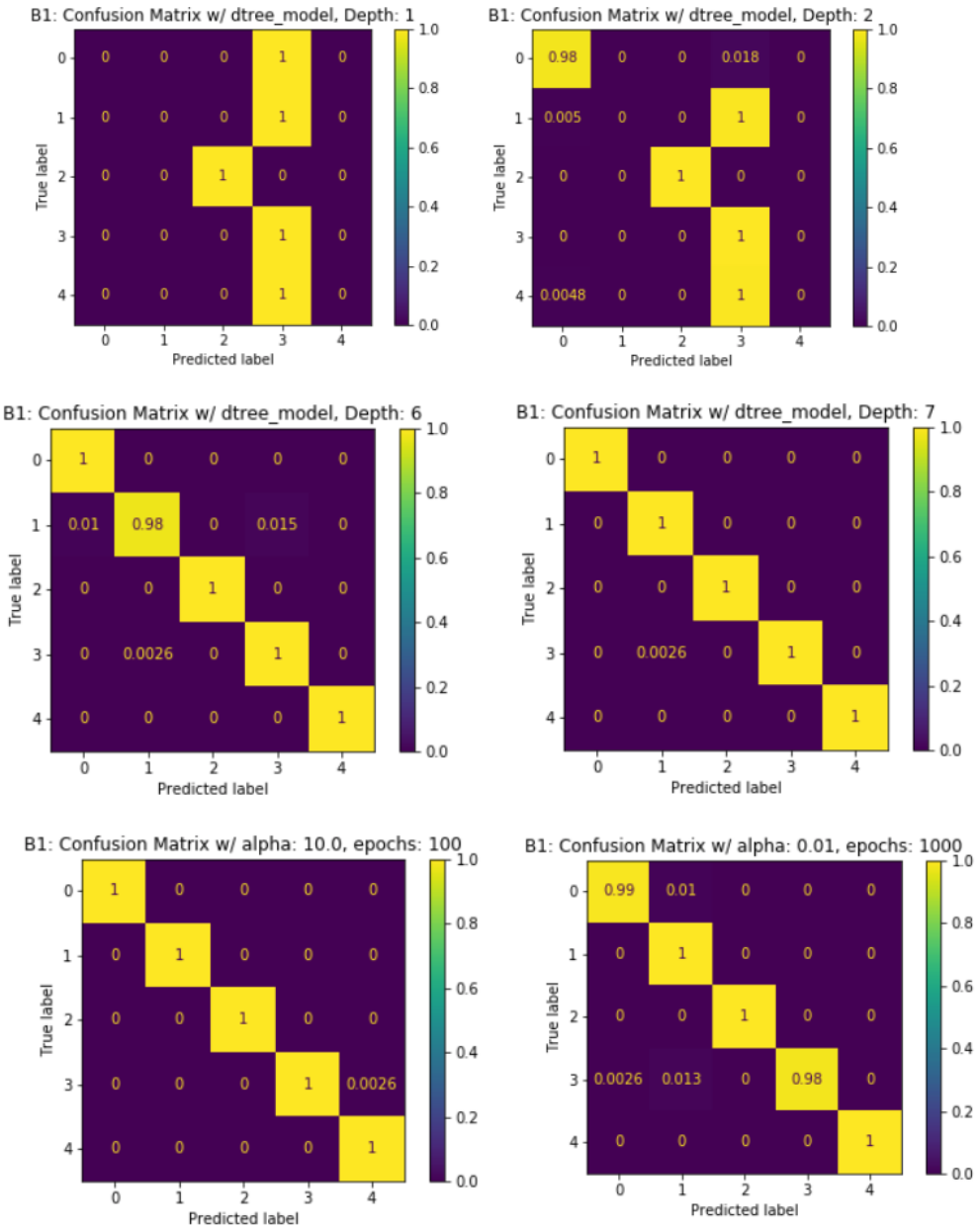


Figure 3: B1 extra results

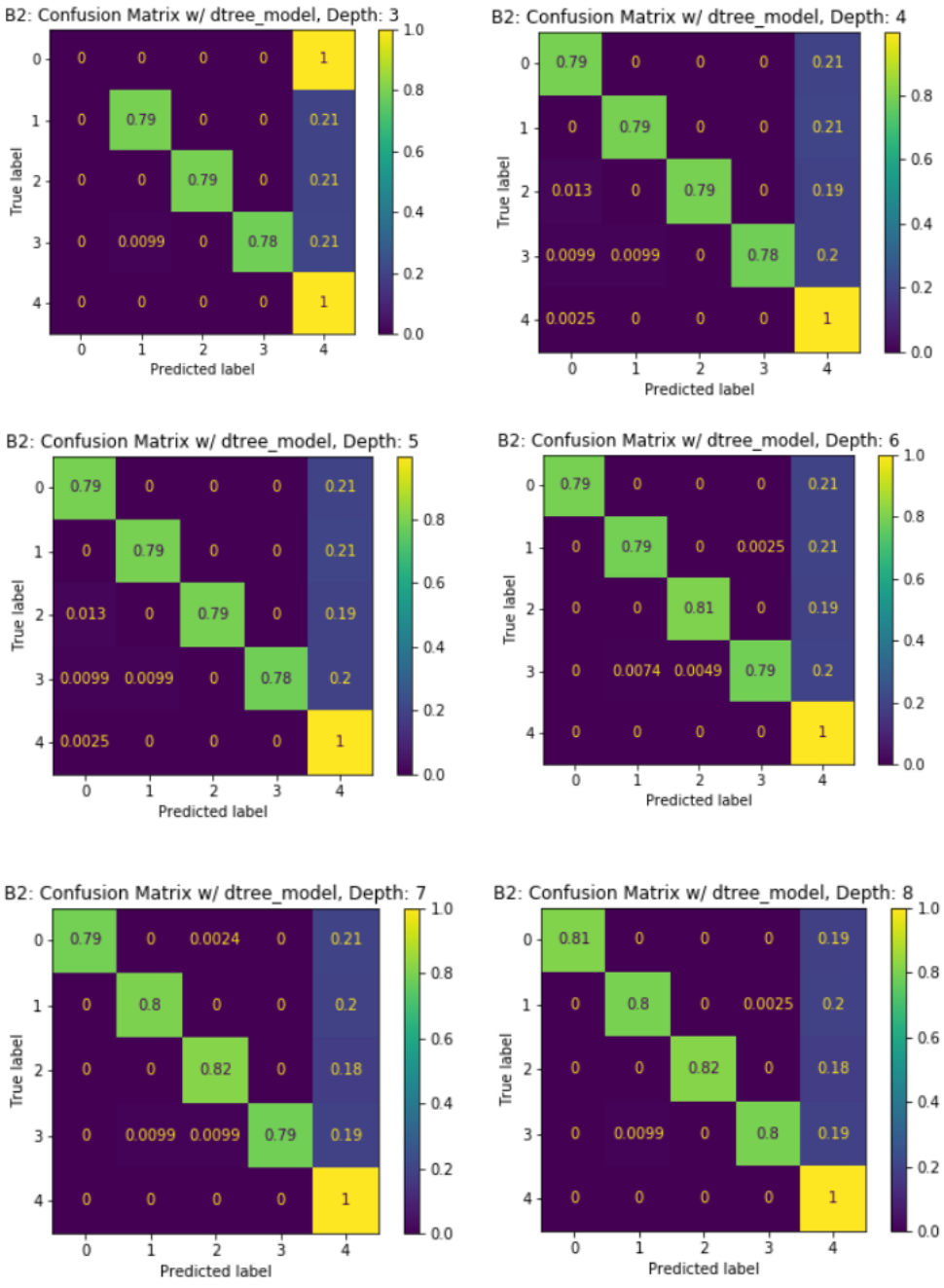


Figure 4: B2 extra results