IN3060/INM460 Computer Vision Coursework report

- Rohit Sunku (210008043) – PG
- https://drive.google.com/drive/folders/1Lm6SPS6ueW0sOuBZxwTbaxvB1n5Q5ZEc?usp=sharing

Data

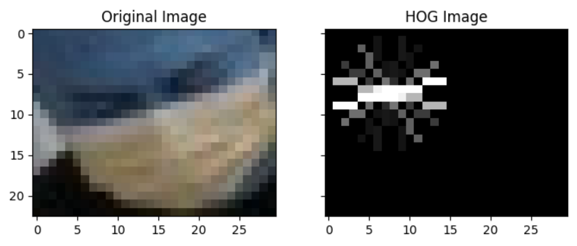
In this project, we divided the images into train and test sets containing 2394 and 458 images, respectively. Each set has two subfolders, Images and Labels, containing .jpeg images and .txt label files. The images depict people during the COVID-19 pandemic wearing masks either fully, partially, or not at all, labelled 0, 1, and 2, respectively. This task involves multiclassification of images. To prepare the data, we pre-processed the images and trained our models using the labelled training data. We then evaluated the models using the labelled testing images. The dataset also contained some images which had poor resolution and were smaller in size. We scale all the images to the same size as a pre-processing step. To use TensorFlow and PyTorch implementations, we had to modify the folder structure to accommodate the requirements of image data generators and image datasets. This included creating subfolders for each class containing their respective images and labels (0, 1 and 2). These modifications allowed us to run PyTorch and TensorFlow neural network models.

There is notable class imbalance in this dataset, with their being 376(0), 1941(1) and 78(2) in the training set and 51(0), 388(1) and 19(2). There are also some incorrect images within the training and testing data for class 2, with some images having people with no masks/ there faces had masks partially fitted on.

Implemented methods

SIFT (Scale-Invariant Feature Transforms) + KNN algorithm (using KMeans and BoVW) – The training process involves iterating the SIFT algorithm through the image data first, creating key-points and descriptors before the KNN algorithm. The main advantage is that this combo includes robustness to noise and occlusion, scale and rotation invariance, and simplicity implementation. However, it is computationally expensive, slow and memory-intensive and is sensitive to lighting conditions. KNN may get superior results with a large training sample size and only 3 different classes. The model was trained on 2394 images, with accuracy (74.2%) which only increased to 74.9% after optimisation, which only increased model performance slightly.

HOG (Histogram of Oriented Gradients) + MLP (Multi-Layer Perceptron) classifier – The same data was used on this model. A baseline MLP model was trained on the data, and it showed to have superior performance to the SIFT Algorithm of 84.7% after applying grid search optimisation on the model. Advantages include the ability to handle non-linear data and high accuracy potential, however there is a tendency for overfitting with MLP and the HOG algorithm is sensitive to variations in illumination.

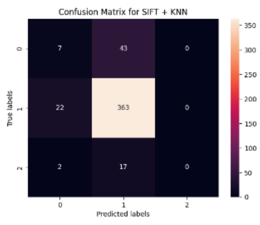


ORB (Orientated FAST and Rotated BRIEF) + SVM (Support Vector Machine) – This model has 66.0% accuracy applying grid search to the model, with its performance being lower than the HOG and SIFT models. The same data is used to train this feature-selector + ML algorithm combination. ORB and SVM has high accuracy and robustness to noise and occlusion, however SVM may not perform too well on large datasets and requires lots of hyper-parameter tuning. Furthermore, ORB may be heavily affected by variations in illumination.

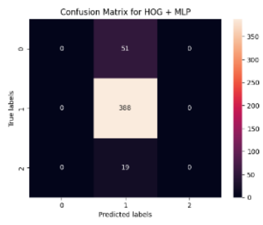
CNN (Convolutional Neural Networks) – Tensorflow Implementations – We edit the data for this model, creating new folders and sub folders for the Image Data Generator to work with images and labels aligned perfectly to work. This CNN model outperformed SIFT and ORB models but not the HOG model with 75.6% accuracy after optimisation using grid search. These have a high accuracy and ability to handle complex data, deciphering complex non-linear relationships more strongly than other machine learning models. However, careful hyper-parameter tuning is needed along with significant computational resources.

Results

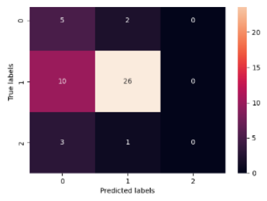
For all our feature selectors, SIFT, HOG and ORB, we do not perform a grid-search since they are instantiated from the cv2 library and are therefore not pickable.



SIFT + KNN Pipeline Optimisation – Best hyperparameters: (n_neighbours = 1, p = 2 (Euclidean distance), weights = uniform). We train three hyper-parameters on a grid search with a 5 K-Fold cross-validation, and the best hyperparameter is 1 neighbour suggesting the data is well separated in the feature space and the closest neighbour if sufficient to accurately classify the image. We can see that this pipeline cannot differentiate class 1 from class 2, and classifies people wearing the mask and people wearing the mask properly to be in the same class.



HOG + MLP Pipeline Optimisation – Best hyperparameters: (hidden_layer_sizes = (50,50), activation = ReLU ,solver = SGD(stochastic gradient descent) ,alpha = 0.01 ,learning rate = constant). Stochastic gradient descent has proven to be a better algorithm due to the size of the dataset and the simplicity of the model because it is a simpler algorithm. This pipeline fails to classify any images with labels 0 and 2, differentiating between the classes, labelling all the testing images in class 1.



ORB + SVM Pipeline Optimisation – Best hyperparameters: (C = 0.1, degree = 2, kernel = poly, gamma = scale (distance between hyperplane and closest points), class_weight = balanced). A lower regularisation parameter is chosen(C = 0.1) for our optimised model because the model decision boundary does not need too complex and a wide margin works well for this problem. The choice of a polynomial kernel with degree 2 means the kernel function can handle complex relationships between input features, and the kernel function will compute the dot product between two polynomial vectors in the original space with a degree of 2. The choice of these two parameters suggests that the data may have a simple non-linear structure, and the decision boundary can be approximated by a quadratic function.

Keras CNN Model Optimisation – Best hyperparameters : (conv_units = 128, dropout = 0.5, activation = “relu”, kernel size = (2,2), fc_1_units = 128)). The Keras library allows us to optimise our model and use sci-kit learn’s grid-search function to optimise our model using our validation data generator. Increasing the number of convolutional layer units, fully connected units, changing the activation function applied to convolutional layers to “relu” while only keeping the output activation layer as softmax and reducing the kernel size improved our model drastically. Here is a comparison of model accuracy, recall and precision between the initial model and optimised model.

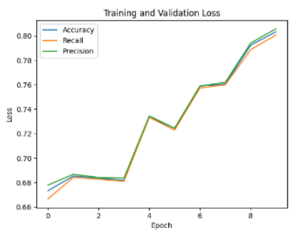


Figure 1: Initial Model (conv_units = 64, dropout = 0.5, activation = "softmax", kernel_size = (3,3), fc_1_units = 128)

Another note is that batch size had a significant impact on this model. Decreasing the batch size from 300 to 30 * improved model accuracy significantly. We modify our notebook to run all models on the GPU rather than the CPU to increase training speeds.

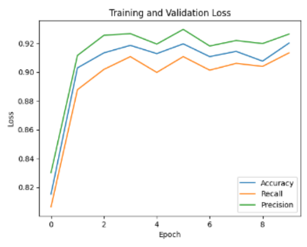
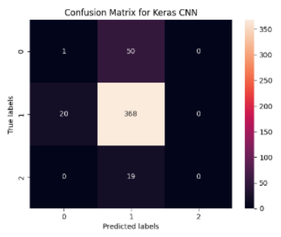
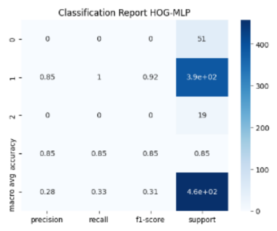


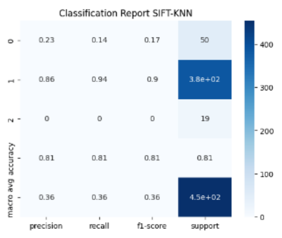
Figure 2: Optimised Model (conv_units = 128, dropout = 0.5, activation = "relu", kernel_size = (3,3), fc_1_units = 128)

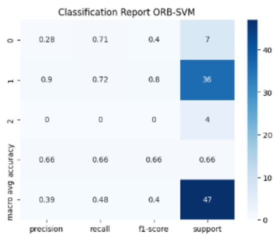


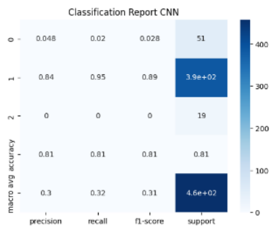
Our Keras model is simpler than our Pytorch model below, with only one convolutional layer with a max pooling layer (2x2 size), the model also adds a layer to flatten the 3D vector map into a 1D feature vector before passing through one fully connected dense layer with a dropout probability of 0.5 to prevent overfitting before a final output layer with 3 units and a SoftMax activation function (3 units for 3 output vectors).

We can see from our confusion matrix that similarly to the HOG-MLP pipeline and the SIFT-KNN pipeline, that there are a high number of false positives for Class 1 (69).









From our classification reports, we can see that HOG + MLP predicts Class 1 the best, however ORB + SVM is better at identifying class 0 images. ORB + SVM has the highest precision indicating is makes fewer false positive predictions and is more accurate at identifying positive instances, the recall score indicates the model is better at identifying all positive instances. A higher f1-score, which is the harmonic mean of precision and recall and the most robust metric for model performance.

	Hamming Loss	Accuracy Score	F1 Score (average = "micro")	Cohen- Kappa co-efficient
ORB + SVM	0.34	0.66	0.66	0.28
HOG + MLP	0.15	0.85	0.31	0.0
SIFT + KNN	0.19	0.81	0.36	0.0
Keras CNN	0.19	0.81	0.80	-0.042

From our comparison of other metrics table, these metrics can provide an insight into the performance of different feature extraction techniques and machine learning models. ORB + SVM has the highest hamming loss, indicating that it has the lowest performance of all three models evaluated as it predicts a higher fraction of the labels incorrectly. HOG + MLP and SIFT + KNN and the Keras CNN have lower hamming losses, with HOG + MLP proving to be the most effective pipeline. This is further emphasised by HOG + MLP also having the highest accuracy score (0.85 > 0.81 (SIFT + KNN), 0.85 > 0.66 (ORB + SVM), 0.85 > 0.81 (Keras CNN)). So far, we can conclude that the hierarchy of models in terms of accuracy is:

1. HOG + MLP
2. KERAS CNN
3. SIFT + KNN
4. ORB + SVM

However, Keras CNN and ORB + SVM have higher F1 score, suggesting they are good at predicting other classes. Class 1 is the dominant class which is predicted most accurately by HOG + MLP however it does not predict any other classes. So far, no models can differentiate from class 2 and class 1. The Cohen-Kappa coefficient of 0.28 on the ORB + SVM pipeline suggests it performs better in terms of inter-rater agreement than other models, with a negative cappa coefficient of the keras CNN suggesting a small chance of misclassification and inter-rater disagreement however this is miniscule. Considering all metrics and also the class imbalance and errors in label 2 photos, we conclude the model hierarchy for overall performance to be:

1. ORB + SVM
2. SIFT + KNN
3. Keras CNN
4. HOG + MLP

Overall, we will conclude that the ORB+SVM pipeline is best option. Despite its low accuracy, there is notable class imbalance in this dataset, and it has proven to be the best model at distinguishing classes. It has the highest recall, precision and f1-score overall based on our classification reports. While the Keras CNN has a higher micro f1-score, this is because the f1 score calculation here is calculated by treating all classes as a single class. Looking at the macro-average f1-score, ORB+SVM has proven to the best pipeline.