Experimentation of MTCNN and SVM on Masked Faces in Both Detection and Classification

Alun Owen, Sarah Lappin and Saumya Roy

Abstract

Due to the ongoing COVID-19 pandemic, many countries have introduced mandatory use of face masks in certain settings (Royo-Bordonada et al., 2020). This paper presents a two-part model which will detect faces in images and then classify them as 'with mask', 'without mask' or 'mask worn incorrectly'. Our research investigates how an existing state-of-the-art pre-trained Multi-Task Cascaded Convolutional Neural Network (MTCNN) performs on such a dataset, and experiment with different levels of transfer learning by freezing layers or whole sub-networks in relation to performance gain. The second part of the research uses a Support Vector Machine (SVM) model to classify the faces into one of three mentioned categories and discuss the performance with various feature selections. Currently there are no research papers that experiment with different degrees of transfer learning with the MTCNN, and works into producing a classifier for whether a face mask is being worn correctly, incorrectly or not worn at all. In conjunction with the different approach of methodology, experimentation are done on an extremely small dataset which can be applicable to real life scenarios since annotation of large scale data is expensive.

1. Introduction

In March 2020, the COVID-19 pandemic was declared (WHO, a). Soon after the World Health Organization (WHO) noted that mask wearing in accordance to other actions such as keeping good hygiene is how we should mitigate the spread within communities (WHO, b). Following this recommendation, many countries have introduced mandatory use of face masks in indoor and public settings (Royo-Bordonada et al., 2020). The pandemic continues to have an unprecedented impact on daily life around the world, and has presented challenges in many areas, including in machine and deep learning.

Face detection is a well established application of machine learning to identify human faces. It has a wide range of uses including security, surveillance and telecommunications (Colmenarez & Huang, 1998). Face detection methods have developed rapidly in recent years, with deep learning methods becoming the leading technique. The Multi-Task Cascaded Convolutional Neural Network (MTCNN) is one example of a state-of-the-art method, it has achieved accuracies up to 85% (Zhang et al., 2016). The introduction of mandatory face masks presents new challenges to face detection as they occlude key features used in detection, such as facial landmarks (e.g. location of the eyes, nose, and mouth), rendering most existing technologies ineffective (Wang et al., 2020). Not only is there a huge need for existing face detection methods to be updated to detect masked faces, there is also increasing demand for technology to assist in the regulation of face mask rules as this has proved challenging for many shops and organisations (Eley & Megaw). This has the potential to have very serious implications on the health and safety of the organisations' employees and the public, so it is incredibly important that this technology is developed rapidly and effectively. For these reason, the experiments in this paper are aligned in motivation with producing better face mask enforcement to help improve public safety in the current COVID-19 pandemic.

The current state-of-the art methods for face detection involve deep learning, which require very large datasets to prevent overfitting (LeCun et al., 2015). To date, there are no large, widely available face detection datasets which include a significant proportion of masked faces, meaning this problem cannot be solved simply by retraining on a new dataset which includes masked faces. To address this, most existing research uses a simulated masked dataset (Wang et al., 2020). These are existing facial recognition datasets with masks artificially overlaid on the face. We theorise that through manipulation of training methodology and training data, current methods can be improved for training data, without the need for artificial data.

To address both of these new problems we have developed a two-part model, which aims to (1) perform face detection on a dataset including masked faces, and (2) determine if the individual is wearing a mask correctly. We investigate how an existing pre-trained MTCNN performs on a dataset which includes masked faces, and propose changes to both the training data and methodology of transfer learning for the MTCNN to improve performance on the masked dataset. In part 2, we have developed a Support Vector Machine (SVM) to classify the faces into three categories: with mask, without mask and mask worn incorrectly, and discuss the performance with various feature selections.

The research questions behind this paper are:

1. How well does a current facial recognition method (MTCNN) perform on a dataset which includes masked faces?
2. Can the MTCNN be improved for masked faces by learning only certain parts of the architecture?
3. Can the training data be altered to create a bias towards masked faces by generally using occluded faces?
4. How well can a classical ML technique (SVM) classify faces into the three classes?

2. Data set and task

Both the mask dataset (MakeML, 2020) and the WiderFace dataset (Yang et al., 2016) will be utilised for the experimentation. All evaluations will be based on the mask dataset while both datasets will be trialled for training and validation.

The mask dataset consists of 853 real-life images of varying sizes. Each image has at least one face present. This small set is separated such that 50% accounts to the training set, 15% for validation and 35% for testing. The images are annotated with the number of faces present, with each face labelled as one of the three classes:

- With mask
- Without mask
- Mask worn incorrectly

WHO recommends masks cover the mouth and nose, and are tied *"securely to minimise any gaps between the face and the mask"* (WHO, 2020). For the purpose of this research, we consider masks to be worn correctly if the meet these recommendations.

In the dataset there are 3232 labelled 'With Mask', 717 labelled 'Without mask' and 123 samples labelled 'Mask worn incorrectly'. Each face is annotated by a bounding box. The dataset needed to be preprocessed from an xml file to a dictionary file type. Undesired attributes such as pose were processed out in the process.

In association with the mask dataset, the WiderFace dataset will be used, as it contains many more examples to learn from. With a train-validation-test split of 40% 10% 50%, Wider Face contains 32,203 images with 393,703 faces in total. We will only use the training and validation sets as we will test on a Masked Face subset. It is important to note that each face is not labelled with the same three attributes of the mask dataset, instead they are labelled with six different attributes, one namely being occlusion level. Each face is attributed to have heavy, partial or no occlusion.

The overall task undertaken will be to produce a model which detects faces with or without a mask, and then classifies as to whether a mask is being worn correctly, incorrectly or not at all. Both detection and classification will be examined exclusively. Regardless of the chosen method of training for the task, all optimum methods are evaluated with the reserved test set from the mask dataset which contains the ground truth for bounding boxes and classification.

Firstly, the detection will be done by creating bounding boxes. It will begin with our baseline, which is the pretrained model, then investigation will be done to see if it can be improved upon, based on its performance on the validation set. Comparison between different trained detectors will be done based on the f1-score. An f1-score is chosen since it is important to consider the false positives and false negatives that a model produces. The detector should produce bounding boxes for faces in a picture regardless of its dimensions. Therefore, using the model with best f1-scores over the validation set, an inference of its general performance is done on the test set.

Regarding the classifier, it uses bounding boxes as input and then produces a classification for one of three listed above. Using the mask dataset, many different arrangements of the classifier are produced with the training set, and then assessed on the validation set through a hyperparameter search. The most effective arrangement is tested on the test set while trained using both training and validation sets. Evaluation for the classifier will be based on the macro f1-score to ensure weighing is done on a class basis rather than being weighted according to each face. This is in the hope of reducing bias due to the unbalanced nature of the dataset.

3. Methodology

For classification, we require images to be annotated with the classes 'mask not worn', 'worn incorrectly' and 'mask worn'.With face detection, however, we have more flexibility in the training data we can use. Although datasets with masked faces are limited, the Wider Face dataset used in the MTCNN contains a variety of occluded faces. These faces are occluded in several different ways, including shadows, face-paint, masks, and helmets. Despite the lack of annotations on the cause of the occlusion, we can utilise this to train the MTCNN to recognise faces which are highly occluded, as they are when covered by a face mask. To allow for this flexibility in training data, the detection and classification is split into two parts, as shown in Figure 1.

3.1. Face detection

As a baseline for face detection, the multi-task cascaded convolutional neural network (MTCNN) (Zhang et al., 2016) will be implemented. This model is chosen over other alternatives since it is one of the best performing published models on the Face Detection Data Set and Benchmark[1]. In the original paper (Zhang et al., 2016), the recall of the MTCNN is significantly better on the harder test set in comparison to similar models. The difference is not

[1] http://vis-www.cs.umass.edu/fddb/results.html#pub

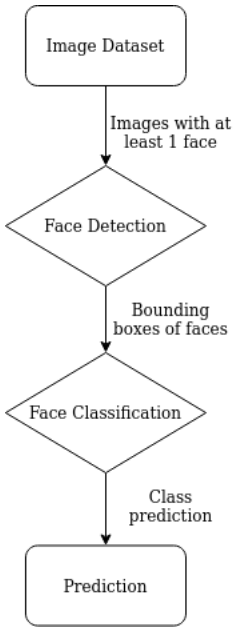

Figure 1. High-level system control flow overview

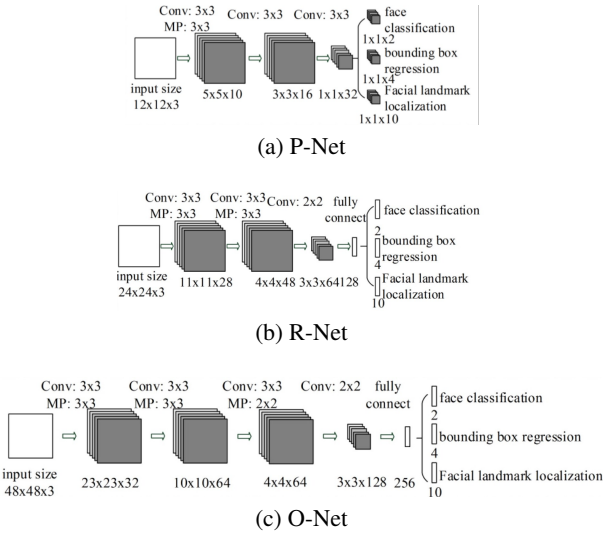

Figure 2. Architecture of the MTCNN in order used in the network. Each image shows the layers for each sub-network. (Zhang et al., 2016)

so significant on the easier test set, however, this paper will be implementing the method for a comparably hard set of images with faces occluded by masks. Regarding the output of the model, it is important to have all three of outputs (i.e. face classification, bounding box regression and landmarks regression). Contrasting to other models, the landmark recognition produces the alignment for faces in parallel with other processes. Computation of the landmarks that work in parallel with the other outputs help with detection where faces can be aligned in any direction.

The architecture of the MTCNN consists of three networks running in series. They are P-Net, R-Net and O-Net respective to their position in the MTCNN pipeline. The architecture of the three networks are seen in Figure 2 with layers for each subnetwork detailed in the subfigures. Its shallow architecture allows for quick, effective computation. The P-Net uses a fully convolutional neural network to produce its three outputs as visualised in Figure 2(a), this first stage produces a large number of possibilities for faces. It then applies non-maximum suppression (NMS) to unify similar boxes. Possible faces outputted by the P-Net is taken as input for the R-Net to refine its decision of possible faces; NMS is applied once again for the output of R-Net. Both P-Net and R-Net contrast in their dimension, however, the main difference is from final layers with fully connected layers used as the final two layers of R-Net. Again, candidate from the R-Net are taken as input for the O-Net, its architecture is similar to the R-Net but consists of an extra convolutional layer. It outputs the final bounding box and facial landmarks for parts of an image it believes a face is located. As for Section 4, some of the layers will be frozen while training to experiment in relation to their performance to the mask dataset. This is to investigate the degree of transfer learning that can be applied to the mask dataset (i.e. Fine-tuning the model by freezing, use weights as initialisation or use the pre-trained model directly).

Generally speaking for the implementation of MTCNN, there are three different variations of thresholds to consider. Firstly for training, there are three threshold values for obtaining training data for each of the sub-networks. For P-Net, randomly cropped sections of the image is compared with the ground truth, if the IOU is larger a certain threshold it is considered a suitable example. If the IOU value is between the upper bound and another threshold, it is considered a partial example. If it less than the third threshold it is a poor example. During inference for the model, three separate thresholds can be adjusted to determine whether a bounding box is declared as a face or not, these thresholds will be denoted as the inference thresholds. Each of the three thresholds are related to its sub-network. Outputs from the bounding box regression for the sub-networks are kept if they are greater than their respective inference threshold and discarded otherwise. The NMS thresholds that similarly determines how lenient the refinement is, comes subsequent to the inference thresholds. NMS thresholds are based on how the bounding boxes overlap rather than against the distribution created by the bounding boxes regression.

3.2. Face classification

In choosing a classifier, we are restricted by the size of dataset we are using. SVMs are known to be effective at generalising (Yue et al., 2003), which is essential when training with such a small dataset, and thus, an SVM was the model of choice for this task. The goal of an SVM is to find a hyperplane such that it separates the data by class such that each class is contained, and through this SVMs can intrinsically identify important features for classification.

Although SVMs are less burdened by the curse of dimen-

sionality, when working with the small datasets and high dimensional data, the effects can be observed (Kolluru, 2013). We consider feature extraction and selection to minimise its effects. Feature extraction can also reduce the run-time of the classifier (Meyer-Baese & Schmid, 2014). The goal is to reduce the dimensionality as far as possible, by creating new, meaningful features from the existing features. We analyse and compare measure-based feature and principal component analysis (PCA) for feature extraction.

Principal component analysis is a widely used technique for feature extraction and selection. In PCA, features are reorganised into a new set of uncorrelated features which maximise variance (Meyer-Baese & Schmid, 2014), the features which represent 95% of variance are then selected for the model input.

For the measure-based feature extraction several predefined features, popular in vision problems, are calculated. The extracted features are: Prewitt edges (Prewitt, 1970), coloured histogram, Haralick Texture (Haralick et al., 1973) and Hu Moments (Hu, 1962). In the pre-processsing, the pixels are scaled to ensure the images are equal sizes. Moments are therefore a useful feature do their invariance to scale and rotation. The Prewitt operator is utilised for edge detection, which allows us to understand the shapes in the image. Texture can be defined as *"a function of spatial variation of the brightness intensity of the pixels"* (Armi & Ershad, 2019). The Haralick texture is calculated on grey-scale and spatial distribution. (Haralick et al., 1973).

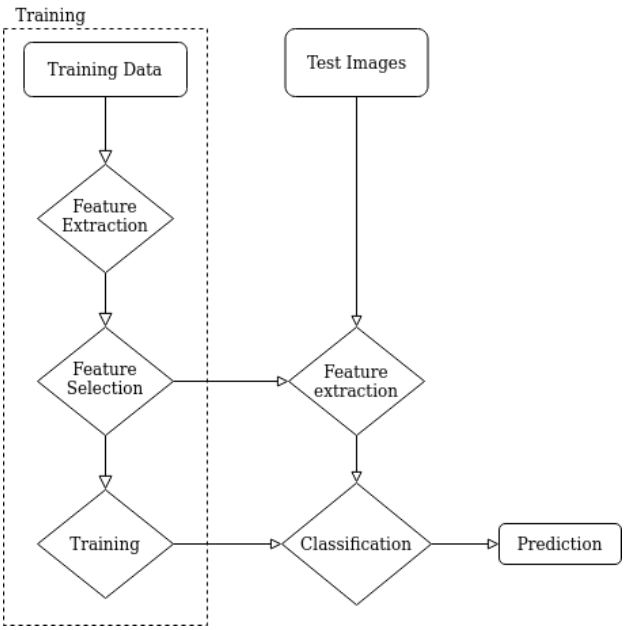

Figure 3. Control flow diagram for Support Vector Machine baseline algorithm.

4. Experiments

4.1. Detector - MTCNN

4.1.1. Freezing Layers

The baseline for the face detector is the pre-trained MTCNN face detector model. All code for the pretrained version of the MTCNN are gathered from the GitHub repository by FacialLab[2]. It was originally trained on the WiderFace and CelebA dataset. Since the mask dataset (MakeML, 2020) mostly contains masked faces, the network will be trained by freezing different layers to see if fine-tuning or training fully can better fit the masked faces than the baseline. Four different experiments for freezing layers were trained with the original hyperparameters (0.6, 0.7, 0.85) chosen by FacialLab and the experiment with the best F_1 score on the validation set was chosen. Figure 2 describes the MTCNN architecture.

For training during experiments, Adam optimizer was used with learning rate 0.01 and momentum 0.9. Due to the small size of the dataset, as found in a similar study (Samanta et al., 2020) with a small dataset, their model was overfitting by epoch 50 and was optimum between 20 and 30. Since their dataset is still larger we tried with less than 20 epochs. It seemed as 10 epochs was fitting the model reasonably after some trial and error. This was enough to reach training convergence.

To evaluate the outputs by the detector model, the IoU threshold for true positives was chosen 0.5 since the bounding boxes in the ground truth of the dataset were tighter than those outputed by the detector. Although the detected face being correct, a higher IoU threshold would label it as false positive. Such a case can be seen in Figure 4. In these images green box marks the ground truth, red box marks the output by the detector, and blue boundary around red box means the output was passed as a true positive. In the top image IoU was 0.7 which labeled a true positive as false positive which was corrected by lowering the IoU threshold to 0.5. This avoided misclassification of detected faces due to variability between annotations.

Experiment 1: In this experiment we used the pre-trained MTCNN network which was our baseline for face detection algorithm to evaluate its performance on the mask dataset.

accuracy	precision	recall	F_1 score
0.61	0.72	0.61	0.66

Table 1. Statistical metrics for network experiment 1

Experiment 2: In this experiment only the face classification, bounding box regression and the facial landmark localisation layers were trained for each p-net, r-net and o-net.

Experiment 3: For this experiment we trained the fully

[2]https://github.com/faciallab/FaceDetector

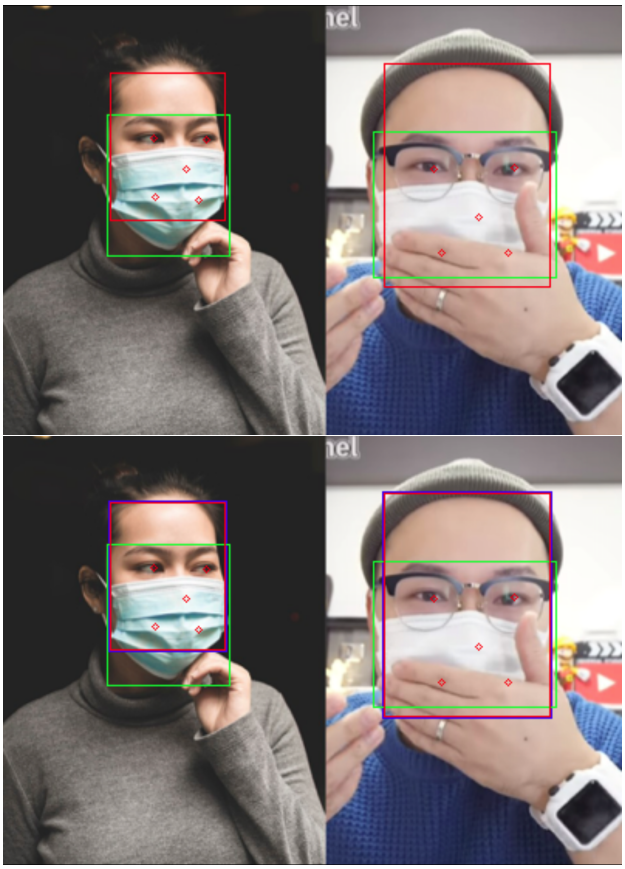

Figure 4. MTCNN output for a test image to show effect of IoU threshold.

accuracy	precision	recall	F_1 score
0.59	0.83	0.59	0.69

Table 2. Statistical metrics for network experiment 2

connected layers in the O-Net and froze the remaining layers from the pre-trained MTCNN network.

accuracy	precision	recall	F_1 score
0.57	0.87	0.58	0.69

Table 3. Statistical metrics for network experiment 3

Experiment 4: For the final experiment the entire pre-trained MTCNN network was re-trained on the train set of Face Mask dataset and tested on the test set.

accuracy	precision	recall	F_1 score
0.67	0.90	0.67	0.77

Table 4. Statistical metrics for network experiment 4

Figure 5. Example of image in occluded Wider Face subset (cropped)

4.1.2. Training on Occluded Wider Face

As discussed in Section 1, a large training set is preferable to avoid overfitting. Adding the Masked Face dataset was to the Wider Face dataset would have little impact due to the high volume of images in the Wider Face dataset, but the occlusion attribute in the Wider Face dataset can be utilised to create a sample of the Wider Face dataset with a larger bias towards occluded faces, and creating a middle ground between the percentage of occluded faces, and a large dataset. For each face in Wider Face, there is an occlusion attribute with annotations 'no occlusion' (0), 'partial occlusion' (1) and 'heavy occlusion' (2). Faces can be occluded by a number of objects or lighting, and notably for this project, Wider Face contains a small sample of masked faces (see 5) which may assist the training. For this test, we extracted images from the training and validation sets in which more than 50% of the faces are highly or semi-occluded. The breakdown of resulting datasets are illustrated in figure 6. Notice that the occluded subset dataset has significantly fairer class balance.

Set	Total	High	Partial	Not
Train	57704	22937	15663	19104
Validation	14342	5537	3978	4827

Table 5. Number of faces in the occluded Wider Face subset, with the occlusion breakdown.

The same experiments as described in Section 4.1.1 were completed on the occluded Wider Face subset. These yielded the following results:

precision	recall	F_1 score
0.71	0.61	0.66

Table 6. Statistical metrics for network experiment 1 on the occluded Wider Face subset

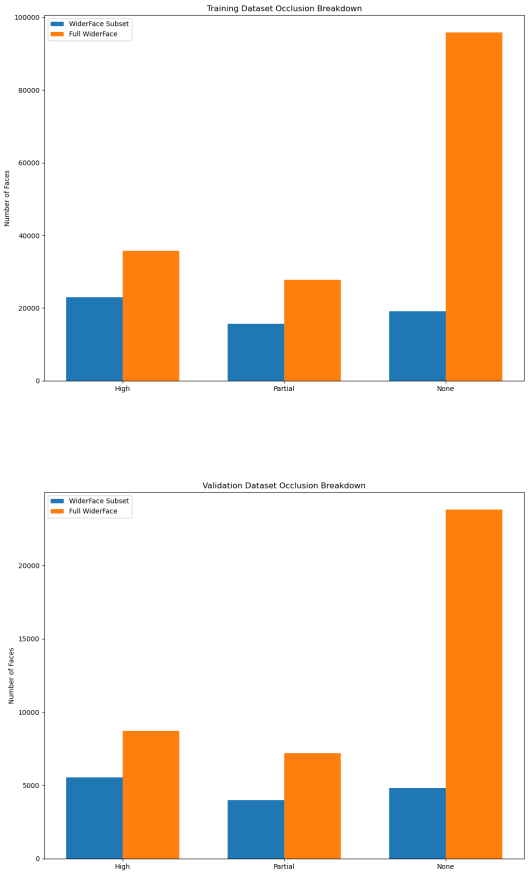

Figure 6. Validation and Training Set breakdown

precision	recall	F_1 score
0.83	0.59	0.63

Table 7. Statistical metrics for network experiment 2 on the occluded Wider Face subset

precision	recall	F_1 score
0.87	0.58	0.69

Table 8. Statistical metrics for network experiment 3 on the occluded Wider Face subset

precision	recall	F_1 score
0.92	0.62	0.74

Table 9. Statistical metrics for network experiment 4 on the occluded Wider Face subset

4.1.3. Comparison and explanation of results

From the results of section 4.1.1 we can see that the pre-trained model performs well on the mask dataset with an F_1 score of 0.66. Freezing different layers of the MTCNN model architecture can perform better on a masked-images heavy dataset than that of the pre-trained model. Experiments 2 and 3 train sub-parts of the the p-net, r-net and o-net on the masked dataset which makes the model perform slightly better. Our model is meant to primarily deal with masked face images, so it was crucial to achieve better performance on this dataset. For the final experiment, the entire MTCNN network was trained on the masked face dataset which significantly improved the model performance. This achieves a F_1 score of 0.77.

A fully trained model achieved a higher F_1 score because of the class distribution of the masked face dataset. The dataset has 3355 masked faces and 717 unmasked faces making the ratio $\approx$ 4.5:1, which leads the model to train heavily on masked faces. Thus out of the four model designs, training the entire network seems most suited for our task.

Similarly to the experiments completed on the Masked Face dataset, experiment 4 yielded the best performance with an F1-Score of 0.74 (see Table 9 when trained on the occluded subset of Wider Face. Experiment 4 achieved the highest precision of all the models, but the model trained on the Masked Faces dataset performed higher on the F1-Score metric. Although the subset of Wider Face has significantly more faces in the dataset, the ratio of masked faces is lower than in the Masked Face dataset. Using the occluded subset of Wider Face did however significantly improve the performance of the pre-trained network (Table 10, suggesting that occluded subset successfully biased the algorithm towards occluded faces.

During training, the inference thresholds by default these were set to 0.6, 0.7, 0.85 (Zhang et al., 2016) respectively. To get the best out of the network models we performed a grid search over the possible hyper-parameter values on the validation set as visualised in Figure 7. To keep the search sensible we chose a range of (0.5 - 1.0) with 0.1 increments. This grid search was done for the model design from experiment 4 with the mask dataset (the best performing model). In figure 7 we can see that if any of the threshold values are equal to 1, the corresponding F_1 score is 0 since the threshold is too tight. From the results the re-trained model performs best on the test set by achieving an F_1 score of 0.77 at thresholds 0.5, 0.6, 0.5 which is marked by the larger yellow dot. The search suggests that using a threshold higher than 0.9 will produce drastically poorer results.

4.2. Classifier - SVM

Experimentation for the SVM begins with producing a baseline, this starts with a generic SVM with the ground truth

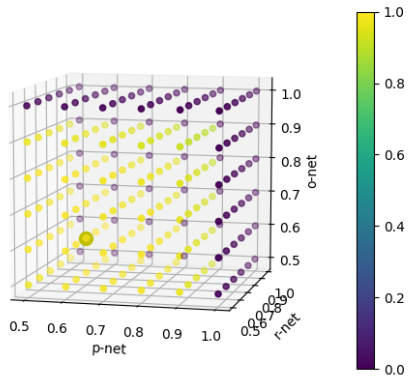

Figure 7. Grid search for Network experiment 4 based on mask dataset, the largest point is the denoted as the optimum

bounding boxes as input. To discover the best possible configuration for the baseline SVM, a hyperparameter search over three different parameters is done. The hyperparameter search is performed as a random search, with 200 iterations over the following selection of hyperparameters:

- The kernel type is chosen from the set of { linear, rbf, polynomial }
- If the kernel type is polynomial, then the degree of polynomial is chosen between { 2, 3, 4, 5 }
- A regularisation parameter which is similar to L2 regularisation { 0.1, 0.2, . . . , 5 }

Every alternative parameter for the SVM is kept to the default values [3]. For reproducibility, the random state for the SVM is set to 9. This is the setting used for all experimentations with SVM. Each iteration is trained on the training set and then evaluated on the validation set. Based on the macro f1-score, the best combination of hyperparameters for a simplistic SVM, is a polynomial kernel of degree two with a regularisation of 3.2. This configuration returns a macro f1-score of 0.7450. The returned precision, recall and f1-score for each of the classes is in the table below.

	Precision	Recall	F1-score
Not Worn	0.8296	0.9573	0.8889
Wrong	0.5455	0.3	0.3871
Correct	0.9681	0.9499	0.9589

The above table shows varying performance across the three different classes, the capability of the results are proportionate to the number of faces belonging to the classes. This reinforces the reasoning for evaluating, based on macro f1-score rather than the weighted average. In fact, the largest f1 score related to the 'Wrong' class from all configurations is 0.4865, which is still considerably lower than the other classes. If a weighted average is taken with respect to the number of instances per class, the f1-score for the best configuration would be 0.93, this is also the value for accuracy over the whole validation set using the best configurations.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Type	Kernel	Degree	Reg	F1
0	poly	2	3.2	0.7450
1	rbf	N/A	3.4	0.5865
2	poly	2	3.7	0.7450
3	linear	N/A	5.0	0.7377
4	linear	N/A	0.2	0.7143

Table 10. Best performing configuration for each type.

To further develop the classifier, the SVM will be evaluated using extracted features and principal component analysis in conjunction with the baseline. Therefore, four different architecture types will be experimented with. The baseline will be denoted as type 0 for the following experimentation. Other architectures are listed below:

1. Using only features detected (Hu moments, Haralick texture, histograms of colours and prewitt edges of original bounding boxes).
2. Both original pixels and features detected.
3. Original pixels with PCA applied.
4. PCA applied to both original pixels from bounding boxes and features detected.

For these four different ways of applying the SVM, the same hyperparameter search as that for the baseline is performed; that is the 200 iteration random search. In the same fashion, the best performing configuration is chosen, based on the macro f1-score for each type. The results can be seen in Table 10

From the results of the hyperparameter search over all types, it is evident that a polynomial kernel of degree two leads to better results when dealing with the pixels from bounding boxes, with no PCA applied. This is with or without the extracted features. It is interesting to note that the performance with just features extracted, performs at a significantly lower standard than all other types, which are in the neighbourhood of 0.71-0.75. This indicates that the SVM itself is intrinsically extracting meaningful data from the input.

Using the configurations of the best performing model on the validation set, a final model will be tested by training on both training and validation sets, and evaluated on the test set. As the best model from Type 0 and Type 2 have equivalent macro f1-scores, both will be tested on the test set. The results for the baseline is seen in Table 11, while the second architecture type has results in Table 12. On the test set, a subtle increase in the macro f1-score is seen in both models (0.7535 and 0.7581 respectively).

	Precision	Recall	F1-score
Not Worn	0.8138	0.9481	0.8758
Wrong	0.7368	0.2917	0.4179
Correct	0.9692	0.9642	0.9667

Table 11. Results on test set for best configuration of Type 0.

	Precision	Recall	F1-score
Not Worn	0.8238	0.9481	0.8816
Wrong	0.7778	0.2917	0.4242
Correct	0.9694	0.9677	0.9685

Table 12. Results on test set for best configuration of Type 2.

5. Related work

The mask dataset used for evaluation is also available on Kaggle[4] where 47 code submissions have been done using the dataset to date of this report. The best rated notebooks created for the dataset use some sort transfer learning models with the two most popular using Faster RCNN (Freitas) and VGG19(Chauhan). However, the faster RCNN doesn't evaluate on a test set while the VGG19 performs detection and binary classification of a mask or no mask, these are incomparable to the results from the MTCNN. It does nevertheless demonstrate that numerous different pre-trained models can be successful for the detecting and classifying the faces.

With respect to training the pre-trained model on the heavy occluded faces from WiderFace, a recent publication demonstrated a method for improving the performance of MTCNN with occluded faces(Bezerra & Gomes, 2018). It combines the MTCNN model with both the DLIB toolkit and homographies, it concludes that the combination of methods improve on the results of the MTCNN on its own for small datasets.

A similar construct of MTCNN and SVM for the context of face masks in another recent report (Qi & Yang, 2020). It builds on the architecture in the original paper (Zhang et al., 2016) by including another network which is applied after the R-Net, this new network works solely on the extraction of masked faces. The paper had great improvements in terms of dealing with the occlusion caused by masks in the context for facial recognition.

In relation to the goal of this paper, a prior publication (Qin & Li, 2020) was based on classification based on identical classed to that used in this paper. They implemented a super-resolution and classification network (SRCNet) that takes input as faces using MTCNN for detection. The results were very promising, reaching upwards of 98% accuracy, however, their dataset was of larger size than that used in this report. As far as we know, the dataset used is now unavailable.

For future work, it would be interesting to investigate whether our findings are equivalent on a larger dataset. To do so, a composition of smaller, relatable datasets which have sufficient information could produce better learning and enforce stronger conclusions. An example of such a dataset could be the very recent MaskedFace-Net(Cabani et al., 2021), it could at the minimum be used to ensure equal examples between classes.

[4]https://www.kaggle.com/andrewmvd/face-mask-detection

6. Conclusions

The MTCNN achieves an average accuracy of 85% on regular face detection methods. When testing this same model on the masked dataset, the accuracy drops dramatically to 61%. Despite the small dataset, when the MTCNN was trained on the Masked Face dataset, we increased this to 67% accuracy. On the Masked Face dataset, the F-1 score was increased marginally in experiment 2 and 3 by training on selective layers, but ultimately the best performance was observed by in experiment 4 by re-training all the layers.

Although the performance was significantly improved when using the occluded Wider Face training and validation sets, when compared to the baseline, training on the Masked Face dataset achieved better results. This leads us to the conclusion that more training data does not always result in better performance. When creating our datasets we must also consider the suitability of the data for the task.

The best results were observed in experiment 4, when trained on the Masked Face dataset. With hyper-parameter tuning, we were able to further improve the performance of the model to have an accuracy 70% and an F1-Score of 0.77. Despite significant improvements from the baseline, the model struggles to match the MTCNN's performance on regular face detection problems. Detecting masked faces has proved challenges with the unavailability of large real-world datasets. Until this is achieved, we encourage further research into the trade-offs between a large dataset and the proportion of masked faces. A next possible step could be to include a simulated masked dataset in the training stages.

Considering that the SVM had a disproportionate amount of data between the classes, it produced a reasonable performance. Its overall results were very promising with a score of around 90% for both dominating classes. On the test results it also achieved an f1-score of around 0.4 for the smaller sized set, this leaves plenty of room for improvement, which is highly dependent on the shortage of data for 'masks worn incorrectly'. It would be interesting to test the performance of the SVM with input from the MTCNN.

References

Advice on the use of masks in the community, during home care and in healthcare settings in the context of the novel coronavirus (covid-19) outbreak, Dec 2020. URL https://www.who.int/publications/i/item/advice-on-the-use-of-masks-in-the-community-during-home-care-and-in-healthcare-settings-in-the-context-of-the-novel-coronavirus-(2019-ncov)-outbreak.

Armi, Laleh and Ershad, Shervan Fekri. Texture image

analysis and texture classification methods - A review. *CoRR*, abs/1904.06554, 2019. URL http://arxiv.org/abs/1904.06554.

Bezerra, G and Gomes, R. Recognition of occluded and lateral faces using mtcnn, dlib and homographies, 2018.

Cabani, Adnane, Hammoudi, Karim, Benhabiles, Halim, and Melkemi, Mahmoud. Maskedface-net–a dataset of correctly/incorrectly masked face images in the context of covid-19. *Smart Health*, 19:100144, 2021.

Chauhan, Nagesh Singh. Mask and social distancing detection using vgg19. https://www.kaggle.com/nageshsingh/mask-and-social-distancing-detection-using-vgg19. Accessed: 2021-03-30.

Colmenarez, Antonio J. and Huang, Thomas S. *Face Detection and Recognition*, pp. 174–185. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. ISBN 978-3-642-72201-1. doi: 10.1007/978-3-642-72201-1_9. URL https://doi.org/10.1007/978-3-642-72201-1_9.

Eley, J. and Megaw, N. Uk retailers face up to challenge of enforcing mask wearing. https://www.ft.com/content/6af0ff7e-cabc-4dd2-bcaf-00a4f83693a2. Accessed: 2021-02-10.

Freitas, Daniel. Pytorch - fasterrcnn. https://www.kaggle.com/daniel601/pytorch-fasterrcnn/notebook#Load-Model. Accessed: 2021-03-30.

Haralick, R. M., Shanmugam, K., and Dinstein, I. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, 1973. doi: 10.1109/TSMC.1973.4309314.

Hu, Ming-Kuei. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2): 179–187, 1962.

Kolluru, Pavan Kumar. Svm based dimensionality reduction and classification of hyperspectral data. Master's thesis, University of Twente, 2013.

LeCun, Yann, Bengio, Y., and Hinton, Geoffrey. Deep learning. *Nature*, 521:436–44, 05 2015. doi: 10.1038/nature14539.

MakeML. Mask dataset, 2020. URL https://makeml.app/datasets/mask.

Meyer-Baese, Anke and Schmid, Volker. Chapter 2 - feature selection and extraction. In Meyer-Baese, Anke and Schmid, Volker (eds.), *Pattern Recognition and Signal Analysis in Medical Imaging (Second Edition)*, pp. 21–69. Academic Press, Oxford, second edition edition, 2014. ISBN 978-0-12-409545-8. doi: https://doi.org/10.1016/B978-0-12-409545-8.00002-9. URL https://www.sciencedirect.com/science/article/pii/B9780124095458000029.

Prewitt, JMS. Picture processing and psychopictorics. *Academic Press, New York*, 1970.

Qi, Chao and Yang, Lei. Face recognition in the scene of wearing a mask. In *2020 International Conference on Advance in Ambient Computing and Intelligence (ICAACI)*, pp. 77–80. IEEE, 2020.

Qin, Bosheng and Li, Dongxiao. Identifying facemask-wearing condition using image super-resolution with classification network to prevent covid-19. *Sensors*, 20 (18):5236, 2020.

Royo-Bordonada, Miguel Angel, García-López, Fernando José, Cortés, Fátima, and Zaragoza, Gustavo Andrés. Face masks in the general healthy population. scientific and ethical issues. *Gaceta Sanitaria*, 2020. ISSN 0213-9111. doi: https://doi.org/10.1016/j.gaceta.2020.08.003. URL https://www.sciencedirect.com/science/article/pii/S0213911120301990.

Samanta, Abhishek, Saha, Aheli, Satapathy, Suresh Chandra, Fernandes, Steven Lawrence, and Zhang, Yo-Dong. Automated detection of diabetic retinopathy using convolutional neural networks on a small dataset. *Pattern Recognition Letters*, 135:293–298, 2020. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2020.04.026. URL https://www.sciencedirect.com/science/article/pii/S0167865520301483.

Wang, Zhongyuan, Wang, Guangcheng, Huang, Baojin, Xiong, Zhangyang, Hong, Qi, Wu, Hao, Yi, Peng, Jiang, Kui, Wang, Nanxi, Pei, Yingjiao, Chen, Heling, Miao, Yu, Huang, Zhibing, and Liang, Jinbi. Masked face recognition dataset and application, 2020.

WHO. Speech from director general of WHO, a.

WHO. WHO's advise to the public. https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public/when-and-how-to-use-masks, b. Accessed: 2021-02-08.

Yang, Shuo, Luo, Ping, Loy, Chen Change, and Tang, Xiaoou. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Yue, Shihong, Li, Ping, and Hao, Peiyi. Svm classification:its contents and challenges. *Applied Mathematics*, 18:332–342, 09 2003. doi: 10.1007/s11766-003-0059-5.

Zhang, Kaipeng, Zhang, Zhanpeng, Li, Zhifeng, and Qiao, Yu. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.