Facial Mask Detection Using AlexNet

Niharika Chintalapati (nchint4)

Kamila Makhambetova (kamilam3)

Weiyang Wang (weiyang7)

Abstract—**Given the current COVID-19 circumstances, there has been an increased research interest focusing specifically on mask detection in facial recognition. In this report, we will be presenting one such approach, where we have used the AlexNet architecture to develop a facial mask recognition tool that aims to detect whether a face in each image is correctly wearing a mask. The model has been trained and tested on the MaskedFace-Net dataset, which contains images of human faces either properly or improperly wearing a mask, organized under various subsets. This model classifies pictures into 3 groups: properly worn masks, improperly worn masks, and no masks. In order to determine whether or not our model is successful, we judged the accuracy of our algorithm, which is able to detect not only whether or not someone is wearing a mask, but also if they are wearing the mask properly (i.e. the mask is completely covering their nose and mouth). We have incorporated 3 different configurations, which can be characterized by the batch size, number of epochs, and optimizer used. In our "best" configuration, which had a batch size of 200, total epochs of 40, and the Adam optimizer, we have attained the highest accuracy of 99.16%. For this reason, we have tuned our pre-trained model with this configuration for real-time face mask detection using a webcam camera.**

I. Introduction

Image recognition and object detection are within the larger field of Deep Learning, which aims to design neural networks that can recognize specific objects within a given image. Image recognition revolves around assigning a label to a given image or video and object detection handles cases where specific objects are identified in an image or video. There are a wide range of applications for these programs, such as automating certain verification tasks, improving security, or aiding in overall productivity by allowing one to quickly and efficiently analyze image data.

For our project, we have developed a facial mask detection tool that aims to detect whether a face in a given image is correctly wearing a mask. This model classifies pictures into 3 groups: properly worn masks, improperly worn masks, and no masks. In order to determine whether or not our model is successful, we judged the accuracy of our algorithm, which is able to detect not only whether or not someone is wearing a mask, but also if they are wearing the mask properly (i.e. the mask is completely covering their nose and mouth).

Given the current conditions due to the COVID-19 pandemic, individuals in many parts of the world must wear a mask to protect themselves and those around them. What makes our project important is that our tool has the potential to automate current masking verification processes, reducing time and human effort. Currently, just within UIUC, humans monitor building access and ensure that individuals are properly wearing masks in high-traffic areas. During class times, there are always lines forming around entrances as people wait to have their access individually verified before being let into the building. Instead, if we have a neural network in place to perform real-time analysis and ensure that the individual is wearing a mask properly, we can much more quickly process verification, avoiding long lines and wait times.

II. Related Work

In the past few years, many related studies have been conducted. In general, most of these studies have used similar data processing practices and attained close training and testing accuracies. However, each study varies based on the network implementation used.

The first study we have considered focused on feature extraction using ResNet50 and classification using decision trees, Support Vector Machine (SVM), and an ensemble algorithm [1]. ResNet50 is a CNN that is 50 layers deep and uses a pretrained network to classify images into object categories. In this study, three different datasets were used: the Real-World Masked Face Dataset (RMFD), the Simulated Masked Face Dataset (SMFD), and the Labeled Faces in the Wild (LFW). The SVM classifier had a testing accuracy of 99.64% with the RMFD, 99.49% with the SMFD, and 100% with the LFW dataset. Similar to this study, we initially also considered implementing a ResNet CNN. However, after continuously seeing low accuracies and the lack of proper implementation, we proceeded to implement AlexNet, which is relatively comparable to ResNet.

The next study we have considered developed an algorithm based on MobileNetV2 [2], which is primarily effective in object detection and segmentation for feature extraction. This study had a 96.85% accuracy in detecting whether or not people were wearing a face mask. While our project does not use MobileNetV2, we have implemented similar steps to their methodology of collecting the data, pre-processing, splitting the data, testing the model, and implementing the model.

Some other studies we have considered aimed to find viable solutions for real-time analysis. Their approaches were different from ours; however, we still considered them to learn more about potential networks and algorithms for future implications of this project. In one such study, the YOLOv3 and faster R-CNN models were used for facial mask detection [3]. YOLOv3 uses a single neural network on an image, which divides the image into regions and considers probabilities in each region for real-time object detection. The faster R-CNN uses a region proposal network to generate boxes/anchors, which are then ranked on the likeliness of containing the object. This study used images of people with and without face masks and generated results by drawing a red or green box around faces to identify whether or not a mask was present. In another study, the YOLOv4 algorithm was used in an actual device [4]. The purpose of the device was to conduct real-time analysis and generate accurate results, even if the subject was moving around.

One more study we have considered combined MobileNetV2 and Single Shot Detector (SSD) [5]. The application of this study was to develop a solution for real-time video analysis in public spaces to ensure individuals are wearing a mask and social distancing.

Another study used Max Pooling, Average Pooling, and MobileNetV2 [6]. The data consisted of over 1845 images collected from webcams and phone cameras, and the results are 96.49% training accuracy and 98.67% validation accuracy for Max Pooling, 95.19% training accuracy and 96.23% validation accuracy for Average Pooling, and 99.72% training accuracy and 99.82% validation accuracy for the MobileNetV2 architecture.

An additional study we have considered aimed to resolve the issues of the lack of relevant datasets, and small intra-class distances and large inter-class distances [7]. To address the former, the study compiled a dataset containing 8635 faces without masks, wrongly worn masks, and correctly worn masks. To address the latter, the study proposed a Context-Attention R-CNN, which extracts distinguishing features in order to expand the intra-class distance and reduce the inter-class distance. The framework attained an average precision of 84.1% on the given dataset.

The final study we have considered uses a transfer learning approach [8], which is an alternative approach we have also considered implementing for our project. The study utilized the Simulated Masked Face dataset and developed a model based on InceptionV3. Image augmentation was also used to improve the training and testing accuracies. The model was able to attain a training accuracy of 99.9% and a testing accuracy of 100%.

III. Data

For our project, we are using the MaskedFace-Net dataset [9], which consists of over 130, 000 images of human faces either properly or improperly wearing a mask. We implemented our model and trained it with a training dataset and validated it on a testing dataset. This model will classify pictures into 2 groups: properly worn masks and improperly worn masks. In order to determine whether our model is successful, we will be judging the accuracy of our algorithm, which should be able to detect not only whether someone is wearing a mask, but also if they are wearing the mask properly (i.e. the mask is completely covering their nose and mouth).

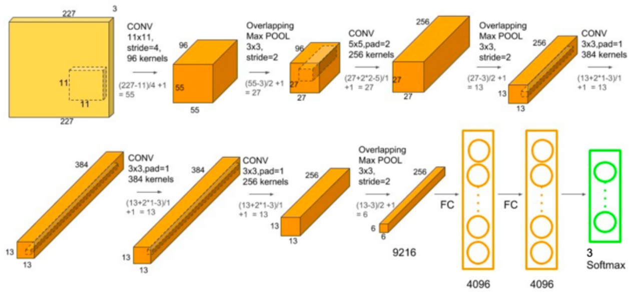
Specifically, in the "correctly masked" subset, there are 67,049 images at a resolution of 1024×1024. In the "incorrectly masked" subset, there are 66,734 images at a resolution of 1024×1024. Within the "incorrectly masked" subset, the images are further classified based on whether the chin is uncovered, the nose is uncovered, or both are uncovered. 51% of the images are "correctly masked" faces and 49% of the images are incorrectly masked faces. Within the "incorrectly masked" faces, 10% are images of an uncovered chin, 80% are images of an uncovered nose, and 10% are images of both an uncovered nose and chin. All the images are in JPG format.

Correctly Masked
Incorrectly Masked (chin uncovered)
Incorrectly Masked (nose uncovered)
Incorrectly Masked (nose and chin uncovered)

During the step of extracting the data, we first imported the data. Then, we transformed the data by resizing the images, transferring them to Tensor, and normalizing. Next, we determined the number of observations and split the data into the training set and the testing set. During the preliminary stage, 60% of the dataset was used to train the model and 40% of the dataset was used to test the model. Finally, we have also specified the batch size and setup loaders.

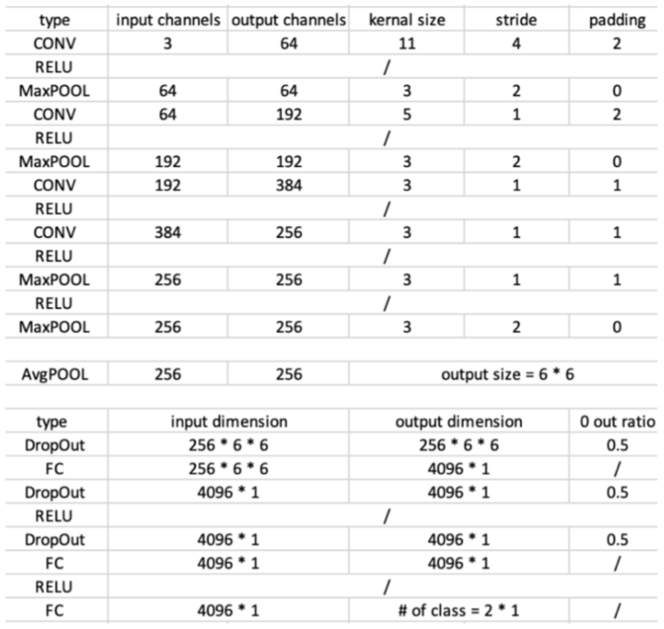
IV. Method

We have implemented a CNN with AlexNet architecture, which can be modeled as the following diagram [10]:


227
3
227
11
11
CONV 11x11, stride=4, 96 kernels
(227-11)/4 +1 = 55
96
55
55
Overlapping Max POOL 3x3, stride=2
(55-3)/2 +1 = 27
96
27
27
CONV 5x5,pad=2 256 kernels
(27+2*2-5)/1 +1 = 27
256
27
27
Overlapping Max POOL 3x3, stride=2
(27-3)/2 +1 = 13
256
13
13
CONV 3x3,pad=1 384 kernels
(13+2*1-3)/1 +1 = 13
384
13
13
CONV 3x3,pad=1 384 kernels
(13+2*1-3)/1 +1 = 13
384
13
13
CONV 3x3,pad=1 256 kernels
(13+2*1-3)/1 +1 = 13
256
13
13
Overlapping Max POOL 3x3, stride=2
(13-3)/2 +1 = 6
256
6
6
9216
FC
4096
FC
4096
3 Softmax

Describing our approach at a high level, the neural network first went for the feature extraction stages, containing 5 convolutional layers combined with the activation layers of ReLU and Max Pooling layers. Next, the classification stages used fully connected layers to classify the images through the extracted features by the convolution layers.

More specifically, in our model, we first extract features by convolutional layers. For each single convolutional layer, the network implements a kernel and scans through the input image by calculating the dot product. Given that the image is passed into the model as a two dimension array, where the values correspond to the color of each pixel, we are able to set different matrices of the kernel and extract the horizontal/vertical edges and other features of the human face, such as the eyes or nose. After each convolutional layer, we have a pooling layer to do down sampling, reducing the amount of data, while keeping the critical features. The ReLU layer, which comes after, serves as a solution to the problem of vanishing gradients and prevents overfitting by adding sparsity and nonlinearity to the network. We have also used Max Pooling layers to progressively reduce the spatial size of the network, while keeping the crucial features. After that comes the classification part, which is similar to the traditional deep neural network used for classification, except that we have the feature to be extracted from the convolutional layer instead of the value of single pixels directly obtained from the graph. Here, we have fully connected layers for classification.



type	input channels	output channels	kernal size	stride	padding
CONV	3	64	11	4	2
RELU			/		
MaxPOOL	64	64	3	2	0
CONV	64	192	5	1	2
RELU			/		
MaxPOOL	192	192	3	2	0
CONV	192	384	3	1	1
RELU			/		
CONV	384	256	3	1	1
RELU			/		
MaxPOOL	256	256	3	1	1
RELU			/		
MaxPOOL	256	256	3	2	0
AvgPOOL	256	256	output size = 6 * 6		

type	input dimension	output dimension	0 out ratio
DropOut	256 * 6 * 6	256 * 6 * 6	0.5
FC	256 * 6 * 6	4096 * 1	/
DropOut	4096 * 1	4096 * 1	0.5
RELU	/		
DropOut	4096 * 1	4096 * 1	0.5
FC	4096 * 1	4096 * 1	/
RELU	/		
FC	4096 * 1	# of class = 2 * 1	/

We have used the hyperparameters batch size, total epochs, and learning rate in the following configurations/combinations of our model:

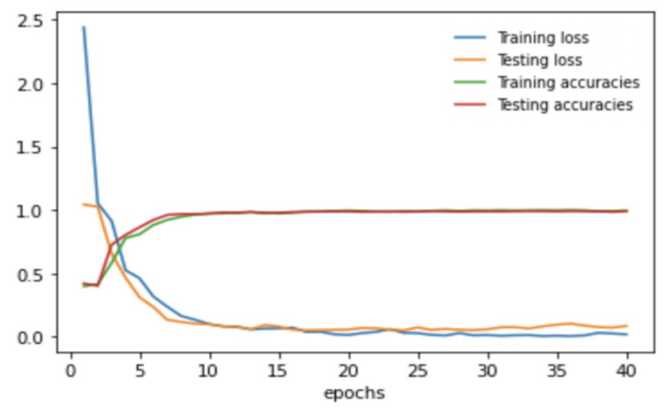
Configuration	Batch Size	Total Epochs	Learning Rate	Optimizer
1	200	40	0.001	Adam
2	300	40	0.0001	SGD with 0.0005 weight decay and 0.9 momentum
3	180	40	0.001	Adam

The two optimizers we have used in our model are the Adam optimizer and the Standard Gradient Descent (SGD) optimizer. Typically, the "better" optimizer is the Adam optimizer as it uses sparse features to converge more quickly. However, the SGD optimizer is better at generalization. Additionally, we opted to optimize the cross-entropy cost as we are dealing with a multiclass classification.

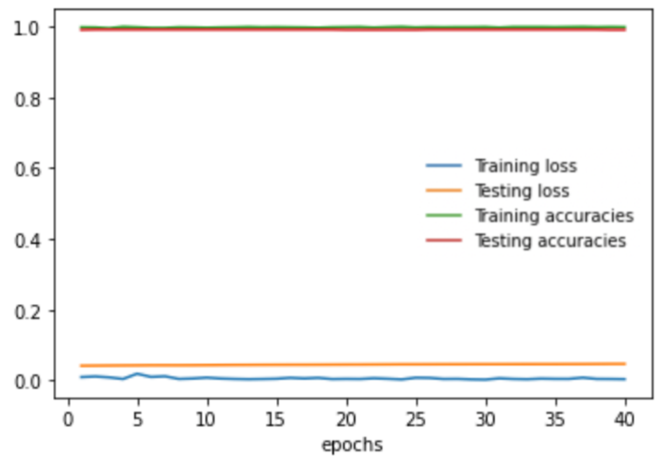
An alternative approach is to use transfer learning. In that case, we would still use convolutional layers for feature extraction and fully-connected layers for classification. However, the difference is that we would be using a pre-trained architecture and weights, as we initialize the convolutional layers with the trained weights and freeze these convolutional layers. As a result, the weights will not be updated in the latter stage of training and we would only be training the fully connected layers, with the last layers modified to match the result class we desire. The approach can be implemented alongside ResNet18 or YOLOv3. Each process is its own distinct, well-designed architecture with dramatically well-trained weights.

V. Results/Discussion

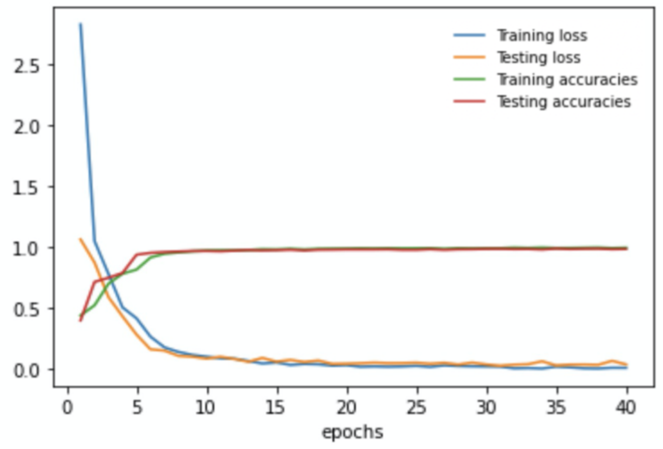
In our first configuration, we obtained the highest accuracy of 0.9916 or 99.16%. The overall losses and accuracies have been plotted below:



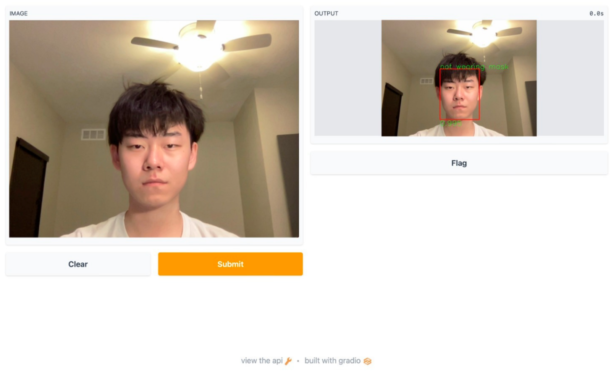
In our second configuration, we obtained the highest accuracy of 0.9915 or 99.15%. The overall losses and accuracies have been plotted below:

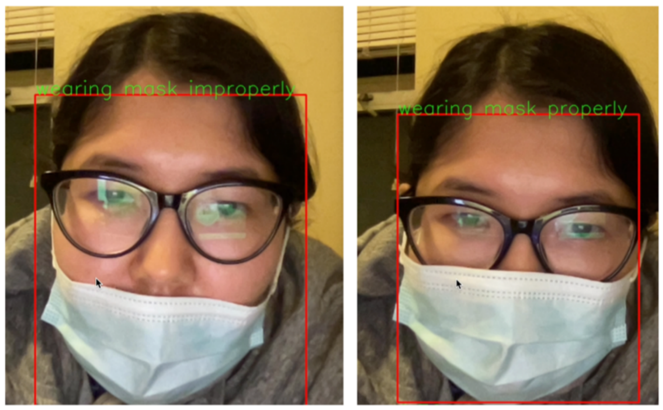


In our third configuration, we obtained the highest accuracy of 0.9899 or 98.99%. The overall losses and accuracies have been plotted below:



Our "best" configuration was our first configuration, which had a batch size of 200, total epochs of 40, and the Adam optimizer. For this reason, we have tuned our pre-trained model with this configuration for real-time face mask detection using a webcam camera.





In our model, the total number of epochs in each configuration remained consistent. To an extent, increasing the number of epochs can improve the accuracy as smaller epoch sizes could result in underfitting. However, if you increase the epoch size beyond a reasonable limit, your model could be overfit. In regard to the batch size, larger batch sizes usually train faster, but they do not converge as quickly as smaller batch sizes. The learning rate also must be considered as an extremely large learning rate could cause the model to converge too quickly to a solution which is not optimal. Typically, a larger batch size will perform more accurately with a higher learning rate. This correlation could explain why the first configuration performed better than the second configuration.

VI. Conclusion and Future Work

Given the current COVID-19 circumstances, our project has the potential to be scaled for real-time analysis, and improve current accessibility and masking procedures in public spaces. However, there are some limitations in our current stage. One limitation we have identified in our project is that the accuracy of our program relies on the type of mask the user is wearing, traffic in the background, and the quality of the image (resolution, brightness, etc.) This can be traced back to the training and testing data we have used. Another limitation we have identified is that we used a pre-trained FaceNet for face recognition and detection, which was not trained by our dataset. Therefore, there are some cases where faces cannot be properly identified on video or real-time camera.

We have determined various future implications for this project, including better scaling the model for real-time analysis, improving the training and testing data with a larger variance of samples, training existing face recognition and detection models with our dataset to improve accuracy, or even implementing our own object detection models.

References (Report)

[1] Loey, M., Manogaran, G., Taha, M. H. N., & Khalifa, N. E. M. (2021). A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic. Measurement, 167, 108288.

[2] Sanjaya, S. A., & Rakhmawan, S. A. (2020, October). Face mask detection using MobileNetV2 in the era of COVID-19 pandemic. In 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI) (pp. 1-5). IEEE.

[3] Singh, S., Ahuja, U., Kumar, M., Kumar, K., & Sachdeva, M. (2021). Face mask detection using YOLOv3 and faster R- CNN models: COVID-19 environment. Multimedia Tools and Applications, 80(13), 19753-19768.

[4] Susanto, S., Putra, F. A., Analia, R., & Suciningtyas, I. K. L. N. (2020, October). The Face Mask Detection For Preventing the Spread of COVID-19 at Politeknik Negeri Batam. In 2020 3rd International Conference on Applied Engineering (ICAE) (pp. 1-5). IEEE.

[5] Yadav, S. (2020). Deep learning based safe social distancing and face mask detection in public areas for covid-19 safety guidelines adherence. International Journal for Research in Applied Science and Engineering Technology, 8(7), 1368-1375.

[6] Shamrat, F. J. M., Chakraborty, S., Billah, M. M., Al Jubair, M., Islam, M. S., & Ranjan, R. (2021, June). Face Mask Detection using Convolutional Neural Network (CNN) to reduce the spread of COVID-19. In 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 1231-1237). IEEE.

[7] Zhang, J., Han, F., Chun, Y., & Chen, W. (2021). A novel detection framework about conditions of wearing face mask for helping control the spread of covid-19. Ieee Access, 9, 42975-42984.

[8] Jignesh Chowdary, G., Punn, N. S., Sonbhadra, S. K., & Agarwal, S. (2020, December). Face mask detection using transfer learning of inceptionv3. In International Conference on Big Data Analytics (pp. 81-90). Springer, Cham.

[9] Cabani, A., Hammoudi, K., Benhabiles, H., Melkemi, M., &amp; Cao, J. (n.d.). MaskedFace-Net. GitHub. Retrieved May 13, 2022, from https://github.com/cabani/MaskedFace-Net

[10] Nayak, S. (2018, June 13). Understanding AlexNet. LearnOpenCV. Retrieved May 13, 2022, from https://learnopencv.com/understanding-alexnet/

References (code)

[11] 12a-pytorch-cnn-mnist.ipynb, Professor James Balamuta Google Collab code, from https://colab.research.google.com/github/fdluiuc/lecture-notebooks/blob/master/12a-cnn-pytorch-mnist.ipynb

[12] Introduction to The Architecture of Alexnet, shipra_saxena, 19 March 2021, from https://www.analyticsvidhya.com/blog/2021/03/introduction-to-the-architecture-of-alexnet/

[13] How to Train an Image Classifier in PyTorch and use it to Perform Basic Inference on Single Images, Chris Fotache, 20 November 2018, from https://towardsdatascience.com/how-to-train-an-image-classifier-in-pytorch-and-use-it-to-perform-basic-inference-on-single-images-99465a1e9bf5

[14] Automatic Face and Facial Landmark Detection with Facenet PyTorch, Sovit Ranjan Rath, 9 Nov 2020, from https://debuggercafe.com/automatic-face-and-facial-landmark-detection-with-facenet-pytorch/

Contributions

Niharika Chintalapati (33.33%) complied the slides and wrote the final report. Kamila Makhambetova (33.33%) processed the data and implemented the main network architecture. Weiyang Wang (33.33%) implemented an alternative network architecture and ran the models.