Face Mask Detection model White paper

By Andres Zamora

Class W251

Professors Esteban Arias-Navarro & Dima Rekesh

August 7, 2020

White paper

- **What does the project do? What problem does it solve?**

For my final project I made a facemask detection model that identified whether someone was not wearing a facemask and sent that image to the ibm cloud to store it, which would allow companies to have record of whether certain employees or customers were not following certain sanitary guidelines within any given space with health restrictions. In order to accomplish this. I first have to create a model that identified whether someone had a facemask or not. Subsequently, I also had to have a base model that recognized faces. In terms of the actual code and after training the model I made a code that would take images instantly so long as you specified the path and a code that would read images in real time and send the images to cloud storage in case the person in the image did not have a face mask on. This would be useful in an environment such as a factory, a large retail store, a mall, an airport or any place where moderate or high amounts of people necessarily need to be in despite the increasing sanitary limitations that should be in place in order to reduce the transmission of COVID-19 until the emergence of a vaccine materializes at scale. The project would best be deployed on both ends of a hallway with good lighting and clear visibility of the faces of people passing by, where people would regularly walk past so that the model is able to capture images of people

walking on both directions whether they be employees, customers, policemen, or travelers. After the COVID-19 Pandemic is surmounted this model will still be useful in places where people still need to wear facemasks and there needs to be a record of people complying with such regulations such as hospitals, research facilities such as laboratories and places where there may exist the risk of hazardous biological contamination.

- **What tools are used to accomplish this?**

To accomplish this a TX2 was used, along with an ibm cloud virtual server instance , an s3 ibm cloud storage bucket, a tensorrt docker container, two tensorflow docker containers mentioned in https://github.com/azamora2/W251/tree/master/FinalProject , three alpine linux containers also mentioned in https://github.com/azamora2/W251/tree/master/FinalProject , python, shellscript to pull the data, and the python packages tensorflow, keras, paho-mqtt, opencv among others(but those were the most important ones). If you wish to see a list of all the packages and modules that were imported please refer to all the .py files and the .pynb files in https://github.com/azamora2/W251/tree/master/FinalProject . Images for training the model were obtained from https://thispersondoesnotexist.com/ , and .png's of facemasks were placed on top of half of the scraped images to train the model for people with and without face masks. The .png's which can be found in https://github.com/azamora2/W251/tree/master/FinalProject were placed manually using https://www.picmonkey.com/ , an online photo editing and design service, which was accessed from a web browser. For the IBM Cloud VSI a V100 server was used with the flavor AC2_8X60X100. And in order to install most of the packages to the containers pip was used and in order to edit most of the code VIM and python was used . A different docker container was

used for training than for detecting the images whether they were real time images of individual images where you specified the path. To train the model I pulled "jupyter/tensorflow-notebook" container, and to detect the images once the model was trained I used "tensorflow/tensorflow:nightly-jupyter".

- **How was it built?**

Below are the instructions on how the model was built and how to run it (you can also find these instructions in a README.md file in https://github.com/azamora2/W251/tree/master/FinalProject):

- If you want to create your own dataset download your own set of images with the *download_faces.sh* script in this file
- Afterwards you can label half of them by putting the png of a facemask on top of it using picmonkey.com
- First you need to run the training in a v100 vsi with flavor AC2_8X60X10

```
ibmcloud sl vs create --datacenter=dal12 --hostname=v100 --domain=dima.com --image=2263543 --billing=hourly --network 1000 --key=1831742 --flavor AC2_8X60X100 --san
```

- Then you need to pull the following docker container and upload the dataset, the *fasmask_detector* folder and the *train_facemask_model.ipynb* on the jupyter container by uploading a zip file and then unzipping it which is the fastest way to do it

```
docker pull jupyter/tensorflow-notebook
docker run -p 8888:8888 jupyter/tensorflow-notebook
```

- Afterwards make sure you run the *train_facemask_model.ipynb* with the appropirate cells to install and import opencv and imutils

- Train the model, and download the *facemask_detector.model* to your computer
- set up alpine bridge containers in your TX2 and in the your VSI just like for HW3 and HW7 explained in this repository https://github.com/azamora2/W251
- pull the tensorrt docker container and run it like in week 05 and upload the *image_taker.ipynb* jupyter notebook
- set up the object storage follow the instructions from the class and in the end to mount it run the following command:

```
sudo s3fs andres-cos-standard-api /mnt/mybucket  -o passwd_file=$HOME/.cos_creds -o sigv2
-o use_path_request_style -o url=https://s3.us-south.objectstorage.softlayer.net
```

where *andres-cos-standard-api* is the name of the bucket (male sure to change it)

```
sudo s3fs andres-cos-standard-api /mnt/mybucket  -o passwd_file=$HOME/.cos_creds -o sigv2
-o use_path_request_style -o url=https://s3.us-south.objectstorage.softlayer.net
```

also make sure you are using the correct url whether it is us-south or us-north

- set up the alpine container in your VSI and the *run_facemask_files_here.ipynb* notebook will take care of saving the images pull the following docker container to run the image detection jupyter notebook

```
docker pull  tensorflow/tensorflow:nightly-jupyter
```

```
docker run --network hw03 --privileged --rm -it --user root -v /mnt/mybucket:/tf/work -p
8888:8888 tensorflow/tensorflow:nightly-jupyter
```

- upload the facemask_detector folder the examples folder the *detect_facemask_in_realtime.py* and the *detect_facemask_indicidual_image.py* and *run_facemask_files_here.ipynb* to the tensorflow/tensorflow:nightly-jupyter container

run the *.py* files in the *run_facemask_files_here.ipynb*

- set up the bridge alpine container in the TX2 and upload and run the *forwarder.py* program inside it after installing vim and paho-mqtt and pip etc.
- after setting up the alpine bridge network and containers in the TX2 like in homework 3 pulling the tensorrt container from lab week05 and uploading the *image_taker.ipynb*. Run the cells in *image_taker.ipynb*

this should cause the images to detect faces in real time and if you are not wearing a mask send those files to the cloud storage

in this case here: https://s3.us-south.cloud-object-storage.appdomain.cloud/andres-cos-standard-api/ also make sure to make this bucket public if you want averyone to be able to see what is inside of it.

- invariably if you only wish to see if the model can detect certain images, place them in the examples folder and run the *run_facemask_files_here.ipynb* cells after you have tweeked line 13 of *detect_facemask_individual_image.py* with the path and name of the image you want to evaluate

```
image_path = "/tf/face-mask-detector4/examples/Con_cubrebocas.jpg"
```

and tweek line 83 of detect_facemask_individual_image.py if you want the evaluated image to have a certain name or be placed in a certain path

```
cv2.imwrite("/tf/face-mask-detector4/facemask_detection_output.jpg", image)
```

- **Sample Data:**

Above we can see the sample of three .jpg images without a mask gotten from

https://thispersondoesnotexist.com/

Above we can see a picture with augmented jpg used for training on people with mask also gotten from https://thispersondoesnotexist.com/. The three different face mask .png's were placed on top of the .jpg's with https://www.picmonkey.com/ are blue black and white facemasks. The specific .png's can be found in

https://github.com/azamora2/W251/tree/master/FinalProject . In class during the presentation

a question was raised whether it would be a good idea to feed the model when training a picture of a women with big earrings to see if the model would be able to identify the facemask despite the big earrings. And as we can appreciate from the picture in the middle the model was indeed fed with data that took this into consideration.

- **The Data**

1200 jpg images were scraped from a generative adversarial network found in https://thispersondoesnotexist.com/ using the shell script found in https://github.com/azamora2/W251/tree/master/FinalProject. Half of them (600) were augmented by placing png's of white blue and black facemasks on top of them using the application called https://www.picmonkey.com/. This data was then put into folder of images with and without face masks respectively and fed to the model for training

- **The model**

Imagenet V2, a pretrained face mask detection model in tensorflow, was used for face detection without a fully connected layer was used as a base model and the head model to be used for face mask detection was placed on top of this model. The layers were initially frozen so that they would not update the first training process. The head model consisted of a CNN model from keras with a pooling layer, a flatten layer, a dense layer with dropout and a softmax layer. The batch size was 64 images with 32 epoch, the solver used was Adam. For further details you can look at the source code in:

https://github.com/azamora2/W251/blob/master/FinalProject/train_facemask_model.ipynb

- **Results**

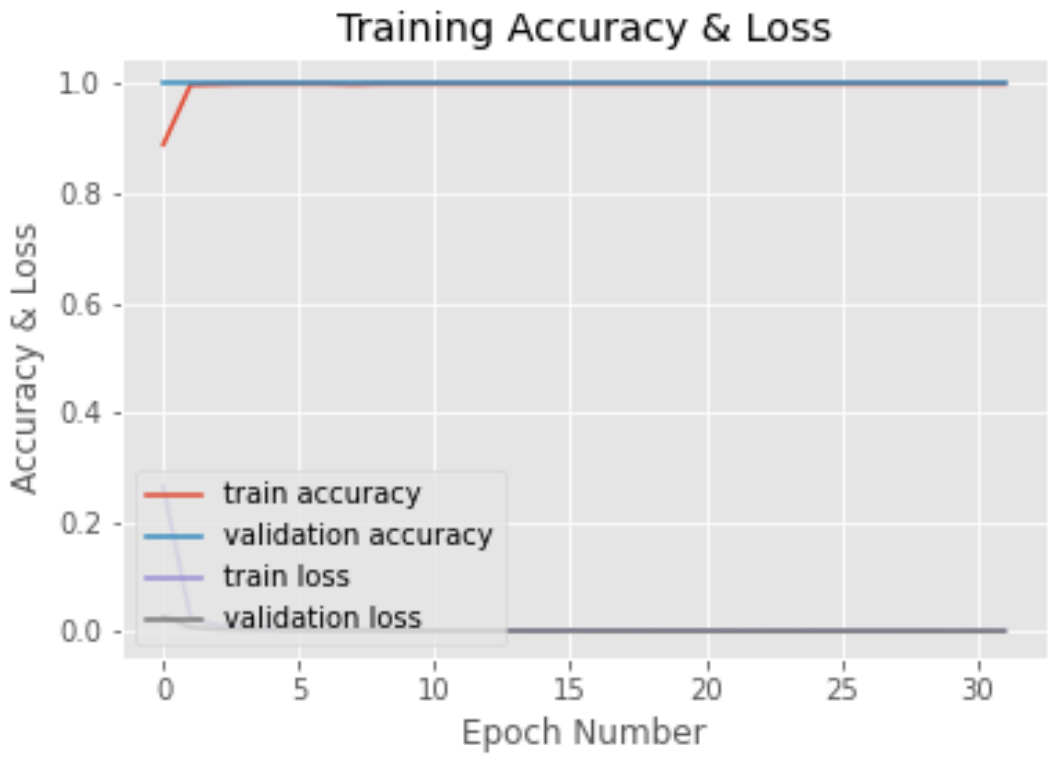
The final results for the training went very well

```
--------------EVALUATING HEAD NETWORK----------------
              precision    recall  f1-score   support

   with_mask       1.00      1.00      1.00       138
without_mask       1.00      1.00      1.00       138

    accuracy                           1.00       276
   macro avg       1.00      1.00      1.00       276
weighted avg       1.00      1.00      1.00       276

--------------------SAVING MODEL---------------------
```



Above are three images of real people that were correctly evaluated with the face mask detection model.

- **challenges and the alternative ways that this problem could be solved**

The major challenge was placing the .png's of face masks on top of the .jpg's manually for the 600 images, although it did yield very good results. Other challenges were finding the appropriate docker containers where you could install all the needed python packages to train the model and carry out the face mask detection. Another challenge during training was being able to have enough memory to train the model, thus an IBM cloud VSI was used as specified in the instructions. The project could potentially alternatively be solved with the library *fastai* a python library for deep learning and artificial intelligence in python. Although it is unclear whether the results would have been as good as they currently are.

- **The potential room for improvements**

The major rooms for improvements for this project would be to look for ways to increase the frame rate currently at around 17 frames per second. Another implementation that would

improve the model is to place pictures with several people in them with and without facemasks to train the model, because so far only images of one person either with or without face masks were used. And finally, to include more patters of face masks in the training, because so far only white, blue, and black face masks were included, so if different patterns of facemasks are shown this could confuse the model making it more unsure whether a person is using a facemask or not if the facemask has a distinctive pattern.

Thank you and if there are any further questions, consult
https://github.com/azamora2/W251/tree/master/FinalProject
Email me at azamora@berkeley.edu, or send me a message through slack at Andres Zamora