**Report for**

**COMP 472 Project Assignment #1**

**Prepared by**

**Peng Zhao 40070966**

**Moben Li 40078435**

**Qichen Liu 40055916**

**Fuqiang Zhai 40072577**

**Project repository:** https://github.com/limoben/COMP472\_AI\_Face\_Mask\_Detector.git

**Concordia University**

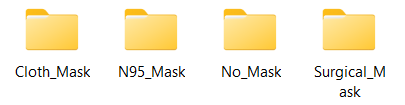
**04/06/2022**

# **1. Contribution Log**

|  |  |
| --- | --- |
| **Group Member** | **Contribution:**  (I) Data Specialist, responsible for creating, pre-processing, loading & analyzing the datasets;  (II) Training Specialist, responsible for setting up and training the CNN;  (III) Evaluation Specialist, responsible for analyzing, evaluating, and applying the generated model;  (IV) Compliance Specialist, responsible for the overall planning, scope verication, and compliance  (in part 2 of the project, this member will also be responsible for the new tasks mentioned below). |
| Peng Zhao  40070966 | (I) Data Specialist, responsible for creating, pre-processing, loading & analyzing the datasets;  (II) Training Specialist, responsible for setting up and training the CNN; |
| Qichen Liu  40055916 | (II) Training Specialist, responsible for setting up and training the CNN;  (III) Evaluation Specialist, responsible for analyzing, evaluating, and applying the generated model; |
| Moben Li  40078435 | (I) Data Specialist, responsible for creating, pre-processing, loading & analyzing the datasets;  (IV) Compliance Specialist, responsible for the overall planning, scope verication, and compliance |
| Fuqiang Zhai  40072577 | (II) Training Specialist, responsible for setting up and training the CNN;  (III) Evaluation Specialist, responsible for analyzing, evaluating, and applying the generated model; |

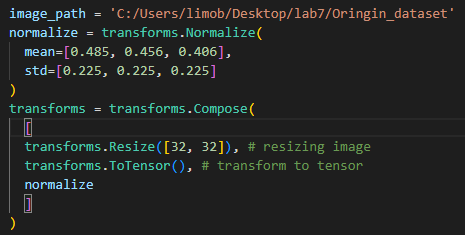
**Dataset**:

The dataset for this project is coming from different sources(see reference). The main source of our image is from Kaggle, which counts approximately 85% of our total image. We download 5 dataset from Kaggle, and manually sort them to have 4 kinds of mask image into 4 different folders.



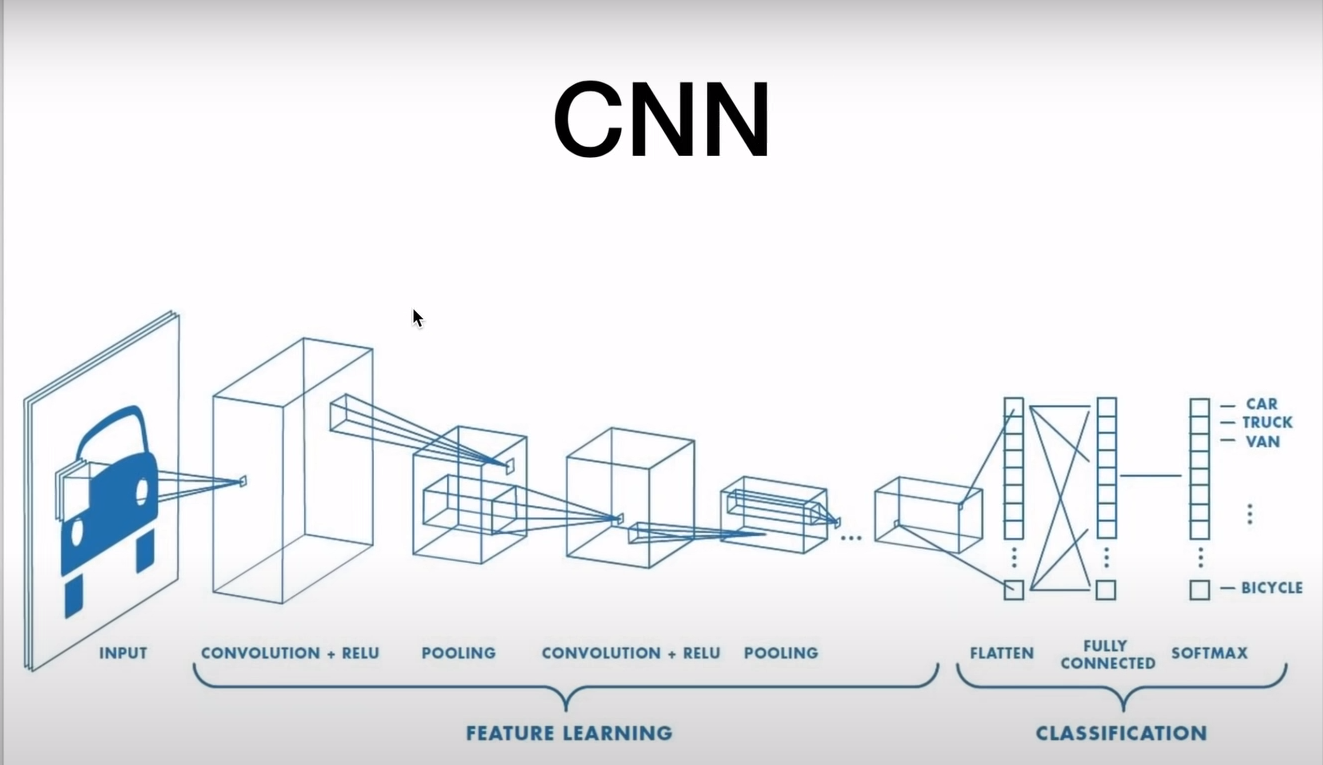
At the end of sorting process, we found out that the N95 mask is much less than other types images. Therefore, we go to google image by searching “N95 mask” key word to get list of image. We pick and save the image to local folder. After we have enough image, we found that the image has a different file type, include .jpg .jpeg .png. We convert all the image to .jpg file type to avoid the potential type error. All the image are in high resolution with full clear face and mask, although some of image are in landscape and some are in portrait. Finally, the 4 class of has 409, 405, 469 and 406 images respectively, a total of 1689 image.

After manually processing the image, we start using python code to further pre-process the image. First, we use pyTorch transforms function to transform the image to the same property, which are resize image to 32 \*32, convert image to tensor, and normalize the image. Then we load the image through torchvision.datasets.ImageFolder function, after that, we use DataLoader to load train data and test data specifictly.

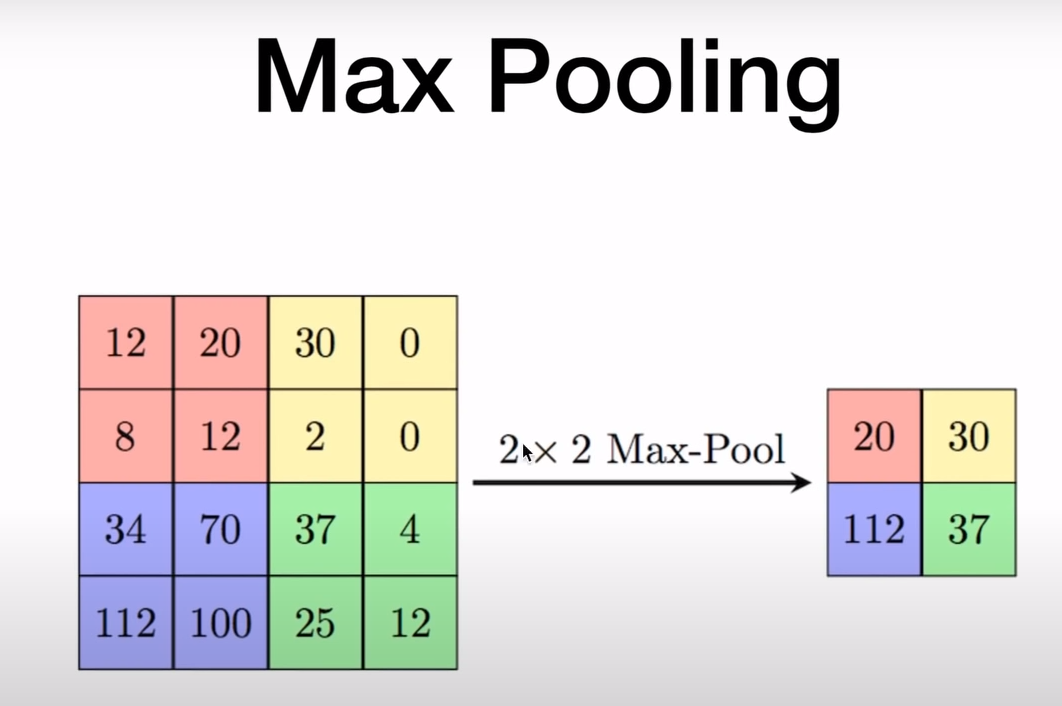




**CNN Architecture**:



The image above shows the the main structure of CNN Architecture which combined with a 1st convolutional layer followed with a relu activation function, then apply the max pooling layer, and 2nd convolutional layer followed with a relu activation and max pooling, after that , we have three different fully connected layers.

****

Max Pooling layer is used to reduce the computational cost by reducing the size of images. also avoid overfitting by providing an abstracted form of the input

In details:(explain some important layers implementation)

**self.conv1 = nn.Conv2d(3, 6, 5)**

the first convolutional layer passed three parameters, 3 stands for 3 color channel, the output channel size is 6 and the kernel size is 5 (5\*5)

**self.pool = nn.MaxPool2d(2, 2)**

**self.conv2 = nn.Conv2d(6, 16, 5)**

the max pooling layer with kernel size 2\*2

For the second convolutional layer, the input channel size is 6,which is the output of the first conv channel, and output channel size is 16.

self.fc1 = nn.Linear(16 \*5 \*5, 120)

For the 1st fully connected layer. the input size is 16\*5\*5 , since the original size of image is 32\*32, and we calculate the result by applying the formula (W-F + 2P)/S + 1, W is input , F is the filter size, P stands for padding value and S is stride.and the result is 16 \* 5 \* 5 after two conv layers and 2 max pooling layers.

self.fc3 = nn.Linear(84, 4)

For the final fully connected layer , with the input 84 stands for the output of the previous fully connected layer.the second parameter value is 4, which stands for four different classes.

Training :

loop over the number of epochs that we set it to 6

loop over the train\_loader , get images and labels of each batch,

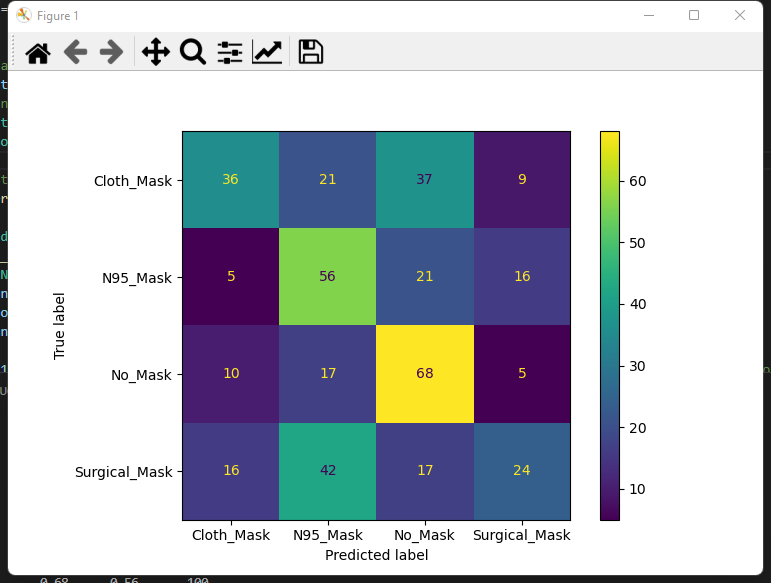
then we do typical forward pass and create loss

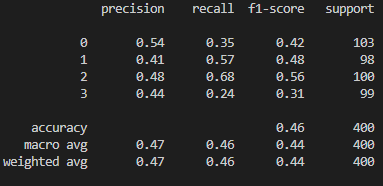
in each loop , we need to empty the gradiance with the function zero\_grad()

and finally calculate the accuracy

**Evaluation**:

For evaluation purposes, we used our implemented CNN model to test the 400 random images from 4 categories of masks based on our dataset. Then we plot the confusion matrix to and calculate the value of accuracy, precision, recall and F1-measure. Following is result we get from the test:

****

****

In the figure above, we can see that the accuracy of our model reaches 46%. This value is acceptable because we have not optimized our model yet. In the second phase of the project, we are going to expand our dataset to improve the accuracy. A larger dataset means that more images can be used to train our model. In addition, we can try to change the learning rate and number of epochs to improve the model accuracy.

In the evaluation process, we found out that as we increase the number of epoch, the accuracy is increasing, the precision and recall are also increasing.

**Reference:**

Reference1:

pytorch training video, (<https://www.youtube.com/watch?v=c36lUUr864M>)

Reference2:

kaggle dataset

dataset1:

https://www.kaggle.com/datasets/dhruvmak/face-mask-detection?resource=download

dataset2:

https://www.kaggle.com/datasets/prithwirajmitra/covid-face-mask-detection-dataset

dataset3:

https://www.kaggle.com/datasets/ashishjangra27/face-mask-12k-images-dataset

dataset4:

https://www.kaggle.com/datasets/perke986/face-mask-segmentation-dataset?select=images

dataset5:

https://www.kaggle.com/datasets/humansintheloop/medical-mask-detection

google image

https://images.google.ca/