COMP 472

Artificial Intelligence

Summer 2022

Group : PG_09

Christopher Cui 40096627 - Compliance Specialist

Sami Ibrahim 40156134 - Data Specialist

Ibrahim Ibrahim 40158162 - Training Specialist

Usama Saleem 40110036 - Evaluation Specialist

Project Repository : https://github.com/chriscui47/Project-1-cnn

Dataset

We primarily built our dataset using two sources which had around 5000 images each. The first source [1] is from a public dataset provided by Humans In the loop, which is “a social enterprise that aims to connect conflict-affected communities to digital work”. The dataset consists of 6k images acquired from the public domain with an extreme attention to diversity, featuring people of all ethnicities, ages, and regions. The second source [2] is from the FaceMask database, where images were “manually collected from Google Images and annotated using Labelling tool in YOLO format”. The Database consists of 4,866 images of people in different environments and situations. We then selected 1600 images, and manually split the images into four different classes by putting them into a separate folder for each class: Person without a face mask (400 images), Person with a “community” (cloth) face mask (400 image), Person with a “surgical” (procedural) mask (400 images), and Person with a “FFP2/N95/KN95”-type mask (300 images). The testing data consisted of 100 images from each class. The validation data was split from the initial 1600 images in a 80/20 ratio, where 80% of the images were used for training and the remaining 20% was used as validation data.

The resolution of the images varied from 450x325 pixels all the way up to 6240x4160 pixels. However, the images were all resized to 150x150 as part of our pre-processing. Furthermore, our pre-processing also converted the images to tensors, and then normalized the images. Our normalization consisted of using the transform. Normalize function, and setting both the mean and standard deviation to 0.5. This ends up normalizing the image to the range [-1,1]. This reduces the skewness and helps the CNN learn faster.

CNN Architecture

Overview

The basis of this architecture has “hidden layers” called convolutional layers, which receives input, transforms the input in some way, and outputs the transformed input to the next layer. In this architecture, this transformation is a convolution operation, and each layer has filters to detect patterns, which get more accurate as the layers progress and increase.

In our architecture, we have 3 CNN Blocks, with each block consisting of 2 convolutional layers and 1 max-pooling layer. In the max pooling layer, we reduce the size of the image to half of the original size before passing it onto the next layer.

We are creating a class that inherits from the nn.Module to define different layers of the network. The first step is to use the nn.Sequential module to create sequentially ordered layers in the network. Each consists of a convolution + ReLU + pooling sequence. In each convolution layer, use LeakyRelu for the activation function and BatchNorm2d to accelerate the training process. [4]

ReLU

As an activation function, the ReLU function was used to prevent the negative values in the output from the input of the network and achieve a non-linear transformation of the data. If the output is negative, then the output is set to zero, and if the output is positive, then the output is set to the original value. The function is $f(x) = \max(0,x)$ and the ReLU is used to speed up training in the neural network. In addition, we hope for ReLU transformation in hopes that the transformed data will be close to linear (regression) or close to linearly separable (classification). In essence, the ReLU used is a compact and efficient way of moving signals from between the layers, resulting in faster computation and a more efficient CNN.

Epoch

We have an epoch of 12, which is the number of times the network is trained. Each epoch refers to when the entire dataset completes one propagation forward and backwards through the network. [3].

If we increase the number of epochs, the accuracy of the network will increase, although with a cost of time and computational power, thus a good balance of 12 is what we have set on.

FC Layer

In our architecture, we implement a fully connected layer which is used to connect the output of the convolutional layer to the fully connected layer. The output of the convolutional layer is a vector size of 82944 and as for the fully connected layer, that size is 1024. In essence, a FC layer is nothing more than a dense network of neurons and the connections between two neurons. We use this to classify an image to a specific category, for instance a mask type, after we have extracted the features from the image using CNN. Once we find the features of an image, we flatten the features found into a 1D layer, which is then used as an input to the FC layer. In the end, the number of neurons resulting in the final output after all the layers will be correlated to the number of categories of classification we are aiming for. In our case, we end up with 4 final neurons. Not only does the FC layer categorize the features along the way, it learns to associate features to a specific label, thus making it an essential part of our CNN.

Skorch

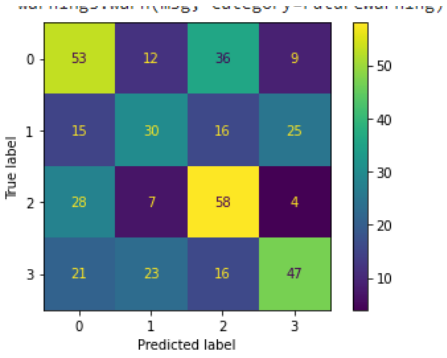
The goal of skorch is to make it possible to use PyTorch with scikit-learn. This is achieved by providing a wrapper around PyTorch that has a scikitlearn interface. Additionally, skorch abstracts away the training loop, a simple net.fit(X, y) is enough. We are using net.fit(train_data, y=y_train)to train our network. Skorch also takes advantage of scikit-learn functions, such as predict. We are using y_pred = net.predict(test_data)to predict the test data.

skorch.NeuralNetClassifier is a Neural Network class used for classification tasks. We Initialize the NeuralNetClassifier class then train the CNN model using the method fit. [4]

Evaluation

Based on the confusion matrix, labels 2 and 3 have relatively good accuracy. The big issue arises when trying to predict cloth masks and n-95/ffp2 masks. There may be a bit of accuracy skewness because the dataset is not completely balanced for label 1 (n95 masks). This is because we have around 75 less training images than the other classes. We recognize this issue and will focus on rectifying this during the second phase of the project. Another thing we must do for the next part of the project is to improve the accuracy which currently sits at 0.46 which is not optimal at all. This can be accomplished by adding more training data (especially N95 class, which is our lowest accuracy).

Another thing we will do is to add more layers to our CNN architecture. With more layers, the neural net would be able to more fine tune the results from more calculations. Specifically, we will aim to increase the number of hidden layers which may increase the accuracy of the results.



	precision	recall	f1-score	support
0	0.45	0.48	0.47	110
1	0.42	0.35	0.38	86
2	0.46	0.60	0.52	97
3	0.55	0.44	0.49	107
accuracy			0.47	400
macro avg	0.47	0.47	0.46	400
weighted avg	0.47	0.47	0.47	400

0: (cloth) face mask

1: 'FFP2N95KN95-type mask'

2: 'Person without a face mask'

3: 'surgical (procedural) mask'

References

[1] H. in the Loop, “Medical mask dataset | Humans in the Loop,” May 28, 2020. https://humansintheloop.org/resources/datasets/medical-mask-dataset/ (accessed Jun. 06, 2022).

[2] M. Vrigkas, E.-A. Kourfalidou, M. E. Plissiti, and C. Nikou, “FaceMask: A New Image Dataset for the Automated Identification of People Wearing Masks in the Wild,” *Sensors*, vol. 22, no. 3, p. 896, Jan. 2022, doi: 10.3390/s22030896. **https://mvrigkas.github.io/FaceMaskDataset/** (accessed Jun. 06, 2022).

[3] “Epoch vs Batch Size vs Iterations | by SAGAR SHARMA | Towards Data Science.” https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9 (accessed Jun. 06, 2022).

[4] COMP472 Artificial Intelligence (Summer 2022)
Lab Exercise #07: Introduction to Deep Learning