

Jeffrey Wolberg

Project Writeup

NOTE: The code that I wrote is contained in the first and second cells. In the second cell, I create a class that helps with readability. All relevant outputs are printed below the second cell. In the second cell, please see the function 'run_pipeline' to see the pipeline of the Traffic Sign classifier.

Dataset Summary: The dataset is of 43 different traffic signs in the German Traffic Sign Dataset. The images are quite small, only at 32x32, which makes extracting useful features from them more difficult, but doable, nonetheless.

Exploratory Visualization: Please look at function 'print_dataset_metrics'.

Preprocessing of images: Usually you would downsize the image, but because they are already so small, that is not necessary. For now, I am just normalizing the input pixel values by dividing each one by 128, and then subtracting one. (**Note:** I tried doing the recommended $(img-128)/128$, however negative values were not allowed (the pixel values just went to 255 instead of -1), THIS SHOULD BE MADE CLEAR TO STUDENTS, IT WOULD SAVE A LOT OF TIME)

Model Architecture: I replicated the architecture given in the class (please see the LeNet function in my code). It involves two convolutional layers (5x5 kernel each, first one outputs 6 feature channels, second one outputs 16 feature channels) and two fully connected layers. I am unsure if adding more hidden layers would increase performance, especially given how performance is already (94%+). Perhaps it would add marginal performance benefits.

Model Training: I set my learning rate to 1×10^{-3} , and I had 50 epochs (50 runs through the data), and my batch size was 128 (I calculated gradient for each optimizer step using only 128 images each time). Therefore, I did $\text{len}(\text{training_dataset})/128$ amount of steps per epoch.

Solution Approach: My validation accuracy was 95%, and my test accuracy was 93% (you can verify the test accuracy by running the code, it will be printed in the output)

Acquiring New Images: For the testing of my new network on traffic signs, I downloaded traffic signs from the web, and resized them to 32x32. (see folder test_images).

Performance on New Images: (see 'test_on_my_images'). Out of 10 new images, it got 9 correct. This fits with my training and test accuracy.

Model Certainty – In the function 'test_on_my_images', you will see the real answer (ground truth), the network's top 5 predicted outputs (in order of confidence) for each image, the softmax probabilities for those 5 outputs, and the logit values for those 5 outputs. In general, the network is quite confident in its first prediction, and the softmax value is usually 95%+.

Visualization of the conv layers of the neural network: I visualized the activations for the conv layers in the network. You can find them in the folder 'layer_visualizations'. In the first image, the top row is the activations after applying a 5x5 convolution of the input RGB image, the second row is after clipping the negative values of the previous conv outputs using a ReLU (in the pic this would be equivalent to clipping all values < 128). The third row is after a 2x2 max pool of the previous output. The second image shows what the activations are after the input image goes through the second conv layer.