Introduction

This traffic sign classifier used the LeNet neural network to classify the traffic signs. The network gave an accuracy of 99.4% on the test set. It gave an accuracy of 63% on the newly downloaded images. Eight images of German traffic signs were downloaded from the internet.

Dataset Exploration

<u>The submission includes a basic summary of the data set.</u>
This is included in the jupyter notebook

<u>The submission includes an exploratory visualization on the dataset.</u>
This is included in the jupyter notebook.

Design and Test a Model Architecture

<u>The submission describes the preprocessing techniques used and why these techniques were chosen.</u>
The first technique used to preprocess the data was to shuffle the data. This ensures that groups of similar images which were added to the training set during labelling will not end up in the same mini batch. This should make the network more robust when it is trained. The data is also shuffled at each epoch during training.

The second technique used was to normalise the pixel values. This is to ensure that the data has zero mean and equal variance between -0.5 and 0.5 to allow for easier optimisation.

<u>The submission provides details of the characteristics and qualities of the architecture, including the type of model used, the number of layers, and the size of each layer. Visualizations emphasizing particular qualities of the architecture are encouraged.</u>
The le-net architecture was used for traffic sign classification. A 32x32 image is fed to the neural network. It firstly goes to a convolutional layer, then a pooling layer, then another convolutional layer, then another pooling layer, then three fully connected layers.

The first convolutional layer converts a 32x32x32 image to a 28x28x6 output. The weights are used to create a filter size of 5x5x3 and six filters are used to try and capture the important features of the traffic sign. The filter depth of 3 is used as the filter works on all three colour channels. The convolutional layer is created using the tf.nn.conv2d function. A stride of 1 and VALID padding is used to create the convolved layer. The is then passed to the ReLu activation function.

The output of the activation function is fed to a pooling layer. The pooling layer uses max pooling. It uses a filter size of 2*2 and a stride of 2 which reduces the output by half. This reduces the size of the input to the next layer and enables the neural network to concentrate on the most important parts of the image. It also prevents overfitting. However, important information which could be used to detect the sign may be lost. Dropout may be a better regularizer. Average pooling could also possibly be used.

The output of the pooling layer is then fed to another convolutional layer, followed by an activation function, and then another max pooling layer. The output of this layer is then flattened to a tensor. This is then fed to the third fully connected layer.

The third layer is a fully connected layer with an input of 400 and an output of 120. This is fed to an activation function followed by another fully connected layer with an input of 120 and output of 84. This is then fed to another activation function followed by the final layer. The final layer takes an input of 84 and gives an output of 43. The output of the neural network are the logits. 43 values or logits are given for each image which corresponds to the input image's score linking it to a specific label.

<u>The submission describes how the model was trained by discussing what optimizer was used, batch size, number of epochs and values for hyperparameters.</u>
The scores (logits) are then turned into probabilities using the softmax function. We then compare the probabilities to the one hot encoded label vectors using the cross entropy function. This is carried out in one line of code.

We then take the average of all the cross entropy losses for the current parameters (weights and biases). We then try to minimise this loss using the Adam optimiser. A learning rate of 0.001 was chosen for the optomiser as this was found to work well in the tested problems in the paper; "Adam: A Method for Stochastic Optimization".

The hyper parameters for weight initialisation were chosen to be values from a truncated normal distribution zero mean and a standard deviation of 0.1 which worked well in the lab. It is also suggested in the lectures to use a small sigma initially which means the distribution is uncertain. This allows the optimisation to become more confident during training.

A batch size of 128 was chosen as it worked well in the lab. The larger the batch size the faster the model will train but batch size is limited by memory.

<u>The submission describes the approach to finding a solution. Accuracy on the validation set is 0.93 or greater.</u>

Accuracy on the validation set was 99.4% using 11 epochs.

Test a Model on New Images

<u>The submission includes five new German Traffic signs found on the web, and the images are visualized. Discussion is made as to particular qualities of the images or traffic signs in the images that are of interest, such as whether they would be difficult for the model to classify.</u>
Eight traffic sign images of traffic signs in Germany were downloaded from the internet. The signs all have varying backgrounds from blue sky to buildings and trees. Two no entry signs are included to see how the classifier works on two very similar images. The image resolution for each image is much larger than the 32x32x3 image fed to the classifier. This is expected to have some effect on the classification of the signs.

Eight images were fed to the classifier. It predicted 5 out of 8 signs correctly. This equates to an accuracy of 63%. The test gave an accuracy of 99.4%. Reasons for this may be the max pooling layer. Additional filters may also help but will slow down the training of the network.

The signs it calculated incorrectly were the first no-entry sign, the 120 kph and 30 kph signs. The 120 kph sign was predicted as a turn right ahead sign with a certainty of 0.994. The correct prediction was the 5th highest probability. The first no-entry sign was predicted as a no passing sign with a certainty of 0.905. The correct prediction was the 5th highest probability. The 30 kph sign was predicted as an ahead only sign with a certainty of 0.998. The correct prediction was the 2nd highest probability.

It was interesting to see that the classifier misclassified the no entry sign. This would suggest that the classifier places particular importance on colour. The training set may contain few faded signs. The viewing angle is also different here. It might be a good idea in future to try and artificially fade and distort the images during pre-processing.

<u>The submission documents the performance of the model when tested on the captured images. The performance on the new images is compared to the accuracy results of the test set.</u>
See above.

<u>The top five softmax probabilities of the predictions on the captured images are outputted. The submission discusses how certain or uncertain the model is of its predictions.</u>
See above.