Team Name: The AInsteins
Team Members: Liz Parker, Colin Craighead, Aren Dalloul

Kaggle Mini Competition: Image Classification

Libraries

We used the Tensorflow backend and Keras library to build a sequential convolutional neural network.

Preprocessing

Data set was very clean, so not much preprocessing was required.

Convolutional Neural Network

To build our baseline model, we decided to follow a CNN architecture of Conv-Conv-Pool-Conv-Conv-Pool. We used a smaller filter number for the first two convolutional layers and a larger filter number for the last two convolutional layers, following the principle to "keep the feature space wide and shallow in the initial stages of the network, and then make it narrower and deeper towards the end." We decided not to use padding because the borders of the image were less important to identifying the sign. We used pooling, and no stride. Finally, we used a constant 3x3 filter size throughout the model.

Model Performance

Our baseline model performance was great, with 98-99% accuracy and about 5% loss, so our only fear was overfitting. We fit the baseline model using only 3 epochs to avoid overfitting.

Exploratory Analysis

Number of filters:

Our baseline model had a pattern of the earlier layers having fewer number of filters than the following layers. We tested this architecture by creating an inverse or backwards model that had more filters in the earlier layers and fewer in the following layers. The backwards model had worse performance on the test data, confirming our strategy to use smaller-to-large filter numbers.

We tested a variable number of filters, from 12 total filters up to around 800 total filters. Looking at a plot of these model's test accuracy and test loss, we found that having around 192 (32+32+64+64) total filters was at the end of the bend of the elbow. This optimal number of total filters would allow us to achieve the best test accuracy and categorical cross entropy without making the model overfit or be excessively large.

Filter size

Our baseline model has a constant 3x3 filter size. We tested using smaller filter size on earlier layers and larger filter size for the following layers - a small-to-large filter size model. We also tested using a larger filter size on earlier layers and smaller filter size for the following layers - a large-to-small model. However, both models had worse performance than using a constant filter size throughout all the layers.

We then tested what the optimal constant filter size would be, and concluded that the 3x3 constant filter size used in the baseline model was the most effective.

Normalization and Regularization Techniques:
We tested the effect of using max batch normalization after every convolutional layer versus minimum batch normalization (only 1-layer) in our model. We found that the number of batch normalization layers had negligible impact on the performance of the model, so we decided to keep the baseline model architecture using batch normalization after each convolution layer.

We then tested the effects of a dropout layer on our model. We found that the model without a dropout layer had minimally improved loss, which makes sense, because the dropout layer removes data to avoid overfitting. It was interesting to see that during training the model with a dropout layer had poorer performance and took longer to catch up to the model with no dropout layer. Because there was negligible statistical difference between the models with and without a dropout layer, we decided not to use a dropout layer in our final model.

Plotting the test accuracy and test categorical cross entropy of models with variable dropout rates, we found that higher dropout rates led to worse model performance.

Conclusion
Because all of our tests supported the architecture of our baseline model, we used the same strategy for our final model. In the normalization and dropout tests, we saw an increase in model performance with more training epochs. So, for our final model, we trained using 10 epochs, and our Kaggle score confirmed that 10 epochs improved performance without too much overfitting.

Contributions
We all worked together to finetune architecture and hyperparameters to build our baseline model with fairly good accuracy. From there Colin explored the effects of the number of filters chosen for each layer. Liz explored the effects of filter size at each layer. Aren explored the effects of normalization and regularization techniques. We all discussed the results to conclude on our final model.