GR5242 Final Project - Project 1 (CIFAR 10)
The effect of pooling method on CNN performance

Fall 2018 Advanced Machine Learning
yj2465 Ying Jin
rl2658 Robin Lee

I. Problem Setting

The objective of this project is to explore the effect of different pooling methods on the performance of Convolutional Neural Network(CNN). Pooling operations, which reduce the spatial dimension of the input volume(the length and the width) to lower the computational cost and to control overfitting, play a central role in current architectures of CNN. A typical CNN is structured as a series of convolutional layers and pooling layers, and max-pooling layer is the most popular choice. This project implements a number of different pooling methods, which are the max pooling, average pooling, and fractional-max pooling. By comparing the accuracy in the test result obtained by different pooling methods, we examine whether different pooling methods yield a substantial difference in the performance of the model.

II. Data Description and Preprocess

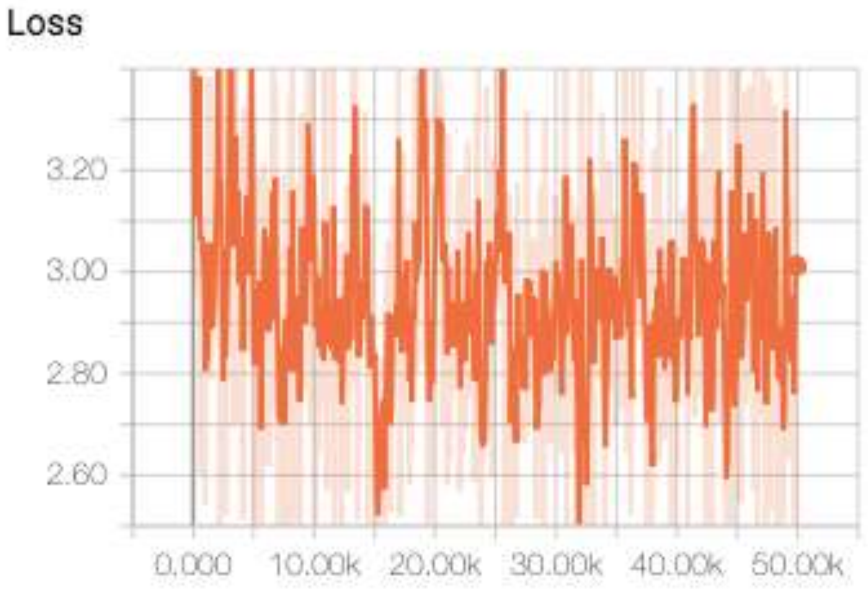
The dataset used in this project to compare the effects of different pooling methods on CNN performance is CIFAR-10 image dataset. This dataset is composed of ten classes of images, having training set with a size of 50,000 (5,000 per class) and test set with the size of 10,000. Each image is an RGB image with a size of 32 x 32 pixels.

We preprocess the dataset by randomly distorting the input images to make CNN more flexible at recognizing and classifying images. This is implemented in the cell named ‘Data Preprocess’ in the notebook. To illustrate the process in detail, images are cropped from 32 x 32 to 24 x 24 pixels by randomly slicing images for each channel. And then, random distortions are applied to artificially increase the size of the dataset for training, implementing horizontal flip, distortion in the brightness and contrast. Finally, the training images are standardized to make the model insensitive to dynamic range. The original training data are distorted by this preprocessing, and then become input in our model.

III. Early Exploration

We start from simple logistic regression (one layer neural network) as a starting point in moving toward more complex networks. Figure 1 shows the movement of loss in the training process. It is obvious that the graph is highly volatile, which means the loss is not reduced as the iteration goes on. This indicates that the logistic regression model is not learning the dataset well, resulting in 33.2% of the test accuracy which is very low. To improve the performance and also to compare the effect of different pooling methods, complex models are implemented in the next section.

Figure 1. L2 Loss of Logistic Regression

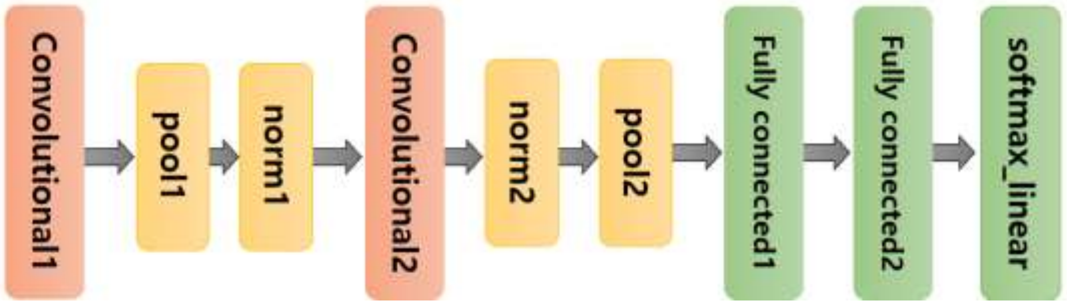


IV. Models and Results

Figure 2 represents the basic structure of the CNN used in this project. We adopted this series of operation from the tutorial on tensorflow from the tensorflow website.[1] The unique part of our model is that we try three different pooling methods (i. Max Pooling, ii. Average Pooling, iii. Fractional Max Pooling), both individually and collectively, to examine whether the difference in pooling methods yields difference in performance. Each of the pooling methods is described in detail in the next part.

[1] https://www.tensorflow.org/tutorials/images/deep_cnn

Figure 2. Graph of CNN



i. Max Pooling

With max pooling method, a filter of size 2x2 with strides 2x2 is applied to the input image. This picks up the maximum value in each grid with a size of 2x2, and the grid moves by 2 pixels. Acting on the hidden layers of the network, this operation discards 75% percents of the dataset, allowing for reasonable computational cost and preventing overfitting. The max pooling method has been a default choice for building convolutional networks due to its characteristic of being fast, quickly reducing the size of hidden layers, and being robust to some distortions. Therefore, we set the performance of CNN with max pooling method as a benchmark when assessing the effect of different pooling methods which will be explained next.

Figure 3 shows the loss in the training process in the model with the max-pooling method. It is clear that the model achieves a huge improvement when compared to the logistic regression in section 2, as the loss function stabilizes after about 5,000k iteration. Also, the evaluation accuracy has gone up to 72.8 %.

ii. Average Pooling

Now we implement average pooling method, which works in the exact same manner with the max pooling in section i except that the filter calculates the average of a window instead of picking up the maximum value.

Figure 4 shows the loss in the training process in the model with average pooling method. The shape of the figure is pretty similar to that of max pooling method. The evaluation accuracy is 71.9%, which is slightly lower than that of max poling method. This indicates that by smoothing the sub-regions in the images, the loss of information is actually more severe compared with simply keeping the largest. This is an interesting finding and would be contrary to our previous assumptions.

Figure 3. L2 loss of CNN with max-pooling method

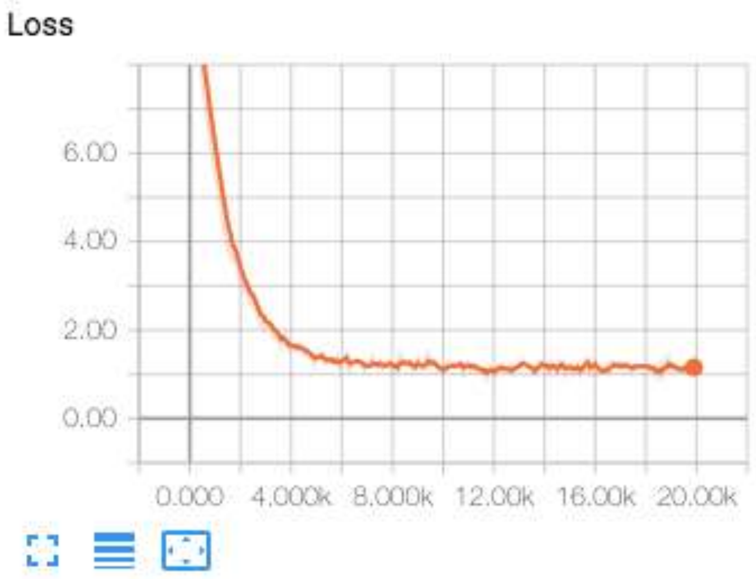
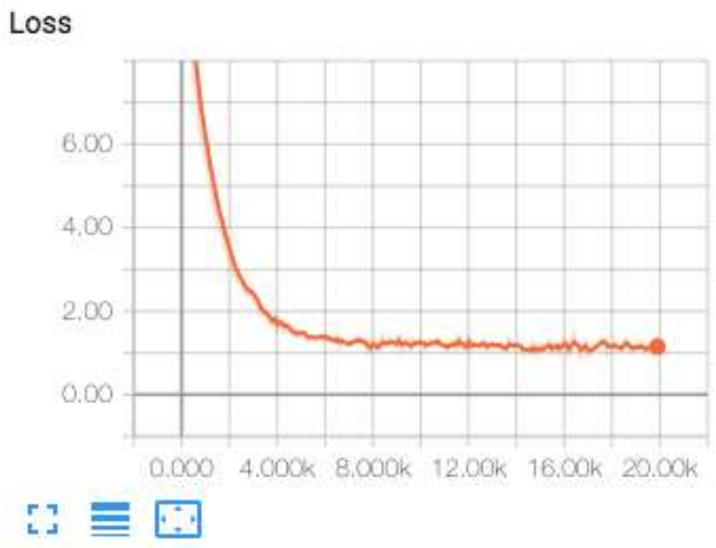


Figure 4. L2 loss of CNN with average pooling method



iii.   Fractional Max Pooling

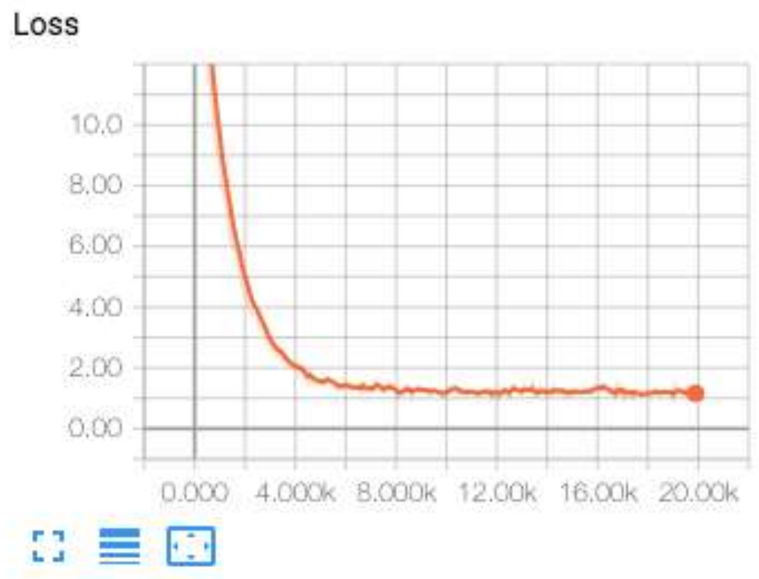
Following Graham 2015[2], we implement fractional max pooling method. This can take non-integer values for the size of a window (width and height) and also introduce a certain randomness in the pooling process. This is an alternative to the pooling scheme in i. that reduce

[2] Graham, B.: Fractional max-pooling. arXiv:1412.6071 (2014).

image dimensions by a factor of 2. We implement 1.44 x 1.44 grid in our model instead of 2x2. This allows a higher number of pooling layers and softer image size reduction throughout the network. Fractional max pooling method also has a stochastic characteristic in that it uses random sequences to generate the pooling regions instead of moving the grid always by 2 pixels.

Figure 5 shows the loss in the training process in the model with fractional max pooling method. The shape of the figure is pretty similar to that of previous pooling method. The evaluation accuracy is 71.8%, which is slightly lower than that of max pooling method and average poling method. Our guess of the result was that the stochastic process introduced more variation of the model, increasing variance, thus error, to a slight degree.

Figure 5. L2 loss of CNN with fractional max pooling



iv. Combination of Pooling Methods

We experiment on combinations of different methods to explore the interaction between different pooling methods. This technique turned out to make a slight improvement in model performance, thus proving the existence of combined effect. For example, in Figure 6 and 7, Average Pooling was combined with Max and Fractional Max respectively, and this new networks both worked better than their sole performance. In evaluation, we get the test accuracy rate of 73.3% and 72.3%, which are also slightly higher than the previous numbers.

Figure 6. L2 loss of CNN with average pooling + max pooling

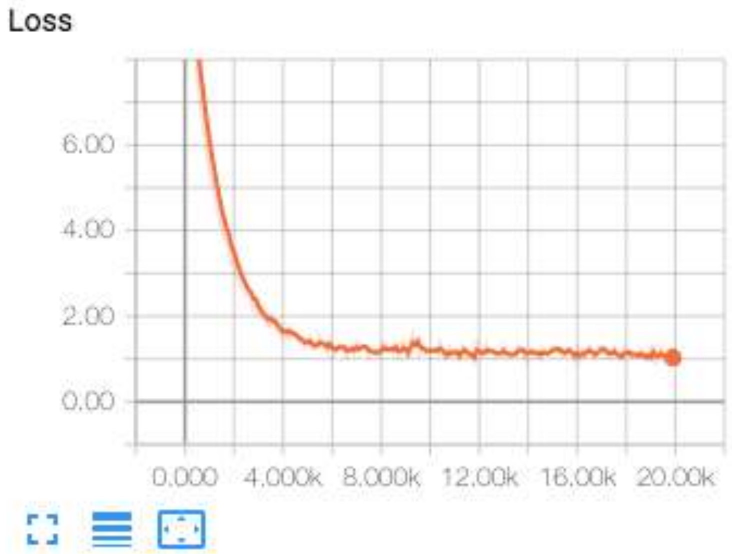
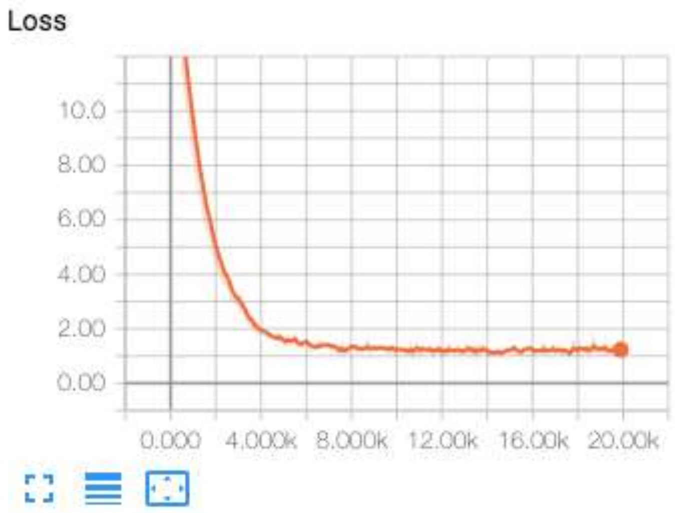


Figure 7. L2 loss of CNN with Average Pooling + Fractional Max Pooling



V. Discussion

We do not discover a significant difference in the test accuracy of the models with different pooling methods. This is somewhat disappointing since this result indicates that the effect of pooling methods are not that huge to make models perform better, being contrary to what we have expected when we were designing our project. Especially, considering that fractional max pooling method is an advanced version of max pooling in that it is gentler and it introduces a degree of randomness to the pooling process, it is even surprising that the prediction accuracy is actually lower than that of traditional max pooling although the difference is negligible. We conclude that the pooling method itself cannot generate a substantial improvement in the performance of CNN. Further methods to improve the model would be changing the grid size from 2x2 to 3x3 to make the pooling regions overlap with each other, or combining the pooling methods with other tuning processes of dropout amount, etc.