EE 569: Homework #5

Issued: 03/23/2020

Problem 1 due: 11:59PM, 04/07/2020

Problem 2 due: 11:59PM, 05/03/2020

General Instructions:

1. Read *Homework Guidelines* for the information about homework programming, write-up and submission. If you make any assumptions about a problem, please clearly state them in your report.
2. You are required to use PYTHON in this assignment. It is recommended to use interface tool KERAS, which is built upon TENSORFLOW, it will be easier for your understanding.
3. TENSORFLOW can also be used in this homework. PYTORCH is an alternative choice if you feel more comfortable with it.
4. DO NOT copy codes from online sources e.g. Github.
5. You need to understand the USC policy on academic integrity and penalties for cheating and plagiarism. These rules will be strictly enforced.

Problem 1: CNN Training on LeNet-5 (100%)

In this problem, you will learn to train a simple convolutional neural network (CNN) called the LeNet-5, introduced by LeCun et al. [1], and apply it to the CIFAR-10 dataset [2]. LeNet-5 is designed for handwritten and machine-printed character recognition. Its architecture is shown in Fig. 1. This network has two *conv* layers, and three *fc* layers. Each conv layer is followed by a *max pooling* layer. Both *conv* layers accept an input receptive field of spatial size *5x5*. The filter numbers of the first and the second *conv* layers are 6 and 16 respectively. The stride parameter is 1 and no padding is used. The two *max pooling* layers take an input window size of *2x2*, reduce the window size to *1x1* by choosing the maximum value of the four responses. The first two *fc* layers have 120 and 84 filters, respectively. The last *fc* layer, the output layer, has size of 10 to match the number of object classes in the CIFAR-10 dataset. Use the popular ReLU activation function [3] for all *conv* and all *fc* layers except for the output layer, which uses softmax [4] to compute the probabilities.

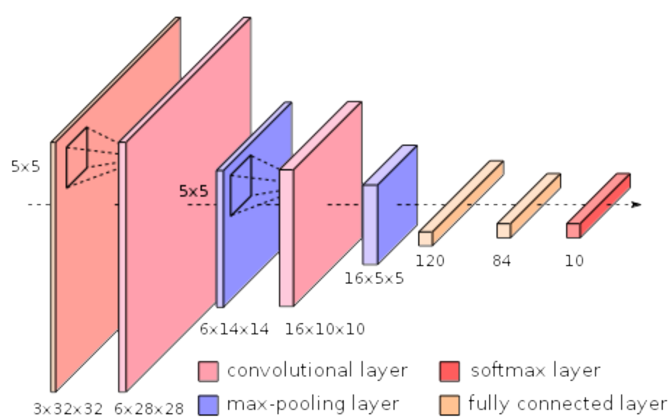

Figure 1: A CNN architecture derived from LeNet-5

The CIFAR-10 dataset consists of 60,000 RGB 32x32 pixel images in 10 classes (with 6000 images per class). It includes a labeled training set of 50,000 images and a test set of 10,000 images. Fig. 2 shows some exemplary images from the CIFAR-10 dataset.

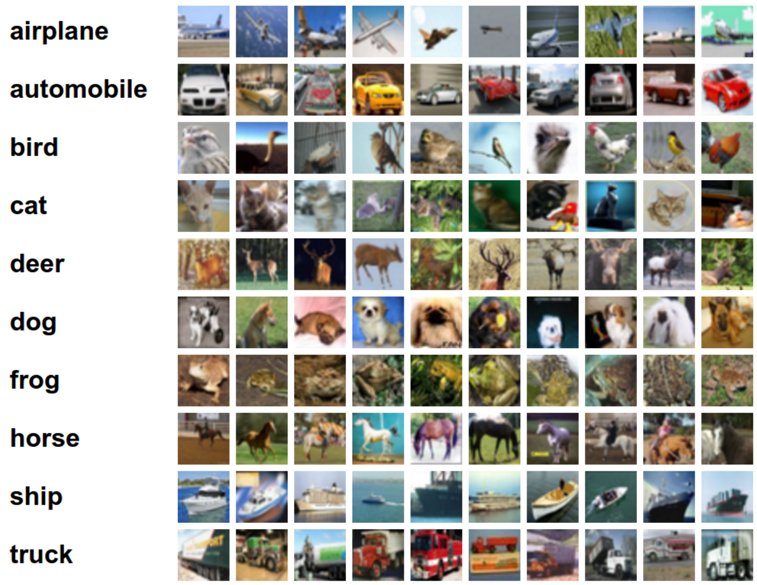

Figure 2: CIFAR-10 images

(a) CNN Architecture (Basic: 20%)

Explain the architecture and operational mechanism of convolutional neural networks by performing the following tasks.

1. Describe CNN components in your own words: 1) the fully connected layer, 2) the convolutional layer, 3) the max pooling layer, 4) the activation function, and 5) the softmax function. What are the functions of these components?
2. What is the over-fitting issue in model learning? Explain any technique that has been used in CNN training to avoid the over-fitting.
3. Why CNNs work much better than other traditional methods in many computer vision problems? You can use the image classification problem as an example to elaborate your points.
4. Explain the loss function and the classical backpropagation (BP) optimization procedure to train such a convolutional neural network.

Show your understanding as much as possible in your own words in your report.

(b) CIFAR-10 Classification (Basic: 50%)

Train the CNN given in Fig. 1 using the 50,000 training images from the CIFAR-10 dataset. You can adopt proper preprocessing techniques and the random network initialization to make your training work easy.

1. Compute the accuracy performance curves using the epoch-accuracy (or iteration-accuracy) plot on training and test datasets separately. Plot the performance curves under 5 different yet representative initial parameter settings (filter weights, learning rate, decay and etc.). Discuss your observations and the effect of different settings.

2. Find the best parameter setting to achieve the highest accuracy on the test set. Then, plot the performance curves for the test set and the training set under this setting.

(c) State-of-the-Art CIFAR-10 Classification (Advanced: 30%)

Check the state-of-art implementation on CIFAR-10 classification in [5]. Select one paper from the list for discussion.

1. Describe what the authors did to achieve such a result. You do not have to implement the network.
2. Compare the solution with LeNet-5 and discuss pros and cons of the two methods.

You can add pictures, flowcharts, and diagrams in your report. If you do so, you need to cite their sources.

Problem 2: EE569 Competition --- CIFAR10 Classification (50%)

Feel free in modifying the baseline CNN in Fig. 1 to improve the classification accuracy obtained in Problem 1(b). For example, you can increase the depth of the network by adding more layers, or/and change the number of filters in some layers. You can augment the dataset. You can also try different activation functions or optimization algorithms. They all have a potential to improve the result. You may need to fine-tune the training parameters to get the training job done. Report the best accuracy that you can achieve and draw the diagram of your network architecture. Describe the training parameter setting to reach this result. Discuss the sources of performance improvement. Your grading in this part will be based on your obtained performance in comparison with other students in the same class.

WARNING: You can borrow the ideas from other papers. But you need to state clearly how you construct the baseline and why you do that. Don't forget to cite their sources. Code copying and other types of plagiarism is strictly forbidden. To be fair for other students, we will double check your code. Please make sure to submit a single python file that is executive.

References

[1] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324
[2] https://www.cs.toronto.edu/~kriz/cifar.html
[3] ReLU https://en.wikipedia.org/wiki/Rectifier_(neural_networks).
[4] Softmax https://en.wikipedia.org/wiki/Softmax_function
[5] CIFAR-10 Leaderboard http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html#43494641522d3130