EQ2425 – Analysis and Search of Visual Data

Project #3: Image Classification using CNNs

due Monday, October 12, 2020, 11:00 pm

Please upload your project in a .zip file to Canvas, the .zip file should contain your project report and codes separately.

1 Objective

The objective of this project is to investigate the preferred network structures and training parameters of convolutional neural networks for recognition of images. The project starts from building and training a three-layer convolutional neural network following the given configuration in Section 2. Then we modify several relevant parameters of the network structure and the training. Based on the performance, we finally choose the preferred configuration.

1.1 Dataset

We use the CIFAR-10 dataset [1] which comprises $50,000$ training examples and $10,000$ test examples of images from 10 classes as training and testing data. Examples in the CIfar dataset are formatted as 32×32 pixel color images.

1.2 Environment

Google Colab (Nvidia T4 GPU)
Tensorflow / Pytorch

1.3 Evaluation

Use the labeled test images as queries. We use the average top-1 recall rate as our criterion for evaluation.

2 Data Pre-processing

For each channel, cast and normalize the pixel values to the range of [-0.5, 0.5].

3 Build the Default Model

3.1 Default Network Structure

Build a three-layer convolutional neural network based on the following configuration:

1. Convolutional layer 1: Filter size: 5×5, stride: 1, zero-padding: 'valid', number of filters: 24

2. ReLu

3. Pooling Layer: Max pooling, filter size: 2×2, stride: 2

4. Convolutional layer 2: Filter size: 3×3, Stride: 1, zero-padding: 'valid', number of filters: 48

5. ReLu

6. Pooling Layer: Max pooling, filter size: 2×2, stride: 2

7. Convolutional layer 3: Filter size: 3×3, Stride: 1, zero-padding: 'valid', number of filters: 96

8. Pooling Layer: Max pooling, filter size: 2×2, stride: 2

9. Fully connected layer 1: Output size: 512

8. ReLu

9. Fully connected layer 2: Output size: 10

10. Softmax classifier.

3.2 Default Training Parameters

First, use a built-in weight initialization function for your chosen implementation framework to initialize the training parameters. Then use a built-in stochastic gradient descent algorithm to train your network based on the following configuration:
Learning rate: 10^{-3};
Size of minibatch: 64;
Training epochs: 300.

4 Network Structure

A. *Number of layers* vs. *Number of filters*
Modify the original network structure based on the following two changes independently: 1. Change the default number of filters of three convolutional layers from $[24, 48, 96]$ to $[64, 128, 256]$. 2. Add a fully connected layer between the original fully connected layer 1 and 2 with the number of output units 128, followed by a ReLu activation function. Report the recalls of the two changes. Select the configuration with the higher recall to proceed.

B. *Filter size*
Based on the preferred configuration from A, change the filter size of convolutional layer 1 and 2 to 7×7 and 5×5, respectively. Report the recall. Select the configuration with the higher recall to proceed.

C. *ReLu* vs. *Leaky ReLu*
Based on the preferred configuration above, change all the activation functions from ReLu to Leaky ReLu. Report the recall rate and explain the reason for the performance difference. Select the configuration with higher recall to proceed.

D. *Dropout* vs. *without Dropout*
Based on the preferred configuration above, add a dropout regulation between the fully connected layer 1 and 2. Set the dropout rate to 0.3. Report the recall. Select the configuration with the higher recall to proceed.

E. *Batch Normalization Layer*
Add batch normalization layer after each activation function ReLu/Leaky ReLu. Report the recall. Select the configuration with the higher recall to proceed.

5 Training Settings

A. *Batch size*
Based on the preferred configuration above, change the batch size from 64 to 256. Report the recall. Also report the change of the training time.

B. *Learning rate*
Based on the preferred configuration above, change the learning rate to 0.1. Report the recall.

C. *Data shuffleing*
Shuffeling the data each epoch. Report the recall.

6 Conclusions

Report the combined network and training configuration which results the best recognition performance. Report which practices give the largest performance improvement.

References

[1] https://www.cs.toronto.edu/ kriz/cifar.html