EQ2425 Analysis and Search of Visual Data

EQ2425, Project 3

Chunyu Wang
chunyuw@kth.se

Jingwen Liu
jingwenl@kth.se

October 13, 2024

Summary

The project focuses on investigating the best configuration for image classification using several three layer Convolutional Neural Networks (CNN). The CIFAR-10 image dataset is used for training and testing. This project begins with building a default three layer CNN, and several modifications are added, including adjusting the number of layers and filters, filter size, activation functions (ReLU vs. Leaky ReLU), dropout, batch normalization, and other training settings.

1 Introduction

In this project, we explore the best configuration of Convolutional Neural Networks (CNN) for image classification among all possible options within the range of this project, specifically applied to the CIFAR-10 dataset. The dataset contains $50,000$ training examples and $10,000$ test examples of 32×32 pixel color images, which is classified into 10 classes.

The goal is to analyze and fine-tune parameters to achieve greater classification accuracy. Starting from a default three-layer CNN model, we perform a series of experiments to investigate the impact of various factors, including adjusting the number of layers and filters, filter size, activation functions, dropout, batch normalization, and other training settings. The aim of having different models is to identify the best configuration that provides the highest recall rate and enhances the overall performance of the model.

2 Problem Description

In this section, the configurations of all CNNs we implemented is introduced.

2.1 Default Network Structure & Training Paramaters

To build a default three-layer CNN for this project, the following configuration is carried out:

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
10. ReLu
11. Fully connected layer 2: Output size: 10
12. Softmax classifier.

After the network structure is built, built-in weight initialization function is used for initializing the training parameters. Then a built-in stochastic gradient descent algorithm is used to train the network based on the following configuration:

- Learning rate: 10^{-3}
- Size of minibatch: 64
- Training epochs: 300. Due to overfitting, the actual epochs applied is 100 or 30.

2.2 Parameter Modification

In this section, the modifications we made to the parameters are introduced.

2.2.1 Number of Layers vs. Number of Filters

The modifications are made respectively:

- Change the default number of filters of three convolutional layers from $[24, 48, 96]$ to $[64, 128, 256]$.
- Add a fully connected layer between the original fully connected layer 1 and 2 with 128 output units, following which a ReLu activation function is added.

The process continues with the configuration of higher recall.

2.2.2 Filter Size

Following from Section 2.2.1, the filter size of convolutional layer 1 and 2 are changed to 7×7 and 5×5 respectively.

The process continues with the configuration of higher recall.

2.2.3 ReLu vs. Leaky ReLu

Following from Section 2.2.2, all the activation functions are changed from ReLu to Leaky ReLu.

The process continues with the configuration of higher recall.

2.2.4 Dropout vs. Without Dropout

Following from Section 2.2.3, a dropout regulation between the fully connected layer 1 and 2 is added.

The process continues with the configuration of higher recall.

2.2.5 Batch Normalization Layer

Following from Section 2.2.4, a batch normalization layer is added after each activation function.

The process continues with the configuration of higher recall.

2.2.6 Other Training Settings

- Batch size: Based on the preferred configuration above in Section 2.2.5, change the batch size from 64 to 256.
- Learning rate: Based on the preferred configuration above in Section 2.2.5, change the learning rate to 0.1.
- Data shuffling: Based on the preferred configuration above in Section 2.2.5, shuffle the data each epoch.

3 Results

As mentioned in Section 2.1, the 300 epochs training may cause the model to overfit. Therefore, we are using 100 or 30 epochs for training.

1. In Section 2.1, the recall rate after training for 100 epochs is 64.37%.
2. In Section 2.2.1, the recall rate after changing the default number of filters and training for 100 epochs is 43.76%. Meanwhile the result for adding a fully connected layer (100 epochs) is 41.47%. Hence, the former model performs better.
3. Continuing with the model in Section 2.2.1 with changing the number of filters, the recall rate after changing the filter size of convolutional layer 1 & 2 (100 epochs) is 13.2%. It has worse performance after modifications, therefore we exclude it for following operations.
4. Continuing with the model in Section 2.2.1, the recall rate after changing all ReLu to Leaky ReLu (30 epochs) is 56.12%.
5. Continuing with the model in Section 2.2.3, the recall rate after adding a dropout regulation between fully connected layer 1 & 2 (30 epochs) is 58.47%.
6. Continuing with the model in Section 2.2.4, the recall rate after adding a batch normalization layer after each activation function (100 epochs) is 75.54%.

The final model adds on from the default model by increasing the default number of filters, changing all ReLu to Leaky ReLu, adding a dropout regulation between fully connected layer 1 & 2, and adding a batch normalization layer after each activation function.

Based on the final model, we make minor modifications to the training settings and collect recall rate respectively.

1. After increasing the batch size to 256, the recall rate is 70.8%.
2. After changing the learning rate to 0.1, the recall rate is 12.05%.
3. After shuffling the data each epoch during training, the recall rate is 74.2%.

4 Conclusions

This project demonstrated the significant impact that modifying CNN structure and parameters can have on the performance of CNN for image classification. The CNN is tested and updated according to its training recall rate.

To prevent overfitting, the maximum training epoch is limited to 100. By increasing the default number of filters, changing all ReLu to Leaky ReLu, adding a dropout regulation between fully connected layer 1 & 2, and adding a batch normalization layer after each activation function as shown in Section 3, significant improvements are achieved over the default CNN model. Overall, this project successfully improved the performance of image classification CNN by modifying its network and parameters.

References

[1] Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, Prentice Hall, 2nd ed., 2002

[2] Tobias Oetiker et al., *The Not So Short Introduction to LaTeX 2_ε*, Available: http://tobi.oetiker.ch/lshort/lshort.pdf, Last accessed: March 17, 2009