Homework1

Difficulty : Easy

Workload : Middle

In this assignment, you need to implement the entire image classification pipeline on Fashion Mnist, including image input, data preprocessing, network forward and backward process, and finally use the model on the test data to evaluate performance. Meanwhile, you will use numpy to implement a variety of machine learning algorithms, and learn to split the parts of the data set and finally apply the algorithms you implement to some specific data..

■ Tasks:

- Fill blank code blocks with codes generated with your own efforts (75%)

- Answer questions using markdown (25%)

■ Details:

✧ Classifier Based on Neighbors and Support Vectors(45 %)

In the implementation part of the KNN algorithm, you not only need to implement the entire pipeline based on the neighbor-based classification algorithm, but also need to compare the performance, by changing the code writing method to achieve several times or even tens of times acceleration. Finally, the cross-validation will help you measure the representation ability of your model under parameter K.

In the implementation part of the svm, you will use a learning algorithm to implement a linear support vector machine and apply the SVM to the features extracted by the neural network. In addition to paying attention to the correctness of the algorithm, you also need to pay attention to the performance of the algorithm. We will guide you through loop calculations and matrix calculations to gain a deeper understanding of the performance of parallel computing.

✧ Transfer Features into Probability and Neural Network(45 %)

In the first part, you will learn the most important techniques for converting vectors into probabilities in deep learning: Logarithm probability. By transforming features into probabilities, we can apply arbitrary neural network structures to classification problems. Finally, you need to independently write the anti-transfer formula of the softmax layer to deepen your theoretical understanding.

In the rest part, you need to design and implement a two-layer neural network structure, including network front-end calculations, learning algorithms, and hyper-parameter tuning. Here you will learn how to debug the structure of a neural network. Finally, neural networks visualization will helps deepen your understanding of the online learning process.

✧ Image Features Understanding with a Higher Level Representations(10 %)

In this section, IPython Notebook features.ipynb will walk you through this exercise, in which you'll examine the improvements you've made with advanced representations relative to using raw pixel values. By using a higher representation than the original pixel, you will end up with a basic understanding of performance improvement (eg color histogram, gradient histogram (HOG) function)

Homework2

Difficulty : Hard

Workload : High

This assignment will help to refine the rest of the neural network you have implemented, including the modular construction of multi-layer neural network structures, the implementation of multiple learning strategies, network regularization of neurons and data. The content includes data and regularization methods on the network side, some neuron inactivation strategies and the application of open source framework in convolutional neural network construction.

▪ Tasks:

- Fill blank code blocks with codes generated with your own efforts (75%)

- Answer questions using markdown (25%)

▪ Details:

✧ FC Layers in Neural Network (20 %)

In this part, you need to implement the modularization of the fully connected network, the forward and backward calculation of ReLU activation, the momentum-based and variance-based neural network learning algorithm. Finally, the parts are spliced to complete the multilayer neural network. Build and compare the training process of various learning strategies to gain a deeper understanding of the importance of network parameter adjustment.

✧ Different Kinds of Normalization and Random Drops(40 %)

In the implementation part of normalizations, you need to implement the forward and backward path of Batch Normalization, and add the BN layer to the previously implemented multi-layer fully connected neural network. By a reasonable initialization method, compare the convergence curve of the network before and after adding the BN layer, you will gain a more comprehensive understanding of the impact of regularization technology on neural network training. Finally, you will implement another Normalization method: Layer Normalization.

Besides, you will learn about the effects of random inactivation of neurons on model training. As a regularization technique, random inactivation of forward and backward is simple and effective to guide network training.

✧ Convolutional Neural Networ (30 %)

Convolutional neural networks, as the most basic technology in the direction of computer vision, influent the development of deep learning. In this section you need to implement a variety of convolution operations including pooling layers. By assembling multiple convolution layers together, the overall deep convolutional neural network architecture is finally realized. Meanwhile, you also need to implement the forward and backward propagation of Spatial batch normalization and Group Normalization, which are specific to the convolutional neural networks. Through experiments you will learn that convolutional neural networks plays an incomparable role and status of image processing.

✧ PyTorch / TensorFlow on Cifar-10 (10 %)

For this last part, you will be working in either TensorFlow or PyTorch, two popular and powerful deep learning frameworks. **You only need to complete ONE of these two notebooks.** You do NOT need to do both, but a very small amount of extra credit will be awarded to those who do so.

Open up either **PyTorch.ipynb** or **TensorFlow.ipynb.** There, you will learn how the framework works, culminating in training a convolutional network of your own design on Cifar-10 to get the best performance you can.