华南理工大学

South China University of Technology

The Experiment Report of Deep Learning

SCHOOL: SCHOOL OF SOFTWARE ENGINEERING

SUBJECT: SOFTWARE ENGINEERING

Author:
Yiqun Wu, Hainan Huang, Yuxuan Gao

Supervisor:
Mingkui Tan

Student ID：
Yiqun Wu : 201720145082
Hainan Huang: 201720145150
Yuxuan Gao: 201720145075

Grade:
Graduate

December 24 2017

Handwritten Digit Recognition based on Shallow Neural Network

Abstract— In this paper, we build a shallow neural network LeNet5 based on pytorch to recognize handwritten digit, understanding of convolution, pooling, ReLu, fully connected and other network layers deeply. We also discuss the experimental results in the paper.

I. INTRODUCTION

MNIST ("Modified National Institute of Standards and Technology") is the de facto “hello world” dataset of computer vision. Since its release in 1999, this classic dataset of handwritten images has served as the basis for benchmarking classification algorithms. As new machine learning techniques emerge, MNIST remains a reliable resource for researchers and learners alike [1].

In machine learning, a convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery [2]. In this experiment, the structure of the convolution neural network is given as follows:

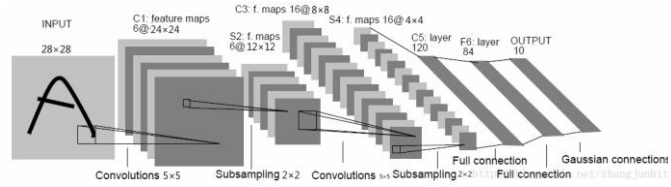

Fig. 1. CNN network structure the experiment used

In this network, there are two convolutions layer and two sampling layer are carried out. After convolutions and sampling the network uses two full connection and then use one Gaussian layer to get result. We create this network by inherit nn.Module in pytorch.

In this experiment, we build a shallow neural network LeNet5 based on pytorch to recognize handwritten digit, and the different experimental parameters were compared and discussed, finally obtained a satisfactory experimental results, we will describe the experimental principle and the steps we detail below, and discuss some of the results.

II. METHODS AND THEORY

The whole experiment is divided into the following steps:

1) Build shallow neural network LeNet5 (A class inheriting from nn.Module) in model.py. Pytorch is used to build this network.
2) Complete the network training and validation process in train.py: Use torch.utils.data.DataLoader, torchvision.datasets.MNIST and torchvision.transforms to load dataset.
3) Instantiate torch.optim.SGD to get the optimizer
4) Instantiate LeNet5 to get net.
5) Input data, which size is batch size, into the network for forward propagation to get predict target.
6) Use torch.nn.CrossEntropyLoss to calculate the loss between predict target and ground truth target.
7) Calculate gradient using loss.backward and optimize the net using optimizer.step.
8) Calculate accuracy of recognition.
9) Repeat step 5 to 8 until all data are inputed to the net.
10) Define the number of selected training epoch, repeat the training process 6—10
11) Save the best model as a .pkl file.
12) Compared with the training , there is no optimization process in validation. The other processes between training and validation are basically the same.
13) Load the trained model (.pkl file) into the main.py application. The implementation of main.py can test the model's performance on the web page.

Next, we will introduce the theory used in the experiment.

A. Convolutional Neural Networks

Convolutional Neural Networks are a special kind of multi-layer neural networks. Like almost every other neural networks they are trained with a version of the back-propagation algorithm. Where they differ is in the architecture.

Convolutional Neural Networks are designed to recognize visual patterns directly from pixel images with minimal preprocessing.

They can recognize patterns with extreme variability (such as handwritten characters), and with robustness to distortions and simple geometric transformations.

1) Sparse Connectivity

CNNs exploit spatially-local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. In other words, the inputs of hidden units in layer m are from a subset of units in layer m-1, units that have spatially contiguous receptive fields. We can illustrate this graphically as follows:

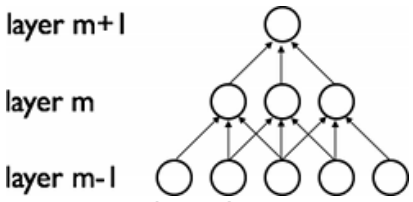

Fig. 2. Sparse Connectivity

2) Shared Weights

In addition, in CNNs, each filter h_i is replicated across the entire visual field. These replicated units share the same parameterization (weight vector and bias) and form a feature map.

B. LeNet

LeNet is a convolutional network designed for handwritten and machine-printed character recognition. Sparse, convolutional layers and max-pooling are at the heart of the LeNet family of models. While the exact details of the model will vary greatly, the figure below shows a graphical depiction of a LeNet model.

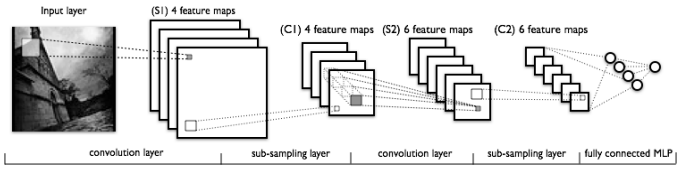

Fig. 3. LeNet model

The lower-layers are composed to alternating convolution and max-pooling layers. The upper-layers however are fully-connected and correspond to a traditional MLP (hidden layer + logistic regression). The input to the first fully-connected layer is the set of all features maps at the layer below.

From an implementation point of view, this means lower-layers operate on 4D tensors. These are then flattened to a 2D matrix of rasterized feature maps, to be compatible with our previous MLP implementation.

III. EXPERIMENT

A. Dataset

This experiment use handwritten digital data set - MNIST, which contains 60,000 hand-written digital images foe training and 10,000 hand-written digital images for validation. Each image is 28 ×28 pixels in size.

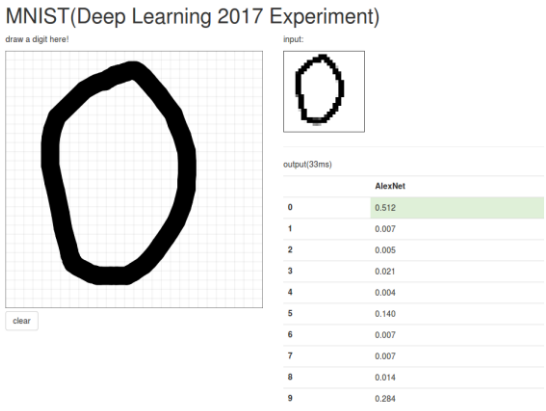

Fig. 4. Recognization result of the number 0

For this paper, we used handwritten digital data set - MNIST to train the model and save the model as net.pkl. The ratio of the training set to the verification set is 6:1. Finally, we used the model to test the online handwritten dataset.

B. Implementation

a. Data preprocessing

First, we divided the handwritten dataset into a 6-1 training set and validation set, and then normalized the data.

b. Implementation of LeNet5

Please refer to the part II.

c. Experimental result

Our lab is based on LeNet5 and the experimental parameters are set as bellows.

TABLE1. Experiment parameter setting

η	iters	batch_num
0.01	10	64

Among them, η represents the learning rate, iters represents the number of iterations and batch_num represents the batch number of images processed.

```
Train Epoch: 1 [0/60000 (0%)]   Loss: 2.195670
Train Epoch: 1 [1000/60000 (2%)]    Loss: 2.300479
Train Epoch: 1 [2000/60000 (3%)]    Loss: 0.018494
Train Epoch: 1 [3000/60000 (5%)]    Loss: 0.250411
Train Epoch: 1 [4000/60000 (7%)]    Loss: 0.008531
Train Epoch: 1 [5000/60000 (8%)]    Loss: 0.666226
Train Epoch: 1 [6000/60000 (10%)]   Loss: 0.414315
Train Epoch: 1 [7000/60000 (12%)]   Loss: 0.620478
Train Epoch: 1 [8000/60000 (13%)]   Loss: 0.000099
Train Epoch: 1 [9000/60000 (15%)]   Loss: 0.016527
Train Epoch: 1 [10000/60000 (17%)]  Loss: 0.012969
Train Epoch: 1 [11000/60000 (18%)]  Loss: 0.000010
Train Epoch: 1 [12000/60000 (20%)]  Loss: 0.010653
Train Epoch: 1 [13000/60000 (22%)]  Loss: 0.027759
```

Fig. 5. The training process of the model

With the model of the ten iterative training, the average loss of the validation set is 0.0734, and the accuracy is 98%. The results of online handwritten digital tests are as follows.

1) The probability of predicting the number 1 is 0.778.

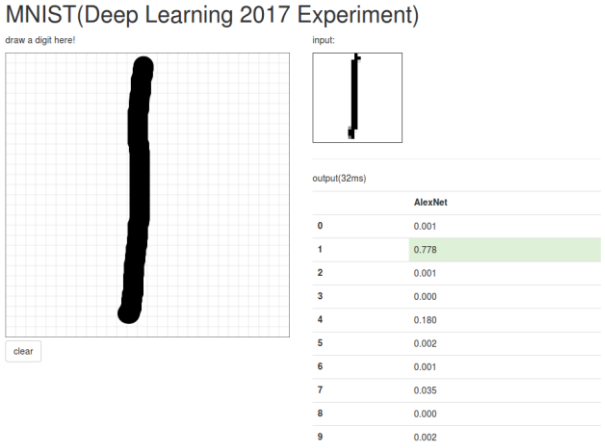

Fig. 6. Prediction result of the number 1

2) The probability of predicting the number 2 is 0.422.

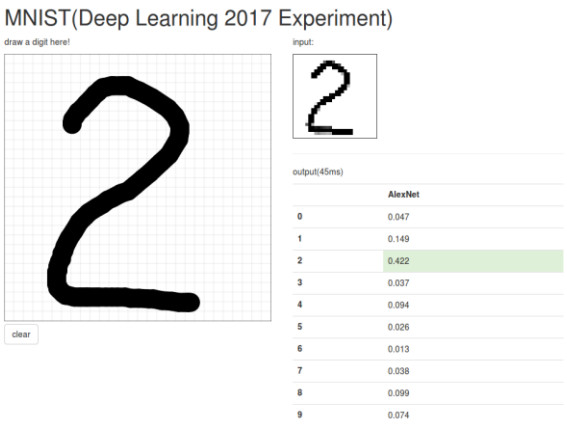

Fig. 7. Prediction result of the number 2

3) The probability of predicting the number 3 is 0.553.

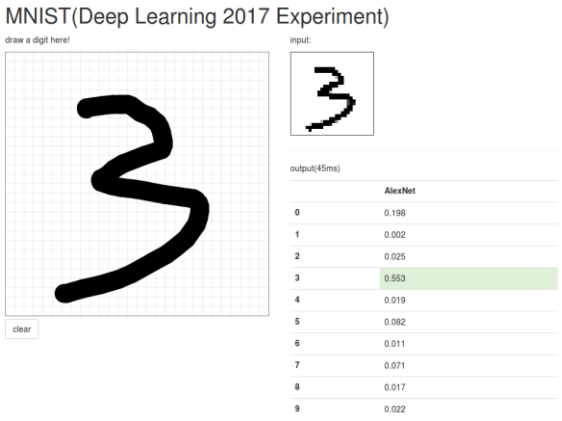

Fig. 8 Prediction result of the number 3

Seen from the above results, the model effect of our training is generally good. For the number 1, 2, 3, we all succeeded in judging the correct results, even if the probabilities were different. Among them, the judgment of the number 1 is more decisive, because the probability is 0.778.

IV. CONCLUSION

In this experiment, we learn the deep neural network construction process and apply the theory into practical problems: the handwritten digit recognition based on shallow neural network. Finally we obtained a satisfactory experimental results.

From this experiment we understand deep learning framework pytorch and preliminarily use this tool in the practical problem.

What's more, we experience the neural network training and testing process, and deepen the understanding of convolution, pooling, ReLu, fully connected and other network layers.

To sum up, this experiment enables us to realize the handwritten digit recognition problem based on shallow neural network, and deepen our understanding of CNN. What's more valuable is that we have exercised teamwork ability. Thank the teacher for arranging this experiment, and let us have the opportunity to improve ourselves.

References:

[1] kaggle. digit-recognizer [EB/OL]. https://www.kaggle.com/c/digit-recognizer

[2] wikipedia.org. Convolutional neural network [EB/OL]. https://en.wikipedia.org/wiki/Convolutional_neural_network