The Image Classification Algorithm Was Implemented on The MNIST Data Set

Wayuan Xiao

Abstract:

In the current era of rapid development of science and technology, the recognition and classification of digital images is the key to solving many problems, such as the application of license plate recognition, document digitization, and remote sensing image surface classification. Based on the MNIST handwritten numerical data set collected by the National Institute of Standards and Technology (NIST), this report uses Python language and PyTorch programming framework to construct a convolutional neural network (CNN) structure and practice and experience the image classification of handwritten digits in the MNIST data set.

Keywords: Convolutional neural networks, Convolution kernel, Data set, Pooling layer

1. Introduction

The MNIST data set is a handwritten digital image set collected by the National Institute of Standards and Technology of the United States of America and compiled by collecting handwritten digital handwriting images from 250 people. The purpose of collecting this data set is to realize the recognition of handwritten digits through algorithms. Deep learning is a machine learning algorithm used to recognize data images. Deep learning simulates the human brain's neural network and is trained on large amounts of data to recognize and classify information such as images, speech, and natural language.

In 1989, LeCun, the first person to solve the problem of handwritten digits on the MNIST data set through convolutional neural networks, used the CNN algorithm to realize the recognition of handwritten digits based on the MNIST data set in his paper "Gradient-Based Learning Applied to Document Recognition"[1] which made MNIST gradually become a well-known handwritten numeric data set. Since 1998, this data set has been widely used in machine learning and deep learning to test the effectiveness of algorithms such as linear classifiers, K-nearest neighbors, support vector machines (SVMs), neural nets, convolutional neural networks, etc.

In this article, a convolutional neural network structure is constructed. Convolutional neural networks (CNNs) are a common deep learning architecture for image recognition. This technology, which enables robots to autonomously identify specific targets, is a major advance in computer vision and greatly improves work efficiency.

2. Related work

2.1Convolutional neural networks (CNNs)

A convolutional neural network is a specialized artificial neural network, and the image is the matrix, so the convolutional neural network replaces the general matrix multiplication method through the convolution operation, and the convolutional layer convolutes the input image and transmits its results to the next layer.[2] A convolutional neural network consists of an input, hidden, and output layer specifically designed to process pixel data and for image recognition and processing.[3] The convolutional neural network contains the following concepts: Kernal, Stride, Padding, Channel, and Feature Map. Moreover, they are mainly composed of the following five layer-level structures: Input layer, Convolutional computing layer(CONV layer), ReLU layer, Pooling layer, and FC layer.

2.2 Input layer

The input layer of the neural network is the first layer of the convolutional neural network and is the only layer that interacts with the outside world. The data input layer preprocesses the raw image data. In an image classification task, each pixel is a neuron in the input layer. Its role is to transform the input data into a format that can be processed inside the neural network.[4]

2.3 Convolutional computing layer(CONV layer)

The convolutional computing layer is the most important in a convolutional neural network, which is locally perceived and shared globally.

The convolution kernel convolutes the input content by the prescribed step size for each move. In addition, we can find that the output result obtained by convoluting the 32*32 input content with a 5*5 convolution kernel is 28*28. We will find that its spatial dimension becomes smaller, so we need to use padding to retain as much of the original input content information as possible, and the operation of filling is to add zeros to the surrounding boundaries of the input content so that the input and output content can always maintain the same spatial dimension. The following diagram illustrates the calculation process of convolution.

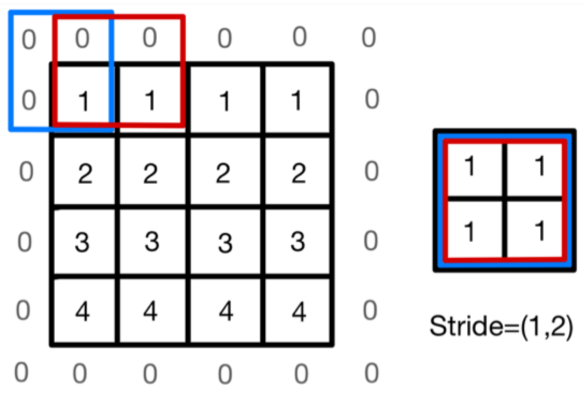

Figure1. CONV stride

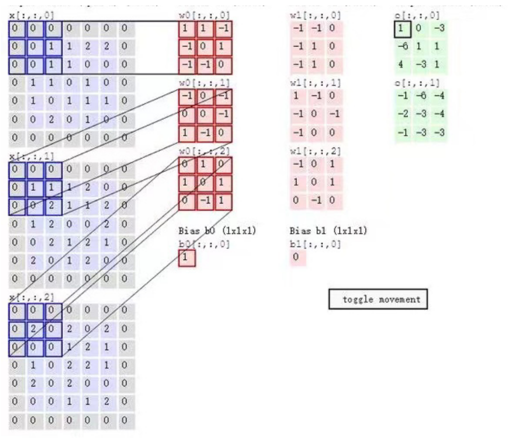

Figure2. CONV toggle movement

2.4 Pooling layer

Pooling[5] plays a role in dimensionality reduction. In convolutional neural networks, pooling operations are often followed by convolutional operations and are used to reduce the dimensionality of feature maps. The basic idea of the pooling operation is to divide the feature graph into several sub-regions and make a statistical summary of each sub-region. There are two most common pooling operations: maximum pooling and average pooling. Through pooling, we can reduce the dimension of the feature map, reduce the number of parameters in the network, avoid the occurrence of overfitting, and improve the calculation speed and operation efficiency of the model.

2.5 Fully connected layer

All neurons between the two layers are heavily connected. In general, the fully connected layer appears at the tail of the convolutional neural network. Please refer to the Official website of the MNIST data set: http://yann.lecun.com/exdb/mnist/.

3. Experimental analysis:

(1)Load the necessary libraries

Add the torch network model and abbreviate it to “nn,” import function, define it as F, import the optimizer and abbreviate it as “optimal,” load torch-vision to operate on the database, load datasets, and transform.

(2) Defining Hyperparameters

Hyperparameters are used to define the model structure or optimization strategy. The difference between parameters and hyperparameters is that the neural network learns parameters, while hyperparameters are set artificially. In this experiment, we need to process 60,000 images, so this definition aims to process 60,000 images in batches to improve our operational efficiency.

The role of BATCH_SIZE is to define the quantity to be processed for each batch. In this experiment, we define the quantity processed per batch as 16.

DEVICE refers to the device used by the computer in this experiment, whether it is trained on the CPU or the GPU.

EPOCHS aims to define the number of training rounds, i.e., to cycle a data set through several rounds of operations. In this experiment, we set 10 rounds of training.

(3) Build a pipeline and process the images

Call the transform library to build a pipeline, and use the composition to transform each input image, including stretching, zooming in, shrinking, rotating, etc. Because the operation in PyTorch is carried out in tensor format, equivalent to a container, we also need to convert the image to tensor format. The effect of normalization for regularization is that when the model is overfitted, regularization can be used to reduce the complexity of the model.

(4) Download, load data

In this experiment, we need to download the data through the code, load it into the model, including 60,000 training sets and 10,000 test sets, and use “datasets” to download the data set. The download order of the training set and the test set is as follows: download the training set first and then download the test set. After downloading the data set, we need to load the data and use DateLoader to load it, in which we need to set the batch_ size hyperparameter. You also need to set shuffle to scramble the image for better accuracy.

(5) Build a network model

Building a network model is the most important part of this experiment. First, you must build Digit, inherit the Module, and call the Module, which defines the properties. Second, we start to build 2D convolution: in the first layer, we set the input channel to 1, the output channel to 10, and the convolution kernel to be 5*5, and the second layer, because the input of the second layer is the output of the first layer, the input channel of the second layer is 10, the output channel is 20, and the convolution kernel is 3*3. In the next place, we need to define a fully connected layer, which is a linear layer. The first fully connected layer has 20*10*10 input channels and 500 output channels.

Similarly, the number of input channels in the second layer is the number of output channels in the first layer. Hence, the second fully connected layer has 500 input and ten output channels. After that, we define forward propagation and perform a convolution operation. We call convolutional layer 1, input batch*1*28*28, and output batch*10*24*24, where the source of 10 is output channel 10, and 24 is obtained by image pixel 28 minus the convolution kernel five plus step 1. Afterward, the output of x is given to the activation function to maintain the output, where the activation function is to add an activation function between all the hidden layers so that the output is a nonlinear function so that the neural network becomes more expressive. Next, we need to set a pooling layer to reduce the dimension of the feature map, where we set the maximum pooling layer to (x,2,2), 2 is the step size, input batch*10*24*24, output batch*10*12*12.

Furthermore, we call convolutional layer 2, input batch*10*12*12, output batch*20*10*10, and then activate it with the above convolutional layer 1. After that, we stretch it and send the stretched data to the fully connected layer. The first layer is input batch* 2000 and output batch* 500, and they are activated again to make the model more fully expressed. The second layer: input batch*500, output batch*10. Finally, the softmax loss function is called to calculate the probability value of each number after classification and finally returns.

(6) Define the optimizer

Create a model Digit and deploy it to the device, and then, create an optimizer using the Adam optimizer, which is used to update the model parameters to optimize the training and test results.

(7) Define the training method

First, you need to upload the model, device, data, and optimizer; secondly, start model training: call train, read data and target, deploy data to the device, and initialize the gradient to zero. Secondly, you need to predict the result after training, then compare the real value with the predicted value, call the cross-entropy loss function, and accumulate to calculate the loss, then call the max function, define the horizontal axis, find the largest subscript of each row on the horizontal axis, to obtain the subscript with the largest probability value, and then perform backpropagation, and then update the parameters, and use the if statement to judge multiple cycles.

(8) Define the test method

In general, the training method is defined as the same. First, you must upload the model, equipment, and test data set. Secondly, the model is verified, and the model_eval is called to verify the model. Next, the correct rate is assumed to be 0 initially, and then the correct rate is counted. Then, the test loss is required, and there is no need to calculate the gradient and backpropagation in the test. The above two items only need to be carried out in training. After testing the loss, we need to read the data and target to the test_loader and then deploy them to the DEVICE. Next, the model is called to test the data, and then the cross-entropy function is used to calculate the test loss, and the subscript with the largest probability value is found. Finally, the accuracy is accumulated.

(9) Invoke training and testing methods

We must also upload the model, hyperparameters, devices, data, and the optimizer.

(10) Consequence

Ultimately, the accuracy rate can reach 98 percent or 99 percent.

4. Conclusion

Using PyTorch to construct a new environment for experimental operation, a two-layer convolutional neural network was built, one of which used a convolutional kernel of 5*5 and a layer of 3*3. A pooling layer was built to reduce the dimensionality of the image, and secondly, the data were trained and tested, and the handwritten numbers in the MNIST data set were classified by setting hyperparameters, backpropagation, and calculation loss. Eventually, the accuracy was between 98% and 99%.

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[2] Charmve. 2021. Explanation of the principles of convolutional neural networks for computer algorithms. Zhihu. May 22. https://mbd.baidu.com/ma/s/Dhd63xfk

[3] Anonymous users. 2020. Convolutional neural networks. Global Encyclopedia. https://vibaike.com/174473/

[4] Srrrrr ran. 2023. What are the roles of the various layers of a neural network. Baidu AI Studio. March 13. https://ml.mbd.baidu.com/r/19D9fWEVS1O?f=cp&u=bb6d2c549690db60

[5] Helloworld188888. Pooling layer. CSDN. https://minipro.baidu.com/ma/qrcode/parser?app_key=y1lpwNoOyVpW33XOPd72rzN4aUS43Y3O&_swebFromHost=baiduboxapp&path=%2Fpages%2Fblog%2Findex%3FblogId%3D130371223&launchid=CFD4DD52-9136-4213-AE59-776BEB20828B