1st World Congress on Industrial Engineering and Operations Management
October 9-11, 2024
Publisher: IEOM Society International, USA DOI: 10.46254/WC01.20240171
Published: October 09, 2024,

Correlation of MNIST Classification Accuracy and Training Time According to Neural Network Depth

Youju Rim, Shin Dong Ho
Student and Professor, My Paul School
12-11, Dowontongmi-gil, Cheongcheon-myeon, Goesan-gun
Chungcheongbuk-do, Republic of Korea
eavatar@hanmail.net

Abstract

In this study, we will observe how MNIST learning results vary depending on the depth of the artificial neural network, that is, the number of hidden layers, and analyze the correlation to describe it. The experiment was performed under the same conditions, with only the number of hidden layers adjusted. As a result of adjusting the hidden layer up to the 10th floor, it was found that a higher number of hidden layers does not necessarily lead to more accurate results.

Keywords

NND, MNIST, Neural Network Depth, NIST and AI

1. Introduction

Ensuring Artificial neural networks are algorithms inspired by neural networks in biology in machine learning and cognitive science that mimic the way biological neurons signal each other. It is mainly used to learn patterns from data and make predictions. An artificial neural network consists of one input layer, one or more hidden layers, and one output layer. The input layer receives data from the outside, and when external data enters the input layer, the input layer forwards the data to the neural network. The hidden layer is located between the input layer and the output layer, and can have multiple layers, and each hidden layer is composed of several nodes, that is, neurons, which are connected to each other and are responsible for information transmission and conversion. The hidden layer learns abstract representations from the input data and models complex patterns through them. The Output layer produces the final output of the neural network. The role of the output layer varies depending on the type of job, but the output layer generates output according to the purpose of the given task.

2. Body

2.1 MNIST

MNIST is a large database of handwritten numbers from 0 to 9. The MNIST database has 60,000 training sets and 10,000 example test sets. These test sets are a subset of the larger sets available from NIST. All NIST images are 28x28 pixel black and white images. MNIST is commonly used to train a variety of image processing systems. The database is also widely used in training and testing in the field of machine learning.

3. Experimental Design

1) Input neurons

Input neurons are equal to the sum of the weighted outputs of all neurons in the previous layer. Each input is multiplied by the weight associated with the synapse that connects the input to the current neuron. In the end, the number of input neurons is equal to the number of features you want to use. Since this study uses the MNIST dataset, we will use 28x28 images. Thus, the number of features and input neurons is 784.

2) **Output neurons**

Output neurons are neurons located in the last layer of the artificial neural network structure. These neurons produce the final output of the network. The number of output neurons are the number of cases you want to predict. In this study, the number of output neurons will be determined to be 10 because the purpose is to classify handwriting from 0~9 in the MNIST dataset.

3) Hidden layer

The hidden layer is the layer located between the input layer and the output layer in the artificial neural network structure, and intermediate calculations are performed. A hidden layer can have multiple layers, and each hidden layer is connected to each other and is responsible for information transmission and transformation. In this study, the hidden layer will be gradually observed from the first floor and the layer will be increased.

4) Hidden layers of neurons

Hidden layers of neurons sit between the input and output of the algorithm, and the function weights the input and passes it to the output through the activation function. There is no specific criterion for determining the number of hidden layers of neurons. The number of hidden layers of neurons is to be found in the experiment. In this study, the number of hidden layers of neurons was set at 50.

5) Loss function

The loss function is a function that measures the error between the model's predicted value and the actual value in machine learning and deep learning models. It is used to train a model by numerically expressing how far the predicted result of the model deviates from the actual value. The purpose of the loss function is to minimize the error between the model's predicted value and the actual value. In this study, we will use cross-entropy loss, which is often used in classification problems among various loss functions.

6) Batch size

Batch size refers to the number of samples of data that machine learning and deep learning models are training at once. If the batch size is large, it will train with a lot of data at a time. It learns quickly, converges to a certain level very quickly, and is less likely to fall into local optima. However, if the batch size is larger than the small batch size, the deviation of the calculated loss value is smaller, and there is a risk of overfitting. Conversely, if the batch size is small, the iteration per epoch is large, so there will be more steps. If the batch size is small, the variance of loss is large because it learns from small data, so the amount of data calculated at once has a regularizing effect, so it is learned a little more diversely and sharply. However, there are many steps, which can lead to local minima, and it takes a long time to learn. In this study, we will experiment with a setting of 32.

7) Number of Epoch

Epoch refers to machine learning and deep learning models that assign an entire training dataset to the model once and update it. More epochs are not always better. As the number of epochs increases, the model learns better on the training data, but if you use too many epochs, the model may become overfit to the training data and perform poorly on test data or new data. Choose the appropriate number of epochs to avoid overfitting. In this study, the number of epochs will be flexibly adjusted up to the optimal point.

8) Optimizer

The optimizer is responsible for finding the parameters when the loss function is at its minimum. There are many different types of optimizers, and in this study, we will be using Adam. Adam is an optimizer who mixes two types of optimizers: Momentum and RMSProp. Adam is an optimizer that gives inertia to the pace of progress and has an adaptive learning rate based on the amount of recent curved changes in the path. It has been proven to work on different neural networks with a wide range of architectures, making it the most widely used optimizer at present.

9) Learning rate

Learning rate is a tuning parameter for the optimization algorithm, which determines the step size in each iteration and moves towards the least loss function. Finding the right learning rate is crucial. Whenever you adjust the hyper-parameters of the network, the learning rate must be adjusted as well. To find the optimal learning rate, it is necessary to start with a low value and gradually increase it, checking the loss of each learning rate to find the optimal value. In this study, we will set the learning rate as 0.001.

10) Activation function

Activation function is a function that converts the sum of the weights of the input signal into the output signal in an artificial neural network. Activation functions determine the output of a neural network and allow the network to learn complex nonlinear relationships. The Activation function takes the input, determines whether each neuron in the neural network will be activated, and generates the output. In this study, we will use the ReLU function, which outputs the value as it is if the input is positive, and 0 if it is negative.

4. Research Process

```
model_hidden_1 = tf.keras.models.Sequential(
    [
        tf.keras.layers.Flatten(input_shape=(28, 28)),
        tf.keras.layers.Dense(50, activation='relu'),
        tf.keras.layers.Dense(10, activation='softmax')
    ]
)
model_hidden_1.summary()
```

Figure 1. Model Design

The model is designed using 'tf.keras.models.Sequential' and consists of an input layer, hidden layer, and output layer. For detailed layer composition, refer to the model design code above.

```
import os
import time

class MyCallBack(tf.keras.callbacks.Callback):

    def on_epoch_end(self, epoch, logs):
        os.makedirs(f'./{epoch}', exist_ok=True)
        model_hidden_1.save(f'./{epoch}/model_hidden_1.keras')


start = time.time()
history_1 = model_hidden_1.fit(
    x=x_train,
    y=y_train,
    epochs=50,
    callbacks=[
        MyCallBack(),
        tf.keras.callbacks.EarlyStopping(monitor='loss', patience=1)
    ],
)
end = time.time()
model1_training_time = end - start
```

Figure 2. Train the model and set up callbacks

For model training, 'tf.keras, callbacks' was used to save the model in the middle of training and set the training end condition. The code above shows model training and callback implementation.

4.1 Experimental results

If you look at the table above, you can see that many layers of hidden layers does not bring an accurate result, but you need to find an appropriate number depending on the situation. Increasing the number of layers of hidden layers can produce more accurate results, but based on the current experimental results, simply increasing the number of layers of hidden layers will not continue to produce accurate results.

Table 1. Hidden layer changes by number of layers

Number of hidden layers	Training time	loss	accuracy
1	113.1314	0.0129	0.9958
2	87.1094	0.0217	0.9923
3	83.7983	0.0276	0.9906
4	108.4202	0.0252	0.9916
5	118.6033	0.0278	0.9912
6	90.0231	0.0392	0.9880
7	170.8114	0.0256	0.9925
8	123.8747	0.0389	0.9883
9	113.9952	0.0510	0.9856
10	129.7146	0.0489	0.9867

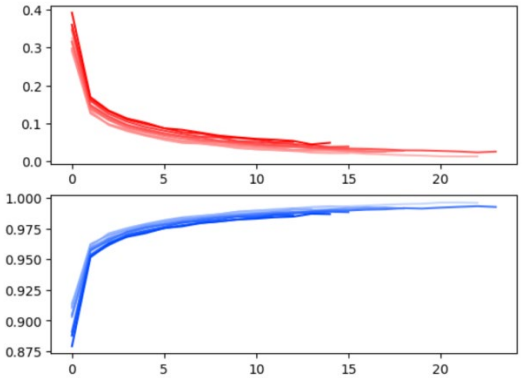

Figure 3. Graph for accuracy and loss by layer

Figure 3 above shows the loss and accuracy of the training data according to the number of layers of the hidden layer. Looking at the graph, it can be seen that the accuracy increases and the degree of loss decreases compared to the number of layers of the hidden layer, but the number of hidden layers does not mean that the accuracy is the highest and the loss is the lowest.

```
test_1 = model_hidden_1.evaluate(x_test, y_test)
test_2 = model_hidden_2.evaluate(x_test, y_test)
test_3 = model_hidden_3.evaluate(x_test, y_test)
test_4 = model_hidden_4.evaluate(x_test, y_test)
test_5 = model_hidden_5.evaluate(x_test, y_test)
test_6 = model_hidden_6.evaluate(x_test, y_test)
test_7 = model_hidden_7.evaluate(x_test, y_test)
test_8 = model_hidden_8.evaluate(x_test, y_test)
test_9 = model_hidden_9.evaluate(x_test, y_test)
test_10 = model_hidden_10.evaluate(x_test, y_test)
```

Figure 4 Test Data Execution Code

The code above tests all the MNIST datasets from 1 to 9 that have been set aside. If you run the above code, you will get the loss and accuracy as shown in the following table.

Table 2. Loss and accuracy by data

MNIST dataset	loss	accuracy
1	0.1239	0.9730
2	0.1257	0.9743
3	0.1307	0.9729
4	0.1049	0.9757
5	0.1369	0.9683
6	0.1333	0.9716
7	0.1353	0.9698
8	0.1152	0.9715
9	0.1111	0.9768

If you look at the table above, you can see that the loss and accuracy of each handwritten numeric data is different. Under the same conditions, that is, even if the number of layers of the hidden layer is the same, there may be differences in the degree of loss and accuracy depending on the data learned.

5. Conclusion

In this study, the MNIST dataset was used to analyze the effect of the number of hidden layers in the neural network model on performance. The results of the experiment showed that an increase in the number of hidden layers does not always guarantee an improvement in performance, and that the model design needs to be optimized to consider the characteristics of the data and the complexity of the problem.

In future studies, we will further observe this study and explore the optimal model design methodology using more diverse datasets and model architectures.

Reference

Pu S., Olshevsky A., Paschalidis I. Ch., Asymptotic Network Independence in Distributed Stochastic Optimization for Machine Learning: Examining Distributed and Centralized Stochastic Gradient Descent, pp.114 – 122, *IEEE signal processing magazine*, 2020

Zavala M., Luis A., Lamichhane B., Zhang L., Haan G., CNN-SkelPose: a CNN-based skeleton estimation algorithm for clinical applications, *Journal of Ambient Intelligence and Humanized Computing*, pp.2369 – 2380, 2020

Chin T. L., Chao H. H., Shi A. C., CNN-Based Hybrid-Order Texture Segregation as Early Vision Processing and Its Implementation on CNN-UM, pp.2277 – 2287, *IEEE*, 2007

Vladimirov N., Brui E., Levchuk A., Al H., Walid F., Vladimir E., Aleksandr B., David, CNN-based fully automatic wrist cartilage volume quantification in MR images: A comparative analysis between different CNN architectures, pp.737 – 751, *Magnetic resonance in medicine: official journal of the Society of Magnetic Resonance in Medicine*, 2023

Yann L.C., Corinna C., Christopher J.C., The MNIST DATABASE of handwritten digits

Yann L.C., Leon B., Yoshua B., and Patrick H., Gradient-Based Learning Applied to Document Reconition, 1988.

Medium.Degining Your Neural Networks, https://towardsdatascience.com/designing-your-neural-networks-a5e4617027ed

TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. https://www.tensorflow.org/tutorials/quickstart/beginner?hl=ko

Taggle.Training a Neural Network? Start here!, https://www.kaggle.com/code/lavanyashukla01/training-a-neural-network-start-here/notebook

John Hunter and 4 others.(2017).Matplotlib Release 2.0.2, https://matplotlib.org/2.0.2/Matplotlib.pdf

Haijian S., Intelligent Automation & Soft Computing 36(3):3595. 2023.

MNIST Handwritten Digit Classification Based on Convolutional Neural Network with Hyperparameter Optimization Drishti B., Akhtar R., Handwritten Digit Recognition of MNIST dataset using Deep Learning state-of-the-art Artificial Neural Network (ANN) and Convolutional Neural Network (CNN), *IEEE.* 2021.

Fatchul A.,Variations in the Number of Layers and the Number of Neurons in Artificial Neural Networks: Case Study of Pattern Recognition, *Journal of Physics Conference Series* 1413(1):012016, 2019.

Biographies

Youju Rim is student in MY PAUL SCHOOL. He is interested in artificial intelligence, deep learning, cryptography, robots, autonomous vehicles, etc., and is conducting related research.

Shin Dong Ho is Professor and Teacher in MY PAUL SCHOOL. He obtained his Ph.D. in semiconductor physics in 2000. He is interested in artificial intelligence, deep learning, cryptography, robots, blockchains, drones, autonomous vehicles, mechanical engineering, the Internet of Things, metaverse, virtual reality, and space science, and is conducting related research.