

# Revision of the LeNet algorithm——Construction of LeNet deformation algorithm based on multi-conditional hyperparameter adjustment

Xiyuan Miao, Shi Zhang\*

Central University of Finance and Economics, 100081, Beijing, China  
miaoxiyuan0422@163.com, 19390035573@163.com

**Abstract:** This paper explores two main issues. First, this paper explores the optimal hyperparameters of the LeNet algorithm under the Fashion-MNIST dataset based on the grid method: where when the learning rate is 0.032, the regularization coefficient is 0.03, the momentum is 0.9, the weight decay parameter is 0.001, and the number of iterative rounds is 50, the model has the best results under the Fashion-MNIST dataset of 10% uniformly sampled samples has the relatively best results, i.e., the test accuracy converges to 85.8%. In addition, this paper improves the LeNet algorithm by constructing a LeNet deformation algorithm based on multi-conditional hyperparameter adjustment, specifically, the learning rate, momentum, and regularization coefficients change with the increase of the number of iteration rounds; in addition, in the construction of the model, the model introduces two blocks containing a convolutional layer, a batch normalization layer (BatchNorm), and a maximum pooling layer, and three linear neuron layers. After tuning, the tested accuracy of the algorithm is 91.5% under the full sample based on the Fashion-MNIST dataset.

**Keywords:** LeNet algorithm; hyperparameter; Fashion-MNIST; pooling layer; convolutional layer

## 1. Introduction

With the development of technology, image processing and image recognition have become more and more important in the modern computer vision field and have a wide impact on various fields and applications [1]. Image recognition and processing are widely used in face recognition, object detection and recognition, image classification, medical image analysis, unmanned vehicles, robot navigation, augmented reality, and other fields [2]. On the other hand, the processing and analysis of large amounts of image data can extract valuable information and patterns [3]. This information can be used in data-driven decision making, market research, business intelligence, and other fields to help companies and organizations make more accurate and targeted decisions [4]. Image processing and image recognition are of importance in various fields, and they provide effective tools for understanding and analyzing image information, driving scientific research, technological innovation, and social progress [5].

Among many image processing algorithms, LeNet is a classic algorithm based on convolutional neural networks. This algorithm is widely used in image recognition tasks, especially handwritten digit recognition. It was proposed by LeCun et al. in 1998 [6]. This algorithm was the first convolutional neural network that was successfully applied to a real task and introduced the concept of convolutional layers as well as pooling layers. And as the demand for image recognition is increasing, LeNet has seen improvements one after another.

LeNet-5 improved version is the earliest LeNet improved version, and the innovation of this improved algorithm is to replace the maximum pooling layer with the average pooling layer, which allows the neural network to learn and extract features efficiently and improve the learning effect. In order to explore the robustness of the LeNet algorithm, regularization as well as the tentative retirement method were also added in the subsequent improved versions of LeNet [7]. In addition, the LeNet algorithm has more variants in terms of structure; the LeNet-300-100 algorithm is a model based entirely on linear neural networks and is mainly used in applications such as recognition of handwritten digits; the LeNet++ algorithm extends and improves the network structure of LeNet. This algorithm has deeper layers and also adds more convolutional and fully connected layers; LeNet Conv4 introduces an additional convolutional layer and has a slight increase in parameter two and complexity compared to LeNet [8,9].

For the numerous LeNet improvement algorithms among which the target datasets as well as the

application scenarios are different, it is difficult to analyze these deformation algorithms. Therefore, this paper explores the prediction accuracy of LeNet algorithms and LeNet deformation-based algorithms based on the Fashion-MNIST dataset, respectively. In addition, based on the deformation of the existing LeNet algorithm, this paper innovatively constructs a LeNet model based on multi-conditional hyperparameter adjustment. After debugging, we found that under the average selection of 10% of the Fashion-MNIST dataset (i.e., 10% of the samples of each of the 10 categories, i.e., each category contains 600 training sets and 100 test sets, and a total of 6000 training sets and 1000 test set samples are taken from the 10 categories, and the training sets and test sets are selected as subsets of the original dataset. The test accuracy of the original LeNet algorithm tends to be 85.8%; while for the many LeNet deformation algorithms, we select LeNet-300-100, LeNet++ algorithm, etc. We find that the test accuracy of these algorithms is close to 85.8%. We found that the test accuracies of these algorithms were all 85% and 84.5%. We analyzed the defects of these algorithms and based on the defects of the above model algorithms, we constructed the LeNet algorithm based on multi-conditional hyperparameter adjustment, i.e., for the hyperparameter learning rate, regularization coefficient, and momentum size dynamically adjusted with the number of cycles; and for the model structure itself, compared with the LeNet algorithm, I added blocks containing convolutional layer, batch normalization, maximum convergence layer, and linear neuron layer. The test accuracy of this algorithm is close to 88.5% with 10% of the data set, which is a good improvement compared to LeNet and LeNet deformation algorithms, while the test accuracy of the model tends to 91.5% with the full sample.

This paper is structured as follows, Section II describes the data set situation of this paper; Section III describes the first experiment of this paper, i.e., for the introduction and debugging of the LeNet model; Section IV describes the second experiment of this paper, i.e., for the construction and analysis of the LeNet deformation model and for the introduction of the multi-conditional hyperparameter-based - LeNet model; Section IV concludes.

## 2. Experiment 1 - Predictive analysis based on the original LeNet model

### 2.1 LeNet model

LeNet is a classical convolutional neural network, proposed by Yann LeCun et al. in 1998. It was the first deep learning model to be successfully applied to the task of handwritten digit recognition.

The network structure of LeNet consists of a convolutional layer, a pooling layer, and a fully connected layer. The convolutional layer extracts the features of the input image by convolutional operations, uses multiple convolutional kernels for feature mapping, and performs nonlinear transformations using an activation function, commonly known as the Sigmoid function. The pooling layer reduces the size of the feature mapping by averaging the pooling operations to reduce the number of network parameters and computation. The convolution and pooling layers are alternated to efficiently extract local and global features of the image. The conceptual diagram is shown in Figure 1:

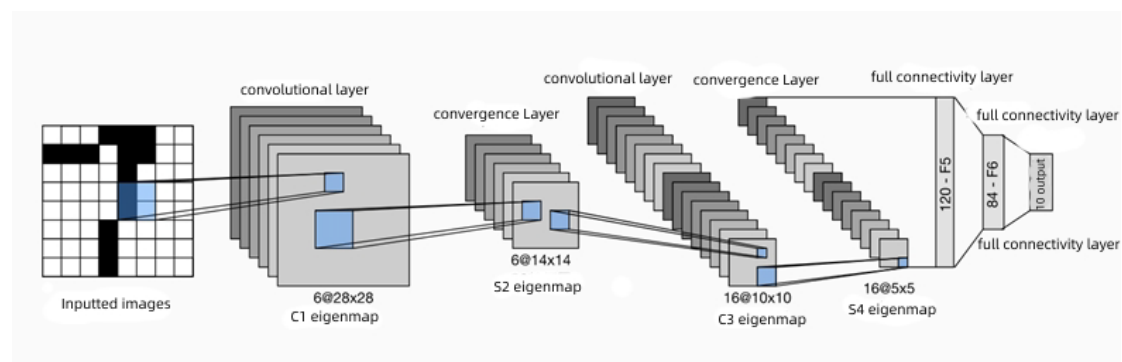


Figure 1: LeNet structure

The LeNet model also includes a fully connected layer that spreads the output of the pooling layer into a one-dimensional vector and integrates features into the final classification result through multiple fully connected layers and activation functions. The LeNet model is designed with the idea of shared weights and parameter sharing in order to reduce the number of parameters in the network and improve the efficiency of the model.

Although LeNet has achieved better performance and accuracy in handwritten digit recognition tasks,

it has a simpler structure compared to modern deep learning models because it is a model proposed in an earlier period. And this drawback has been continuously weakened in subsequent improvements of the LeNet model.

## **2.2 Optimization of the original model - based on the grid method**

In the field of machine learning and deep learning, the grid method is widely used to select the best combination of hyperparameters. Hyperparameters including learning rate, batch size, regularization parameters, etc. play an important role in deep learning. The selection of these hyperparameters has an important impact on the performance and generalization ability of the model.

The grid method traverses the given hyperparameter space by exhaustive search, tries all possible combinations, and then evaluates the performance of the model under each set of hyperparameters based on predefined evaluation metrics. Finally, the combination of hyperparameters that makes the evaluation metrics optimal is selected as the best hyperparameters.

The following are the general steps to obtain the optimal hyperparameters using the grid method:

Step 1: Define hyperparameter space: Determine the hyperparameters to be adjusted and their value ranges.

Step 2: Create hyperparameter combinations: Based on the hyperparameter space, all possible hyperparameter combinations are generated. This can be achieved by a grid search algorithm, which permutes the possible values of the hyperparameters.

Step 3: Build the model: Based on each set of hyperparameter combinations, the model is built on the training set and the performance of the model is evaluated on the validation set. Cross-validation can be used to evaluate the model more accurately.

Step 4: Evaluate the model performance: Evaluate the model performance on the validation set based on predefined evaluation metrics, such as accuracy, F1 score, etc.

Step 5: Select the best hyperparameters: Based on the evaluation results, the combination of hyperparameters that makes the evaluation metrics optimal is selected as the best hyperparameters. This is usually a matter of finding the maximum or minimum value, such as finding the maximum accuracy or the minimum mean square error.

Step 6: Evaluation on test set: The model with the best hyperparameters is finally evaluated on an independent test set to assess the performance of the model on real data.

## **2.3 Analysis of model results**

### **2.3.1 Sample situation**

In training the LeNet model, I load the Fashion-MNIST dataset and select a subset of the original dataset for the training and test sets. 10% of the samples are taken from each of the 10 categories, i.e., each category contains 600 training sets and 100 test sets, for a total of 6000 training sets and 1000 test sets for the 10 categories.

### **2.3.2 Model analysis**

Based on the basic structure of LeNet, we train the number of iterations, learning rate, regularization coefficients, momentum, and weights of LeNet separately to construct the parameter space. To determine whether there is overfitting or underfitting of the model, I make judgments based on the training accuracy. The prerequisite assumptions are as follows:

*Prerequisite assumption 1: If the training accuracy exceeds 95% and does not exceed 97%, then the model is in a state of neither overfitting nor underfitting.*

After initial training, we set the hyperparameter space. We compare the maximum test accuracy of each parameter space. And select the top 20% of the maximum test accuracy (sorted from largest to smallest) hyperparameter space as the potentially optimal hyperparameters. Based on these potentially optimal hyperparameters, we perform further refinement in the The hyperparameter space and the corresponding maximum test accuracy are shown in Table 1:

Table 1: Initially determined hyperparameter space

Maximum Test Accuracy	Hyperparameter space (learning rate, regularization factor, momentum)
84.5	(0.02, 0.02, 0.7)
85	(0.02, 0.02, 0.8)
86.6	(0.02, 0.02, 0.9)
83.7	(0.02, 0.03, 0.7)
85.9	(0.02, 0.03, 0.8)
86.9	(0.02, 0.03, 0.9)
85.8	(0.02, 0.04, 0.7)
84.9	(0.02, 0.04, 0.8)
85.9	(0.02, 0.04, 0.9)
85.6	(0.03, 0.02, 0.7)
87.2	(0.03, 0.02, 0.8)
84.6	(0.03, 0.02, 0.9)
85.1	(0.03, 0.03, 0.7)
85.7	(0.03, 0.03, 0.8)
86.1	(0.03, 0.03, 0.9)
84.6	(0.03, 0.04, 0.7)
85.1	(0.03, 0.04, 0.8)
87	(0.03, 0.04, 0.9)
85.5	(0.04, 0.02, 0.7)
86.9	(0.04, 0.02, 0.8)
85.3	(0.04, 0.02, 0.9)
84.5	(0.04, 0.03, 0.7)
85.7	(0.04, 0.03, 0.8)
86.8	(0.04, 0.03, 0.9)
85.9	(0.04, 0.04, 0.7)
84.9	(0.04, 0.04, 0.8)
86.6	(0.04, 0.04, 0.9)

Ultimately, the hyperparameters are trained to obtain the following results shown in Table 2:

Table 2: Hyperparameters

Hyperparameters	Hyperparameter value
learning_rate	0.032
epoch	50
l2_lambda	0.03
momentum	0.9
weight_decay	0.001

The training and testing accuracies of the model are shown in Figure 2 and Table 3:

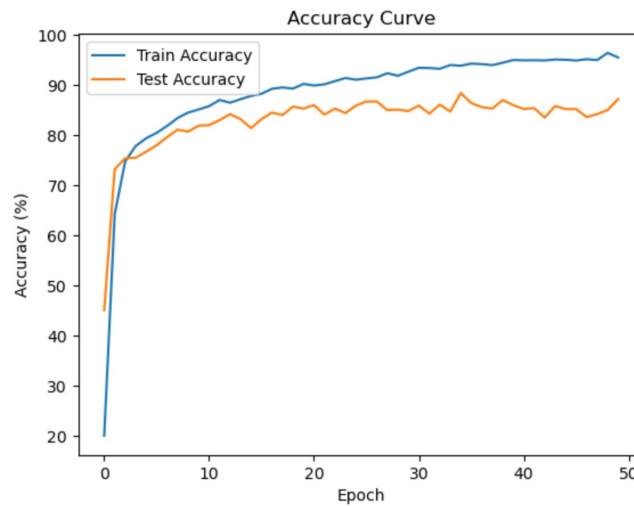


Figure 2: Training and testing accuracy curve of LeNet model

Table 3: Test accuracy of LeNet model

<i>epoch</i>	<i>Test_accuracy</i>	<i>epoch</i>	<i>Test_accuracy</i>
1	45.1	26	86.7
2	73.2	27	86.7
3	75.4	28	85
4	75.5	29	85.1
5	76.7	30	84.8
6	78	31	85.9
7	79.6	32	84.3
8	81.1	33	86.1
9	80.7	34	84.7
10	81.9	35	88.4
11	82	36	86.4
12	83	37	85.6
13	84.2	38	85.3
14	83.2	39	87
15	81.4	40	86
16	83.2	41	85.2
17	84.5	42	85.4
18	84	43	83.5
19	85.7	44	85.8
20	85.3	45	85.2
21	86	46	85.2
22	84.1	47	83.6
23	85.3	48	84.2
24	84.4	49	85
25	85.9	50	87.2

According to Table 1, the test error tends to be 85.8% when the number of iterations becomes larger. When exploring the deficiencies of the model, the experiments of hyperparameters have less negative impact on the LeNet model due to the use of the grid method for tuning the LeNet model. The structure of the LeNet model is relatively simple, and it is difficult to learn the laws of the data completely through only a small number of convolutional layers and convergence layers. Therefore, we need to adjust the model structure.

### 3. Experiment 2 - Multi-conditional hyperparameter-based - improved LeNet model for predictive analysis

Based on the LeNet model, we found that the structure of the existing LeNet model is relatively simple and needs to be enriched. The following is the introduction of the existing LeNet deformation algorithm.

#### 3.1 Introduction to the existing LeNet model

##### 3.1.1 LeNet-300-100

LeNet-300-100 is composed based on a neural network architecture for image classification tasks. It is a variant of the LeNet family of networks. LeNet-300-100 is mainly used to handle simpler image classification problems such as handwritten digit recognition.

The network structure of LeNet-300-100 is relatively simple and contains two fully connected hidden layers. Its input is a spreading image, usually grayscale, of size 28x28. The first layer of the network is a fully connected layer with 300 neurons, followed by a fully connected layer with 100 neurons. Finally, the network outputs the final classification result through a fully connected layer with 10 neurons. sigmoid activation function is used for the hidden layer of LeNet-300-100, while softmax activation function is generally used for the output layer to obtain the probability distribution of each class. The network learns the weight parameters by a back-propagation algorithm with the goal of minimizing the cross-entropy loss function. Its conceptual diagram is shown in Figure 3:

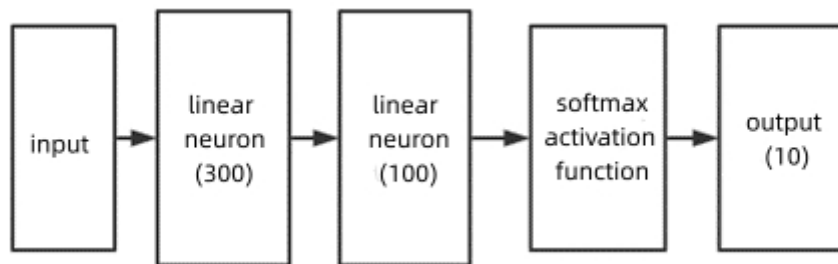


Figure 3: Model structure of LeNet-300-100

##### 3.1.2 LeNet-Conv4

LeNet-Conv4 is a deep convolutional neural network model that is based on an improved version of the LeNet model. Its conceptual diagram is shown in Figure 4:

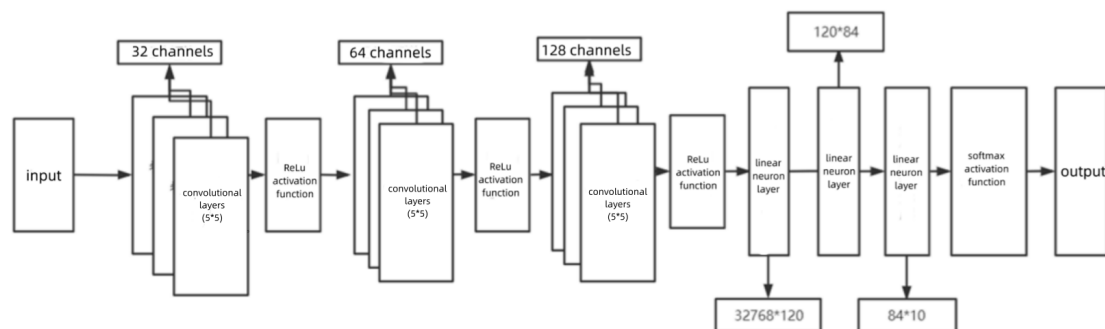


Figure 4: Model structure of LeNet-Conv4

One of the main improvements of LeNet-Conv4 is the introduction of a deeper convolutional layer structure. Compared to LeNet's two convolutional layers, LeNet-Conv4 adds an additional convolutional layer, allowing the network to learn more complex feature representations. In addition, the model introduces more convolutional kernels and pooling layers. LeNet-Conv4 uses more convolutional kernels in each convolutional and pooling layer, which increases the network's perceptual field and feature extraction capabilities. In addition, LeNet-Conv4 also uses larger size convolutional kernels. Compared with the smaller convolutional kernels in LeNet, LeNet-Conv4 uses larger convolutional kernel sizes to better handle the spatial structure in the input image. Finally, LeNet-Conv4 also introduces more fully connected layers and increases the depth of the network. Such design improvements increase the nonlinear capability and expressiveness of the model, allowing the network to learn more abstract and

advanced feature representations.

### 3.1.3 LeNet++

LeNet++ is a network model that improves and extends on LeNet. The concept diagram is displayed in Figure 5.

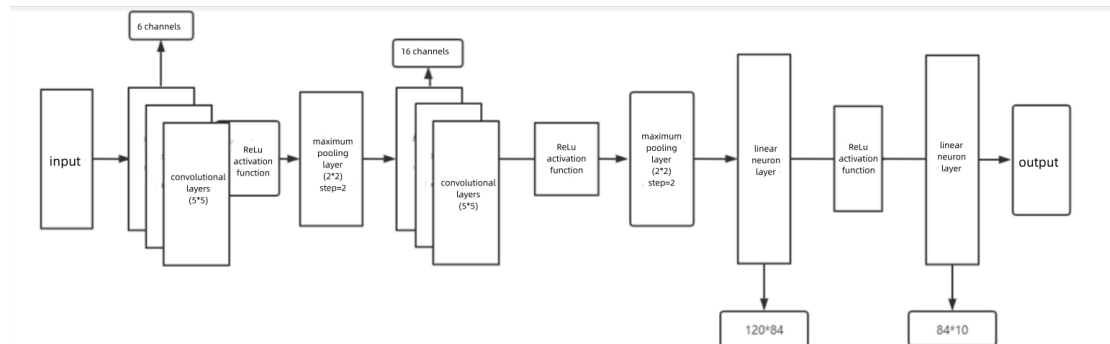


Figure 5: Model structure of LeNet++

LeNet++ extends the depth and complexity of the network by adding more convolutional layers and fully connected layers. By adding additional convolutional layers, LeNet++ can better capture abstract features in images, thereby improving the accuracy of image classification or recognition tasks. Also, increasing the number of fully-connected layers and the size of neurons allows the network to more fully learn and represent complex feature relationships.

## 3.2 Analysis of existing models

### 3.2.1 Sample situation

In the comparison of deformation algorithms, I still choose 10% of uniform classification samples as training and testing samples.

### 3.2.2 Model analysis

We debugged LeNet-300-100, LeNet++, and LeNet-Conv4 with the hyperparameters as well as the training test accuracy, the results are displayed in Table 4, Figure 6, Table 5, Figure 7, Table 6 and Figure 8:

Table 4: LeNet-300-100 prediction forecast accuracy

epoch	Test_accuracy	epoch	Test_accuracy
1	76.4	16	84.3
2	79.2	17	84.5
3	82.6	18	84.6
4	81.9	19	84.6
5	82.5	20	83.4
6	84	21	85.3
7	85.3	22	83.2
8	83	23	85.8
9	85.7	24	83.3
10	83	25	84.9
11	84.2	26	84.5
12	83.6	27	85.5
13	84.3	28	84.1
14	83.9	29	84
15	84	30	86.2

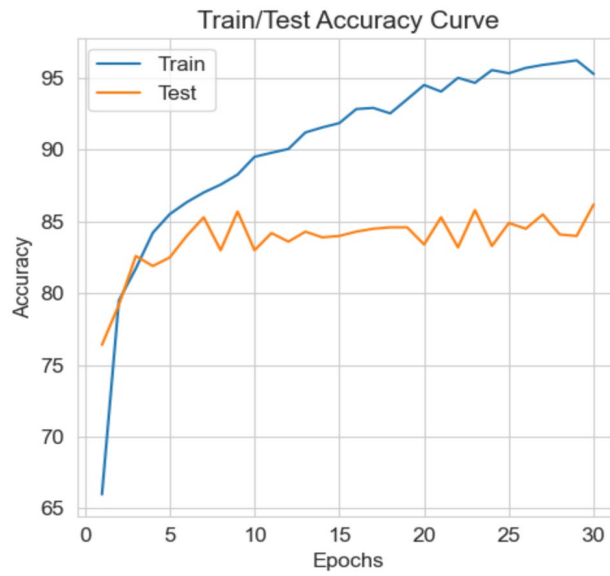


Figure 6: LeNet-300-100 training and testing accuracy

Table 5: LeNet++ prediction forecast accuracy

epoch	Test_accuracy	epoch	Test_accuracy
1	51	16	84.1
2	68.8	17	85.4
3	74.1	18	83.1
4	78.5	19	83.7
5	78.4	20	84.5
6	80.3	21	85.1
7	81.5	22	84.1
8	83.9	23	83.2
9	82.9	24	85.5
10	83	25	84.6
11	84.5	26	84.7
12	84.6	27	83.8
13	83	28	82
14	84.4	29	84.7
15	83.7	30	83.6

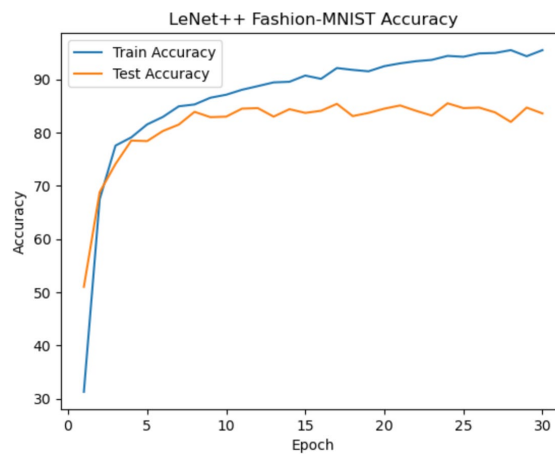


Figure 7: LeNet++ training, testing accuracy



Table 6: LeNet-Conv4 prediction forecast accuracy

epoch	Test_accuracy	epoch	Test_accuracy
1	71	16	32.1
2	64.8	17	33.6
3	65.6	18	32.5
4	67.2	19	52.7
5	64	20	57.4
6	68.3	21	22.3
7	59.3	22	19.2
8	55.2	23	17.4
9	68.6	24	17.4
10	59.9	25	17.4
11	61.9	26	18.5
12	68.4	27	18.4
13	56.9	28	18.4
14	64.4	29	18.4
15	56.1	30	18.4

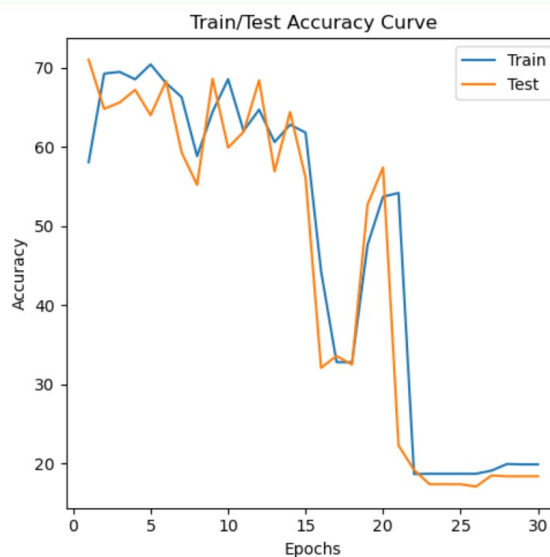


Figure 8: LeNet-Conv4 training, testing accuracy

For the LeNet-300-100 model two linear neuron layers were chosen and their test accuracy was close to 85%. This result indicates that the linear neuron layer has a reference value in training the Fashion-MNIST dataset, but the accuracy is not high because the model is too simple. For the LeNet-Conv4 model, I believe that the model's mechanism is not applicable to the Fashion-MNIST dataset.

The existing improved LeNet models other than the LeNet-300-100 model have a more complex structure, but these models are not efficient in learning. According to the variation of the accuracy of the model test set, the accuracy variation becomes smaller and smaller as the number of iterations becomes larger. The accuracy variation receives a variety of factors, and sufficient learning will cause the accuracy to converge to a fixed value. In addition, a single hyperparameter can also lead to accuracy variation. Because as the model learns more and more adequately, the learning rate and regularization coefficients of the model need to change as the number of iterations increases in order to continue learning the information implied by the data, thus making the model learning more efficient and effective.

**3.3 Construction of LeNet deformation algorithm based on multi-condition hyperparameter adjustment**

Based on the above model analysis, LeNet deformation algorithm has different problems. And we modify the LeNet model based on the above problems. The hyperparameters of the improved model are shown in Table 7.

Table 7: Hyperparameters of the model design and their interpretations

Hyperparameters	Meaning
learning_rate	Learning Rate
l2_lambda	Regularization factor
epoch	Number of iterations
momentum	Momentum
weight_decay	Weight decay
batch_size	Number of samples per iteration of training

In terms of hyperparameter setting, since the extent to which the model improves by learning and thus improving the model in different iteration rounds varies, we adjust the hyperparameters according to the iteration rounds condition, as defined below.

$$hyper\ parameters = \begin{cases} a(epoch \in (a_i, a_j)) \\ b(epoch \in (a_j, a_k)) \\ \dots \dots \end{cases} \quad (1)$$

We choose the learning rate, the regularization factor, and the momentum as tunable hyperparameters. The number of iterations as well as the weight decay coefficients do not change.

In terms of model construction, unlike the LeNet algorithm, I construct two two blocks. These two blocks contain a convolutional layer, a batch normalization layer (BatchNorm), and a maximum pooling layer, respectively. These two blocks are adjacent to each other, and BatchNorm can speed up the training, improve the gradient propagation, and enhance the generalization ability of the model. And after the two blocks, we introduce four linear neuron layers. The first three linear neuron layers are connected to the activation function after the first three linear neuron layers. The conceptual diagram is displayed in Figure 9:

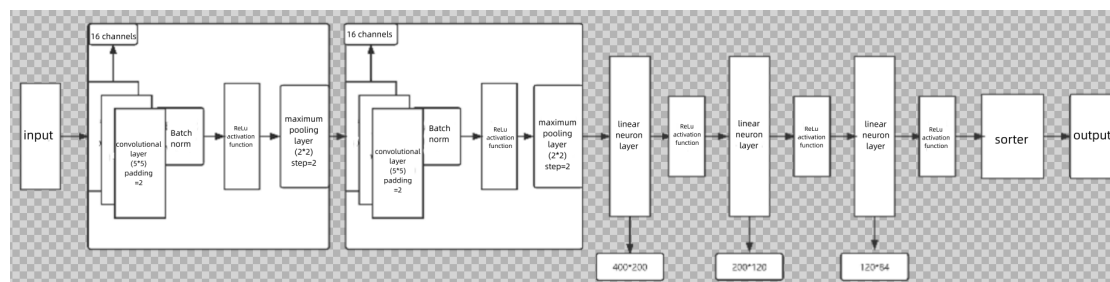


Figure 9: Multi-conditional hyperparameter-based - improved LeNet model structure

**3.4 Analysis based on multi-conditional hyperparameters - improved LeNet model**

**3.4.1 Sample situation**

In this model, we select 10% uniform classification samples as well as full samples as training and testing samples, respectively.

**3.4.2 Model analysis**

According to the above model, we gave 10% of the equal classification samples as well as the full

sample, respectively. We trained the hyperparameters. Their specific values are shown in Table 8:

Table 8: Hyperparameters

Hyperparameters	Numerical value
epoch	30
weight_decay	0.001
batch_size	32

For the learning rate, regularization factor, and momentum of the model, they are taken as shown in the following equations:

$$learning\_rate = \begin{cases} 0.05(epoch \in [0,11)) \\ 0.01(epoch \in [11,16)) \\ 0.005(epoch \in [16,30]) \end{cases} \quad (2)$$

$$l2\_lambda = \begin{cases} 0.03(epoch \in [0,11)) \\ 0.09(epoch \in [11,16)) \\ 0.12(epoch \in [16,30]) \end{cases} \quad (3)$$

$$momentum = \begin{cases} 0.7(epoch \in [0,11)) \\ 0.4(epoch \in [11,16)) \\ 0.1(epoch \in [16,30]) \end{cases} \quad (4)$$

According to the above hyperparameters, the model training effect is shown in Figure 10 and Figure 11:



Figure 10: Training, testing accuracy curve of LeNet deformation algorithm based on multi-conditional hyperparameter tuning (10% sample)

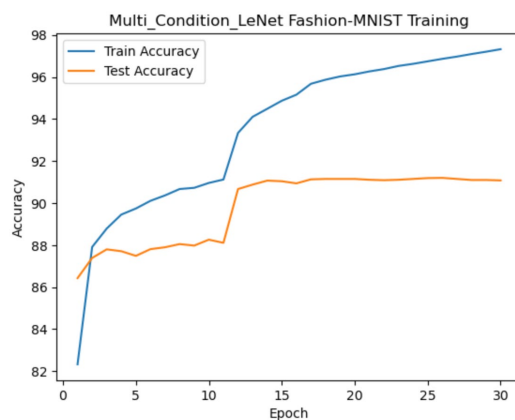


Figure 11: Training and testing accuracy curves of LeNet deformation algorithm based on multi-condition hyperparameter adjustment (full sample)

The test accuracies are displayed in Table 9 and Table 10:

*Table 9: Accuracy of the test set at 10%*

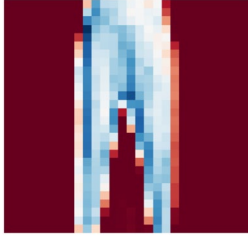
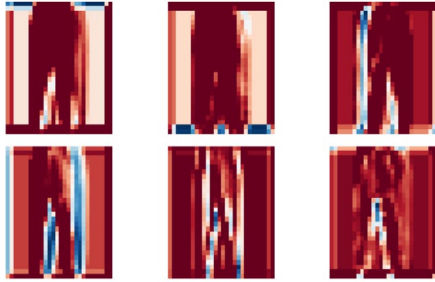
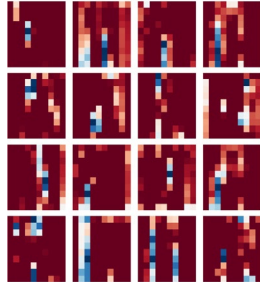

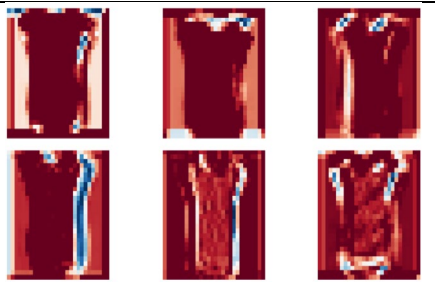
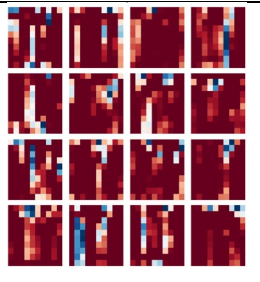

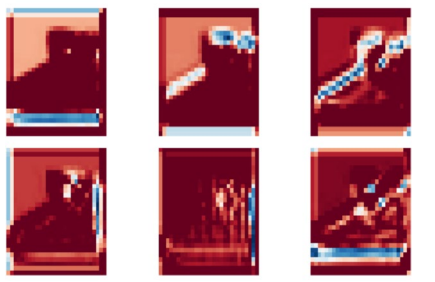
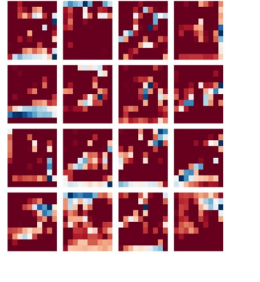
<i>epoch</i>	<i>Test_accuracy</i>	<i>epoch</i>	<i>Test_accuracy</i>
1	73.1	16	87.4
2	79.5	17	87.6
3	82.1	18	87.8
4	83	19	87.6
5	82.1	20	87.5
6	82.6	21	87.1
7	84.8	22	87.7
8	86.1	23	87.2
9	85	24	87.2
10	85.6	25	87.3
11	86.1	26	87.1
12	87	27	87.2
13	87.5	28	87.3
14	87.3	29	87.5
15	87.5	30	87.1

*Table 10: Accuracy of the test set under the full sample*

<i>epoch</i>	<i>Test_accuracy</i>	<i>epoch</i>	<i>Test_accuracy</i>
1	86.43	16	90.94
2	87.39	17	91.13
3	87.8	18	91.15
4	87.71	19	91.15
5	87.49	20	91.15
6	87.81	21	91.11
7	87.9	22	91.09
8	88.05	23	91.11
9	87.98	24	91.15
10	88.26	25	91.19
11	88.11	26	91.19
12	0.9067	27	91.15
13	90.88	28	91.1
14	91.07	29	91.1
15	91.04	30	91.08

In addition, based on a 10% uniform category sample, I output the images corresponding to the activation values separately. However, due to the large number of output channels, the images of all channels are shown in the code. The report only shows the images of activation values for some categories. As shown in Table 11:

Table 11: First and second activation layer images

Input	First activation layer image	Second activation layer image
		
Pants	Pants	Pants
		
T-shirt	T-shirt	T-shirt
		
Boots	Boots	Boots

This model adopts the LeNet model and the problem of deformation of the LeNet model. On the one hand, this model incorporates hyperparameters that are classified conditional on the number of iterations, including regularization coefficients, learning rate, and momentum size; on the other hand, this model introduces two blocks to complicate the LeNet model. Each block includes a convolutional layer, a batch normalization, an activation function, and a maximum pooling layer. After the two blocks, the model incorporates several more linear neuron layers. Based on the accuracy in the table, the modification of the model improves the learning and prediction efficiency.

In addition, to explore the role of the parameter case based on the number of iterations as a condition, we compare the original model with the classification parameters and the original model with the fixed parameters, respectively, and the comparison results are shown in Table 12.

Table 12: Accuracy of the test set at 10% (single hyperparameter)

epoch	Test_accuracy	epoch	Test_accuracy
1	73.1	16	86.9
2	79.5	17	85.8
3	82.1	18	84.2
4	83	19	85.8
5	82.1	20	84.9
6	82.6	21	84.6
7	84.8	22	86.6
8	86.1	23	85.9

9	85	24	85.1
10	85.6	25	85.7
11	86.1	26	84.7
12	84.9	27	86.4
13	84.4	28	85.3
14	85.8	29	85.5
15	86.9	30	85.7

The hyperparameters are shown in Table 13:

Table 13: Single hyperparameter case

Hyperparameters	Numerical value
epoch	30
weight_decay	0.001
batch_size	32
learning_rate	0.05
momentum	0.7

The following is a comparison of the test accuracy curves, which is shown in Figure 12:

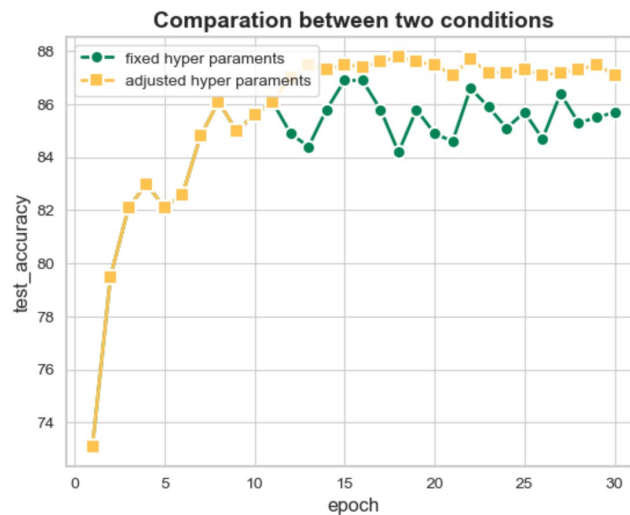


Figure 12: Curve comparison of test accuracy

I found the former to be more accurate than the latter, which also implies that the variation of parameters at different numbers of iterations is something to be concerned about.

#### 4. Conclusion

In this paper, we conduct experiments on LeNet based on the Fashion-MNIST dataset and construct a LeNet deformation algorithm based on multi-conditional hyperparameter tuning based on LeNet. Based on the original LeNet model, we tuned the parameters by grid method, and finally the test accuracy of the original model converged to 85.8%. In addition, we explored the LeNet deformation algorithm and explored the reasons for the low accuracy of the LeNet deformation model. We constructed the LeNet deformation algorithm based on multi-conditional hyperparameter adjustment. We found that the accuracy of the new model has been improved. We found that the new model has a more complex neural network structure, which is a factor that the original LeNet model does not have; in addition, we proposed that the hyperparameters should change with the number of iterations based on the change of the test accuracy curve. This also improves the accuracy of model training. Finally, the test accuracy of the model converges to 91.5% based on the Fashion-MNIST full sample data.

## References

- [1] E. R. Davies *Machine Vision: Theory, Algorithms, Practicalities [M]*. 2005.
- [2] A. R. Pathak, M. Pandey and S. Rautaray(2018) *Application of Deep Learning for Object Detection. Procedia Computer Science, 132, 1706-1717.*
- [3] Y. Zhang , E. Bieging , H. Tsui , and J. Jiang (2010) *Efficient and Effective Extraction of Vocal Fold Vibratory Patterns from High-Speed Digital Imaging. Journal of Voice Official Journal of the Voice Foundation,24(1),21-29.*
- [4] Y. Niu, L. Ying, J. Yang, M. Bao and C. B. Sivaparthipan(2021). *Organizational business intelligence and decision making using big data analytics. Information Processing & Management,58(6), 102725.*
- [5] J. Liu, H. Chang, Y.L. Forrest and B. Yang(2020) *Influence of artificial intelligence on technological innovation: Evidence from the panel data of china's manufacturing sectors." Technological Forecasting and Social Change 158, 120142.*
- [6] Y. Lecun and L. Bottou(1998) "Gradient-based learning applied to document recognition." *Proceedings of the IEEE, 86(11),2278-2324.*
- [7] S. Ghosh, R. Shet, P. Amon, A. Hutter and A. Kaup (2018), *Robustness of Deep Convolutional Neural Networks for Image Degradations, 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, 2916-2920.*
- [8] X. Dai, H. Yin, and N. K. Jha. (2020) *Incremental Learning Using a Grow-and-Prune Paradigm with Efficient Neural Networks. IEEE Transactions on Emerging Topics in Computing.*
- [9] Y. Wen, K. Zhang, Z. Li and Y. Qiao. (2016). *A Discriminative Feature Learning Approach for Deep Face Recognition. European Conference on Computer Vision Springer, Cham.*