Computing, Performance and Communication Systems (2025)
Clausius Scientific Press, Canada

DOI: 10.23977/cpcs.2025.090108
ISSN 2371-8889 Vol. 9 Num. 1

Research on Optimization of Deep Learning in Handwritten Digit Recognition

Xueju Hao

School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, China
1450946940@qq.com

Keywords: Handwritten Digit Recognition; MNIST Dataset; Multilayer Perceptron; Lightweight Convolutional Neural Network; Dropout; Hyperparameter Optimization

Abstract: Aiming at the problem of balancing model accuracy and generalization ability in handwritten digit recognition tasks, this study takes the MNIST dataset as the research object, systematically compares the recognition performance of Multilayer Perceptrons (MLP) and Lightweight Convolutional Neural Networks (CNN). It optimizes model structures by adjusting the number of network layers, neurons, and convolution kernels, introduces Dropout regularization to suppress overfitting, and analyzes the impact of hyperparameters such as learning rate and batch size on model performance. Experimental results show that the lightweight CNN, relying on its advantage in spatial feature extraction, achieves a basic model recognition accuracy of 97.2%, significantly outperforming MLP's 95.8%. After structural optimization and Dropout regularization, the test accuracy of the lightweight CNN is improved to 98.6%, and overfitting is effectively alleviated. Among hyperparameters, the learning rate has the most significant impact on model convergence speed; when the optimal learning rate is 0.001, the model can quickly reach stable accuracy. This research provides an efficient lightweight model solution for handwritten digit recognition tasks, which is of reference value for image recognition applications in low-resource scenarios.

1. Introduction

Handwritten digit recognition, as a classic task in the fields of pattern recognition and computer vision, has extensive application demands in practical scenarios such as postal mail sorting, automatic processing of financial documents, and educational examination marking. Traditional handwritten digit recognition methods rely on manually designed feature extraction operators, such as contour features and texture features, which have limitations such as weak generalization ability and sensitivity to noise, making it difficult to adapt to complex situations involving different writing styles and handwritten deformations. With the development of deep learning technology, end-to-end recognition schemes based on neural networks have significantly improved recognition accuracy and robustness by virtue of their automatic feature learning capabilities, becoming the mainstream research direction in this field.

The MNIST dataset, as a standard test set for handwritten digit recognition[1], contains a large number of labeled samples, providing a unified benchmark for model training and performance comparison. Currently, Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN) are the two most widely used model types. MLP, based on the backpropagation error learning mechanism[2], has a simple structure and efficient training but lacks the ability to extract spatial features of images. CNN captures local features and spatial correlations through convolution operations, resulting in better recognition performance, but its network complexity can easily lead to overfitting. Based on this, this paper focuses on the MNIST dataset, conducts a comparative study on the recognition performance of MLP and lightweight CNN, optimizes network structure parameters, introduces regularization techniques, and analyzes the impact of hyperparameters to construct a lightweight recognition model that balances accuracy and generalization ability, providing technical support for the engineering application of handwritten digit recognition.

2. Related Technical Foundations

2.1 Dataset Introduction

The MNIST dataset consists of 60,000 training samples and 10,000 test samples, each being a 28×28 pixel grayscale image corresponding to 10 digit categories (0-9). Images in the dataset have undergone normalization, with pixel values ranging from 0 to 255, and are center-aligned, with a black background and white digit regions. In the experiment, the data is further preprocessed: pixel values are normalized to the range [0, 1] to reduce the impact of numerical differences on model training; labels are converted using one-hot encoding to facilitate the calculation of cross-entropy loss functions.

2.2 Core Model Structures

The Multilayer Perceptron (MLP) is a fully connected neural network composed of an input layer, hidden layers, and an output layer. The input layer receives a 784-dimensional vector obtained by flattening 28×28 pixels. The hidden layers implement feature mapping through linear transformations and activation functions, and the output layer uses the Softmax activation function to output probability distributions over 10 categories. In the experiment, ReLU is selected as the activation function for hidden layers, whose nonlinear characteristics can alleviate the gradient vanishing problem and improve model training efficiency.

The Lightweight Convolutional Neural Network (CNN) takes "convolution-pooling-fully connected" as its core structure. The convolution layer extracts local spatial features by sliding 3×3 convolution kernels over the input image; the pooling layer uses max-pooling operations to reduce the dimension of feature maps while retaining key features, thereby reducing computational load; the fully connected layer maps the pooled features into a one-dimensional vector, and finally outputs classification results through the Softmax layer. The lightweight CNN reduces the number of convolution kernels and network layers to lower model complexity while ensuring recognition performance, referring to the design idea of efficient lightweight networks[3,4], making it suitable for resource-constrained scenarios.

2.3 Dropout Regularization Technology

Hyperparameters are preset parameters before model training, directly affecting model training efficiency and recognition performance[6]. This paper focuses on four key hyperparameters: the learning rate controls the step size of each parameter update, affecting model convergence speed and stability; batch size determines the number of samples in each training iteration, balancing training efficiency and memory usage; the number of epochs represents the number of training rounds over the entire dataset, needing to avoid underfitting due to insufficient iterations or overfitting due to excessive iterations; Dropout probability determines the proportion of neurons deactivated during training, being a key parameter affecting overfitting suppression.

3. Experimental Design

3.1 Experimental Environment

The experiment is implemented using Python 3.8 programming language and the TensorFlow 2.5 deep learning framework[5] for model construction and training. The hardware environment includes an Intel Core i7-10700 processor, 16GB memory, and an NVIDIA GeForce RTX 3060 graphics card, with GPU acceleration used to speed up the model training process.

3.2 Model Design and Optimization Scheme

The basic MLP model is structured as "input layer-hidden layers-output layer", with 784 neurons in the input layer, 2 initial hidden layers (128 neurons each), and 10 neurons in the output layer. For structural optimization, the number of hidden layers is adjusted to 1-3, and the number of neurons per layer is set to 64, 128, and 256 to construct MLP models of different complexities.

The basic lightweight CNN model is structured as "convolutional layer 1-pooling layer 1-convolutional layer 2-pooling layer 2-fully connected layer-output layer". Convolutional layer 1 uses 32 3×3 convolution kernels with a stride of 1 and "Same" padding; pooling layer 1 adopts 2×2 max-pooling with a stride of 2; convolutional layer 2 uses 64 3×3 convolution kernels with the same parameters as convolutional layer 1; pooling layer 2 has the same structure as pooling layer 1; the fully connected layer is set with 128 neurons; the output layer has 10 neurons. For structural optimization, the number of convolutional layers is adjusted to 2-3, the number of convolution kernels is set to 16, 32, and 64, and the number of neurons in the fully connected layer is set to 64, 128, and 256.

Both model types introduce Dropout regularization with probabilities of 0.2, 0.3, 0.4, and 0.5 to compare overfitting suppression effects under different probabilities. During training, the Adam optimizer is used to minimize the cross-entropy loss function, and the impacts of learning rates (0.0001, 0.001, 0.01), batch sizes (32, 64), and epochs (50, 80, 100) on model performance are explored based on hyperparameter optimization research[6].

3.3 Evaluation Metrics

Accuracy is used as the core evaluation metric, defined as the ratio of correctly recognized samples to the total number of samples in the test set, reflecting the overall recognition ability of the model. Meanwhile, the trends of training accuracy and test accuracy during training are recorded to analyze the degree of overfitting and convergence speed of the model.

4. Experimental Results and Analysis

4.1 Performance Comparison of Basic Models

Table 1 shows the comparison results of recognition performance between the basic MLP and lightweight CNN. It can be seen that the lightweight CNN achieves a test accuracy of 97.2%, 1.4 percentage points higher than MLP’s 95.8%, verifying the advantage of CNN in extracting spatial features of images. MLP treats images as one-dimensional vectors, losing spatial correlations between pixels and making it difficult to capture local structural features of digits. In contrast, the lightweight CNN effectively extracts local features such as edges and contours of digits through convolution operations and retains key information through pooling operations, significantly improving classification accuracy. The training accuracy of both models is higher than the test accuracy, indicating slight overfitting, which needs further optimization through regularization techniques.

Table 1 Comparison of Recognition Performance of Basic Models

Model Type	Training Accuracy (%)	Test Accuracy (%)	Training Time (min/50 epochs)
MLP	98.5	95.8	8.2
Lightweight CNN	99.1	97.2	15.6

4.2 Results of Network Structure Optimization

Table 2 shows the optimization results of the MLP structure. For 1 hidden layer, as the number of neurons increases from 64 to 256, the test accuracy rises from 94.3% to 95.5%, but the improvement slows down when the number exceeds 128. When the number of hidden layers increases to 2, the model accuracy reaches a peak of 95.8%; continuing to increase to 3 layers, the test accuracy drops to 95.2%, with a significant increase in training time. This is because excessive hidden layers increase model complexity, easily causing overfitting and redundant calculations. Therefore, the optimal structure for MLP is 2 hidden layers with 128 neurons each.

Table 2 Optimization Results of MLP Structure

Number of Hidden Layers	Number of Neurons per Layer	Training Accuracy (%)	Test Accuracy (%)
1	64	97.2	94.3
1	128	98.1	95.1
1	256	98.3	95.5
2	128	98.5	95.8
3	128	98.8	95.2

Table 3 Optimization Results of Lightweight CNN Structure

Number of Convolutional Layers	Number of Convolution Kernels (per layer)	Number of Neurons in Fully Connected Layer	Test Accuracy (%)
2	16,32	128	96.1
2	32,64	128	97.2
2	64,128	128	97.3
3	32,64,64	128	97.8
3	32,64,64	256	97.6

Table 3 shows the optimization results of the lightweight CNN structure. As the number of convolution kernels increases from 16 to 64, the test accuracy rises from 96.1% to 97.5%, but excessive kernels lead to parameter redundancy and insignificant accuracy improvement. Increasing the number of convolutional layers to 3 raises the test accuracy to 97.8%, but increases training time by 20%. The fully connected layer with 128 neurons achieves the best performance; too many or too few neurons reduce accuracy. Therefore, the optimal structure for the lightweight CNN is 3 convolutional layers (with 32, 64, and 64 kernels respectively), 2 pooling layers, and 1 fully connected layer (128 neurons), achieving a test accuracy of 97.8%.

4.3 Effects of Dropout Regularization

Table 4 shows the impact of Dropout on the performance of optimally structured models. For MLP, when the Dropout probability is 0.3, the test accuracy increases from 95.8% to 96.5%, and the training accuracy decreases from 98.5% to 97.8%, alleviating overfitting. When the probability exceeds 0.3, the test accuracy gradually decreases because excessive neuron deactivation impairs the model's learning ability. For the lightweight CNN, a Dropout probability of 0.3 achieves the best effect: the test accuracy increases from 97.8% to 98.6%, and the gap between training and test accuracy narrows to 0.5 percentage points, effectively suppressing overfitting. Thus, the optimal Dropout probability for both models is 0.3.

Table 4 Impact of Dropout on Model Performance

Model Type	Dropout Probability	Training Accuracy (%)	Test Accuracy (%)	Overfitting Degree (Training-Test, %)
MLP	0.0	98.5	95.8	2.7
MLP	0.2	96.2	96.2	1.9
MLP	0.3	96.5	96.5	1.3
MLP	0.5	96.1	96.1	0.8
Lightweight CNN	0.0	97.8	97.8	1.5
Lightweight CNN	0.2	98.2	98.2	0.8
Lightweight CNN	0.3	98.6	98.6	0.3
Lightweight CNN	0.5	98.0	98.0	0.2

4.4 Analysis of Hyperparameter Impact

Table 5 shows the impact of hyperparameters on the performance of the optimized lightweight CNN. With a learning rate of 0.001, the model achieves a test accuracy of 98.6% and the fastest convergence speed, stabilizing after 50 epochs. A learning rate of 0.01 leads to fast initial convergence but later oscillations, reducing the test accuracy to 97.9% due to excessively large steps that prevent parameters from converging to optimal values. A learning rate of 0.0001 results in extremely slow convergence, reaching 98.0% accuracy only after 80 epochs due to excessively small steps that reduce training efficiency. A batch size of 64 yields a slightly higher test accuracy than 32, with a 30% reduction in training time, as larger batches improve GPU parallel computing efficiency and reduce gradient noise. The model accuracy peaks at 80 epochs; increasing to 100 epochs shows no significant improvement and even causes slight overfitting due to over-training.

Table 5 Impact of Hyperparameters on Lightweight CNN Accuracy

Learning Rate	Batch Size	Number of Epochs	Test Accuracy (%)	Convergence Epochs
0.0001	64	80	98.0	75
0.001	64	80	98.6	45
0.01	64	80	97.9	30
0.001	32	80	98.4	50
0.001	64	50	98.2	45
0.001	64	100	98.5	45

5. Conclusion and Outlook

Based on the MNIST dataset, this paper conducts research on the optimization of deep learning in handwritten digit recognition. By comparing the recognition performance of MLP and lightweight CNN, optimizing network structure parameters, introducing Dropout regularization, and analyzing hyperparameter impacts, the following conclusions are drawn: The lightweight CNN, relying on its advantage in spatial feature extraction, significantly outperforms MLP in recognition performance; the optimal lightweight CNN structure consists of 3 convolutional layers, 2 pooling layers, and 1 fully connected layer. Dropout regularization effectively suppresses overfitting with an optimal probability of 0.3. Among hyperparameters, the learning rate has the most significant impact on model performance; the optimal combination is a learning rate of 0.001, batch size of 64, and 80 epochs, achieving a test accuracy of 98.6%.

The lightweight CNN model constructed in this study balances accuracy and efficiency, making it suitable for resource-constrained embedded devices and real-time recognition scenarios. Future research directions can be further expanded: introducing data augmentation techniques to expand training samples and improve the model's robustness to noise and deformation; combining techniques such as BatchNorm[7] to optimize the training process and accelerate model convergence; exploring more efficient lightweight network structures such as MobileNet[3] and ShuffleNet[4] to further reduce computational complexity while ensuring accuracy; applying the model to practical handwritten digit recognition tasks such as bank document processing and educational marking systems to verify its engineering practicability.

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (2002): 2278-2324.

[2] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." nature 323.6088 (1986): 533-536.

[3] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).

[4] Zhang, Xiangyu, et al. "Shufflenet: An extremely efficient convolutional neural network for mobile devices." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[5] Abadi, Mart ń, et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." arXiv preprint arXiv:1603.04467 (2016).

[6] Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." The journal of machine learning research 13.1 (2012): 281-305.

[7] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." International conference on machine learning,2015.