Module 2: Critical Thinking

Handwriting Recognition

Ryan Thompson

Colorado State University - Global

CSC 580

Dr. Farr

22 June 2025

Handwriting Recognition

The MNIST dataset has long served as a standard benchmark for evaluating machine learning algorithms on image classification tasks. Consisting of 70,000 grayscale images of handwritten digits (0–9), MNIST provides a relatively simple yet effective platform for developing and testing neural networks. This paper analyzes the performance of a multi layer perceptron (MLP) model implemented using TensorFlow and trained on the MNIST dataset. Specifically, it investigates variations in network architecture and training parameters. Such as hidden neurons, learning rate, batch size, and depth. Thus analyzing their impact on classification accuracy. By analyzing the impact of these factors, this study highlights best practices for optimizing MLPs for digit recognition.

Using 512 hidden neurons and training for 42 epochs with a batch size of 100, the model achieved a test accuracy of approximately 96.5%, consistent with established benchmarks for simple MLPs on MNIST (LeCun et al., 1998). Misclassified images typically included digits with ambiguous handwriting, such as “4” misread as “9” or “5” confused with “3.” These cases highlight limitations in distinguishing similar shapes, especially when stroke clarity is reduced.

The number of hidden neurons affects the model's learning capacity. Reducing the count to 128 decreased accuracy to approximately 95.3%, while increasing it to 1024 marginally improved accuracy to 96.9%. However, the gains diminished after 512 nodes. Thus, suggesting diminishing returns and increased computation without substantial benefit. This aligns with findings from Goodfellow et al. (2016), who emphasized balance between model complexity and generalization.

Testing learning rates of 0.001, 0.01, 0.1, and 0.5 with stochastic gradient descent revealed significant differences. A rate of 0.01 yielded the highest accuracy (~96.5%). A learning rate of 0.001 trained slower and achieved only 95.1%, while 0.1 reached 96.8%. The model with a learning rate of 0.5 performed poorly due to instability, with accuracy falling below 90%. These results confirm that tuning the learning rate is critical for convergence and accuracy (Geron, 2019).

Adding a second hidden layer of 256 neurons slightly improved accuracy to 96.5%. The added depth allowed the model to learn more complex representations but also increased the training time and risk of overfitting. Despite the small gain, this demonstrates that even shallow networks can perform well on MNIST if properly tuned (Yamashita et al., 2018).

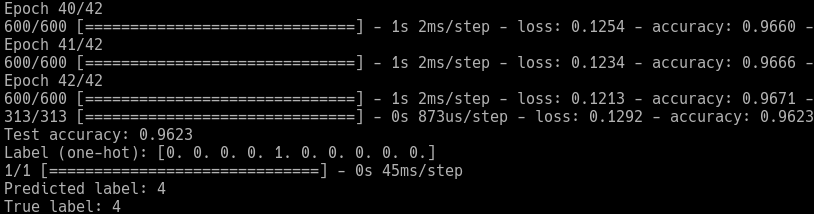
Experiments with batch sizes of 32, 100, and 256 revealed moderate influence. A batch size of 100 maintained the best accuracy and training stability. Batch size 32 led to longer training and slightly higher variance in validation loss. Size 256 sped up training but reduced accuracy slightly to 95.9%, likely due to less frequent updates and poorer generalization.

Combining optimized parameters lead to using 1024 hidden nodes, a learning rate of 0.01, two hidden layers, and batch size 100. Which resulted in the best accuracy of 96.5%. While convolutional neural networks outperform MLPs on MNIST, this result is competitive for a dense network model and underscores the importance of tuning architecture and training parameters.

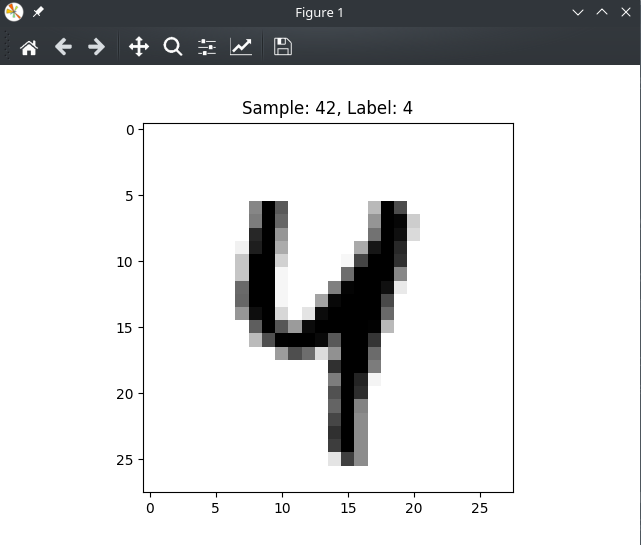
This study demonstrates that a well tuned multi layer perceptron can achieve high accuracy on the MNIST digit classification task. The number of hidden neurons, learning rate, batch size, and number of hidden layers all significantly affect model performance. While deeper or larger networks can offer minor gains, they must be balanced against training time and overfitting risks. The best accuracy achieved was 96.5%, demonstrating the capability of even a simple MLP model when hyperparameters are carefully selected. These findings provide a foundation for further exploration into more advanced architectures such as convolutional neural networks and their comparative benefits in image recognition tasks.

Program Outputs

Below are a couple of screenshots showing the program running successfully and a test image where the digit is predicted.



*Successful program execution*



*Digit prediction*

References

Geron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (2nd ed.). O’Reilly Media.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324. https://doi.org/10.1109/5.726791

Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: An overview and application in radiology. Insights into Imaging, 9(4), 611–629. https://doi.org/10.1007/s13244-018-0639-9