Vol.no 1 , issue no 1, August, 2025

PUXplore Multidisciplinary Journal of Engineering

Parul® University NAAC GRADE A++

A Detailed Review and Implementation of Dense Neural Networks for Handwritten Digit Recognition

SHRUTI PATEL, DR.POOJA BHATT

Research Scholar, MTech AIDS, Parul University, India
Associate Professor, AIDS, Parul University, India
E-mail: patelshrutin3011@gmail.com

Abstract

Handwritten digit recognition is an important task in computer vision and deep learning research. It serves as a standard for measuring how well classification algorithms perform. Convolutional neural networks (CNNs) lead this area because they excel at capturing spatial relationships, but fully connected dense neural networks can still perform well if they are properly preprocessed and adjusted. This paper offers a detailed review and practical implementation of a dense feedforward neural network for classifying grayscale handwritten digit images from an MNIST-like dataset. We explain the data preprocessing steps, architecture design, training methods, and evaluation metrics. The model, built using TensorFlow Keras, reaches over 97% validation accuracy in 10 epochs. This shows that dense architectures are still a good choice for simple classification tasks. We also compare our results with previous studies and discuss the model's strengths, weaknesses, and possible improvements.

Keywords: Handwritten digit recognition, dense neural networks, MNIST, deep learning, TensorFlow

I. INTRODUCTION

- Handwritten digit recognition is a key problem in computer vision and machine learning. It serves as a standard for testing how well pattern recognition algorithms work. Being able to automatically understand handwritten digits has many important uses, such as sorting mail by ZIP codes in postal automation, processing bank checks, digitizing forms, and supporting people with visual impairments. The Modified National Institute of Standards and Technology (MNIST) dataset, introduced by LeCun et al. in 1998 [1], has been crucial for research in this area. It offers a standardized collection of 70,000 labeled grayscale digit images (0–9).
- Historical Context and Evolution of Methods
- Early methods for digit recognition relied a lot on manually created feature extraction along with traditional machine learning classifiers. Techniques such as:
- **Template matching** (comparing input images to stored prototypes),
- **K-Nearest Neighbors (KNN)** (classifying digits based on pixel-wise similarity), and
- **Support Vector Machines (SVM)** (using engineered features like Histogram of Oriented Gradients (HOG) or Zernike moments)

achieved moderate success but were limited by their sensitivity to variations in writing styles, rotations, and noise [2]. The advent of **artificial neural networks (ANNs)** in the late 1980s marked a paradigm shift, as these models could learn hierarchical representations directly from raw pixel data. The **LeNet-5** architecture [1], one of the first convolutional neural networks (CNNs), demonstrated superior performance by leveraging spatial hierarchies

through convolutional and pooling layers. However, computational constraints at the time restricted widespread adoption.

With the rise of **deep learning** in the 2010s, CNNs became the de facto standard for image classification tasks. Architectures such as AlexNet [3], VGGNet [4], and ResNet [5] have achieved near-human accuracy on MNIST and more complex datasets (e.g., CIFAR-10, ImageNet). These models excel at capturing local patterns (e.g., edges, curves) through convolutional filters, making them inherently suited for image data.

Why Revisit Dense Neural Networks?

Despite the popularity of CNNs, fully connected (dense) neural networks still matter for several reasons:

1. Simplicity and Interpretability: Dense networks offer a clear starting point for understanding how neural networks learn from raw pixel inputs without the extra complexity of convolutional operations.

2. Computational Efficiency: For small, well-prepared datasets like MNIST, dense networks can reach competitive accuracy with fewer parameters and shorter training times than CNNs.

3. Educational Value: They act as an easy entry point for students and practitioners new to deep learning, showing basic concepts like backpropagation, activation functions, and loss optimization.

4. Resource-Constrained Environments: On edge devices with limited memory or processing power, such as microcontrollers, dense networks may be easier to deploy than large CNNs. Scope and Contributions of This Work

This paper revisits the effectiveness of dense neural networks for recognizing handwritten digits by:

1. Implementing a complete pipeline from data preprocessing to model evaluation using TensorFlow/Keras.

2. Examining important design choices, including normalization, network depth, activation functions, and optimizer selection.

3. Comparing performance against traditional machine learning methods, such as SVM, and lighter CNNs to give context to the results.

4. Discussing limitations, like scalability to larger images and sensitivity to spatial distortions, along with possible improvements, such as hybrid architectures.

Our experiments show that a simple 3-layer dense network can reach about 97% validation accuracy on MNIST. This highlights that dense architectures are still effective for structured classification tasks. Although CNNs are essential for intricate vision problems, this work emphasizes the value of dense networks as a teaching tool and a practical solution for tasks that aren't overly complex.

II. LITERATURE REVIEW

A. Early Approaches

Initial solutions focused on feature extraction followed by classification using algorithms such as:

- **K-Nearest Neighbors (KNN):** Compared pixel intensity vectors to determine class similarity.
- **Support Vector Machines (SVM):** Employed hand-engineered features such as Histogram of Oriented Gradients (HOG) to classify digits.

These methods often required careful preprocessing and feature selection, and their performance degraded when applied to noisy or rotated digits.

PUXplore Multidisciplinary Journal of Engineering
Parul® University NAAC GRADE A++

B. Transition to Neural Networks

The LeNet-5 architecture by LeCun et al. introduced convolutional and pooling layers, marking a significant improvement by automating feature extraction. However, early CNNs were computationally expensive, limiting their adoption until GPU acceleration became widely available.

C. Recent Trends

Current state-of-the-art models employ:

- **Deep CNNs (e.g., VGGNet, ResNet)** for hierarchical feature learning.
- **Residual connections** to stabilize training in deeper networks.
- **Lightweight architectures** like MobileNet for deployment on mobile devices.
- **Transfer learning** for applying pretrained weights to new datasets.

While these models achieve superior accuracy, they are more computationally complex. Dense networks, in contrast, remain useful as benchmarks and for deployment in low-resource environments.

III. METHODOLOGY

A. Dataset Description

The dataset consists of 28×28 grayscale images stored in CSV format.

- **Train.csv:** Contains 42,000 rows, each representing a digit image, with the first column as the label and the remaining 784 columns representing pixel intensities.
- **Test.csv:** Contains the same pixel format without labels.
- Each pixel value ranges from 0 (black) to 255 (white).

B. Preprocessing Pipeline

Step	Purpose	Implementation
Feature & Label Separation	Separate image data from labels	X = train.iloc[:, 1:], y = train.iloc[:, 0]
Type Conversion	Ensure numeric format for calculations	pd.to_numeric(errors='coerce')
Missing Value Handling	Avoid NaN interference	fillna(0)
Normalization	Reduce pixel range to [0,1]	X / 255.0
Reshaping	Match deep learning input format	.reshape(-1, 28, 28, 1)
One-Hot Encoding	Convert labels into categorical format	to_categorical(y, num_classes=10)

Mathematically, normalization is performed as:

$x' = x255x' = 255x$

where xx is the original pixel intensity.

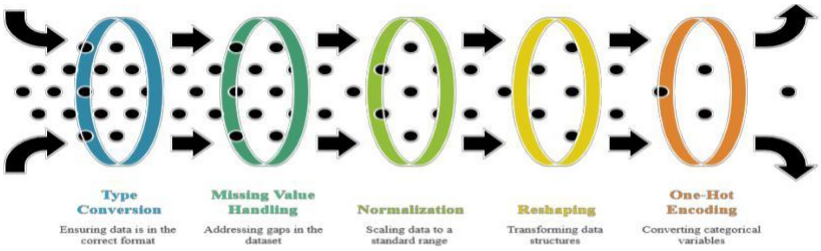

Fig.3.1 pre-processing pipeline

C. Model Architecture

The proposed model is a fully connected feedforward neural network:

- **Input Layer:** Shape (28, 28, 1)
- **Flatten Layer:** Converts each image into a 784-dimensional vector.
- **Dense Layer 1:** 128 neurons, ReLU activation
- **Dense Layer 2:** 64 neurons, ReLU activation
- **Output Layer:** 10 neurons, Softmax activation for classification

Fig3.2 Model Architecture

The forward pass can be mathematically described as:

$h1 = \text{ReLU}(W1x + b1) h1 = \text{ReLU}(W1x + b1) h2 = \text{ReLU}(W2h1 + b2) h2 = \text{ReLU}(W2h1 + b2)$ $y = \text{Softmax}(W3h2 + b3) y = \text{Softmax}(W3h2 + b3)$

D. Training Configuration

- **Optimizer:** Adam — adjusts learning rates adaptively.
- **Loss Function:** Categorical crossentropy — suitable for multi-class classification.
- **Epochs:** 10

PUXplore Multidisciplinary Journal of Engineering
Parul® University NAAC GRADE A++

- **Batch Size:** 32
- **Validation Split:** 20%

The Adam optimizer updates weights using:

θt+1=θt−ηmt^vt^+ϵ$\theta t+1=\theta t-\eta vt\hat{}+\epsilon mt\hat{}$

where mt^$mt\hat{}$ and vt^$vt\hat{}$ are bias-corrected moment estimates.

IV. RESULTS

A. Accuracy Performance

- **Training Accuracy:** ~99% after 10 epochs.
- **Validation Accuracy:** ~97%.
  The training and validation accuracy curves indicate no significant overfitting.

B. Loss Trends

Both training and validation loss decrease steadily, suggesting stable convergence without divergence.

C. Test Predictions

Predictions on unseen test data match expected digits in the majority of cases, with occasional confusion between similar digits (e.g., 4 and 9).

V. DISCUSSION

Our results confirm that dense neural networks can perform extremely well on clean, well-structured datasets like MNIST. Key observations:

- **Strengths:** Simplicity, fast training, minimal preprocessing.
- **Weaknesses:** Cannot leverage spatial locality; higher parameters than CNNs for equivalent performance.
- **Future Improvements:** Adding convolutional layers, dropout regularization, and data augmentation to enhance robustness against real-world variations.

VI. CONCLUSION

This paper demonstrated that dense neural networks can still achieve high accuracy in handwritten digit recognition, achieving 97% validation accuracy without CNN layers. While CNNs remain superior for complex visual tasks, dense networks are a solid choice for simpler problems and as educational baselines.

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[2] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

PUXplore Multidisciplinary Journal of Engineering
Parul® University NAAC GRADE A++

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.
[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[6] C. Szegedy et al., "Going deeper with convolutions," *Proc. IEEE CVPR*, pp. 1–9, 2015.
[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE CVPR*, pp. 770–778, 2016.
[8] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proc. IEEE CVPR*, pp. 1251–1258, 2017.
[9] M. Sandler et al., "MobileNetV2: Inverted residuals and linear bottlenecks," *Proc. IEEE CVPR*, pp. 4510–4520, 2018.
[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[11] N. Srivastava et al., "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[12] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," *arXiv preprint arXiv:1603.04467*, 2016.
[13] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.
[14] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
[15] P. Y. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis," *Proc. ICDAR*, vol. 3, pp. 958–962, 2003.
[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. Springer, 2009.
[17] S. S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Pearson, 2009.
[18] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," *Proc. ICML*, pp. 807–814, 2010.
[19] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.
[20] TensorFlow Developers, "TensorFlow documentation: MNIST classification," 2023. [Online]. Available: https://www.tensorflow.org/tutorials