PHYSICSAccess Research Paper
ISSN Online: 2756-3898, ISSN Print: 2714-500X https://doi.org/10.47514/phyaccess.2026.6.1.010

Comparative Analysis of Classical Machine Learning and Deep Learning Architectures for Handwritten Digit Classification

Tamakloe A Vivian[1], Maikusa S Abdulrahman[1], and Eli A Jiya[1]

[1] Department of Computer Science, Federal University Dutsin-Ma, Katsina State, Nigeria

Corresponding E-mail: jiyaeli@fudutsinma.edu.ng

Received 12-03-2026
Accepted for publication 17-04-2026
Published 21-04-2026

Abstract

Handwritten digit recognition remains a core problem in computer vision, with both classical machine learning (ML) and deep learning (DL) models widely applied but rarely compared under identical conditions. This study presents a controlled evaluation of Logistic Regression, Support Vector Machine (SVM), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and a hybrid CNN–LSTM using the MNIST dataset. Data were normalized, reshaped, and encoded within a reproducible Python pipeline executed in a CPU-only environment. Classical models used flattened inputs, while DL models leveraged spatial structure. Results show a clear progression in performance: Logistic Regression (92.00%), SVM (97.59%), LSTM (98.53%), and CNN/CNN–LSTM (98.96%). CNN achieved the best overall metrics (precision 0.9897, recall 0.9896, F1-score 0.9896, AUC 0.999926) with 225,034 parameters, a training time of 209.77 seconds, and an inference time of 2.32 milliseconds. Statistical analysis confirmed CNN's superiority. The findings demonstrate that architectures aligned with data structure, particularly CNNs exploiting spatial locality, deliver optimal performance. While classical models remain competitive, their reliance on flattened representations limits effectiveness. The near-ceiling MNIST results further suggest the need for more challenging benchmarks to distinguish advanced models.

Keywords: Machine Learning; Image Classification; Spatial Images; Handwritten Digit Recognition; MNIST dataset.

I. Introduction

Handwritten digit recognition remains a fundamental benchmark problem in computer vision and machine learning, with practical applications in postal mail sorting, bank cheque verification, document digitization, and automated data entry systems [1]. The MNIST dataset has historically served as a benchmark for evaluating classification algorithms due to its standardized format, grayscale normalization, and balanced distribution across ten-digit classes [2]. Its simplicity and reproducibility have made it an essential dataset for validating both traditional and modern learning algorithms [3].

Early approaches to handwritten digit classification relied on Classical Machine Learning (ML) algorithms such as Logistic Regression (LR) and SVM, which have shown strong performance in structured classification problems [4]. However, these approaches require flattening image data into one-dimensional vectors, which removes spatial relationships among pixels [5]. As a result, they cannot automatically learn hierarchical spatial features, limiting their robustness and scalability for complex image tasks [6].

Deep Learning (DL) architectures, particularly CNNs, address these limitations by preserving spatial structure and

learning hierarchical representations directly from raw images, leading to excellent performance and improved generalization. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks have also been adapted for image classification by modelling images as sequential data [7]. Hybrid CNN-LSTM architectures combine spatial and sequential feature extraction to enhance performance.

Previous studies have explored various deep learning architectures for handwritten digit recognition using both archival and benchmark datasets. A comparative evaluation of CNN, LSTM, and a hybrid CNN–LSTM on an archival digit dataset derived from NOAA weather logs (approximately 26,000 images), alongside open-source handwritten digits, showed that all models achieved high recognition accuracy; however, the hybrid CNN–LSTM was both faster and marginally more accurate than the standalone CNN, indicating that combining spatial and sequential modeling can yield incremental performance gains [8].

In another study using the MNIST dataset, CNN, ResNet, DenseNet, and a capsule-enhanced CNN were compared. The integration of a Capsule Network (CapsNet) module into the CNN baseline produced the highest accuracy of 99.75%, outperforming the other architectures while requiring fewer training samples to achieve strong performance, thereby demonstrating the advantage of capsule structures in capturing pose information [9].

Similarly, a hybrid CNN–Vision Transformer model evaluated on MNIST combined convolutional layers with a transformer-based attention mechanism. Tested on both clean and noisy digit images, the hybrid model achieved superior recognition accuracy compared to standalone CNN and Vision Transformer baselines, highlighting the robustness of combining spatial feature extraction with attention mechanisms [10].

Although these studies consistently demonstrate that deep learning models outperform classical machine learning approaches in image classification tasks, there remains limited consensus on the practical trade-offs among accuracy, computational complexity, scalability, and generalization when models are evaluated under identical experimental conditions. The lack of structured and reproducible comparative analyses constrains evidence-based model selection, particularly for real-world deployment in resource-constrained environments [2].

Rather than proposing a new architecture, this study presents a controlled, reproducible, and statistically grounded comparison of classical (LR, SVM) and deep learning models (CNN, LSTM, and hybrid CNN–LSTM) under identical conditions, using MNIST as a standardized benchmark despite its near-ceiling performance. While previous studies [8-10] have compared these approaches on handwritten digit classification, most focus primarily on accuracy metrics, with few offering an integrated evaluation that combines accuracy, ROC–AUC, statistical significance testing, and model complexity analysis under uniform experimental conditions. This study addresses this gap by proposing a multi-dimensional comparative framework, incorporating statistical validation and architectural analysis to provide a clearer understanding of performance complexity trade-offs often neglected in previous studies.

II. Methods

This study adopts a controlled experimental framework comprising data preprocessing, model development, and comparative evaluation. Fig. 1 visualizes the method's proposed system architecture.

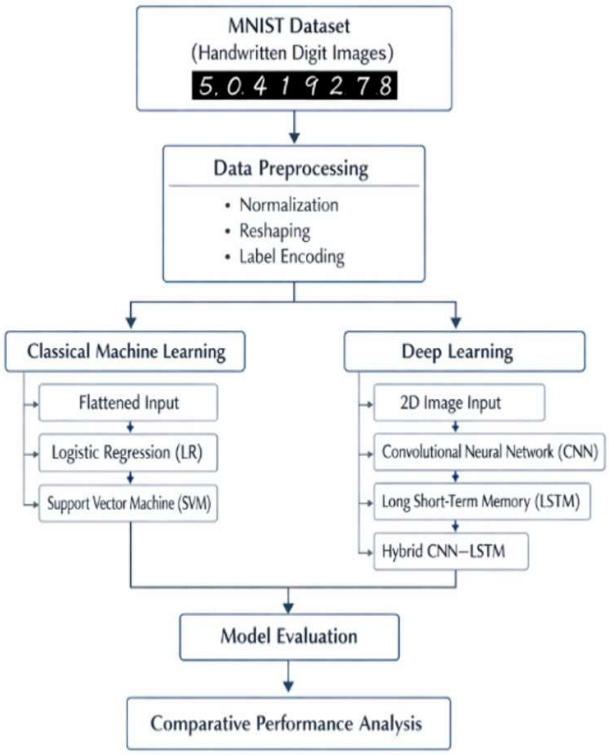

Fig. 1. Proposed Handwritten Digit Classification System Architecture.

A. Data Source

The study relied exclusively on secondary data obtained from the publicly available MNIST repository. As previously stated, the use of MNIST in this study is not for benchmarking accuracy improvements, but for enabling controlled, reproducible, and statistically grounded comparisons across heterogeneous model architectures. The dataset consists of 70,000 grayscale images; each image has 784 features when flattened (28 × 28); pixel values range from 0 to 255, and multi-class labels range from 0 to 9. The dataset is balanced, with each class containing approximately equal observations, making it suitable for macro-average performance evaluation. Fig. 2 shows a sample image of the digits in the MNIST dataset, and Fig. 3 shows the class distribution across the 10 digits in the MNIST dataset.

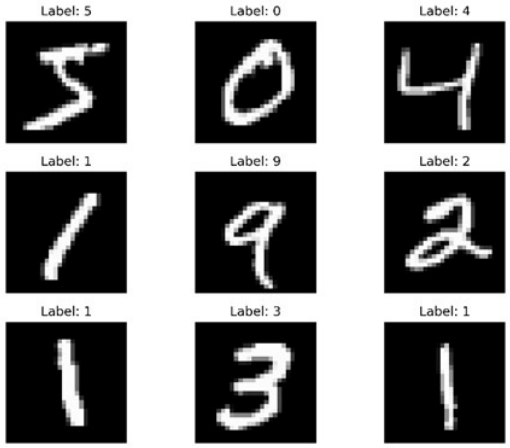

Fig. 2. Sample Images of MNIST Dataset.

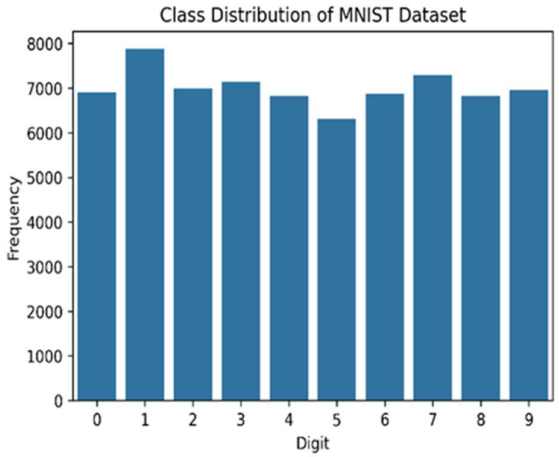

Fig. 3. Class Distribution of MNIST Dataset.

B. Data Preparation

Preprocessing steps applied include: Normalization: Pixel values scaled to the range (0,1) by dividing by 255 to improve convergence during training. Reshaping: Classical ML models used flattened vectors of size 784; CNN models used 2D image tensors of shape (28, 28, 1); LSTM models treated image rows as sequential inputs. Label Encoding: For multi-class ROC analysis, class labels were converted into one-hot encoded format using label binarization. Train-Test Split: The predefined dataset split was maintained to ensure comparability with existing literature [2], [4], [6].

C. Model Development Environment

The experiment was conducted using the Python Programming Language and Jupyter Notebook via Anaconda Environment. The following libraries were used: Scikit-learn (for classical ML models and metrics), TensorFlow/Keras (for DL models), NumPy and Pandas (for data manipulation), and Matplotlib for visualization, and executed in a Central Processing Unit (CPU) only computational environment, with no access to Graphics Processing Unit (GPU) acceleration. This implies that all model training and inference processes were executed using standard central processing resources. While the absence of GPU acceleration does not affect the correctness or predictive performance of the models, it has significant implications for computational efficiency. CNN, LSTM, and hybrid CNN-LSTM architectures typically benefit from parallel processing capabilities offered by GPUs, especially during backpropagation and matrix operations. Consequently, the reported training times in this study may be higher than those observed in GPU-enabled environments. However, the use of a CPU-only setup provides a realistic evaluation scenario for deployment in resource-constrained environments, where specialized hardware may not be available. Therefore, the computational results presented in this study reflect practical performance considerations relevant to real-world applications.

D. Model Development Algorithms

The five machine learning algorithms developed and implemented for classification include: Logistic Regression (LR), Support Vector Machine (SVM, RBF), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and CNN-LSTM Hybrid. LR is a statistical classification model used to predict a discrete outcome (often binary) from a set of input features. In multiclass settings (e.g., 10-digit classification), LR is extended using either a one-vs-rest (OvR) strategy or the SoftMax function in multinomial LR to produce class probabilities [11]. LR is widely employed as a baseline classifier due to its computational efficiency, convex optimization objective, and well-established statistical properties [12]. SVM is a large-margin classifier that seeks a hyperplane separating data classes with maximum margin [12]. SVMs can incorporate kernel functions (Radial Basis Function (RBF or Gaussian), polynomial, and sigmoid kernels) to construct nonlinear decision boundaries. For image classification tasks such as MNIST, linear SVMs often achieve strong performance when combined with suitable preprocessing, normalization, or dimensionality reduction techniques [13].

CNN architecture is specifically designed for processing grid-structured data such as images [13]. Its architecture included: multiple Convolutional layers for feature extraction; ReLU activation; Max pooling for reducing the spatial dimensionality of feature maps; Fully connected dense layers and SoftMax output layer for the final classification. CNNs automatically learn hierarchical and translation-invariant feature representations from raw pixel data, eliminating the need for manual feature engineering. This automatic feature learning capability has led to significant performance improvements in image classification tasks [5].

In this study, the CNN architecture starts with an input layer that accepts a 28×28 grayscale image of a handwritten digit, producing an output shape of 28×28×1. Followed by a convolutional layer with 32 filters of size 3×3 and ReLU activation, which extracts low-level spatial features and

outputs a feature map of size 26×26×32. A max pooling layer with a 2×2 pool size then reduces the spatial dimensions to 13×13×32. A second convolutional layer with 64 filters of size 3×3 and ReLU activation extracts deeper image features, yielding an output shape of 11×11×64. Another 2×2 max pooling layer further reduces the dimensions to 5×5×64. The flatten layer converts these feature maps into a vector of length 1600, preparing the data for the dense layers. A dense layer with 128 neurons and ReLU activation learns high-level representations, producing an output of size 128. To prevent overfitting, a dropout layer with a rate of 0.5 is applied, retaining the shape of 128. Finally, the output layer uses 10 neurons with a SoftMax activation to classify the digit into one of the classes 0 through 9. The CNN architecture used in this study is designed to effectively extract spatial features from images in the MNIST dataset. Convolutional layers capture local patterns such as edges and textures, while pooling layers reduce dimensionality and computational complexity. The fully connected layers perform high-level reasoning, and the SoftMax output layer produces probability distributions over the ten-digit classes.

LSTM networks are a specialized form of Recurrent Neural Network (RNN) designed to model sequential data and capture long-range temporal dependencies [13]. They are also applied in image-related problems that contain an inherent sequential structure, including video analysis and line-based handwriting recognition [13]. Recent studies emphasize that LSTMs effectively model temporal dynamics directly from raw sequential inputs without requiring extensive handcrafted feature extraction, making them powerful when temporal context is essential [14]. The Long Short-Term Memory (LSTM) network was implemented to evaluate the performance of sequential learning on image data. Since LSTM models are designed for temporal sequence processing, each 28 × 28 image was reshaped into a sequence of 28-time steps with 28 features per step. The architecture consists of an LSTM layer with memory cells that retain important sequential dependencies, followed by a fully connected dense layer for classification. The final output layer uses the SoftMax activation function to produce class probabilities for the ten-digit classes. Although LSTMs are effective for sequence modeling, their application to image data may result in loss of spatial information, as the two-dimensional structure of the image is transformed into a one-dimensional sequence.

Hybrid CNN-LSTM architectures integrate the spatial feature extraction capability of CNNs with the sequential modelling strength of LSTM networks to enhance representation capability. In handwritten text recognition, a CNN encodes segments of a line image into feature embeddings, and an LSTM subsequently models the character sequence along the horizontal axis. This allows the CNN to learn spatial representations while the LSTM captures sequential dependencies within the encoded features [13]. Hybrid CNN-LSTM models can be employed when digit images are interpreted as ordered sequences or have spatial and temporal structure [15]. The hybrid CNN–LSTM architecture combines the strengths of convolutional and sequential learning approaches. In this model, the CNN component is first used to extract spatial features from the input images. These extracted feature maps are then reshaped into sequences and passed to an LSTM layer, which captures sequential dependencies within the features. The final classification is performed using a fully connected dense layer with SoftMax activation. This hybrid approach aims to leverage both spatial feature extraction and sequence modeling capabilities. However, for the MNIST dataset, which primarily contains spatial patterns rather than temporal dependencies, the added complexity of the LSTM component may not significantly improve performance over a standalone CNN.

E. Model Hyperparameter Configuration

The hyperparameter configurations used for the implemented models are as follows: For the LR model, the lbfgs solver was employed as the optimization algorithm due to its efficiency in handling multinomial loss. L2 regularization was applied to prevent overfitting by penalizing large coefficients. To guarantee convergence, a maximum of 1000 iterations was trained. To effectively manage multi-class classification, the multinomial option was chosen. A balanced trade-off between fitting the training data and preserving generalization was achieved by setting the regularization strength parameter, C, to 1.0. For SVM, the Radial Basis Function (RBF) kernel was used to capture non-linear relationships within the data. The regularization parameter C was set to 1.0, controlling the trade-off between maximizing the margin and minimizing classification error. The gamma parameter was set to scale, allowing the algorithm to automatically adjust the kernel coefficient based on the data distribution. And probability estimation was enabled to allow the model to output class probabilities alongside predictions.

The CNN model was designed with 2 convolutional layers for feature extraction, using 32 and 64 filters, respectively. A kernel size of 3×3 was used to capture local spatial features, while a pooling size of 2×2 was applied for down-sampling and dimensionality reduction. The fully connected layer consisted of 128 dense units, followed by a dropout rate of 0.5 to reduce overfitting. The Adam optimizer was utilized with a learning rate of 0.001 to ensure efficient gradient-based optimization. Dropout Rate of 0.5 to prevent overfitting by randomly disabling neurons. The model was trained with a batch size of 64 over 10 epochs. Additionally, the ReLU activation function was applied in hidden layers to introduce non-linearity, while the SoftMax function was used in the output layer to generate class probability distributions. For the LSTM model, the input data was structured into a sequence format of shape (28, 28) to capture temporal dependencies. The model included 128 LSTM units to learn sequential patterns, followed by a dropout rate of 0.2 for regularization. A dense layer with 64 units was used for classification. Like

the CNN, the Adam optimizer with a learning rate of 0.001 was adopted, and the model was trained for 10 epochs.

The Hybrid CNN–LSTM model combined both spatial and sequential feature learning. It incorporated 2 convolutional layers with 32 and 64 filters, a kernel size of 3×3, and a pooling size of 2×2 for feature extraction and down-sampling. The extracted features were then passed to an LSTM layer with 64 units for sequence modeling. A dense layer with 64 units was used for final classification. To prevent overfitting, a dropout rate of 0.5 was applied. The model was optimized using the Adam optimizer with a learning rate of 0.001 and trained for 10 epochs. Overall, these hyperparameter settings were selected to balance model complexity, computational efficiency, and generalization performance on the MNIST dataset. Fig. 4 below visualizes the architecture of CNN, LSTM, and the hybrid model.

F. Performance Metric

Accuracy, Receiver Operating Characteristic (ROC), and AUC (using One-vs-Rest strategy for multi-class) were used to ensure comprehensive evaluation. The Macro-Average AUC was reported to ensure equal weighting across all classes. In addition are Precision, Recall, and F1-scores.

III. Results and Discussion

In this section, the results obtained from the comparative evaluation of classical ML models (LR and SVM) and DL architectures (CNN, LSTM, and CNN-LSTM Hybrid) for handwritten digit classification are presented. The performance was assessed using classification accuracy, Macro AUC, statistical significance testing, and computational complexity analysis.

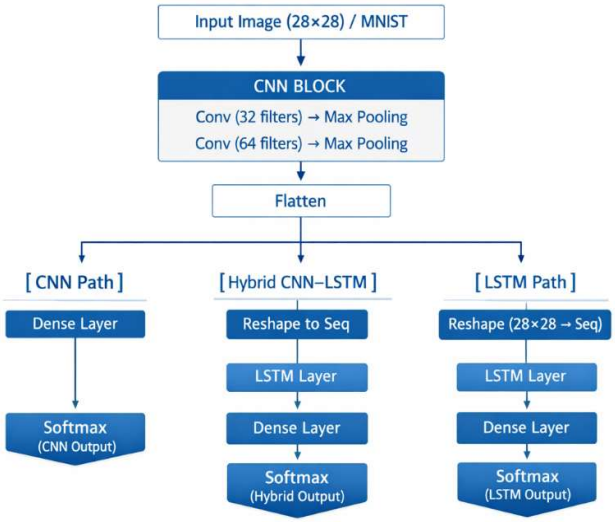

Fig. 4. Unified Architecture of CNN, LSTM, and Hybrid CNN-LSTM Models.

Table I: Comparison of Implemented Models' Performance

Model	Accuracy (%)	Precision	Recall	F1-Score	Macro AUC
LR	92.00	0.9206	0.9209	0.9207	0.994329
SVM	97.59	0.9759	0.9759	0.9758	0.999457
LSTM	98.53	0.9836	0.9834	0.9834	0.999670
CNN	98.96	0.9897	0.9896	0.9896	0.999926
Hybrid CNN -LSTM	98.96	0.9863	0.9861	0.9862	0.999867

A. Performance Comparison

The classification performance of all models is summarized in Table I, and their accuracy is visualized in Fig. 5. The experimental results demonstrate a clear performance progression from classical machine learning models to deep learning architectures. Logistic Regression achieved an accuracy of 92.00% with an F1-score of 0.9207, representing the lowest performance among the evaluated models. Support Vector Machine (SVM) significantly improved performance, achieving 97.59% accuracy and an F1-score of 0.9758.

Deep learning models further enhanced classification performance. The LSTM model achieved 98.53% accuracy and an F1-score of 0.9834. The CNN and CNN–LSTM Hybrid models both achieved the highest accuracy of 98.96%. Among all models, CNN produced the highest precision (0.9897), recall (0.9896), F1-score (0.9896), and Macro AUC (0.999926), thereby demonstrating superior overall performance on classification of MNIST digits.

1) Analysis of Classical Machine Learning Models

Logistic Regression, as a linear classifier, assumes linear separability in the feature space. Since handwritten digit images contain complex spatial patterns and nonlinear variations, their performance is comparatively limited. Although its Macro AUC (0.994329) indicates strong class ranking capability, the lower accuracy and F1-score suggest insufficient modeling capacity for capturing intricate pixel relationships.

The SVM model demonstrated substantial improvement over Logistic Regression. This improvement can be attributed to its ability to construct nonlinear decision boundaries (e.g., via kernel functions) and effectively operate in high-dimensional feature spaces. However, both classical models rely on flattened pixel vectors and do not explicitly exploit spatial locality within images, which restricts their representational power.

2) Analysis of Deep Learning Architectures

The LSTM model outperformed classical approaches, achieving 98.53% accuracy. While LSTMs are primarily

designed for sequential data modeling, treating image rows or columns as sequences enables the model to capture contextual dependencies between pixels. Nevertheless, LSTM architectures are not inherently optimized for spatial feature extraction, which may explain their slightly lower performance compared to CNN-based models.

The CNN model achieved the best overall results across all evaluation metrics. CNNs are particularly well-suited for image classification tasks since the convolutional filters capture local spatial features such as edges and stroke patterns.

The CNN–LSTM Hybrid model achieved the same classification accuracy (98.96%) as the standalone CNN but showed slightly lower precision and F1-score. This indicates that incorporating sequential modeling does not provide significant additional benefit for handwritten digit classification, where spatial feature extraction is the dominant requirement. The marginal difference suggests that CNN alone sufficiently captures discriminative patterns in the dataset.

3) Macro AUC Interpretation

Fig. 5 shows that all models achieved exceptionally high Macro AUC values (>0.99), indicating excellent class separability. Even Logistic Regression demonstrated strong ranking capability despite lower classification accuracy. The near-perfect AUC values for CNN (0.999926) and CNN–LSTM (0.999867) suggest that misclassifications are minimal and likely occur only in visually ambiguous digits.

The CNN achieved the highest classification accuracy (98.96%), Macro AUC, precision, recall, and F1-score, indicating that the CNN achieved consistently high scores across all digit classes.

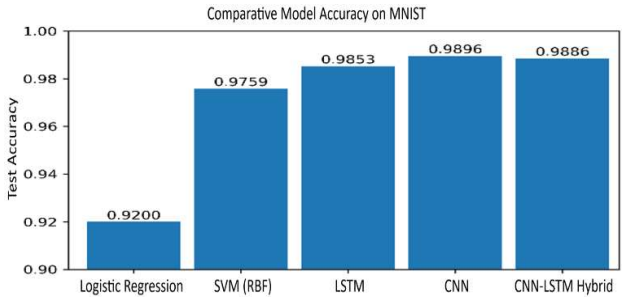

Fig. 5. Comparative Models Accuracy on MNIST.

B. Precision, Recall, and F1-score (PRF) Analysis of CNN

The PRF of the best model (CNN) was analyzed to demonstrate the model's ability to accurately distinguish between handwritten digits across all classes and is visualized by a heatmap shown in Fig. 6. Most digits exhibit near-perfect scores across all evaluation metrics, indicating that CNN effectively captures the spatial patterns inherent in handwritten digit images. The uniformity of the heatmap further confirms the robustness of the CNN architecture in minimizing both false positives and false negatives across the dataset.

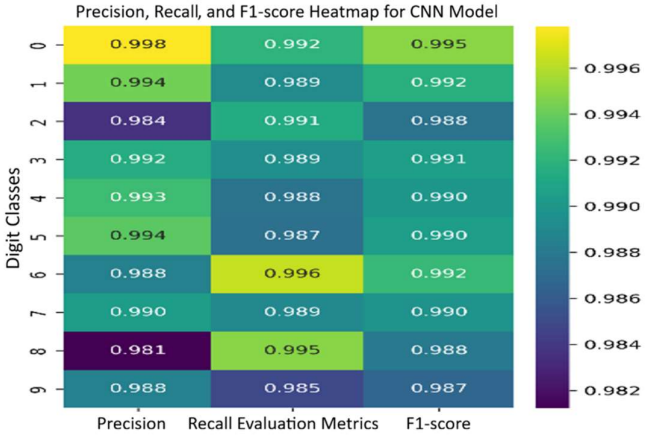

Fig. 6. Precision, Recall, and F1-score of CNN (best implemented model).

C. Confusion Matrix Analysis

The confusion matrix shown in Fig. 7 below provides insight into the classification performance across different digit classes using the CNN model.

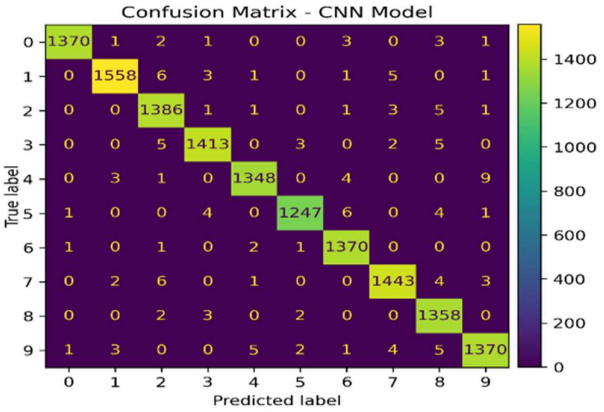

Fig. 7. Confusion Matrix of CNN (best implemented model).

The matrix is strongly diagonal-dominant, indicating that most predictions are correct. Large values on the diagonal (e.g., 1370, 1558, 1413, etc.) indicate high classification accuracy. Off-diagonal values are very small, indicating a low misclassification rate. These represent correctly classified samples: Class 0 (1370), Class 1 (1558 (highest)), Class 2 (1386), Class 3 (1413), Class 4 (1348), Class 5 (1247 (lowest among correct predictions)), Class 6 (1370), Class 7 (1443), Class 8 (1358) and Class 9 (1370). Classes 1 and 7 are the easiest to classify, while Class 5 is relatively harder for the CNN to correctly classify. Even though errors are small, they show confusion trends as 2 is misclassified as 8 / 7 / 3, likely due to similar shapes, 3 is misclassified as 2 / 5 / 8, common overlap in handwritten digits, 4 is misclassified as 9 (9 samples) strongest confusion observed, 5 is misclassified as 6 / 3 / 8, indicating poor separability for class 5, 7 is misclassified as 2 / 8 / 9 and 9 is misclassified as 4 / 8 / 7. Classes like 5 and 4 have more spread-out errors, indicating

slightly lower recall, and Class 1 has almost no confusion, indicating very high recall. Class 5 is slightly problematic. Errors observed reflect the visual similarity in handwritten digits, which is expected in datasets like MNIST. CNN effectively learned discriminative spatial features.

D. ROC Curve Analysis

ROC curves were used to evaluate the discriminative ability of the models across different threshold values. The CNN achieved the highest Area Under the Curve (AUC) as shown in Fig. 8, indicating superior classification capability of MNIST digits.

The ROC curve with an AUC of 0.99, combined with the steep initial slope, shows that the CNN can correctly identify the positive class with very few false alarms. Even at a low threshold, the model rarely misclassifies negatives as positives. The ROC curve complements the learning curves (accuracy/loss over epochs) by showing that the final trained model not only converged well but also achieves near-perfect separability. The ROC curve corroborates that performance, demonstrating that the high accuracy is not due to a skewed threshold but reflects genuine class separation ability. Despite having 225,034 parameters and an inference time of 2.32 ms, the CNN delivers a nearly ideal ROC curve, highlighting that its computational efficiency does not come at the expense of discriminatory power. It is a robust, well-calibrated classifier, capable of making highly accurate predictions with minimal misclassifications.

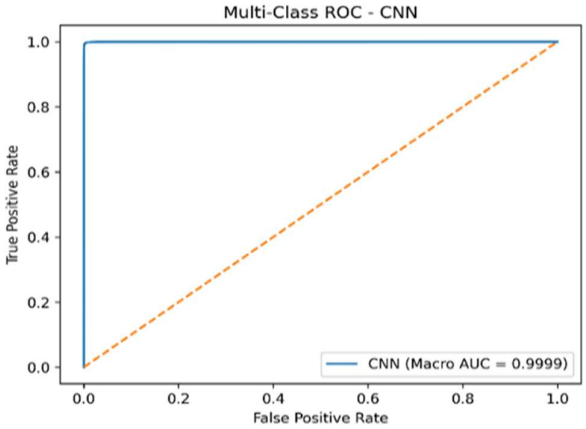

Fig. 8. ROC Curve of CNN (best implemented model).

E. Statistical Significance Testing of Model

McNemar's Test and Bootstrap Confidence Interval test was conducted to determine the significant performance difference between models. Table II shows a comparison of the statistically significant differences of CNN against other models.

Table II: Comparative Statistical Significance Testing of CNN (best implemented model) against other models

Comparison	Chi^2	p-value	Mean Diff	95% Ci	Significance	Direction
CNN vs LR	878.24	≈ 0.0	0.06849	[0.06428, 0.07271]	$p < 0.0001$	CNN > LR
CNN vs SVM	110.93	≈ 0.0	0.01358	[0.01114, 0.01600]	$p < 0.0001$	CNN > SVM
CNN vs LSTM	12.15	0.00049	0.00371	[0.00157, 0.00579]	$p < 0.001$	CNN > LSTM
CNN vs Hybrid	4.36	0.037	0.00299	[0.00086, 0.00500]	$p < 0.05$	CNN > Hybrid

All three comparison instances show statistically significant differences ($p < 0.05$ in all cases), with CNN achieving higher accuracy than the alternative model in every instance. The performance advantage of CNN is the largest against SVM (by about 1.36%), followed by LSTM (about 0.37%), and smallest but still significant against the Hybrid model (by about 0.30%). Also, the confidence intervals for all differences are entirely positive, providing strong evidence that CNN is the best model among those tested on the classification of MNIST digits. The performance of CNN models can be theoretically attributed to their ability to preserve spatial locality through convolutional operations. Unlike classical machine learning models that treat input features independently, CNNs exploit local correlations between neighboring pixels, enabling hierarchical feature extraction. This structural alignment between model architecture and data representation explains the observed performance advantage.

F. Computational Efficiency

To evaluate the computational efficiency of the models, their accuracy, parameter counts, training time, and inference time are compared, as shown in Table III. Insights from Table III inform the efficiency comparison summarized in Table IV.

Table III: Parameters, Training, and Inference Time of Implemented Models

Model	Accuracy (%)	Parameters	Training Time (s)	Inference Time (ms)
LR	92.00	7850	180.17	0.09
SVM	97.59	4580912	314.04	73.14
CNN	98.96	225,034	209.77	2.32
LSTM	98.53	89,290	245.04	4.44
Hybrid	98.96	56,650	279.03	3.98

From Table III, it can be deduced that CNN and Hybrid are the most well-rounded models, with CNN providing top accuracy with fast inference, and moderate training time. The Hybrid model is particularly efficient: it matches CNN's accuracy while using significantly fewer parameters, making it ideal for deployment in memory-constrained environments or when model size matters. LR remains the choice for extreme speed requirements, but at the cost of accuracy. SVM

is clearly inefficient for this task, given its high resource consumption without corresponding accuracy gains.

Table IV: Conclusion of Implemented Models Computational Efficiency

Metric	Best Efficient Model(s)	Reason
Accuracy/ Parameter	Hybrid	Achieves the highest accuracy-to-parameter efficiency among top-performing models
Training Speed	LR, CNN	LR trains fastest; CNN offers a good balance of speed and accuracy
Inference Speed	LR, CNN, Hybrid	LR is fastest, but CNN and Hybrid are also very fast (2–4 ms)
Memory Footprint	LR, Hybrid, LSTM	Hybrid uses only 56k parameters while achieving top accuracy

G. Model Learning Behavior

The learning behavior of the CNN model was analyzed using the training accuracy (TA), validation accuracy (VA), and the corresponding loss curves across training epochs. Figs. 9 and 10 illustrate the TA against VA, and training loss (TL) against validation loss (VL) of the CNN model across the training epochs, respectively, thereby providing insight into the learning behavior and generalization capability of the model during training. The observed trend indicates that the model progressively improved its predictive capability as training progressed. The steady increase in TA, combined with the continuous reduction in training loss, demonstrates that the CNN successfully learned discriminative features necessary for handwritten digit classification. The VL initially decreased and later stabilized with minor fluctuations, suggesting that the model achieved convergence after several training epochs.

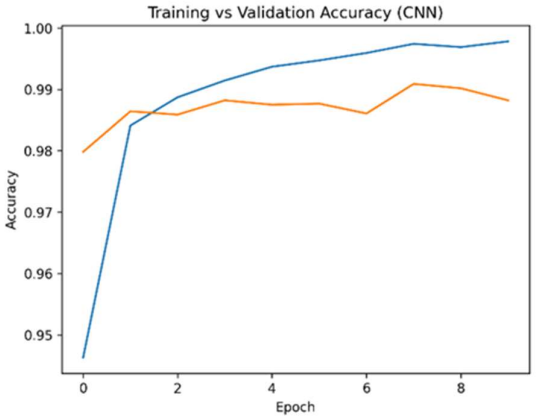

Fig. 9. Training Accuracy vs Validation Accuracy of CNN (best implemented model).

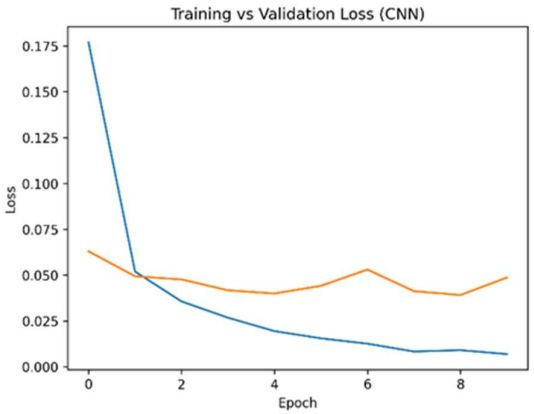

Fig. 10. Training Loss vs Validation Loss of CNN (best implemented model).

Although a slight divergence between training and validation loss can be observed in the later epochs, the difference remains relatively small, indicating that the model did not suffer from severe overfitting and retained good generalization capability. Also, the confusion matrix in Fig. 7 shows that most predictions lie along the diagonal, confirming that the CNN accurately classified most samples from the MNIST dataset.

From Table III, the CNN achieved top accuracy (98.96%) with 225,034 parameters, training in 209.77 s, and inference at 2.32 ms. The learning curves in Figs. 8 and 9 show that this computational investment was well-used as the model converged quickly, meaning training time was not wasted on slow learning. The stable validation performance indicates that the parameter count and architecture are well-matched to the task, with no excess capacity leading to overfitting, and no underfitting due to insufficient complexity. CNN demonstrates effective learning efficiency: it reaches state-of-the-art accuracy with moderate computational cost, and the learning curves confirm a well-behaved, generalizable training process.

H. Model Architecture vs Dataset Structure

The performance results shown in Table V reveal a clear structural alignment pattern. Models that preserved spatial structure achieved the highest classification performance.

Table V: Implemented Model Types vs Dataset Structure Alignment and Accuracy.

Model Type	Structural Alignment with Dataset	Test Accuracy (%)
CNN (spatial)	Strong	98.96
Hybrid	Moderate	98.96
LSTM (sequential)	Weak	98.53
SVM (flattened)	Weak	97.59
Logistic Regression	Weak	92.00

CNN, which is specifically designed for spatial feature extraction, significantly outperformed both LSTM and the Hybrid model ($p < 0.05$). Although the Hybrid model incorporated spatial extraction via CNN layers, the additional sequential modeling component did not enhance performance and, in fact, resulted in statistically inferior accuracy compared to CNN alone. This demonstrates that architectural inductive bias plays a critical role in determining classification performance. When the structural assumptions of a model align with the inherent structure of the dataset, predictive performance improves significantly.

I. Comparison of Results with Previous Works

The Logistic Regression accuracy of 92.00% obtained in this study is consistent with established benchmarks reported in previous studies [2], [4], [6], confirming the correctness of the implementation and providing a reliable baseline. Similarly, the Support Vector Machine result of 97.59% aligns with widely reported performance ranges, where SVMs typically achieve accuracies in the high 90% range for handwritten digit classification tasks.

The CNN accuracy of 98.96% closely matches the result reported by [18] (98.962%), thereby validating the effectiveness of the architecture and training procedure adopted in this work. This agreement reinforces the consensus that CNNs remain among the most effective models for spatial image classification tasks.

The hybrid CNN–LSTM model achieved the same accuracy as the standalone CNN, which is consistent with findings from CNN-LSTM hybrid model studies [8], where only marginal improvements over CNN were observed. This result indicates that while sequential modeling can be integrated without performance degradation, it does not provide a measurable advantage for the MNIST dataset.

More advanced architectures, such as Capsule Network models [9], have reported higher accuracies (up to 99.75%), highlighting the potential benefits of enhanced feature representation techniques. Similarly, hybrid approaches combining CNN with Vision Transformer mechanisms [10] demonstrate improved robustness under noisy conditions.

Table VI presents a comparative summary of the accuracy results obtained in this study alongside those reported in previous works.

Table VI: Comparison of Results from this Study with those Reported in Previous Studies.

Models	Results from this study	Results from other studies with references	
LR	92.00	91.7%	[16]
SVM	97.59	97.1%	[17]
CNN	98.96	98.96%	[18]
LSTM	98.53	97.14%	[19]
Hybrid CNN-LSTM	98.96	~99.0%	[20]

IV. Conclusion

CNN-based models outperform classical techniques such as SVM in handwritten digit classification by effectively capturing spatial hierarchies inherent in image data. However, the performance differences observed on the MNIST dataset are relatively modest due to its simplicity and near-ceiling accuracy.

This study provides a controlled, reproducible, and statistically grounded comparison of classical and deep learning models, demonstrating that alignment between model architecture and data structure is critical for optimal classification performance. While classical models remain competitive, their reliance on flattened representations limits their ability to fully exploit spatial information.

Future work should extend this framework to more complex and diverse handwritten datasets to better evaluate model generalization under challenging conditions. Further investigation into cross-dataset transferability, deployment-oriented metrics such as latency and energy consumption, and the integration of explainable artificial intelligence techniques could enhance the practical applicability and interpretability of classification systems.

Reference

[1] A. Fateh, M. Fateh, and V. Abolghasemi, “Multilingual handwritten numeral recognition using a robust deep network joint with transfer learning,” Information Sciences, vol. 581, pp. 479–494, 2021, doi: 10.1016/j.ins.2021.09.051.

[2] O. Voloshchenko, and M. Plechawska-Wójcik, “Comparison of classical machine learning algorithms in the task of handwritten digits classification,” Journal of Computer Sciences Institute, vol. 21, pp. 279–286, 2021, doi: 10.35784/jcsi.2723.

[3] T. T. Zin, S. Thant, M. Z. Pwint, and T. Ogino, “Handwritten character recognition on Android for basic education using convolutional neural network,” Electronics, vol. 10, no. 8, p. 904, 2021, doi: 10.3390/electronics10080904.

[4] P. S. Uma-Priyadarshini, and A. S. Vickram, “Performance analysis for handwriting identification system using logistic regression algorithm compared with support vector machine,” in AIP Conference Proceedings, vol. 3267, no. 1, p. 020048, 2025, doi: 10.1063/5.0266033.

[5] X. Zhao, L. Wang, Y. Zhang, X. Han, M. Deveci, and M. Parmar, “A review of convolutional neural networks in computer vision,” Artificial Intelligence Review, vol. 57, p. 99, 2024, doi: 10.1007/s10462-024-10721-6.

[6] L. M. Seng, B. B. C. Chiang, Z. A. Abdulsalam, G. Y. Tan, and H. T. Chai, “MNIST handwritten digit recognition with different CNN architectures,”

Journal of Applied Technology and Innovation, vol. 5, no. 1, 2021, doi: 10.65136/jati.v5i1.195.

[7] Z. Takáč, L. Horanská, and N. Krivonakova, "Enhancing LSTM for sequential image classification by modifying data aggregation," in 2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2021, pp. 1–6, doi: 10.1109/FUZZ45933.2021.9494493.

[8] N. LeBlanc, and I. Valova, "Comparison and evaluation of data composition and deep learning models in archival handwritten digit classification," Journal of Machine Intelligence and Data Science, vol. 3, no. 4, pp. 52–67, 2022, doi: 10.11159/jmids.2022.001.

[9] F. Chen, Z. Luo, N. Chen, H. Mao, H. Hu, Y. Jiang, X. Pan, and H. Zhang, "Assessing four neural networks on handwritten digit recognition (MNIST)," Journal of Computer Science Research, vol. 6, no. 3, pp. 17–22, 2024, doi: 10.30564/jcsr.v6i3.6804.

[10] V. Agrawal, J. Jagtap, S. Patil, and K. Kotecha, "Performance analysis of hybrid deep learning framework using a vision transformer and convolutional neural network for handwritten digit recognition," MethodsX, vol. 12, p. 102554, 2024, doi: 10.1016/j.mex.2024.102554.

[11] D. AbdElminaam, F. Essam, H. Samy, J. Wagdy, and S. Albert, "MLHandwrittenRecognition: Handwritten digit recognition using machine learning algorithms," Journal of Computing and Communication, vol. 2, no. 1, pp. 9–19, 2023, doi: 10.21608/jocc.2023.284152.

[12] R. Karakaya, and S. Kazan, "Handwritten digit recognition using machine learning," Sakarya University Journal of Science, vol. 25, no. 1, pp. 65–71, 2021, doi: 10.16984/saufenbilder.801684.

[13] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," Journal of Big Data, vol. 8, no. 1, p. 53, 2021, doi: 10.1186/s40537-021-00444-8.

[14] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," Neurocomputing, vol. 452, pp. 48–62, 2021, doi: 10.1016/j.neucom.2021.03.091.

[15] M. M. Taye, "Theoretical understanding of convolutional neural network: Concepts, architectures, applications, and future directions," Computation, vol. 11, no. 3, p. 52, 2023, doi: 10.3390/computation11030052.

[16] M. Thahiruddin, S. Khotijah, M. Fajar, and A. E. Farras, "A Robustness Study of Multi-Layer Perceptrons and Logistic Regression to Data Perturbation: MNIST Dataset," Zeta-Math Journal, vol. 10, no. 1, pp. 39–50, 2025, doi: 10.31102/zeta.2025.10.1.39-50

[17] A. Rajput, and A. K. Singh, "Handwritten digit recognition accuracy comparison using KNN, CNN and SVM," Educational Administration Theory and Practice, vol. 30, no. 2, pp. 638–643, 2024, doi: 10.53555/kuey.v30i2.1676.

[18] Md. A. Hossain, and Md. M. Ali, "Recognition of handwritten digit using convolutional neural network (CNN)," Global Journal of Computer Science and Technology, vol. 19, no. 2, p. 27, 2019, doi: 10.34257/gjcstdvol19is2pg27.

[19] P. Muthukumar, A. Prabhu, R. A. Karthika, K Eswaramoorthy, V. U Rani, and V. K. Reshma, "Handwritten text recognition from image using LSTM integrated with pixel shifting optimization algorithm," in 2024 International Conference on Advancement in Renewable Energy and Intelligent Systems (AREIS), Dec. 2024, pp. 1–6, doi: 10.1109/AREIS62559.2024.10893651.

[20] F. M. Husari, and M. A. Assaad, "Intelligent handwritten identification using novel hybrid convolutional neural networks – long-short-term memory architecture," Cihan University-Erbil Scientific Journal, vol. 8, no. 2, pp. 99–103, 2024, doi: 10.24086/cuesj.v8n2y2024.