ISSN:2782-8484

FEDPOLAD Journal of Science & Agricultural Technology
(FEDPOLADJSAT); Vol. 4, ISSUE 1. OCTOBER, 2024 Edition
Website: https://seemjournals.fedpolyado.edu.ng/index.php/fedpoladjsat

Proudly Sponsored by: tetfund TERTIARY EDUCATION TRUST FUND
TETFund/ESS/POLY/ADO-EKITI/ARJ/2

DEEP LEARNING APPROACHES TO HANDWRITTEN DIGIT RECOGNITION: A STUDY USING CNN AND ANN WITH THE MNIST DATASET

[1]Sunday Akinwamide, [2]Taiwo Fele and [3]Kayode Akinlekan Akintoye
[1, 2 & 3]Department of Computer Science, The Federal Polytechnic, Ado Ekiti, Nigeria
E-mails: [1]akinwamide_so@fedpolyado.edu.ng, [2]fele_ta@fedpolyado.edu.ng; [3]akintoye_ka@fedpolyado.edu.ng

Abstract
This paper presents a comparative study of Deep Learning approaches, specifically Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), for handwritten digit recognition using the MNIST dataset. The study aims to evaluate and contrast the performance of these models in terms of accuracy and loss across various training epochs. Both ANN and CNN models were trained for 5, 10, 15, 20, 25, and 30 epochs to observe their learning behaviour and effectiveness. The results demonstrate that the ANN model achieved a maximum accuracy of 98.34% with a loss of 0.0644 after 20 epochs. In contrast, the CNN model attained a peak accuracy of 99.40% with a significantly lower loss of 0.0233 after 20 epochs. These findings underscore the superior performance of CNNs in handling complex image recognition tasks, highlighting their potential for applications requiring high precision in pattern recognition.

Keywords: *Handwritten Digit Recognition, Deep Learning, Convolutional Neural Networks, Artificial Neural Networks, MNIST Dataset*

1. Introduction

Handwritten digit recognition has been a vital area of study in computer vision and pattern recognition for many years. This technology has diverse applications, such as sorting postal mail, processing bank checks, and digitizing historical documents. The emergence of deep learning has greatly enhanced the accuracy and efficiency of these recognition systems, enabling them to achieve performance levels close to those of humans in numerous tasks (Wan et al., 2013).

Among the numerous datasets used for benchmarking digit recognition algorithms, the MNIST dataset is one of the most extensively utilized and researched. With 60,000 training images and 10,000 test images of handwritten digits, MNIST offers a standardized framework for assessing and comparing various machine learning models (LeCun & Cortes, 2010).

In this study, we explore two prominent deep learning approaches for handwritten digit recognition: Convolutional Neural Networks (CNNs) and Artificial Neural Networks (ANNs). CNNs have become highly popular because of their capability to autonomously learn spatial hierarchies of features, making them especially effective for image-related tasks. Conversely, ANNs, although a more traditional approach, remain significant and are frequently used as a benchmark for performance comparison (Ciresan et al., 2012).

The main goal of this paper is to examine the effectiveness of CNN and ANN for handwritten digit recognition using the MNIST dataset. We intend to offer a thorough analysis of their performance, emphasizing the strengths and weaknesses of each method. This study aims to contribute to the ongoing research in deep learning and provide valuable insights for practitioners and researchers engaged in similar projects.

In the subsequent sections, we will describe the architecture and implementation of both CNN and ANN models, present the experimental results, and discuss the implications of our findings. By comparing these two methods, we aim to provide a deeper understanding of their capabilities and offer guidance for future advancements in the field of handwritten digit recognition.

The rest of the paper is organized as follows: Section 2 offers a thorough review of relevant literature. Section 3 details the methodology. Section 4 presents the experiments and findings, and Section 5 discusses the results and concludes the study.

2. Related Work

Initial methods for recognizing handwritten digits depended on feature extraction techniques combined with traditional machine learning algorithms, such as Support Vector Machines (SVM) and k-Nearest Neighbours (k-NN) (A. K. Jain et al., 2000). The emergence of deep learning, especially Convolutional Neural Networks (CNNs), has led to substantial gains in accuracy and efficiency. Prior studies have shown that CNNs surpass traditional methods and simpler neural network architectures in performance (Krizhevsky et al., 2012). This research expands on these findings by offering an in-depth comparison between CNN and ANN models. (Upender & Pasupuleti, 2021) introduced recognition and classification of real time handwritten digits by training and testing Convolutional Neural Network (CNN) with MNIST dataset. The model achieved an accuracy of 99.59% with a batch size of 192, 200 epochs and 25 steps per epoch. (Hossain & Ali, 2019) developed a model that will be able to identify and determine the handwritten digit from its image with better accuracy using the concepts of Convolutional Neural Network and MNIST dataset. Among 10,000 test cases, their model misclassified a total of 85 digits after 8 epochs which correspond to 99.15% recognition accuracy. (Seng et al., 2021) implemented the training and testing of CNN ResNet-18 architecture with MNIST dataset. The CNN ResNet-18 architecture was then modified with five PyTorch's pre-trained models including GoogLeNet, MobileNet v2, ResNet-50, ResNeXt-50, Wide ResNet-50 to reveal the best architecture for handwritten digits' recognition in terms of accuracy, training time, top-1 error, top-5 error and model size on all the five models. It was found that the accuracy of ResNet-18 was 96% with a training time of 874 seconds. MobileNet v2 was the best among these 5 models having top-1 error of 15.278, top-5 error of 0.5380 and training time of 826 seconds respectively all in 10 epochs. (Nandan et al., 2020) proposed a simple novel approach to recognize the MNIST dataset handwritten digits by using ensemble learning as it improved convergence by decreasing the complexity of the model. Out of the 60,000 gray-scale images in the training dataset, the training dataset was again divided into train set and validation set. The train set had 54,000 gray-scale images whereas the validation set contained 6,000 gray-scale images (10% of the training set). The base learners were trained using the simple CNN model in Keras for 15 epochs. The MNIST training dataset of 54,000 gray-scale images was divided based on class label wise split and randomly split into 2, 3 and 4 base learner models. When an ensemble model system was compared to that of a single learner model, a significant time gain ranging from

50% to 80% of training time was observed in both random split and class-wise split in 2, 3 and 4 base learner model ensemble system.

The F1 score of the ensemble model was preserved in a random based split because each base learner had enough training data which contains all the class labels to train with a significant amount of time gain. The accuracy got decreased as the number of base learners was increasing in a class based split because each base learner was not getting enough training data of all class labels to train the model. (M. Jain et al., 2021) developed handwritten digit recognition using CNN and MNIST dataset. With 5 epochs, batch size of 200 and a learning rate of 0.02, the model achieved an accuracy of 99.16%. (Goh & Ab Ghafar, 2021) proposed the implement of Convolutional Neural Network (CNN) architecture (First Model) for handwritten digit recognition system, improve the CNN architecture (Second Model) to increase the accuracy of handwritten digit recognition system and analyse the performance of the proposed CNN architecture with MNIST dataset. The final accuracy of the model was 98.99%. (Bharadwaj et al., 2020) implemented Convolution Neural Networks for handwritten digit recognition using MNIST dataset of 70,000 digits. The result of the model after 2 epochs reached 98.21% training accuracy and 98.51% validation accuracy with 5% training loss and 4% validation loss. (Ziweritin et al., 2020) proposed an effective digit recognition system utilizing neural networks and support vector machine (SVM) data mining classification models. This system is capable of recognizing digits ranging from 0 to 9 in the MNIST dataset from pixilated or raster images.

The models were successfully trained and tested, achieving accuracy levels of 94% for the SVM and 99% for the neural network, all implemented using Python. (Garg, 2019) developed models that offer efficient and reliable methods for recognizing handwritten numerals in the MNIST dataset by comparing the accuracies and performance metrics of Support Vector Classification (SVC), Logistic Regression (LR), and Random Forest (RF). The accuracy results for these models were: LR at 88.97%, SVM at 98.3%, and RF at 96.3%. (Pashine et al., 2021) developed handwritten digit recognition using the MNIST dataset with Support Vector Machines (SVM), Multi-Layer Perceptron (MLP), and Convolutional Neural Network (CNN) models. The study aimed to compare the accuracy and execution time of these models to determine the most effective one for digit recognition. The experimental results indicated that SVM achieved an accuracy of 94.01%, MLP achieved 98.85%, and CNN achieved 99.31%.

In the literature review, previous studies have demonstrated the superiority of Convolutional Neural Networks (CNNs) over traditional machine learning methods and simpler neural network architectures for image recognition tasks, particularly on the MNIST dataset. However, the literature underscores the importance of model architecture, training strategies, and evaluation metrics in optimizing performance. This paper aims to build upon these findings by conducting a comparative analysis of CNN and ANN models, focusing on their validation accuracy, validation loss, training times, and suitability for handwritten digit recognition using the MNIST dataset.

3. Methodology

Deep learning, a branch of machine learning, utilizes neural networks with numerous layers (hence "deep") to model and interpret complex patterns within large datasets. It employs these multi-layered networks to automatically learn and represent intricate patterns from extensive datasets. This approach can learn directly from raw data without requiring manual feature extraction. Combined with backpropagation and the capacity to handle large-scale data, deep learning has become essential in many advanced applications (Gu et al., 2018).

To predict hand written digits in this research, we are using ANN and CNN because these neural networks are a shared weights architecture that is most popularly used for analysing images. The major difference between ANN and CNN is the absence of convolutional and pooling layers in ANN. CNN is built with the help of convolutional layer, pooling layer, dropout layer, flatten layer, fully-connected layer and activation functions (Albawi et al., 2017).

Convolutional layer

The convolutional layer is essential for feature extraction, converting input data into feature maps. By applying filters (feature detectors) to the input, the convolutional layer generates feature maps that emphasize specific patterns and features. These feature maps are vital for constructing spatial hierarchies and capturing the local structure of the data, allowing CNNs to handle complex tasks like image and object recognition (Zeiler & Fergus, 2014).

Pooling Layer

The pooling layer is crucial for decreasing the spatial dimensions of feature maps, which helps achieve spatial invariance and lowers the network's computational complexity. Through operations like max pooling or average pooling, it compacts the information in the feature maps while preserving the most significant features. This process enables the CNN to effectively manage complex tasks such as image and object recognition (He et al., 2016).

Dropout Layer

The dropout layer is a highly effective regularization technique used in neural networks, including both ANNs (Artificial Neural Networks) and CNNs (Convolutional Neural Networks). It helps these networks generalize better by preventing overfitting. By randomly deactivating neurons during training, dropout compels the network to learn more robust features, reducing the risk of over-relying on specific neurons. This leads to a model that performs better on new, unseen data (Baldi & Peter, 2013). Dropout is a method where, during each training iteration, a portion of the neurons in the neural network is randomly deactivated. This means that with each forward and backward pass, the network effectively operates with a different architecture.

Flatten Layer

The Flatten layer in neural networks, especially in Convolutional Neural Networks (CNNs) and Artificial Neural Networks (ANNs), is vital for converting multi-dimensional input data into a one-dimensional vector. This conversion is essential to link the convolutional or pooling layers, which handle multi-dimensional data, to the fully connected (dense) layers that require one-dimensional input (Deng, 2014). The Flatten layer bridges the gap between convolutional/pooling layers and fully connected layers. By transforming multi-dimensional arrays into one-dimensional vectors, it ensures that the features extracted by the previous layers are effectively used in the final classification or regression tasks, allowing the dense layers

ISSN:2782-8484
FEDPOLAD Journal of Science & Agricultural Technology
(FEDPOLADJSAT); Vol. 4, ISSUE 1. OCTOBER, 2024 Edition
Website: https://seemjournals.fedpolyado.edu.ng/index.php/fedpoladjsat
Proudly Sponsored by: tetfund TERTIARY EDUCATION TRUST FUND
TETFund/ESS/POLY/ADO-EKITI/ARJ/2

to fully utilize the rich spatial information captured by the convolutional layers.

Fully-connected Layer

The Fully-connected layer is crucial in neural networks for integrating and interpreting the features extracted by preceding layers. It is responsible for the final decision-making process in the model, facilitating high-level reasoning, classification, and regression tasks. By combining and processing the learned features, Fully-connected layers are pivotal in making precise predictions and ensuring the model's adaptability and effectiveness across various problem types (Krizhevsky et al., 2012).

Activation Functions

Activation functions are crucial in neural networks as they introduce non-linearity, facilitate the approximation of complex functions, break symmetry, regulate neuron outputs, and ensure proper gradient flow during training. They are essential for effectively learning and representing complex data patterns, significantly enhancing the network's capacity to tackle various problems. Notable activation functions include Sigmoid, ReLU (Rectified Linear Unit), and Softmax (Glorot & Yoshua, 2010).

Dataset

In this study, we utilized the MNIST dataset which consists of 70,000 handwritten digits, comprising 60,000 training images and 10,000 test images. Each grayscale image is 28x28 pixels. These images were collected from American Census Bureau employees and American high school students (LeCun et al., 1998). The MNIST data is stored in the IDX file format, and an example is illustrated in Figure 1.

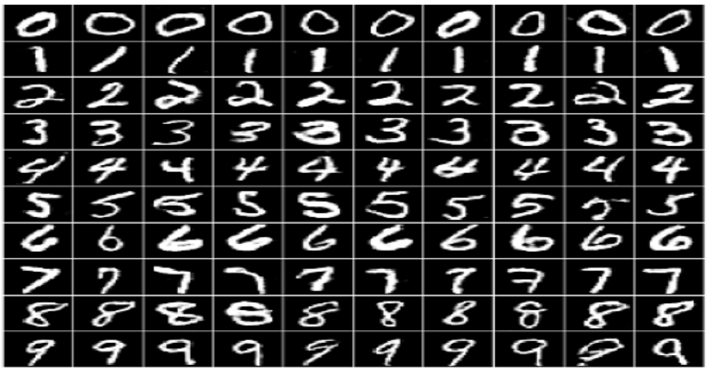

Figure 1: Sample MNIST data

3.1 Data Pre-processing

When loading the dataset from tensorflow.keras.datasets, it is already divided into (X_train, y_train) and (X_test, y_test), with shapes ((60000, 28, 28), (60000,)) for the training set and ((10000, 28, 28), (10000,)) for the test set. As shown in Figure 2, the initial layer (input layer) of our model uses convolution. In convolution, each pixel acts like a neuron, so we reshape the images to ensure each pixel value maintains its unique spatial position. This transformation converts a grayscale matrix of size 28x28 into a tensor with dimensions 28x28x1.

3.2 Model Architectures

3.2.1 Convolutional Neural Network (CNN)

As shown in figure 2, the CNN architecture used in this study includes:

FEDPOLAD Journal of Science & Agricultural Technology (FEDPOLADJSAT) is a Bi-Annual Publication Series of the Federal Polytechnic, Ado-Ekiti, Ekiti State. For more details, kindly visit https://seemjournals.fedpolyado.edu.ng/index.php/fedpoladjsat. All critics, reviews, correspondence, or submission of articles for scholarly publication in the next edition of this Journal should be forwarded to seemjournal@fedpolyado.edu.ng. For more enquiries, please contact +234 806 701 4621 or +234 803 506 0823.

ISSN:2782-8484

FEDPOLAD Journal of Science & Agricultural Technology
(FEDPOLADJSAT); Vol. 4, ISSUE 1. OCTOBER, 2024 Edition
Website: https://seemjournals.fedpolyado.edu.ng/index.php/fedpoladjsat

Proudly Sponsored by: tetfund TERTIARY EDUCATION TRUST FUND
TETFund/ESS/POLY/ADO-EKITI/ARJ/2

- Two convolutional layers with ReLU activation and max-pooling layers.
- A flattening layer to convert the 2D feature maps to a 1D vector.
- A fully connected (dense) layers with ReLU activation.
- A drop-out layer having rate of 0.5 for preventing overfitting
- An output layer with softmax activation for classification.

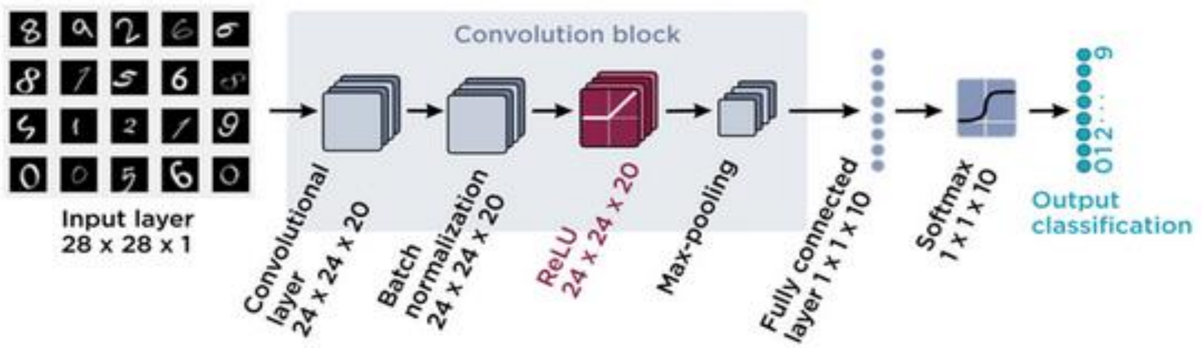

Figure 2: Convolutional Neural Network Architecture

3.2.2 Artificial Neural Network (ANN)

Considering Figure 3, the ANN architecture adopted in this research includes:

- An input layer with 784 neurons (28x28 pixels flattened).
- Two hidden layers with 256 neurons in the first layer, 128 neurons in the second layer and ReLU activation in the 2 hidden layers.
- Each of the two hidden layer has a dropout layer of rate 0.2
- An output layer with 10 neurons (one for each digit) and softmax activation.

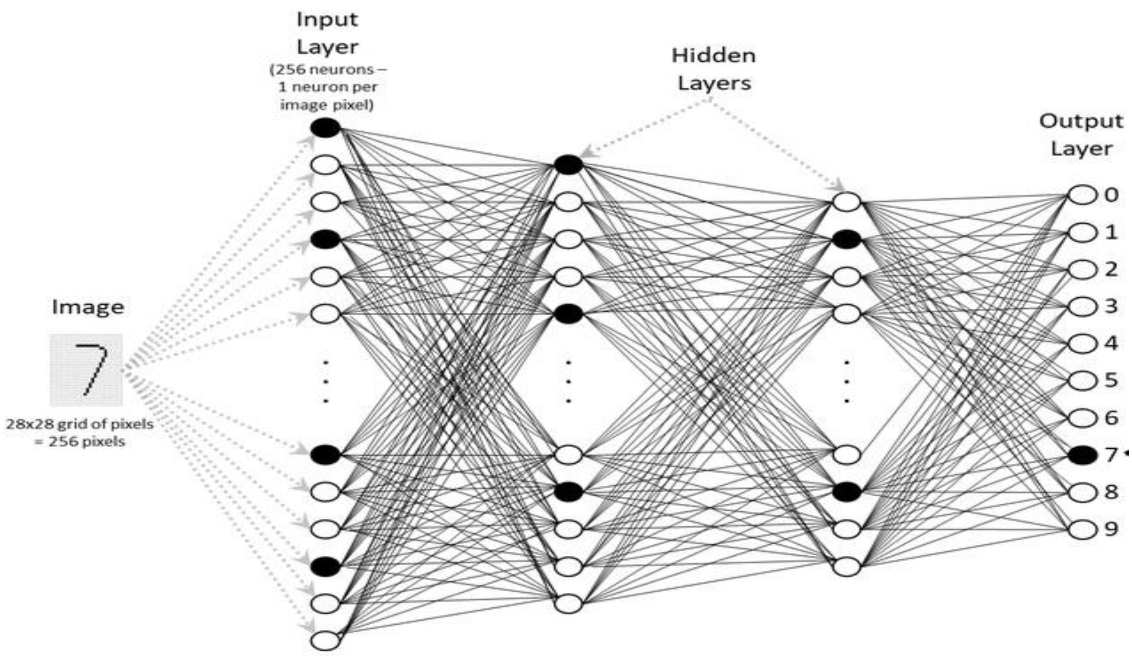

Figure 3: Artificial Neural Network Architecture

ISSN:2782-8484

FEDPOLAD Journal of Science & Agricultural Technology
(FEDPOLADJSAT); Vol. 4, ISSUE 1. OCTOBER, 2024 Edition
Website: https://seemjournals.fedpolyado.edu.ng/index.php/fedpoladjsat

Proudly Sponsored by: tetfund TERTIARY EDUCATION TRUST FUND
TETFund/ESS/POLY/ADO-EKITI/ARJ/2

3.3 Training Process

To achieve the training task, ANN and CNN models were developed as illustrated in the Figures 2 and 3. These models were constructed using TensorFlow and Keras libraries. The MNIST dataset was employed for training the models. Upon loading the dataset from tensorflow.keras.datasets, it was split into training and test sets. Various models with different hyperparameters, as detailed in Table 1, were tested using the Python programming language. The experiments were conducted on Anaconda's Jupyter Notebook, running on a Windows PC.

Table 1: Description of the Models

	Hyper-parameters
ANN	
Flatten layer	input_shape = (28, 28, 1)
Hidden Layer 1	units = 256, activation = ('relu'), Dropout(rate = 0.2)
Hidden Layer 2	units = 128, activation = ('relu'), Dropout(rate = 0.2)
Output Layer	units = 10, activation = ('softmax')
Total Trainable Parameters	235,146
CNN	
Convolutional Layer 1	filters = 32, kernel_size = (3, 3), activation = ('relu'), input_shape = (28, 28, 1)
Max Pooling Layer 1	pool_size = (2, 2)
Convolutional Layer 2	filters = 64, kernel_size = (3, 3), activation = 'relu',
Max Pooling Layer 2	pool_size = (2, 2)
Flatten Layer	()
Fully Connected Layer	units = 128, activation = ('relu'), Dropout(rate = 0.5)
Output Layer	units = 10, activation = ('softmax')
Total Trainable Parameters	225,034

Both models were trained using the Adam optimizer and the sparse categorical cross-entropy loss function. The training process included 5, 10, 15, 20, 25, and 30 epochs, with a batch size of 32 and 1875 steps per epoch. EarlyStopping, with a patience of 5, was utilized to prevent overfitting during the training of the models.

4. Experiments and Results

4.1 Experimental Setup

The proposed ANN and CNN models were trained for different epochs, specifically 5, 10, 15, 20, 25, and 30, to compare their accuracy and loss on the test dataset. The experimental outcomes, showing the various accuracies and losses for each epoch, are summarized in Table 2.

Table 2: Results of the proposed Models

Epoch	**Validation Accuracy**	**Validation Loss**	**Train Accuracy**	**Train Loss**
	ANN			
5	0.9787	0.0725	0.9889	0.0339
10	0.9810	0.0652	0.9922	0.0250
15	0.9812	0.0662	0.9936	0.0195
20	**0.9834**	**0.0644**	0.9954	0.0142
25	0.9812	0.0684	0.9966	0.0119
30	0.9826	0.0683	0.9968	0.0108
	CNN			
5	0.9931	0.0236	0.9965	0.0120

10	0.9936	0.0241	0.9987	0.0048
15	0.9937	0.0225	0.9986	0.0046
20	**0.9940**	**0.0233**	0.9992	0.0023
25	0.9939	0.0281	0.9993	0.0019
30	0.9931	0.0290	0.9994	0.0019

Based on Table 2, the ANN model attained a maximum accuracy of 98.34% after 20 epochs with a loss of 0.0644, while the CNN model achieved a peak accuracy of 99.40% after 20 epochs with a loss of 0.0233. The validation accuracies and losses for both ANN and CNN were plotted against epochs in Figures 4 (a) & (b) and Figures 5 (a) & (b) respectively. Figures 6 (a) and (b) present the confusion matrices at the highest accuracies for both ANN and CNN, illustrating the correct and incorrect predictions for each digit in the test dataset.

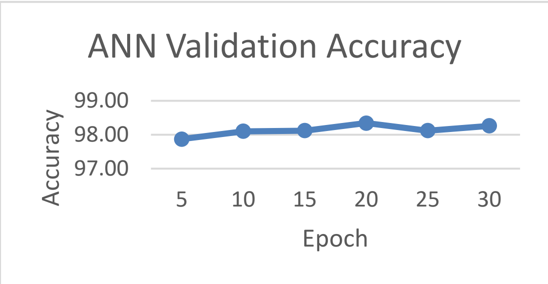

Figure 4(a):

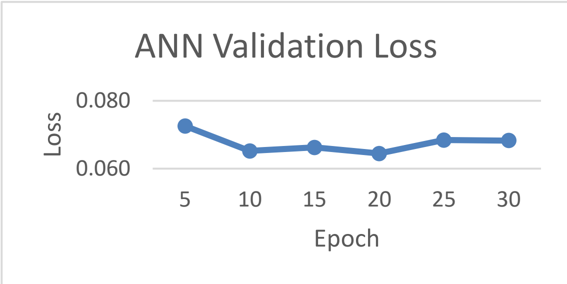

Figure 4(b):

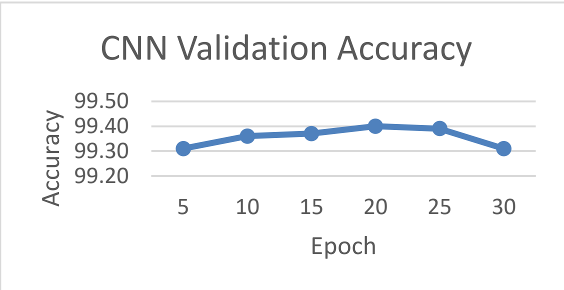

Figure 5(a):

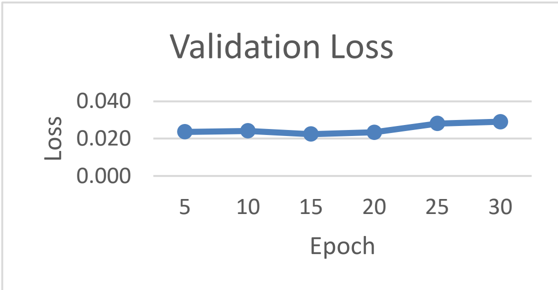

Figure 5(b):

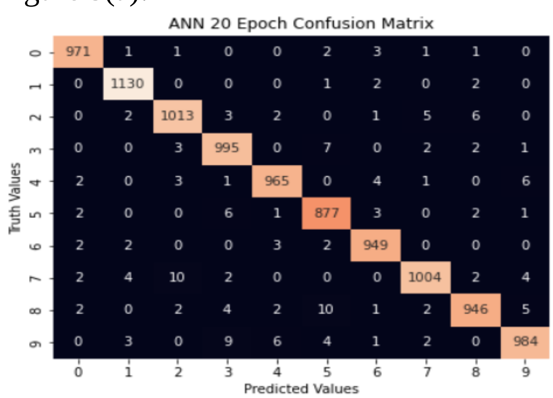

Figure 6(a):

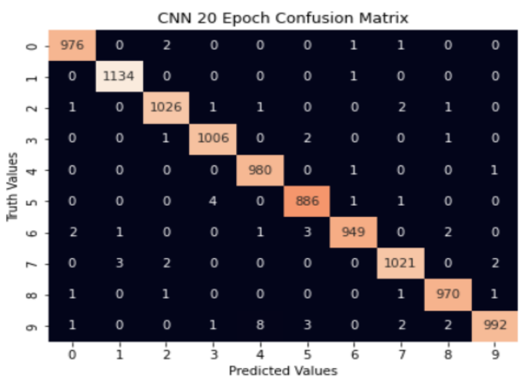

Figure 6(b):

The accuracies of various related studies using MNIST dataset, previously reviewed in this research and proposed by different authors, are compared with the accuracy of the proposed method, as detailed in Table 3.

Table 3: Accuracy comparison for various related methodologies

S/N	Author	Methodology	Accuracy
1.	Hossain & Ali, (2019)	CNN	99.15%
2.	Seng et al., (2021)	CNN ResNet-18	96%
3.	M. Jain et al., (2021)	CNN	99.16%
4.	Goh & Ab Ghafar, (2021)	CNN	98.99%.
5.	Bharadwaj et al., (2020)	CNN	98.51%
6.	Ziweritin et al., (2020)	ANN and SVM	99%
7.	Garg, (2019)	SVC, LR and RF	98.3%
8.	Pashine et al., (2021)	SVM, MLP and CNN	99.31%
9.	**Proposed Research**	**ANN and CNN**	**99.40%**

Table 3 clearly shows that our proposed research achieved the highest accuracy compared to all the reviewed related papers.

5. Conclusion

This study showcases the superior performance of CNNs compared to ANNs for handwritten digit recognition on the MNIST dataset, with the CNN model achieving a test accuracy of 99.40%, significantly higher than the ANN model's 98.34%. CNNs also demonstrate faster convergence and better generalization. Their ability to automatically extract features from raw images makes CNNs more suitable for image recognition tasks, as they can capture spatial hierarchies in the data, resulting in higher accuracy and efficiency. This paper highlights the potential of deep learning techniques in digit recognition tasks and provides a foundation for future research in this area. Next, we propose a model capable of recognizing the 36 English alphabetic characters using deep learning techniques.

References

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017). Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)*, 1–6.

Baldi, P., & Peter, J. S. (2013). Understanding Dropout. *Advances in Neural Information Processing Systems 26*.

Bharadwaj, Y. S., Rajaram, P., Sriram, V. P., Sudhakar, S., & Prakash, K. B. (2020). Effective handwritten digit recognition using deep convolution neural network. *International Journal of Advanced Trends in Computer Science and Engineering*, *9*(2), 1335–1339.

Ciresan, D. C., Meier, U., & Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3642–3649. https://doi.org/10.1109/CVPR.2012.6248110

Deng, L. (2014). Deep Learning: Methods and Applications. *Foundations and Trends in Signal Processing 7.3-4, 7*, 197–387.

Garg, V. (2019). Handwritten Digit Classification using Machine Learning Models. *International*

Research Journal of Engineering and Technology (IRJET), *6*(11), 3593–3597.

Glorot, X., & Yoshua, B. (2010). Understanding the Difficulty of Training Deep Feedforward Neural Networks. *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 249–256.

Goh, K. Y., & Ab Ghafar, A. S. B. (2021). Handwritten Digit Recognition System using Convolutional Neural Network (CNN). *Progress in Engineering Application and Technology*, *2*(1), 578–592.

Gu, J., Wang, Z., Kuen, J., Ma, L., ..., & Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition*, *77*, 354–377.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Hossain, M. A., & Ali, M. M. (2019). Recognition of handwritten digit using convolutional neural network (CNN). *Global Journal of Computer Science and Technology*, *19*(2), 27–33.

Jain, A. K., Duin, R. P., & Mao, J. (2000). *Statistical pattern recognition: A review. IEEE Transactions on pattern analysis and machine intelligence*. *22*(1), 4–37.

Jain, M., Kaur, G., Quamar, M. P., & Gupta, H. (2021). Handwritten digit recognition using CNN. *In 2021 International Conference on Innovative Practices in Technology and Management (ICIPTM)*, 211–215.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems*. *25*.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeCun, Y., & Cortes, C. (2010). *MNIST handwritten digit database. AT&T Labs*. Available Online: http://yann.lecun.com/exdb/mnist/

Nandan, K. V. P., Panda, M., & Veni, S. (2020). Handwritten digit recognition using ensemble learning. *In 2020 5th International Conference on Communication and Electronics Systems (ICCES)*, 1008–1013.

Pashine, S., Dixit, R., & Kushwah, R. (2021). Handwritten digit recognition using machine and deep learning algorithms. *arXiv Preprint arXiv:2106.12614*.

Seng, L. M., Chiang, B. B. C., Salam, Z. A. A., Tan, G. Y., & Chai, H. T. (2021). MNIST handwritten digit recognition with different CNN architectures. *J. Appl. Technol. Innov*, *5*(1), 7–10.

Upender, K., & Pasupuleti, V. S. K. (2021). Real time handwritten digits recognition using convolutional neural network. *In 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 69–73.

Wan, L., Zeiler, M., Zhang, S., LeCun, Y., & Fergus, R. (2013). Regularization of neural networks using DropConnect. *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 1058–1066.

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. *European Conference on Computer Vision*, 818–833.

Ziweritin, S., Ukegbu, C. C., & Ezeorah, E. U. (2020). Building Data Mining Classification Model For Pixilated Digit Recognition System. *SSRG International Journal of Computer Science and Engineering (IJCSE)*, *7*(10), 6–12.