Date of Submission June 11, 2025.

Image Classification using Artificial Neural Networks

AMRIT KANDEL[1], (THA078BEI004), PRASISH TIMALSINA[2], (THA078BEI026)

[1]Department of Electronics and Computer Engineering, Thapathali Campus (e-mail: amrit.tha078bei004@tcioe.edu.np)

[2]Department of Electronics and Computer Engineering, Thapathali Campus (e-mail: prasish.tha078bei026@tcioe.edu.np)

ABSTRACT This lab report outlines the application of Artificial Neural Networks (ANNs) for image classification using the MNIST handwritten digit dataset. The ANN was developed using the Keras Sequential API with dense layers using ReLU activation functions and a softmax output layer for multi-class classification. The model attained an accuracy of 96.51%, while precision and recall metrics averaged 0.97 across all digit classes. Performance measures such as the confusion matrix, recall, and precision were utilized in determining model effectiveness. The findings demonstrate the power of ANNs for classifying images, pointing toward their potential application in real-life situations despite challenges like computational requirements and data needs.

INDEX TERMS Artificial Neural Networks (ANN), Image Classification, MNIST Dataset, Performance Metrics, Confusion Matrix, Precision, Recall, Keras, Deep Learning

I. INTRODUCTION

A. ARTIFICIAL NEURAL NETWORKS (ANN)

An Artificial Neural Network (ANN) is a type of machine learning that is inspired by the way the human brain processes information. It consists of layers of interconnected nodes, or neurons, that work together to analyze and interpret data. Each connection between neurons has a weight that is adjusted during training to improve the network's accuracy.

In a typical ANN, the input is taken in the first layer, and it is passed along one or more hidden layers. The hidden layers perform computations and transformations using mathematical functions known as activation functions. Then the output layer provides the output, for instance, a prediction or classification.

ANN training involves a procedure called backpropagation, where the network learns from its output and the correct solution by adjusting the weights in a way that reduces the error. This learning is repeated many times for different datasets, and the performance of the model improves gradually.

ANNs have widespread uses in real-world problems such as image and speech recognition, medical diagnosis, and financial forecasting. Their ability to learn from data and identify complex patterns makes them extremely helpful in artificial intelligence. They can, however, be computationally intensive and require enormous amounts of data to work efficiently.

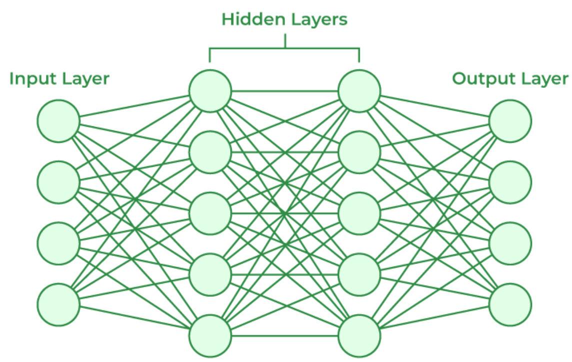

FIGURE 1. **Artificial Neural Network Architecture**

B. PERFORMANCE METRICS

Performance metrics are used to evaluate the performance of a machine learning model at making predictions. They quantify the success of the model and are especially important in classification problems. Some of the popular ones include Accuracy, Precision, Recall, and Confusion Matrix.

1) Confusion Matrix

A Confusion Matrix is a table showing the summary of the performance of a classification model. It tells us the number of correct and incorrect predictions made with respect to the

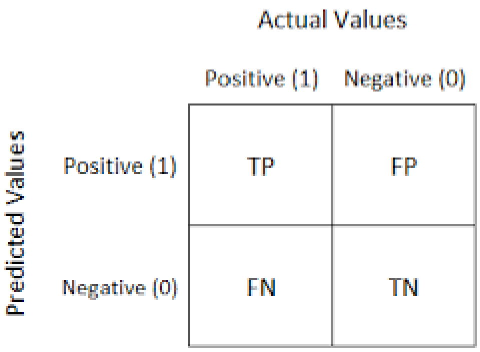

FIGURE 2. Confusion Matrixx

actual labels. The most important elements are:

- True Positive (TP): Model predicts Positive, and it's actually Positive
- True Negative (TN): Model predicts Negative, and it's actually Negative
- False Positive (FP): Model predicts Positive, but it's actually Negative
- False Negative (FN): Model predicts Negative, but it's actually Positive

2) Accuracy

Accuracy computes the percentage of total correct predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

3) Precision

Precision tells us how many of the predicted positive are indeed correct.

$$Precision = \frac{TP}{TP + FP}$$

4) Recall

Recall (also called Sensitivity or True Positive Rate) computes the number of actual positives which were correctly identified.

$$Recall = \frac{TP}{TP + FN}$$

II. DATASET DESCRIPTION

We used MNIST (Modified National Institute of Standards and Technology) Handwritten Digit dataset. It is a standard in the machine learning and computer vision community. It includes 70,000 grayscale images of handwritten digits of size 28x28 pixels. It is divided into 60,000 training images and 10,000 test images, with each image labeled with the appropriate digit it represents, from 0 through 9. MNIST has become a standard for evaluating classification algorithms due to its simplicity, cleanliness, and balanced classes. While fairly simple, it is nevertheless a basic dataset for evaluating new image classification models and techniques.

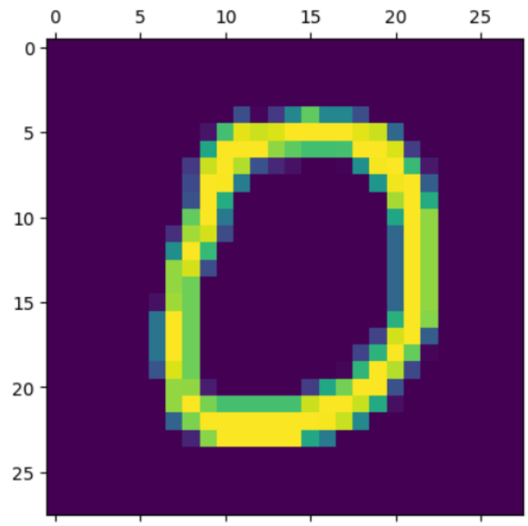

FIGURE 3. MNIST dataset sample

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 20)	15,700
dense_1 (Dense)	(None, 50)	1,050
dense_2 (Dense)	(None, 10)	510

Total params: 17,260 (67.42 KB)

Trainable params: 17,260 (67.42 KB)

Non-trainable params: 0 (0.00 B)

FIGURE 4. Model Summary

III. EXPERIMENTATION MODEL

The model is built using the Sequential API in Keras, which supports linear stacking of layers. The input layer is automatically defined by the input_shape=(784,) argument in the first Dense layer, where the input image (28x28 pixels) is flattened into a 784-dimensional vector. The flattened vector is fed into a fully connected (dense) layer of 20 neurons with a ReLU (Rectified Linear Unit) activation function. ReLU adds non-linearity, allowing the model to learn complex patterns.

Following the first dense layer, a second dense layer of 50 neurons using again the ReLU activation function is added. This provides extra capacity to the model to learn even more abstracted representations of the input data. The final layer is a dense layer of 10 neurons, which corresponds to the 10 digit classes (0-9). It uses the softmax activation function to output a probability distribution over the classes for multi-class classification.

The model is optimized using the Adam optimizer, a quick stochastic gradient descent optimizer that is specifically

313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 926us/step

	precision	recall	f1-score	support
0	0.98	0.98	0.98	980
1	0.98	1.00	0.99	1135
2	0.97	0.98	0.97	1032
3	0.94	0.99	0.96	1010
4	0.97	0.96	0.97	982
5	0.95	0.96	0.96	892
6	0.98	0.97	0.97	958
7	0.98	0.95	0.96	1028
8	0.97	0.94	0.96	974
9	0.96	0.96	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

FIGURE 5. **Result Summary**

ideal for this type of problem. The loss function used is sparse_categorical_crossentropy, appropriate when the target labels are integers (not one-hot encoded). Accuracy is also tracked while training to provide an easy measure for monitoring performance. This setting is highly normal and effective for the MNIST dataset and strikes a good balance between ease and performance. The summary of the model is given in Figure 4.

IV. RESULTS

In lab, we used different performance metrics like Accuracy, precision and recall and observed the result. While selecting the accuracy as our performance metrics, we got the accuracy of 96.51%. We used sklearn library to measure the overall performance of the model. The summary of the result is given in Figure 5

V. DISCUSSION

The experiment worked well to demonstrate the potential of ANNs in identifying handwritten digits with great accuracy (96.51%). The model's performance was evaluated on the basis of precision, recall, and a confusion matrix, which returned similar performances for all classes of digits. In particular, digits like "1" had near-perfect recall (1.00), while others like "3" and "5" had comparatively lower precision, which can be due to the similarity in handwriting.

Knowledge of ReLU activation functions and the Adam optimizer enabled effective training, and the softmax output layer elegantly managed multi-class probabilities. The model's simplicity (e.g., only two hidden layers), however, may limit its ability to generalize to more complex datasets. Deeper architectures (e.g., CNNs) or data augmentation may be explored in future research to enhance performance.

VI. CONCLUSION

This experiment reconfirmed the feasibility of ANNs for image classification, as evidenced by the high accuracy and balanced precision-recall scores on the MNIST dataset. The experiment confirmed the importance of performance metrics in model robustness assessment and indicated possibilities for further improvement, such as addressing class-specific misclassifications. ANNs can be computationally demanding, but their flexibility and learnability make them very valuable for machine learning applications. Future research can attempt to tune architectures or scale to more diverse datasets for better generalization.

•••