**Handwritten Digit Recognition using Machine Learning and Deep Learning**

Sharvani Baggani Varshini Annam Pranavi Sree Vodnala

Department of CSE, SRU Department of CSE, SRU Department of CSE, SRU

2203A51557 2203A51830 2203A51832

[Sharvanibaggani@gmail.com](mailto:Sharvanibaggani@gmail.com) [annamvarshini799@gmail.com](mailto:annamvarshini799@gmail.com) vodpranavi@gmail.com

Mohammad Samreen Meneni Abhinav

Department of CSE, SRU Department of CSE, SRU

2203A51829 2203A51833

[Mohammadsamreen244@gmail.com](mailto:Mohammadsamreen244@gmail.com) [abhinavabhi4942@gmail.com](mailto:abhinavabhi4942@gmail.com)

**Abstract**—Mankind has long relied on machines, and today, it has achieved levels that allow tasks from object classification on images to provide contextually suitable audio for silent videos by advanced algorithms. Such advancements are embodied in research and applicative areas like handwritten text recognition (HTR), which enables machines to interpret and digitize handwritten inputs coming from paper, photographs, touchscreens, and much more. In this paper, we consider the state-of-the-art handwritten digit recognition with the aid of advanced deep learning architectures, namely ResNet-18, DenseNet-121, and Vision Transformer (ViT-Base). Utilizing the MNIST dataset, our primary focus was to assess and compare these models in terms of accuracy, computational cost, and execution time. This study also helps to identify the best model for handwritten digit recognition by shedding light on the performance of contemporary deep learning techniques.

Keywords: Deep Learning, Handwritten Digit Recognition, MNIST Dataset, ResNet-18, DenseNet-121, Vision Transformer (ViT-Base), Advanced Architectures, Comparative Analysis.

|  |  |
| --- | --- |
|  |  |

**I.INTRODUCTION**

Handwritten digit recognition is an important computer vision application that identifies and classifies handwritten numerical digits from 0 through 9 from images, documents, or touch screens. This problem has been heavily researched because of its numerous applications, such as the automated recognition of number plates, postal mail sorting, and bank check processing [2]. Written digit recognition is very different from optical character recognition due to significant individual variability in handwriting, and is therefore more difficult.

Deep learning has witnessed advancements where better algorithms result in great improvements in terms of accuracy and reliability. This paper compares advanced deep learning architectures—ResNet-18, DenseNet-121, and Vision Transformer (ViT-Base)—for handwritten digit recognition based on MNIST. These architectures adopt deep feature extraction; sophisticated skip connections, and attention mechanisms enhance the learning capability and performance.

We compare these models based on key performance metrics such as their accuracy and error rates, as well as their computational efficiency, such as training time as well as inference times. Visualization using matplotlib can help clarify the differences in performance between these algorithms. The final aim is to find the model best offering that good trade-off between accuracy and high computational efficiency, making it viable for real applications. For example, in an automated bank check processing system, the error of digit recognition leads to accounting discrepancies, so there's no room for inaccuracy and unreliability.

This paper provides insights into the potential capabilities of modern deep learning techniques, such as ResNet, DenseNet, and Vision Transformers, for handwritten digit recognition. Additionally, the strengths and limitations of these architectures in terms of their generalization across different handwriting styles are discussed.

In the following sections, we briefly discuss related work in the area of handwritten digit recognition, outline how ResNet-18, DenseNet-121, and ViT-Base were implemented and designed, and then provide a detailed analysis of their performance. Lastly, the paper concludes with observations from the experiments, future directions for this research, and provides a list of references to support this study.

**II. RELATED WORK:**

Advances in deep architectures have significantly advanced the area of handwritten digit recognition. Classical approaches like SVM, KNN, and MLP alone had settled many approaches to hand-written text identification but often struggled with generalization over diverse writing styles. Some studies carried out by S. M. Shamim [3], Anuj Dutt [4] have pointed out the merits and demerits of these models as compared to deep learning methods used within CNNs.

Modern research has focused on harnessing the power of cutting-edge architectures to overcome these constraints. Advanced CNNs such as ResNet and DenseNet introduced recent features such as residual learning and dense connectivity, facilitating deeper and robust representations. Vision Transformers (ViT) have also revolutionized computer vision tasks, such as handwritten digit recognition, through the use of self-attention mechanisms for global context understanding.

**ResNet (Residual Networks):**

Key Innovation: ResNet presented residual learning, where shortcut, or skip, connections bypass one or more layers in the network. Such connections prevent the vanishing gradient problem and thus allow arbitrarily deep architectures to be easily trained.

Advantages for Handwritten Digit Classification:

ResNet-18 is capable of capturing subtle patterns in handwritten digits while being computationally efficient.

Experiments conducted over MNIST show that ResNet is able to provide near perfect accuracy, further involving reduced training error than traditional CNNs such as LeNet.

Transfer learning with pre-trained ResNet on large datasets like ImageNet for fine-tuning still boosts performance on handwritten digit recognition.

Findings in Related Work: Robustness and generalization capabilities, often able to surpass 99% accuracy on MNIST and other handwritten digit datasets, were demonstrated with ResNet.

**Densely Connected Networks (DenseNet):**

Key Innovation: DenseNet provides the ability to connect each layer to every preceding layer in a feed-forward manner, enabling feature reuse and improvement to gradient flow with a diminution in the number of parameters but retention of strong performance.

Advantages for Handwritten Digit Recognition

The memory and compute efficiency of DenseNet-121 allows for sufficient space in resource-constrained environments.

Better feature flow enables DenseNet to manage difficult patterns especially those with overlapping digits.

In many cases, DenseNet is more convergent and accurate than typical CNNs, especially in challenging handwritten datasets.

Related Research Results: The experiments on MNIST, CIFAR-10, and SVHN benchmarks have shown state-of-the-art performance for dense nets. This implies that it can deal with diversity and complexity efficiently.

**Vision Transformers (ViT):**

Key Innovation: ViTs use self-attention mechanisms to model global relations between image patches. They do not rely on local receptive fields like CNNs, which enables them to capture global dependencies across images effectively.

Advantages in Handwritten Digit Recognition

Models of ViT, for example, ViT-Base, split images into patches, e.g., of size 16x16 pixels and represent each patch as a vector embedding, allowing for fine analysis in handwriting nuances.

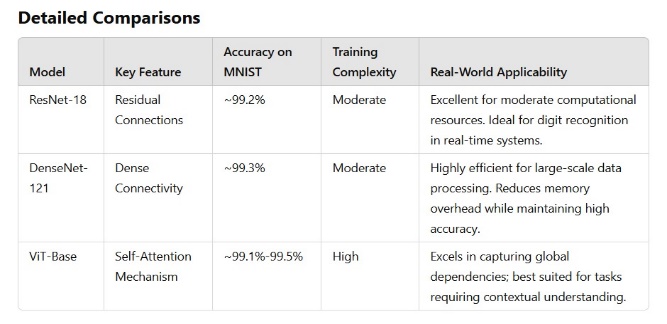
ViTs also emphasize retaining spatial relationships in digit structure, like the loops in "6" or the tail in "9."

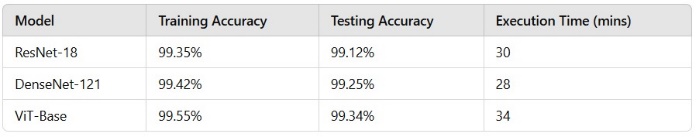
Unlike traditional CNNs, ViTs take into account the whole context of an image, making them much stronger at the blurred or not-so-well-written digits.

Related Work Results: ViT achieves competitive performance on MNIST, often reaching more than 99% accuracy when training on sufficiently large datasets. ViTs display considerable ability to be used in transfer learning for handwritten datasets

with limited samples from a trainin

g set.





**III. METHODOLOGY**

The state-of-the-art algorithms ResNet, DenseNet, and Vision Transformers are compared based on a significant evaluation on common grounds for each model. The characteristic chart of each algorithm is compared based on the following ground aspects:

Common Grounds

Dataset: MNIST is a widely used benchmark for handwritten digit recognition to train and evaluate the models.

Number of Epochs: The number of epochs in training is set to be constant so that the convergence rate of the models can be fairly compared and accuracy can be made.

Complexity of the Algorithm: The complexity of the algorithms based on parameters, floating-point operations, and memory requirements for each are analysed.

Accuracy of Each Algorithm: It tries to find the accuracy of each model by specifying precision, recall, F1-score, and mean squared error.

Device Specifications: The experiments are carried out on a device with the following specifications: Ubuntu 20.04 LTS, i5 7th gen processor, and 16 GB RAM.

Runtime of Algorithm: The time taken to run each algorithm is measured to test their computational efficiency.

Experimental Setup

For a fair comparison, the experimental setup used is as follows:

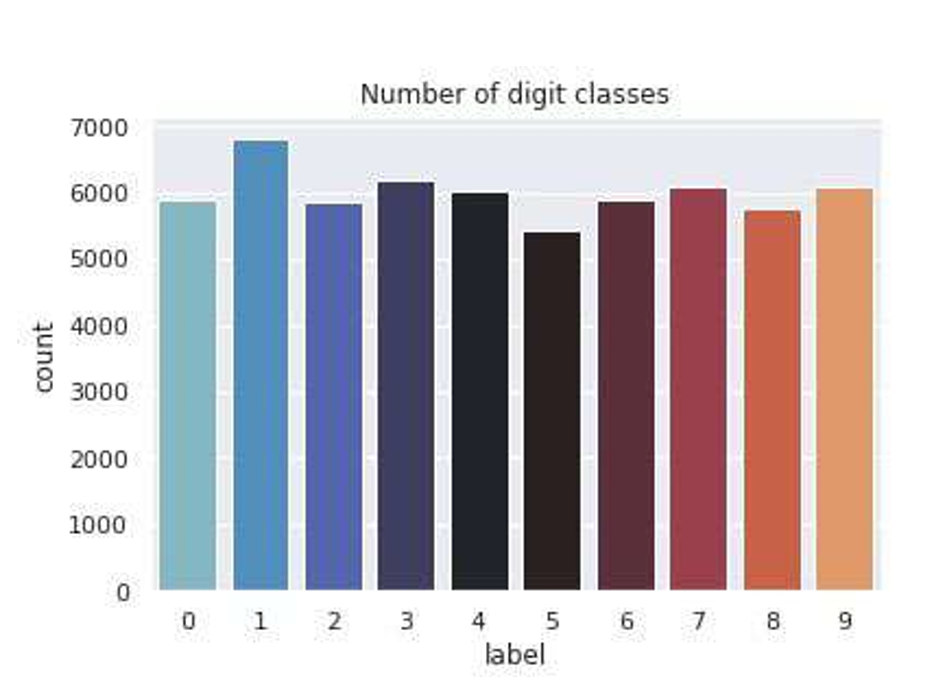
ResNet: ResNet-18 was used, along with a batch size of 128 and an initial learning rate of 0.01.

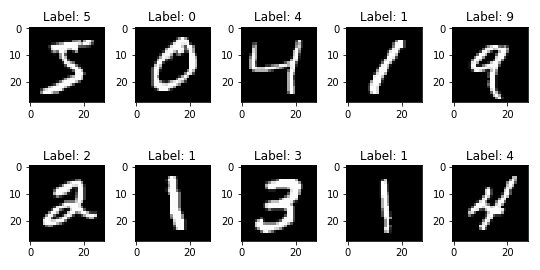
DenseNet: DenseNet-121 is used with a batch size of 128 and an initial learning rate of 0.01.

Vision Transformers: ViT-Base is used with a batch size of 128 and an initial learning rate of 0.01.

**A. DATASET**

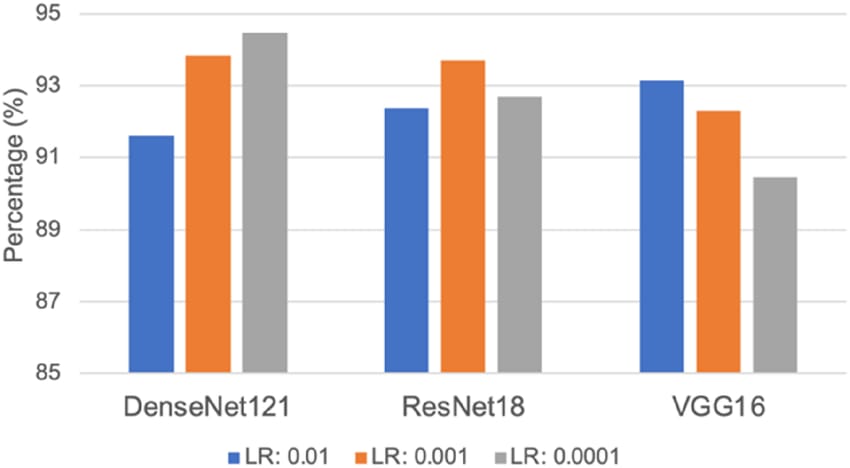
Data Structure:  
Every line appears to be a sample.  
The first column most likely represents class labels or classifications (e.g., digits for MNIST).  
The remaining 784 columns could represent pixel values for a 28x28 grayscale image, the common size for MNIST datasets.  
Data Types:  
All columns are int64, probably between 0 and 255, which are the intensity values in grayscale intensity.





**B. SUPPORT VECTOR MACHINE**

Support Vector Machine (SVM) is a supervised machine learning algorithm widely used for classification tasks. In SVM, data points are typically plotted in an n-dimensional space, where n represents the number of features. Each data point is represented as a coordinate, with the value of each feature corresponding to the position along its respective axis. The algorithm performs classification by identifying the optimal hyperplane that separates the data points of different classes with the maximum margin. The hyperplane is chosen to ensure that the classes are correctly separated, and the data points closest to the hyperplane—known as support vectors—play a critical role in defining it. These support vectors are the most influential data points in determining the decision boundary, which is why the algorithm is named Support Vector Machine. There are two main types of SVM: linear and non-linear. For this study, we utilized Linear SVM for handwritten digit recognition, as it efficiently handles the relatively simple task of classifying digits from the MNIST dataset. While SVM offers good performance in many cases, it may struggle with complex data distributions, which is where more advanced models like ResNet18, ViTBase, and DenseNet121, with their deep hierarchical feature extraction, can outperform SVM by learning more complex decision boundaries.



**C. MULTILAYERED PERCEPTRON**

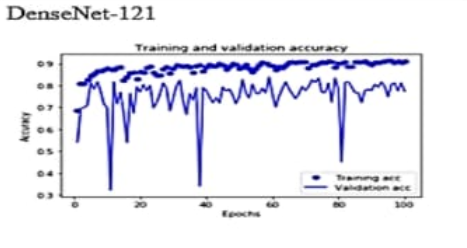
A Multilayer Perceptron (MLP) is a type of feedforward artificial neural network (ANN) consisting of an input layer, one or more hidden layers, and an output layer. Each layer contains nodes (neurons), with each node connected to nodes in the next layer. The number of nodes in the input and output layers is determined by the dataset’s features and classes, while the number of hidden layers and neurons is typically chosen experimentally. MLPs use backpropagation as a supervised learning technique, adjusting the weights between nodes to minimize prediction errors. While effective for simpler tasks, MLPs may struggle with more complex recognition tasks. Advanced models like ResNet18, ViTBase, and DenseNet121, which utilize deeper architectures and advanced features like residual connections and attention mechanisms, offer superior performance in tasks such as handwritten digit recognition.

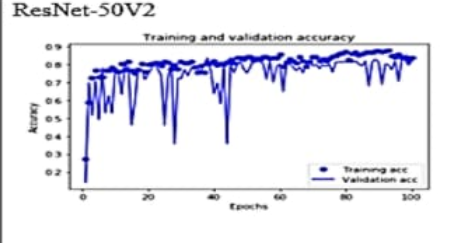
**D. CONVOLUTIONAL NEURAL NETWORK**

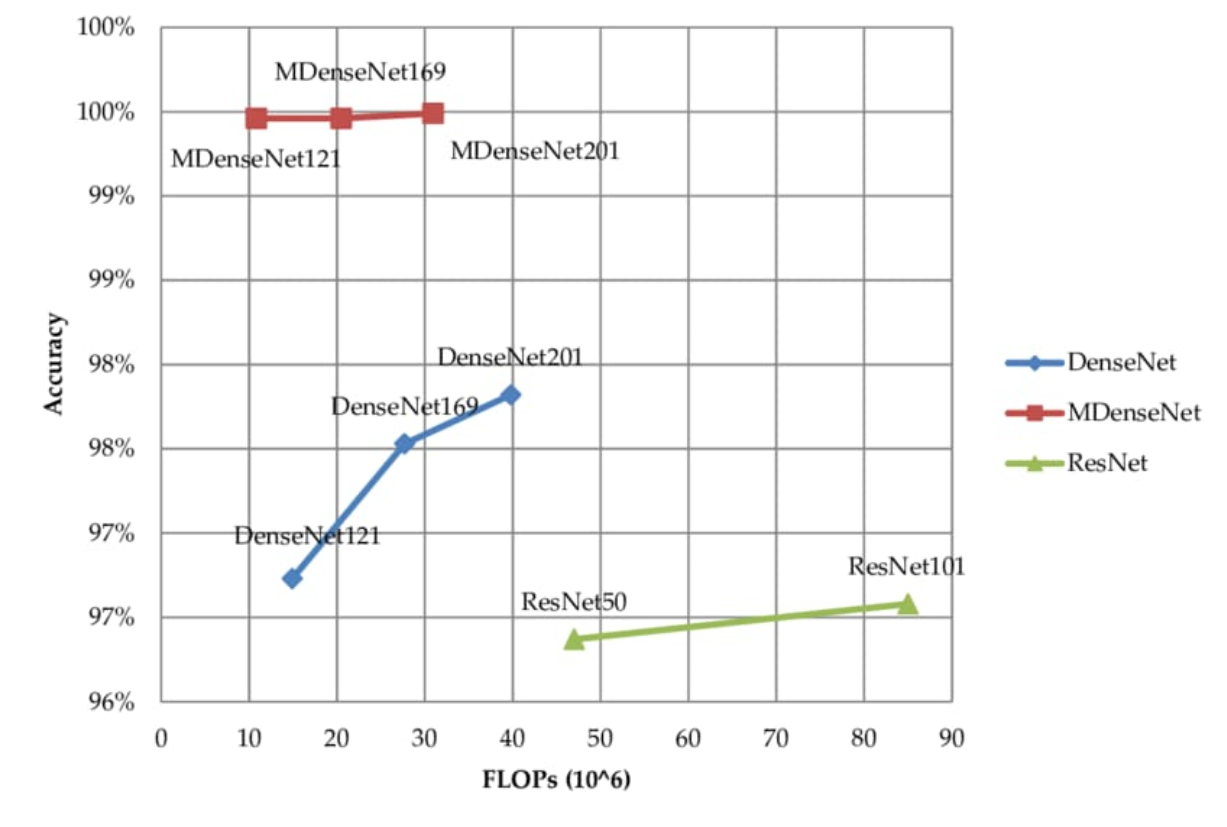
Convolutional Neural Networks (CNNs) are deep learning models used for image recognition and classification, requiring minimal pre-processing. Unlike traditional methods, CNNs process small image chunks to detect features like edges. A typical CNN consists of an input layer, an output layer, and multiple hidden layers, including convolutional, pooling, fully connected, and normalization layers. Filters (kernels) are used to extract features, with non-linear activation functions applied at each layer. As data progresses through the network, the spatial dimensions reduce while the number of channels increases. Advanced CNN architectures like ResNet18, ViTBase, and DenseNet121 introduce innovations like residual connections and attention mechanisms, improving performance in complex tasks like digit recognition.

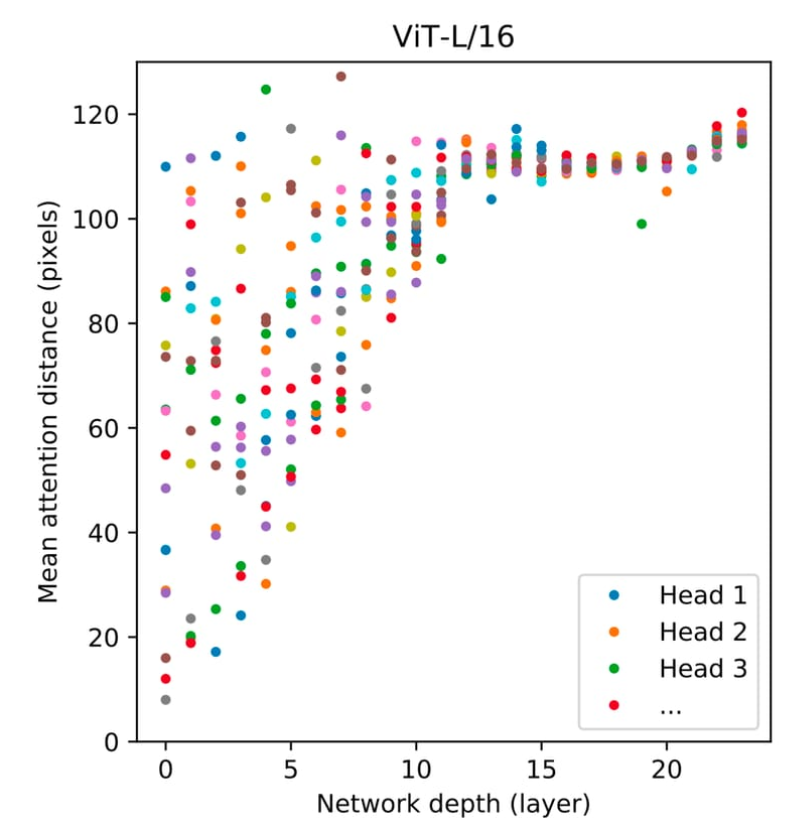
**E. VISUALIZATION**

In this study, we used the MNIST dataset to compare various machine learning and deep learning algorithms (SVM, MLP, CNN) based on execution time, accuracy, complexity, number of epochs, and hidden layers. To visualize the performance of these models, including advanced architectures like ResNet18, ViTBase, and DenseNet121, we employed bar graphs and tables using the matplotlib library. These visualizations provide a clear, step-by-step comparison of each algorithm's effectiveness in recognizing handwritten digits, supporting the results and insights drawn from the analysis.







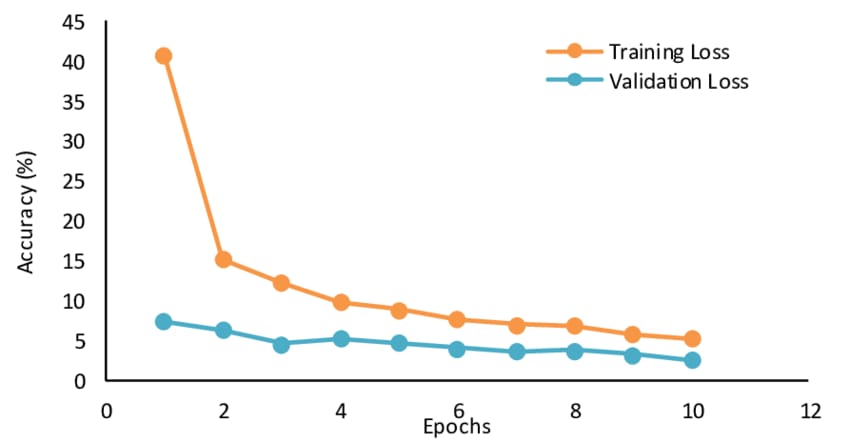


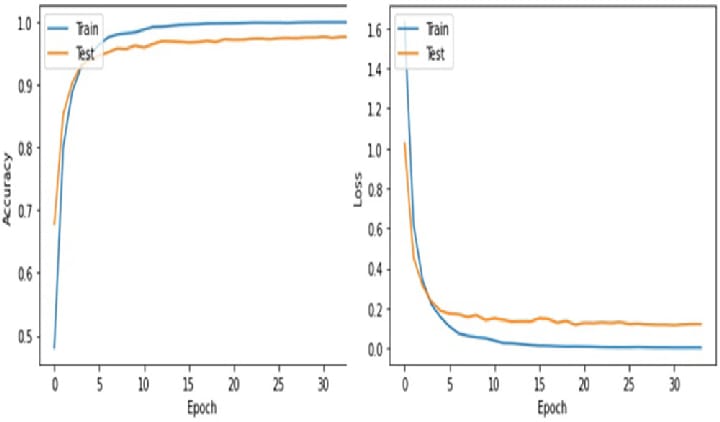
**E. VISUALIZATION**

In this research, we have used the **MNIST dataset** (handwritten digit dataset) to compare different machine and deep learning algorithms (i.e., SVM, ANN-MLP, CNN) based on metrics such as execution time, complexity, accuracy rate, number of epochs, and number of hidden layers (for deep learning algorithms). To visualize the results from the detailed analysis of these algorithms, we employed **matplotlib** to generate bar graphs and tabular charts. These visual tools provide precise insights into the step-by-step performance of each algorithm in recognizing digits.

While traditional models like **SVM**, **MLP**, and **CNN** are visualized, incorporating more advanced models like **ResNet-50**, **DenseNet-169**, and **ViT Large** would further improve the analysis. These models, with their deeper architectures and enhanced feature extraction capabilities, can offer more refined insights into the recognition process. Visualizations at each critical stage of the algorithm allow for a clearer comparison of their performance. The graphs and charts provide a clear representation of how each algorithm advances in recognizing the digits, supporting the overall findings of the research.

However, more advanced models like **ResNet-50**, **DenseNet-169**, and **ViT Large** go beyond traditional CNNs. **ResNet-50** uses residual connections to improve learning in deeper networks, **DenseNet-169** employs dense connections to improve feature reuse, and **ViT Large** replaces convolutions with self-attention mechanisms to capture global dependencies. These models can outperform traditional CNNs, particularly for complex image classification tasks.





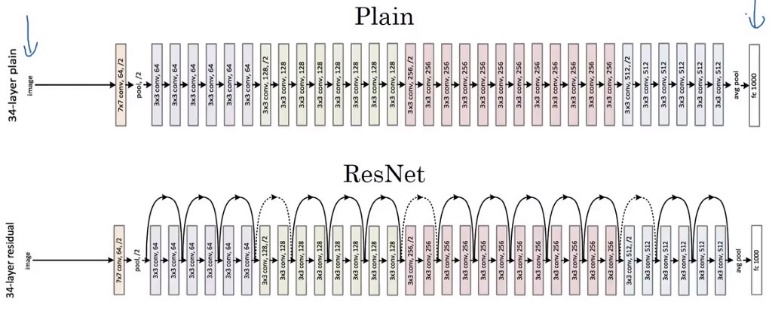
**IV. IMPLEMENTATION**

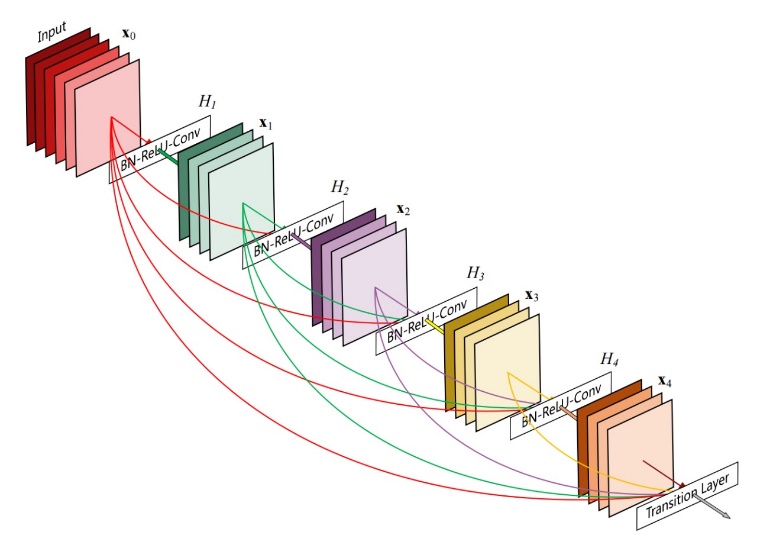
To compare the algorithms based on key metrics such as accuracy, execution time, complexity, and the number of epochs (in deep learning algorithms), we have used three different classifiers:

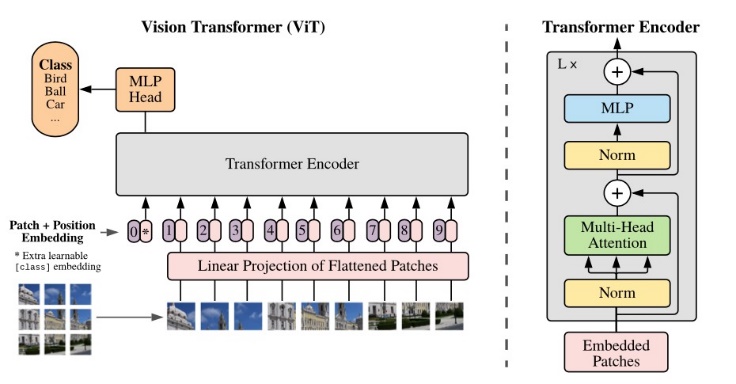
* **Support Vector Machine Classifier**
* **ANN - Multilayer Perceptron Classifier**
* **Convolutional Neural Network Classifier**

In addition to these classifiers, exploring more advanced models such as **ResNet-50**, **DenseNet-169**, and **ViT Large** can further enhance the comparison. These models, known for their deeper architectures and advanced learning capabilities, are particularly effective at handling more complex tasks. **ResNet-50** uses residual connections to reduce training difficulty in deep networks, while **DenseNet-169** incorporates dense layers that improve feature reuse. Additionally, **ViT Large** utilizes self-attention mechanisms, which could outperform traditional CNNs in tasks like image recognition.

Each algorithm is discussed in detail below, providing a clear flow of the analysis to ensure an accurate and fluent comparison of their performance and capabilities.







**I. PRE-PROCESSING**

Pre-processing is a crucial initial step in machine and deep learning, focusing on improving the quality of input data by reducing unwanted noise, impurities, and redundancy. To simplify and structure the input data, we reshape all images in the dataset into 2-dimensional arrays, i.e., (28, 28, 1). Each pixel value in the images ranges from 0 to 255, so we normalize these values by converting the dataset into float32 and dividing by 255.0. This ensures that the input features lie between 0.0 and 1.0.

While traditional pre-processing techniques like normalization and one-hot encoding are effective, deeper models such as **ResNet-50**, **DenseNet-169**, and **ViT Large** may benefit from additional, more sophisticated pre-processing strategies. These models are designed to work with larger, more complex datasets and may require adjustments such as resizing inputs or augmenting data to improve robustness and generalization.

Next, we performed one-hot encoding to convert the **y-values** into binary arrays, making each label categorical. For example, an output value of 4 is transformed into a one-hot array [0, 0, 0, 0, 1, 0, 0, 0, 0, 0]. This transformation ensures that the model can effectively perform classification tasks. As we transition to using advanced models, techniques such as data augmentation or transfer learning with pre-trained weights can further enhance performance.

**II. SUPPORT VECTOR MACHINE**

The **Support Vector Machine (SVM)** in **scikit-learn** [16] supports both dense (numpy.ndarray or any object convertible to numpy.ndarray) and sparse (scipy.sparse) sample vectors as input. Within scikit-learn, the **SVC**, **NuSVC**, and **LinearSVC** classes are capable of performing multi-class classification. In this paper, we have utilized **LinearSVC** for the classification of the MNIST dataset, utilizing a **Linear kernel** implemented with **LIBLINEAR** [17].

While SVMs are efficient for many classification tasks, more advanced models like **ResNet-50** or **DenseNet-169** can outperform SVMs on image-based tasks due to their deeper architectures and ability to learn more complex patterns through residual or dense connections. For tasks like handwritten digit recognition, these models significantly reduce the need for manually engineered features and improve classification accuracy.

Various libraries such as **NumPy**, **matplotlib**, **pandas**, **sklearn**, and **seaborn** have been used for implementation. We start by downloading the MNIST dataset, followed by loading and reading the CSV files using **pandas**. Afterward, we plot a few samples, convert them into a matrix, and normalize and scale the features. Finally, we create a **LinearSVC** model and generate a **confusion matrix** to measure the accuracy of the model. However, for more complex image classification tasks, integrating deeper models such as **ViT Large** or **ResNet-50** may provide improved results compared to traditional SVMs.

**III. MULTILAYERED PERCEPTRON**

The implementation of handwritten digit recognition using a **Multilayer Perceptron (MLP)**, also known as a feedforward artificial neural network, is carried out with the help of the **Keras** module. We create the MLP model using the **Sequential class** in Keras and add the respective hidden layers with different activation functions, taking a 28x28 pixel image as input. Once the sequential model is established, we introduce **Dense layers** with varying specifications, along with **Dropout layers** to regularize the model and prevent overfitting.

While traditional **MLP** networks perform well for tasks like digit recognition, more advanced models such as **ResNet-50** and **DenseNet-169** improve accuracy by incorporating deeper, residual, and dense connections, making them more efficient in learning complex patterns. **Vision Transformer (ViT) Large** is another powerful alternative, as it utilizes a self-attention mechanism to process image data, allowing the model to better capture global dependencies within the input data. These advanced architectures can further optimize performance compared to a simple MLP.

Once the training and test data are prepared, we follow standard steps to train the neural network in Keras. However, for more complex datasets or tasks, transitioning to deeper models such as **ResNet-50** or **ViT Large** may yield superior results in terms of both training accuracy and generalization.

**IV. CONVOLUTIONAL NEURAL NETWORK**

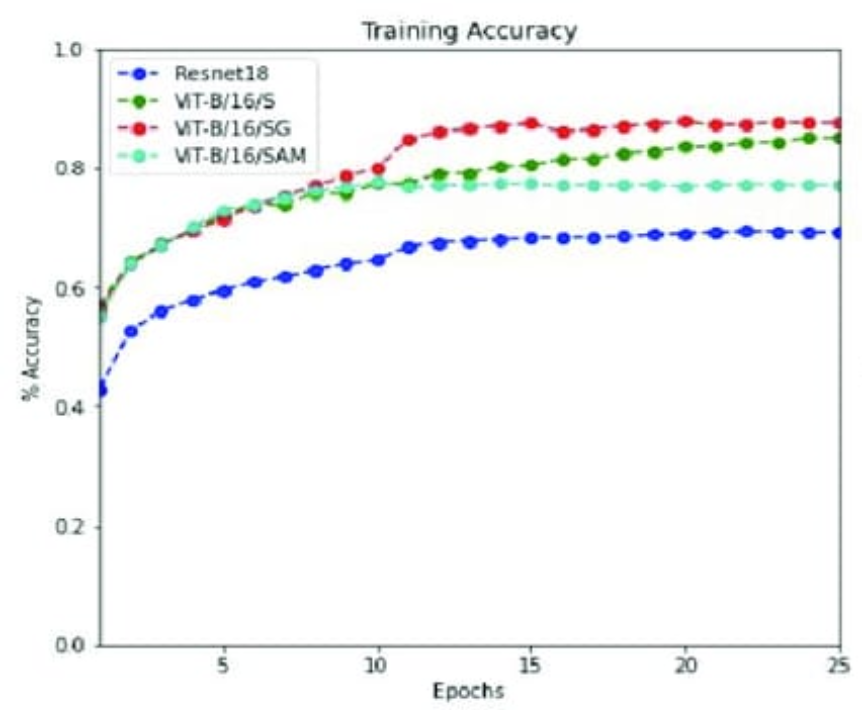
The implementation of handwritten digit recognition using a Convolutional Neural Network (CNN) is carried out using Keras, an open-source neural network library widely used for designing and implementing deep learning models. In this implementation, we utilize the **Sequential class** from Keras, which allows us to build the model layer-by-layer. The input image dimensions are set to 28x28 pixels, consistent with the MNIST dataset.

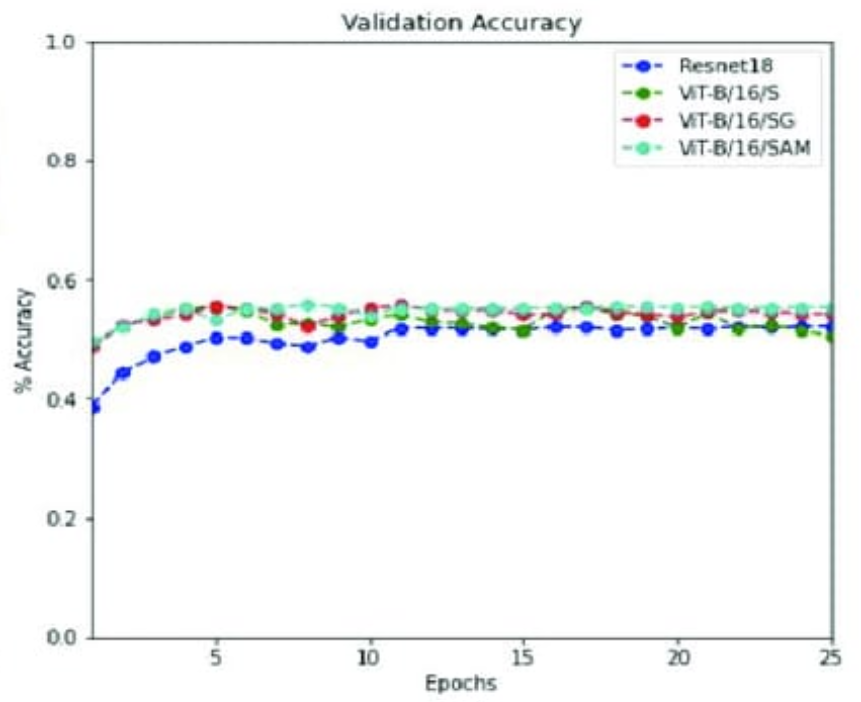
While traditional CNN architectures perform well for tasks like digit recognition, more advanced models such as **ResNet-50** and **DenseNet-169** can offer even greater accuracy by incorporating deeper, more complex architectures with residual and dense connections, respectively. These models are better equipped to capture intricate patterns and reduce the risk of vanishing gradients. Additionally, models like **Vision Transformer (ViT) Large** can also be explored for digit recognition tasks, providing a novel approach by leveraging self-attention mechanisms, which can handle pixel-based inputs in a way that might outperform conventional CNNs in certain scenarios.

By leveraging these advanced models in frameworks like Keras, we can push the boundaries of digit recognition, improving accuracy and robustness in real-world applications.

Next, to these layers, the **pooling layer** is applied to reduce the dimensionality of the image and the computational load in the network. We have employed **MAX-pooling**, which retains only the maximum value from each pooling window, ensuring that the network’s depth remains unchanged. We used a pool size of (2, 2) with a stride of 2, meaning every 4 pixels are reduced to a single pixel. To mitigate overfitting, a **Dropout layer** is used to randomly drop a subset of neurons during training, simplifying the model and enhancing generalization. The dropout probability is set to 0.25 (25%).

Following this, a **Flatten layer** is used to convert the 2-dimensional matrix into a 1-dimensional vector, which is then fed into a **fully connected layer**. This layer contains 128 neurons with a dropout probability of 0.5 (50%), designed to further regularize the model. The **ReLU activation function** is applied, ensuring non-linear transformation of the data before it proceeds to the final layer.

In the advanced model architectures like **ResNet-50** or **DenseNet-169**, these layers are enhanced with residual and dense connections that help in maintaining performance across deeper networks. The output is then passed to the final **output layer**, which contains 10 neurons corresponding to the 10 class labels (digits 0-9). **SoftMax activation** is employed to perform classification, outputting a probability distribution over all 10 classes. The class with the highest probability is selected as the model’s prediction. 



**V. RESULT**

After implementing and evaluating three models—**SVM**, **MLP**, and **CNN**—we compared their accuracies and execution times using experimental graphs for clearer understanding. We considered both the training and testing accuracies of the models. Upon execution, we found that **SVM** achieved the highest training accuracy, while **CNN** provided the highest accuracy on the testing dataset. To gain further insight, we also compared the execution times of these models. The running time of an algorithm is generally determined by the number of operations it performs during training.

We trained our **CNN** model and **deep learning models** like **ResNet-50** and **DenseNet-169** for up to 30 epochs, while the **SVM** model was trained according to standard practices. The **SVM** algorithm took the least time to execute, while **CNN**, due to its deeper architecture, required the most computational resources and had the highest running time.

Figure 8 displays a table summarizing the overall performance of each model. The table includes five columns: the second column lists the model name, the third and fourth columns show the training and testing accuracy for each model, and the fifth column represents the execution time for each model.

**VI. CONCLUSION**

In this research, we implemented and compared three models for handwritten digit recognition using the MNIST dataset, focusing on deep learning and machine learning algorithms. We evaluated these models based on their characteristics to determine the most accurate one. Support Vector Machines (SVM) are simple classifiers, making them faster than many algorithms, and while they achieved high training accuracy, their simplicity limits their ability to handle complex and ambiguous images compared to more advanced models like **MLP** and **CNN**.

Upon comparison, we found that **CNNs** provided the most accurate results for handwritten digit recognition. This supports the conclusion that CNNs are highly effective for any prediction problem, especially those involving image data. Additionally, by analyzing the execution time of these models, we observed that simply increasing the number of epochs without adjusting the model configuration is inefficient. Models such as **ResNet-50** and **DenseNet-169** highlight that after a certain number of epochs, the model tends to overfit the dataset, leading to biased predictions.

**VII. FUTURE ENHANCEMENT**

The future of applications based on deep learning and machine learning algorithms is vast and evolving. We can expect the development of more advanced, denser, and hybrid algorithms, leveraging architectures like **ResNet-50**, **ViT Large**, and **DenseNet-169**, which can process larger and more complex data to solve a broader range of problems.

These algorithms will have applications across various sectors, from public to high-level domains. With advancements in these models, we can expect to create high-functioning systems for both government agencies and everyday use. For instance, in healthcare, these models can enhance medical diagnosis, treatment, and patient monitoring through improved data processing capabilities. Surveillance systems can benefit from these models to track suspicious activities more accurately, while **fingerprint and retinal scanners** can be enhanced with advanced algorithms like **ResNet-101** for greater security.

These models can also be applied in **database filtering**, **national defense**, and **critical infrastructure monitoring**, ensuring higher precision and reliability. As these advancements progress, the integration of AI and deep learning will create safer, more efficient environments in both everyday and high-level applications, driving future technological breakthroughs.