ISSN: 2182-2069 / E-ISSN: 2182-2077

# Performance Evaluation of Transfer Learning VGG16 in Handwritten Text Using Word Beam Search and Language Model

K.P. Kavitha[1*], and Dr.V.B. Kirubanand[2]

[1*]Research Scholar, Department of Computer Science, CHRIST (Deemed to be University), Bengaluru, India. kp.kavitha@res.christuniversity.in, https://orcid.org/0009-0000-8528-7845

[2]Faculty, Department of Computer Science, CHRIST (Deemed to be University), Bengaluru, India. kirubanand.vb@christuniversity.in, https://orcid.org/0000-0003-0792-548X

Received: February 15, 2025; Revised: March 26, 2025; Accepted: May 06, 2025; Published: May 30, 2025

## Abstract

This study evaluates the performance of transfer learning using the VGG16 model for handwritten text recognition, integrating Word Beam Search decoding and language modeling techniques. The VGG16 model, pre-trained on large-scale datasets, serves as a feature extractor for handwritten text images, capturing intricate patterns and structures inherent in handwriting. To convert these visual features into textual information, the system employs a Recurrent Neural Network (RNN) trained with the Connectionist Temporal Classification (CTC) loss function, producing a matrix of character probabilities for each time-step. The Word Beam Search algorithm is utilized for decoding these probabilities into coherent text, effectively constructing recognized text by referencing a predefined dictionary and addressing challenges such as arbitrary character strings and varying handwriting styles. The integration of language models incorporates context which further sharpens the output and improves precision and trustworthiness of recognition systems. Experimental results demonstrate that this combined approach significantly improves recognition performance, highlighting the efficacy of transfer learning and advanced decoding strategies in handwritten text recognition. This involves analyzing its effectiveness across various datasets. Transfer learning leverages pre-trained models, like VGG16, to address challenges such as limited labeled data and extensive training times.

**Key Considerations:**

**Dataset Characteristics:** The diversity and size of datasets impact the accuracy.

**Model Adaptation:** Modifying VGG16's architecture to suit specific datasets can enhance recognition accuracy.

**Performance Metrics:** Evaluations should consider accuracy, computational efficiency, and the model's ability to generalize across different handwriting styles. In summary, assessing VGG16's transfer learning performance in handwritten text recognition requires a comprehensive analysis of dataset properties, model adjustments, and evaluation metrics to achieve optimal results.

**Keywords:** VGG, RNN, CTC, HTR, CNN, NN, LM, FSM.

*Journal of Internet Services and Information Security (JISIS)*, volume: 15, number: 2 (May), pp. 536-551.
DOI: *10.58346/JISIS.2025.I2.037*

*Corresponding author: Research Scholar, Department of Computer Science, CHRIST (Deemed to be University), Bengaluru, India

# 1 Introduction

Research in the areas of computer vision and artificial intelligence has focused heavily on handwritten text recognition (HTR). It involves converting handwritten documents, notes, and manuscripts into machine-readable text. Striving through OCR systems, OCR faces variations, distortion, and inconsistencies of handwriting styles, making the task cumbersome. Deep learning, however, has greatly boosted the precision of recognizing handwritten texts.

In this research, This study primarily aims to develop a method for handwritten word recognition using feature extraction using the VGG16 model, a (DCNN) Indriyani et al., (2023). The deep architecture and capability to capture hierarchical features of VGG16 make it a popular choice for image processing Trivedi et al. (2023).

Using VGG16, we can pull out important patterns from images of handwritten text, which makes recognition more accurate.

Additionally, language models and Word Beam Search decoding techniques are integrated to improve text prediction. These models refine character predictions by incorporating linguistic probabilities, reducing errors, and ensuring a more accurate reconstruction of handwritten text.

This study aims to explore the efficiency of this VGG16 model in HTR, optimize feature extraction techniques, and improve recognition accuracy through deep learning and language models. The results of this research can be applied to digitizing historical manuscripts, automating document processing, and developing intelligent handwriting recognition systems.

This study contributes by evaluating VGG-16's accuracy and performance on sizable, well-known datasets Carter and Heinriksen (2023). The findings will aid in the more precise detection of handwritten text indicators.

# 2 Methods

The following Four steps comprised the research framework: identifying the mortgage datasets, preprocessing the images, training, and evaluating the VGG16 algorithm., as shown in Figure 1.

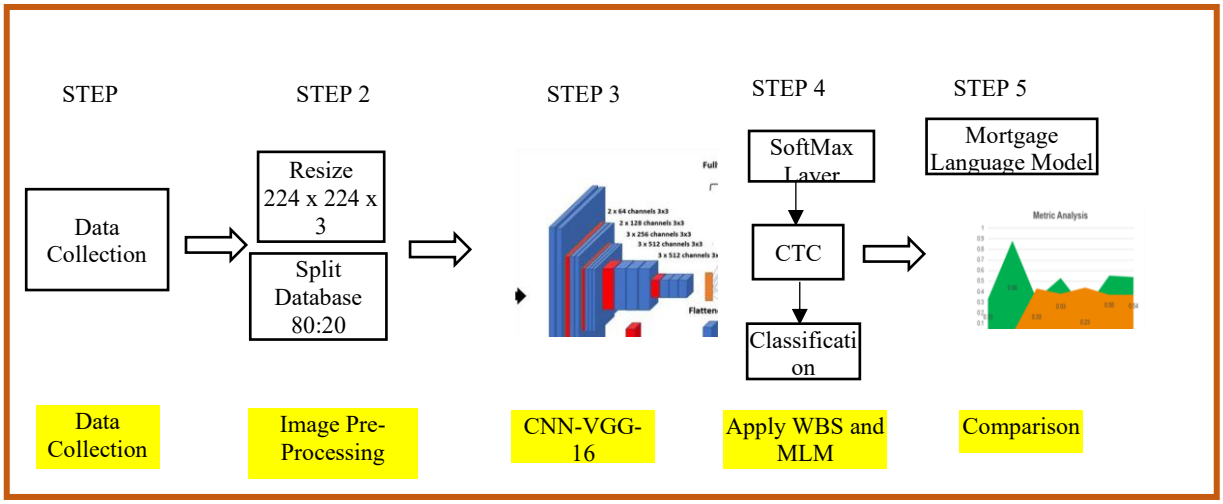


Figure 1: The Proposed Model Framework

**Data Collection**

The effectiveness of any deep learning model, including VGG16 for handwritten text recognition, its closely related to quality and diversity for the dataset used by training and validation. For this study, a custom dataset was created that included mortgage-associated handwritten documents so as to test the performance of the VGG16 model. This database comprises of scanned mortgage forms, loan agreements, financial records, and handwritten notes commonly found in banking and financial services.

**1. Data Sources**

- The mortgage images were sourced from:
- Public mortgage datasets are available through open-source repositories and research institutions.
- Financial institutions' anonymized records, ensuring privacy and compliance with legal regulations.
- Manually collected handwritten mortgage documents, created and digitized to simulate real-world variations in handwriting.

**2. Image Acquisition & Preprocessing**

- To ensure consistency and accuracy, the collected images underwent pre-processing, including:
- Grayscale conversion: Standardizing all images to grayscale for uniform feature extraction.
- Noise removal: Applying denoising filters to remove artifacts, smudges, and distortions.
- Resizing: Adjusting images to a fixed resolution of 224×224 pixels, compatible with the VGG16 model.
- Binarization: Converting images to black-and-white format to enhance contrast and improve text extraction.

**3. Annotation and Labeling**

- Each mortgage document was labeled with the corresponding text transcription to train the deep learning model effectively. The labels included:
- Loan numbers, names, and financial details extracted from handwritten fields.
- Handwritten signatures and numerical values converted into structured text format.
- Character-wise segmentation for training character recognition models.

**4. Dataset Splitting**

- The dataset was divided into: training set (80%) – Used for training the VGG16 model with diverse handwriting samples. (Figure 2)
- Validation set (10%) – Used for fine-tuning and hyperparameter optimization.
- Testing set (10%) – Used for final evaluation and performance benchmarking.

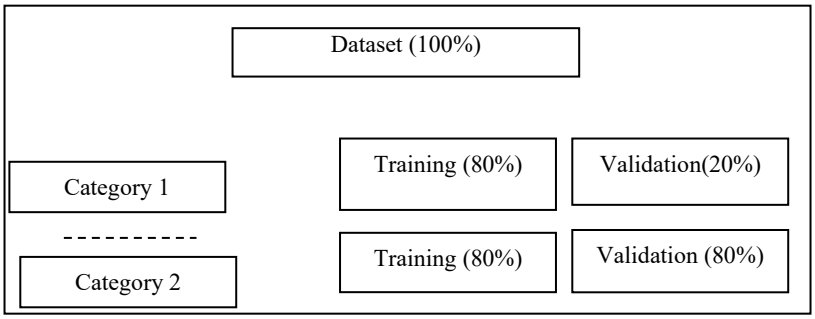


Figure 2: Distribution of Data Set

## 5. Challenges in Data Collection

- Handwriting variations: Different writing styles, slants, and irregularities required robust preprocessing.
- Noisy and low-quality scans: Older mortgage documents had faded ink, creases, and artifacts.
- Class imbalance: Some handwritten fields appeared more frequently than others, requiring data augmentation techniques.

### Application of the CNN

CNN with the field of mainframe vision and approach of revolutionary pattern related to the fields of processor vision and pattern recognition. Its an ability for automatically learn through various features followed by the input images specifically thorough the image task. Below are several key applications of CNNs, including specific use cases in handwritten text recognition and beyond. Digit and character recognition (e.g., MNIST, IAM datasets). Automatic form reading (e.g., bank cheques, exam papers, mortgage forms). Document digitization using models like VGG16 and CRNNs. Postal address or zip code detection on envelopes

CNNs are trained to classify images into predefined categories, powering: Object detection (e.g., cats vs. dogs, medical diagnosis) Scene understanding (e.g., urban vs. rural) and Facial recognition systems. CNNs combined with techniques like Region-based CNNs (R-CNN) or YOLO are used in: Self-driving cars (detecting pedestrians, traffic signs), Video surveillance and security and Augmented reality applications.

Its is an important steps to develop the CNN algorithm is helps fot to involved the various of use of pooling and convolution networks.

Convolution Operations: The cornerstone operation of a CNN is convolution which consists of passing a small matrix known as the kernel or filter over the image to produce a feature map. Important picture features, such as edges, textures, and patterns, can be removed with the use of this method. It is possible to capture a wide range of features by using numerous filters, each of which learns to recognize a different set of features during training. Comparing fully connected networks to convolution networks, fully connected networks have a lot more factors and are harder to compute.

Pooling Operations: Pooling is the down sampling operation for to reduce the three-dimensional dimension contains the feature maps contains the important informations. Maximizing the value from a

specified window of the feature map is the most popular pooling method. This helps prevent overfitting and makes the model robust against input image distortions and minor translations.

Combined Impact: Convolution along with pooling operations forms the backbone of CNNs. While convolution layers fetch certain characteristics of data, pooling layers condense the extracted features, allowing deeper networks to learn more abstract representations of the data. As seen, it is extremely important for image classification, object detection, and recognizing handwritten text where the spatial hierarchies of features need to be learned efficiently.

In this case, to extract features (edges, curves, strokes) from the handwritten text image first when we input a handwritten image (like a scanned document) into VGG16, the first convolution layers detect basic shapes — like straight lines and edges. Pooling helps by focusing on the most prominent features of handwriting. It also makes the model robust to minor variations, such as changes in pen pressure, slant, or spacing. It uses many convolutions and pooling layers in sequence. These layers work together to transform a raw handwritten image into a meaningful high-dimensional vector (feature map). The feature vector is then passed to a classifier or a sequence model (like an RNN or Word Beam Search Decoder) to produce text output.

By removing only the most significant features, convolution and pooling techniques simplify the situation.. For instance, using many convolution layers makes it simple to compress an input signal with 75,000 features to 512 features. Over the past ten years, a number of CNN-based designs have been put out. In order to address a basic handwritten digit recognition problem, the Lenet-5 topology had been the first to suggest a CNN-based architecture Zhang et al. (1990).

The earlier ideas of deep neural networks and back-propagation laid the work's foundation. The availability of a sizable, tagged image dataset known as Image net allowed CNN-based algorithms to make a significant advancement Sutskever et al. (2014). The dataset, which was started in 2009 by a Stanford University artificial intelligence team, now has about 14 million categorized images. In 2014, the Image net Large Scale Visual Recognition Challenge was won by the second-most famous CNN architecture, Alex net Krizhevsky et al. (2012). Originally suggested by Simonyan et al. Simonyan and Zisserman (2015), the VGG16 model won first place within the Image completely Large Scale Visual for localizing objects and placed second for object classification.

ILSVRC 2014: Recognition Challenge. Inception Net Szegedy et al. (2017), ResNet He et al. (2016), Inception-v4 and Inception-Res Net Szegedy et al. (2017), Mobile Net Abbas et al. (2022), Mobile Net V2. Sandler et al. (2018). Efficient Net Tan and Le (2019), and Xception Net Chollet (2017) are just a few of the additional CNN architectures that have been put forth since then. Two essential parts make up the CNN architecture's building block: the pooling layer and the convolution layer. The algorithm's performance can be enhanced by adjusting the sieve size, stuffing, pace, beginning function, and layer connections Vardhan and Bhattacharya (2025). Prior to applying the classification task, the raw visual signal must be transformed into a simpler representation in order to enhance the algorithm's performance. Many algorithms can be trained and their effectiveness evaluated with the abundance of images available in the Image net database.

Most of the researcher should be evaluate the concern architecture, which is helps for to established the weight parameters with after training used by images. Followed by the computer vision, transfer learning used for various methods, its for to adopt the knowledge acquired through training among the network followed by the comparable situations Nayak and Raghatate (2024). In ref Abdel-Jaber et al. (2022) describes about, top layer should be modified through output consists of to reduce from 1000 to

pre trained network Geng (2024). Which is initially noted as the categorised the 1000 classes utilized by the binary classifications.

Additionally, all or part of the network layers' weight parameters must be adjusted. The pre-trained weight that was used to solve the prior problem, however, is where the learning process starts rather than from blank. Large datasets like Image Net were used to train the majority of model architectures, including VGG16, Inception Net, mobile net, and Xception Net. Large datasets like Image Net were used to train many architectures, including VGG16, Inception Net, mobile net, and Xception Net. A better model was created as an outcome of the significant computational expense associated with training. Abdel-Jaber et al. (2022) used information from the Image net Deng et al. (2009) database to make a thorough examination of how well pre-trained models performed on computer vision tasks. The transfer strategy is widely used in computer vision because it makes it possible to produce accurate models more quickly Vaila et al. (2019).

A CNN classification task typically comprises two segments: a classifier and a feature producer. The input of feature generator is takes the input raw images are should performed as stack convolution followed by layers and pooling. The main objective of this research focused on image which is contains the large amount of data. Which is consists of vector machines and decision tree classifiers. Other option, which is directly connected to other layers followed by the feature generators. Support vector machines or decision tree classifiers can achieve this objective. An alternative approach is to append fully connected layers on top of the feature generator. A layer is deemed fully connected if all the activations from the layer beneath are linked to each of its neurons; that is, every neuron in the layer immediately below. Within the fully connected layer, an optimization range was established by the researcher, and a number of layers did exist which he could manually optimize. Concerning the whole network’s overfitting problem, attention must be given to the breadth of the fully connected layers.

With deep learning techniques, the model determines the most relevant structures of the contribution data by itself during the model training phase. Unlike classical AI systems which extract features from an image based on the input and the intended classification, deep learning models, in contrast, start from the CNN-based feature generator and modify weight parameters to construct hierarchical features. The network collects the input data; this data serves as the input for the classifier. A specific weight parameter value will capture a particular problem domain pattern in the Problem domain. These values can be transferred into a different problem domain using transfer learning. A pre-trained model is used by the transfer learning process.

Finally, we decide what features to use based on the dataset's size and how similar it is to the one that was already learned. From comparing the pre-trained dataset with our unique dataset, any of these four transfer learning problems indicated below could develop where the problem definition dataset would be: 1) sizable but not exactly like the pre-trained set 2) sizable and quite similar to the previously used dataset 3) tiny but very similar to the past trained dataset 4) tiny and unlike the pre-trained set-up Within the framework of deep learning, a 1000 image collection including labels for every class is deemed small.

Scheduling a subset of problems within a pre-trained dataset is called dataset similarity. For instance, if one wants to extract features of dogs and birds using a pre-trained network trained on the Imagenet dataset, the dataset is said to be somewhat comparable as the Imagenet dataset comprises classes for both dogs and birds. This being said, we nonetheless decided to use the mortgage dataset for classification in this study. This let us come across our transfer learning issue resulting from a small dataset different from the pre-training set. This work used a pre-trained CNN with sixteen convolutional layers, VGG16. Ioffe and Szegedy (2015).

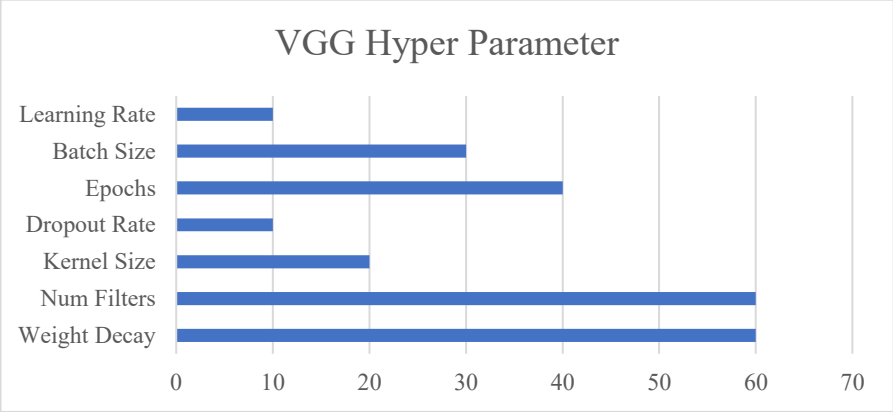


Figure 3: The Study was Conducted Using the Research Framework

This study compares and evaluates the effectiveness of different architectures in identifying Mortgage related handwritten text datasets. The thorough assessment was conducted using the VGG16 architecture's accuracy on numerous datasets using the hyperparameters referenced in Figure 3.

- **Learning Rate (LR):** Controls how much the weights update during training.
- Smaller values (e.g., 0.0001) prevent overfitting in fine –tuning. Controls the step size for optimization.
- **Batch Size:** Number of samples per training batch. 16 (Denotes how many images are processed in parallel) Commonly set to 16, 32, or 64.
- **Number of Epochs:** The number how many times the network processes the complete dataset.
- **Dropout Rate:** Probability of neurons being dropped during training to prevent overfitting.
- **Optimizer:** Algorithm used for weight updates (e.g., Adam, SGD).
- **Kernel Size:** The size of convolutional filters (usually 3x3 in VGG16).
- **Number of Filters:** The number of convolutional filters per layer.
- **Weight Decay (L2 Regularization):** Prevents overfitting by penalizing large weights**.**

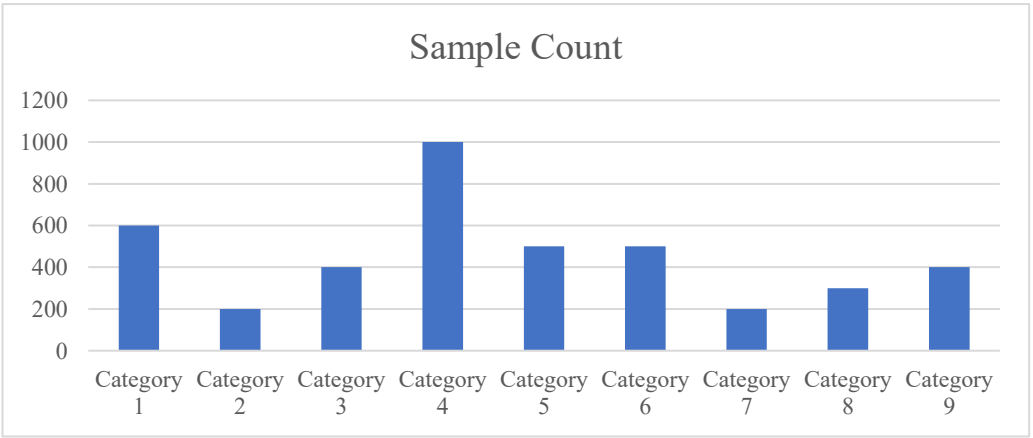


Figure 4: Distribution of Different Data Set

In Figure 4, we can see how multiple data sets are distributed in relation to different categories. In the case of the modified class count of the dataset, the multiclass confusion matrix was utilized to assess the robustness of VGG16. The experimental results in every case is presented in a confusion matrix that incorporates n categories which is presented in Table I.

Table 1: Multiclass Confusion Matrix

| | | Predicted Number | | | |
|---|---|---|---|---|---|
| | | Class 1 | class 2 | …... | class n |
| Actual number | class 1 | $x_{11}$ | $x_{12}$ | … | $x_{1n}$ |
| | class 2 | $x_{12}$ | $x_{22}$ | … | $x_{2n}$ |
| | …... | … | …. | … | …. |
| | class n | $x_{n1}$ | $x_{n2}$ | … | $x_{nn}$ |

Moreover, applying the calculations from equations 1 through 4, I derived the total numbers of (TFN), (TFP), (TTN), (TTP) with regard to class I for assessing the performance metrics of the VGG16 model.

$$TFN_i = \sum_{j \neq i}^{n} xij \quad (1)$$

$$TFP_i = \sum_{j=1}^{n} xij \quad (2)$$

$$TTN_i = \sum_{j=1}^{n} \sum_{k=1}^{n} xjk \quad (3)$$

$$TTP_i = \sum_{j=1}^{n} xij \quad (4)$$

Additionally, each class I's precision (P), recall (R), and specificity (S) were determined using equations 5, 6, and 7. Equations 8 and 9 demonstrate the computation of the overall accuracy and F1-Score, respectively.

$$P_i = \frac{TTP_{all}}{TTP_{all} + TFP_i} \quad (5)$$

$$R_i = \frac{TTP_{all}}{TTP_{all} + TFN_i} \quad (6)$$

$$S_i = \frac{TTN_{all}}{TTN_{all} + TFP_i} \quad (7)$$

$$Accuracy = \frac{TTP_{all}}{Total\ Number\ of\ Testing} \quad (8)$$

$$F_1 - Score = \frac{TTP_{all}}{Total\ Number\ of\ Testing} \quad (9)$$

## 3 Result and Discussion

This work evaluated the performance of the VGG16 transfer learning model in recognizing handwritten text across nine categories of mortgage-related words. Each More than 3000 images were in the database, and 2,181 of them were assessed in total. The datasets were separated based on how many classifiers and photos they contained. Subsequently, the VGG16 model's resilience was evaluated.

1) with over a thousand images.
2) With fewer than a thousand images
3) With an altered class count

The performance of the CCN transfer learning model, VGG16, in handwritten text recognition was evaluated in this work using nine categories of mortgage words. Each more than 3000 images were in the database, and 2,181 of them were assessed in total. The datasets were separated based on how many classifiers and photos they contained. Subsequently, the VGG16 model's resilience was evaluated.

1) with over a thousand images.
2) with fewer than a thousand images.
3) with an altered class count.

Table 2: Relationship between Category and Class

| Data | category Name | Acc | Pres | Res | F1-score |
|---|---|---|---|---|---|
| D1 | Borrower-class 4 | 93.87 | 90.9 | 88.7 | 89.82 |
| D2 | Investor-class 3 | 94.84 | 94.5 | 94.7 | 94.82 |
| D3 | Lendor- class 4 | 80.1 | 80.8 | 79.9 | 90.34 |
| D4 | Mortgage Title-class 3 | 94.87 | 95.2 | 89.8 | 92.41 |
| D5 | Mortgage Title-class 2 | 97.99 | 97.7 | 97.3 | 97.47 |
| D6 | Loan Type-class 3 | 92.67 | 89.4 | 85.8 | 87.55 |
| D7 | Loan Type-class 2 | 97.68 | 97.2 | 97 | 97.22 |
| D8 | Type of credit- class 4 | 88.3 | 86 | 66 | 74.7 |
| D9 | Occupancy- class 3 | 93.09 | 93.7 | 94 | 93.86 |

Table. 2 demonstrates relationship between the performance with the terms of accuracy, recall, f1 score, precision. The algorithm's accuracy was between 80.10% and 97.99%. When compared to datasets with three categories, the algorithm did better on datasets with two categories.

Table 3: Various Categories and Achieved Accuracy Precision, Recall and F-score

| **Data** | **Category Name** | **Acc.** | **Pres.** | **Rec.** | **F1-Score** |
|---|---|---|---|---|---|
| D1 | Borrower – class 4 | 93.87 | 90.9 | 88.7 | 89.82 |
| D2 | Investor – class 3 | 94.84 | 94.5 | 94.7 | 94.82 |
| D3 | Lendors – class 4 | 80.1 | 80.8 | 79.9 | 80.34 |
| D4 | Mortgage Title – class 3 | 94.87 | 95.2 | 89.8 | 92.41 |
| D5 | Mortgage Bank –class 2 | 97.99 | 97.7 | 97.3 | 97.47 |
| D6 | Loan Type – class 3 | 92.67 | 89.4 | 85.8 | 87.55 |
| D7 | Loan Purpose – class 2 | 97.68 | 97.2 | 97 | 97.22 |
| D8 | Type of Credit – class 4 | 88.3 | 86 | 66 | 74.7 |
| D9 | Occupancy – class 3 | 93.09 | 93.7 | 94 | 93.86 |

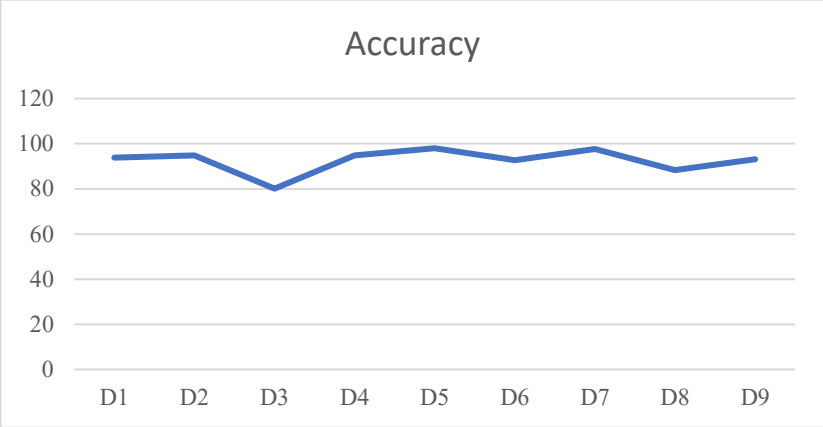


Figure 5: Relationship Between the Different Categories in the Dataset Vs Accuracy of the Algorithm

Table 4: Relationship between Category and Class

| **Data** | **Category Name** | **Acc.** | **Pres.** | **Rec.** | **F1-Score** |
|---|---|---|---|---|---|
| D1 | Borrower – class 4 | 93.87 | 90.92 | 88.72 | 89.82 |
| D2 | Investor – class 3 | 94.84 | 94.48 | 94.68 | 94.82 |
| D3 | Lendors – class 4 | 80.10 | 80.78 | 79.91 | 80.34 |
| D4 | Mortgage Title – class 3 | 94.87 | 95.17 | 89.81 | 92.41 |
| D5 | Mortgage Bank –class 2 | 97.99 | 97.65 | 97.29 | 97.47 |
| D6 | Loan Type – class 3 | 92.67 | 89.43 | 85.75 | 87.55 |
| D7 | Loan Purpose – class 2 | 97.68 | 97.21 | 97.02 | 97.22 |
| D8 | Type of Credit – class 4 | 88.30 | 85.98 | 66.03 | 74.7 |
| D9 | Occupancy – class 3 | 93.09 | 93.69 | 94.03 | 93.86 |

Table. 4 demonstrates the accuracy metric of the data set. The algorithm's accuracy was between 80.10% and 97.99%. When compared to datasets with three categories, the algorithm did better on datasets with two categories.

Table 5: Various Categories and Achieved Accuracy Precision, Recall and F-score

| Data | category Name | Acc | Pres | Res | F1-score |
|---|---|---|---|---|---|
| D1 | Borrower-class 4 | 93.87 | 90.9 | 88.7 | 89.82 |
| D2 | Investor-class 3 | 94.84 | 94.5 | 94.7 | 94.82 |
| D3 | Lendor- class 4 | 80.1 | 80.8 | 79.9 | 90.34 |
| D4 | Mortgage Title-class 3 | 94.87 | 95.2 | 89.8 | 92.41 |
| D5 | Mortgage Title-class 2 | 97.99 | 97.7 | 97.3 | 97.47 |
| D6 | Loan Type-class 3 | 92.67 | 89.4 | 85.8 | 87.55 |
| D7 | Loan Type-class 2 | 97.68 | 97.2 | 97 | 97.22 |
| D8 | Type of credit- class 4 | 88.3 | 86 | 66 | 74.7 |
| D9 | Occupancy- class 3 | 93.09 | 93.7 | 94 | 93.86 |

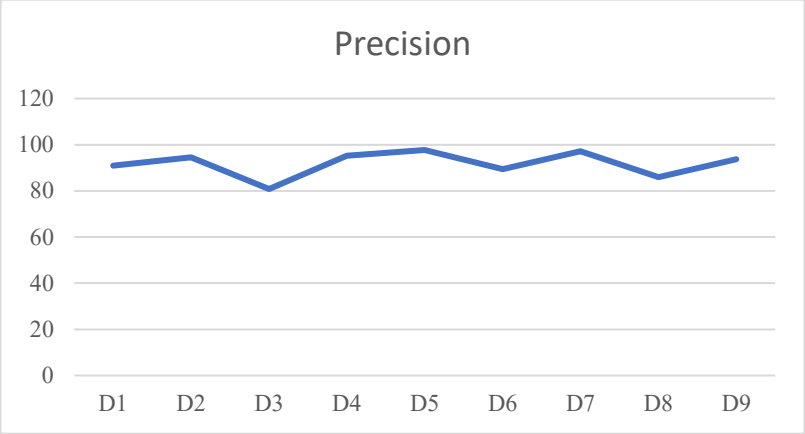


Figure 6: Relationship between the Different Categories in the Dataset and the Precision of the Algorithm

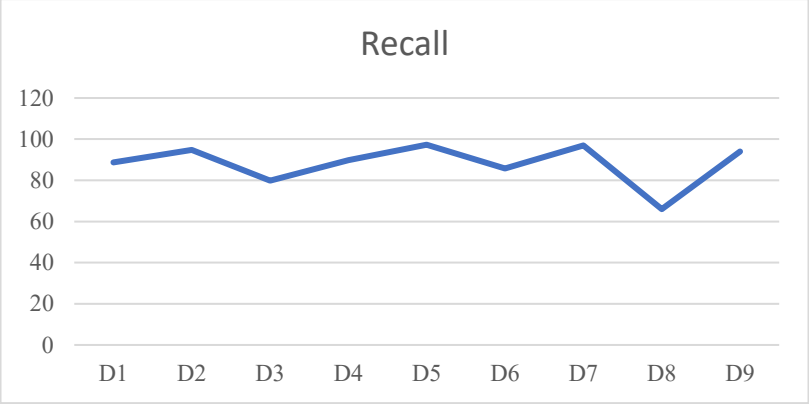


Figure 7: Relationship Between the Different Categories in the Dataset and the Recall of the Algorithm

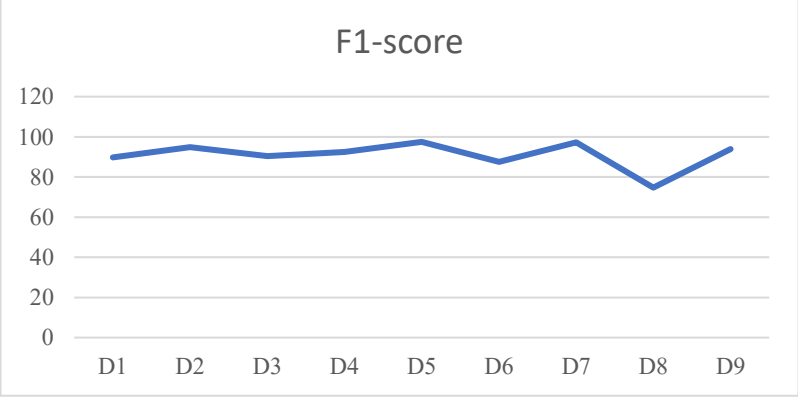


Figure 8: Correlation Between the Different Categories in the Dataset and the Scores of the Algorithm

In datasets with four classifiers, the algorithm performed the poorest. The dataset was categorized into two categories to verify the VGG16 algorithm's performance: training (80%), validation (20%) in figure 8.

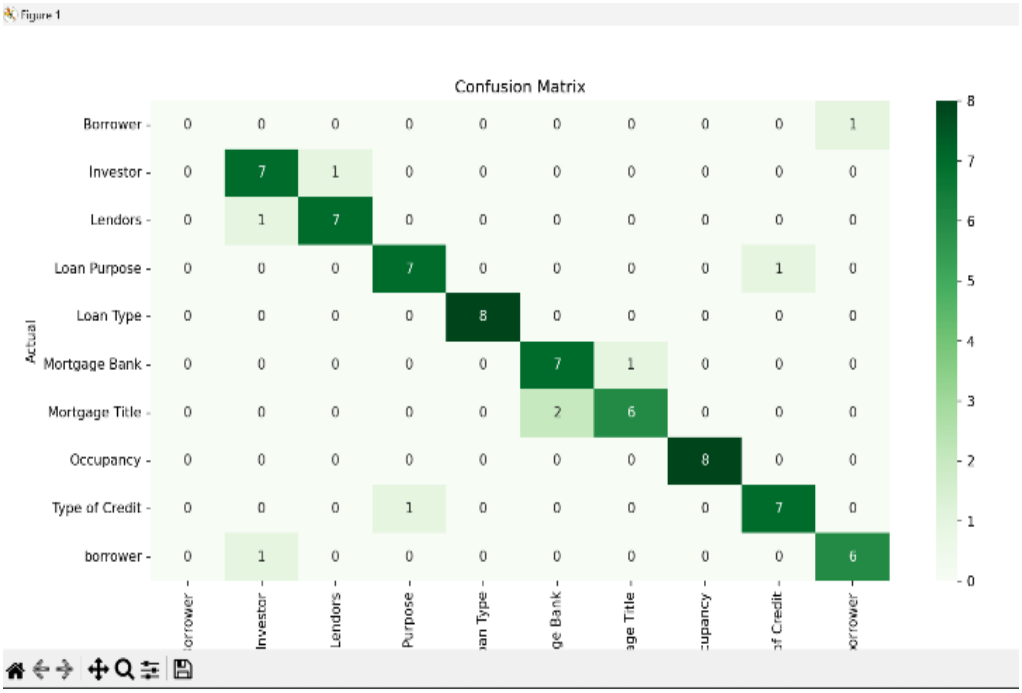


Figure 9: The Various Metrics related to Accuracy, F1 Score, Recall, Precision

Table 6 indicate the various types of accuracy metric related to different categories. The VGG16 algorithm's testing accuracy analysis in datasets with over 3,000 images is demonstrated. We compared the VGG16 algorithm's accuracy across the datasets D1, D2, D3, D4, D5, D6, D7, D8 and D9 categories for the purpose of this review. After testing, the VGG16 algorithm's performance ranged from 97.99% to 88.22% in datasets including over 1,000 mages. The results of this investigation show that as the count of images and the epoch increased, the algorithm's detection accuracy improved for both testing and validation. Figure. 9 shows how accurate the VGG16 method is on datasets with fewer than 5000 images. This analysis was conducted using datasets, which includes 930, 1,044, 2,012, and 2,800 snapshots, respectively. With a range of 80.10 to 93.84%, the VGG16 algorithm's mean accuracy for the test data was 94.31%. With a range of 80.10-93.84%, the algorithm's accuracy was 94.31%. With the exception of the D5 dataset, which displayed an accuracy of 80.10%, the accuracy differences within each dataset were quite minimal, as illustrated in Figure. 9. Nevertheless, additional investigation revealed that the extremely high class difference ratio in the D5 dataset was the reason for its poor accuracy. The VGG16 algorithm's performance various class and describes Figure. 8. The accuracy of the datasets with two and three classes did not differ significantly, as shown in Figure. 8, however the accuracy drops by 9.57% when the dataset comprises four classes.

## Comparison Between Epoch Vs Accuracy for various Categories

Table 6: Category 1 &2 and Accuracy Achieved with Various Epochs

| 09 | epoch | Trained Samples | Validated Samples | Correct Predictions | Incorrect Predictions | Accuracy (%) |
|---|---|---|---|---|---|---|
| Category1 | 10 | 50 | 20 | 15 | 5 | 75 |
| Category1 | 20 | 50 | 20 | 16 | 4 | 80 |
| Category1 | 30 | 50 | 20 | 16 | 4 | 80 |
| Category1 | 20 | 100 | 30 | 25 | 5 | 83 |
| Category1 | 40 | 100 | 30 | 26 | 4 | 86.67 |

| | | | | | | |
|---|---|---|---|---|---|---|
| Category1 | 60 | 100 | 30 | 26 | 4 | 86.67 |
| Category1 | 50 | 150 | 40 | 34 | 6 | 85.00 |
| Category1 | 70 | 150 | 40 | 35 | 5 | 87.50 |
| Category1 | 100 | 150 | 40 | 37 | 3 | 92.50 |
| Category1 | 100 | 200 | 50 | 47 | 3 | 94.00 |
| Category1 | 150 | 200 | 50 | 48 | 2 | 96.00 |
| Category1 | 200 | 200 | 50 | 49 | 1 | 98.00 |
| Category2 | 10 | 50 | 20 | 14 | 6 | 70.00 |
| Category2 | 20 | 50 | 20 | 15 | 5 | 75.00 |
| Category2 | 30 | 50 | 20 | 16 | 4 | 80.00 |
| Category2 | 20 | 100 | 30 | 25 | 5 | 83.33 |
| Category2 | 40 | 100 | 30 | 26 | 4 | 86.67 |
| Category2 | 60 | 100 | 30 | 27 | 3 | 90.00 |
| Category2 | 50 | 150 | 40 | 34 | 6 | 85.00 |
| Category2 | 70 | 150 | 40 | 36 | 4 | 90.00 |
| Category2 | 100 | 150 | 40 | 37 | 3 | 92.50 |
| Category2 | 100 | 200 | 50 | 47 | 3 | 94.00 |
| Category2 | 150 | 200 | 50 | 48 | 2 | 96.00 |
| Category2 | 200 | 200 | 50 | 49 | 1 | 98.00 |
| Category1 & 2 | 250 | 400 | 55 | 54 | 1 | 98.18 |

Table 7: Category 3,4, and 5 Accuracy Achieved

| Data Info | Epoch | Trained Samples | Validated Samples | Correct Predictions | Incorrect Predictions | Accuracy (%) |
|---|---|---|---|---|---|---|
| Category 3 | 10 | 320 | 40 | 30 | 10 | 75 |
| Category 3 | 50 | 320 | 40 | 34 | 6 | 85 |
| Category 3 | 100 | 320 | 40 | 35 | 5 | 87.5 |
| Category 3 | 150 | 320 | 40 | 36 | 4 | 90 |
| Category 3 | 200 | 320 | 40 | 37 | 3 | 92.5 |
| Category 4 | 10 | 800 | 50 | 35 | 15 | 70 |
| Category 4 | 50 | 800 | 50 | 37 | 13 | 74 |
| Category 4 | 100 | 800 | 50 | 40 | 10 | 80 |
| Category 4 | 150 | 800 | 50 | 45 | 5 | 90 |
| Category 4 | 200 | 800 | 50 | 46 | 4 | 92 |
| Category 5 | 10 | 400 | 30 | 23 | 7 | 76.67 |
| Category 5 | 50 | 400 | 30 | 25 | 5 | 83.33 |
| Category 5 | 100 | 400 | 30 | 25 | 5 | 83.33 |
| Category 5 | 150 | 400 | 30 | 27 | 3 | 90.00 |
| Category 5 | 200 | 400 | 30 | 28 | 2 | 93.33 |

Table 8: Category 6,7,8 and 9 Accuracy Achieved with Various Epochs with Various Epochs

| Data Info | Epoch | Trained Samples | Validated Samples | Correct Predictions | Incorrect Predictions | Accuracy (%) |
|---|---|---|---|---|---|---|
| Category 3 | 10 | 320 | 40 | 30 | 10 | 75 |
| Category 3 | 50 | 320 | 40 | 34 | 6 | 85 |
| Category 3 | 100 | 320 | 40 | 35 | 5 | 87.5 |
| Category 3 | 150 | 320 | 40 | 36 | 4 | 90 |
| Category 3 | 200 | 320 | 40 | 37 | 3 | 92.5 |
| Category 4 | 10 | 800 | 50 | 35 | 15 | 70 |
| Category 4 | 50 | 800 | 50 | 37 | 13 | 74 |
| Category 4 | 100 | 800 | 50 | 40 | 10 | 80 |
| Category 4 | 150 | 800 | 50 | 45 | 5 | 90 |
| Category 4 | 200 | 800 | 50 | 46 | 4 | 92 |
| Category 5 | 10 | 400 | 30 | 23 | 7 | 76.67 |
| Category 5 | 50 | 400 | 30 | 25 | 5 | 83.33 |
| Category 5 | 100 | 400 | 30 | 25 | 5 | 83.33 |
| Category 5 | 150 | 400 | 30 | 27 | 3 | 90.00 |
| Category 5 | 200 | 400 | 30 | 28 | 2 | 93.33 |

Analyzing the results, we can reasonably conclude that the dataset's number of images and their corresponding classes correlate in some way with the functioning of the VGG16 algorithm; the algorithm's accuracy increased proportionately with the number of images for datasets above 800. The VGG16 model achieved an average accuracy of 93.7 percent. Although the VGG16 algorithm, with its lesser architectural complexities, outperformed prior experiments, it did fall behind middle-range efficiency for much of this dataset.However, more efficient execution offset this during the higher epoch count.The algorithm's accuracy was impacted by the variety of categories in a dataset; datasets with four classes showed inferior performance from the VGG16 model. The VGG16 architecture has the advantage of having merely six layers in depth in comparison to other well-known transfer learning techniques. The identifying process is quick because of the thin layer. It can be used on handheld computers with modest specifications because of its quick time. With disparate quantities and ratios between groups, the dataset is not optimal under real-world circumstances. In light of the experiment, The amount of images used for the training dataset, the number of classes, and the defined epochs all affect the accuracy achieved using VGG16. All these parameters plays a major role as shown in Table 6, 7 and 8.

## 4 Conclusion and Future Scope

This study set out to evaluate the VGG16 algorithm's performance across several dataset classifications. The experimental findings validated the VGG16 algorithm's excellent accuracy in handwritten text recognition, particularly in the mortgage industry.

In addition, the study validated the VGG16 architecture's resilience when used on datasets with different class and class ratios in the mortgage area. We did not investigate how a VGG16 algorithm's performance might be degraded by ratios of particular classes class ratios considered “high”.

## Acknowledgement

This paper was completed successfully under the guidance of Dr. Kirubanand V.B., Christ (Deemed to be University), as a part of Computer Science Department.

## References

[1] Abbas, T., Razzaq, A., Zia, M. A., Mumtaz, I., Saleem, M. A., Akbar, W., ... & Shivachi, C. S. (2022). Deep neural networks for automatic flower species localization and recognition. *Computational intelligence and neuroscience*, *2022*(1), 9359353. https://doi.org/10.1155/2022/9359353

[2] Abdel-Jaber, H., Devassy, D., Al Salam, A., Hidaytallah, L., & El-Amir, M. (2022). A review of deep learning algorithms and their applications in healthcare. *Algorithms*, *15*(2), 71.

[3] Carter, E., & Heinriksen, L. (2023). Performance Analysis of Ceramic Membranes in Treating Textile Wastewaters. *Engineering Perspectives in Filtration and Separation*, 13-15.

[4] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1251-1258).

[5] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

[6] Geng, Y. (2024). Comparative Study on Physical Education Learning Quality of Junior High School Students based on Biosensor Network. *Natural and Engineering Sciences*, *9*(2), 125-144. https://doi.org/10.28978/nesciences.1569219
[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
[8] Indriyani, I., Giriantari, I. A. D., Sudarma, M., & Widyantara, I. M. O. (2023). Facial Skin Type Detection for Race Classification using Convolutional Neural Network and Haar Cascade Method. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 14*(2), 41-58. https://doi.org/10.58346/JOWUA.2023.I2.004
[9] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448-456). pmlr.
[10] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*.
[11] Nayak, A., & Raghatate, K. S. (2024). Image segmentation and classification of aquatic plants using convolutional neural network. *International Journal of Aquatic Research and Environmental Studies, 4,* 14-19. https://doi.org/10.70102/IJARES/V4S1/3
[12] R. Vaila, J. Chiasson, & V. Saxena,(2019). *Deep Convolutional Spiking Neural Networks for Image Classification.*CEUR Workshop Proceedings, 2546.
[13] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4510-4520).
[14] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. https://doi.org/10.48550/arXiv.1409.1556
[15] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). ImageNet classification with deep convolutional neural networks. *In Advances in Neural Information Processing Systems 27.*
[16] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2017, February). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 31, No. 1). https://doi.org/10.1609/aaai.v31i1.11231.
[17] Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105-6114). PMLR.
[18] Trivedi, J., Devi, M. S., & Solanki, B. (2023). Step Towards Intelligent Transportation System with Vehicle Classification and Recognition Using Speeded-Up Robust Features. *Archives for Technical Sciences/Arhiv za Tehnicke Nauke*, (28). https://doi.org/10.59456/afts.2023.1528.039J
[19] Vardhan, H., & Bhattacharya, R. (2025). The Impact of Sustainable Practices on Business Performance. *International Journal of SDG's Prospects and Breakthroughs*, *3*(1), 15-21.
[20] Zhang, Z.-K., Liu, C., Zhang, Y.-C., & Zhou, T. (1990). Handwritten digit recognition with a back-propagation network. *Advances in Neural Information Processing Systems.*

## Authors Biography

**K.P.Kavitha** is a postgraduate researcher in Computer Science in CHRIST (Deemed to be University), Bangalore, specializing in deep learning and document intelligence. Her research focuses on developing computational approaches for industry-grade handwritten text recognition, leveraging the Word Beam Search algorithm integrated with language modeling techniques. Their recent work emphasizes extracting structured data from unstructured handwritten documents to meet enterprise-level accuracy and

scalability standards. Their broader interests include computer vision, OCR systems, and the application of neural architectures in real-world document automation pipelines.

**Dr.V.B. Kirubanand** is currently working as associate professor in CHRIST (Deemed to be University), Bangalore. He completed his undergraduate in Electronics and Masters in Computer Application from Bharathiar University, Coimbatore. He received his Master's in philosophy from Periyar University, Salem and Doctoral degree from Anna University, Chennai. He possesses two decades of experience in teaching and expertise in Wireless Networks, Web Technology and Internet of Things. Dr.V.B. Kirubanand has several books publications to his credit and has a good track record of several research publications in Scopus and high impact factor journals. Several Patent published and also Design granted. He organised several National and International conferences, seminars and webinars. He also acted as a resource person for many conferences and seminars in reputed institutions. He is also part of various consultancy projects and research projects. Project received from Karnataka panchayat commissionerate Bangalore for solid waste management project and Received from ministry of justice and empowerment, New Delhi. He acted as Senate Member of Bharathiar University in the year 2001-2004. He has received various awards like Best teaching award in TVET polytechnic, Ethiopia,Vocational Service Excellence Award from Rotary Club, Coimbatore, India.(2010).Best Teaching Award,Sri krishna College of engineering and technology, Coimbatore, India (2014) and Best Alumni Award at Erode arts and science college , Erode, India.(2018).Lifetime Research Achievement Award,New Delhi,(2022) Inspired Excellence Award, New Delhi(2023). National level best faculty award from Nehru group of institutions, coimbatore, Tamilnadu 2024.