A novel model to detect and categorize objects from images by using a hybrid machine learning model

Nilambar Sethi[1], Vetukuri Venkata Siva Rama Raju[2], Venkata Srinivas Lokavarapu[3], Ravi Babu Devareddi[4], Shiva Shankar Reddy[3], Silpa Nrusimhadri[5]
[1]Department of Computer Science and Engineering, Gandhi Institute of Engineering and Technology University, Gunupur, India
[2]Koneru Lakshmaiah Education Foundation, Vaddeswaram, India
[3]Sagi RamaKrishnam Raju Engineering College, Bhimavaram, India
[4]GITAM School of Technology, GITAM University, Hyderabad, India
[5]Shri Vishnu Engineering College for Women (A), Bhimavaram, India

Article Info

Article history:

Received Mar 3, 2024
Revised Oct 19, 2024
Accepted Oct 23, 2024

Keywords:

Artificial intelligence
Computer vision
Deep learning
Machine learning
Object detection
Visual geometry group

ABSTRACT

As humans, we can easily recognize and distinguish different features of objects in images due to our brain's ability to unconsciously learn from a set of images. The objectives of this effort are to develop a model that is capable of identifying and categorizing objects that are present within images. We imported the dataset from Keras and loaded it using data loaders to achieve this. We then utilized various deep learning algorithms, such as visual geometry group (VGG)-16 and a simple net-random forest hybrid model, to classify the objects. After classification, the accuracy obtained by VGG16 and the hybrid model was 84.7% and 89.6%, respectively. Therefore, the proposed model successfully detects objects in images using a simple net as a feature extractor and a random forest for object classification, achieving better accuracy than VGG16.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Shiva Shankar Reddy
Assistant Professor, Sagi RamaKrishnam Raju Engineering College
Bhimavaram, Andhra Pradesh, India
Email: shiva.csesrkr@gmail.com

1. INTRODUCTION

Computers have a unique way of interpreting visuals. While human brains understand images, computers break them into numerical values and use algorithms to identify significant elements in still photos, videos, graphics, and live feeds. This process is known as computer vision. Image processing algorithms computers can interpret and comprehend visuals from a single or series of pictures. Figure 1 shows the various object detection in an image. By analyzing and filtering millions of user-uploaded photos, machine vision can accurately detect people and automobiles on the road [1].

Numerous companies are investing in image recognition technology to analyze visual data as computer vision advances. This technique is utilized in various fields, such as medical picture analysis, autonomous object identification in automobiles, and facial recognition in security systems [2]. Using machine vision, artificial intelligence (AI), and trained algorithms, a program or system can identify objects, people, locations, and activities in photographs [3]. Image recognition has become increasingly popular due to advancements in machine learning and processing capacity [4]. The aim is to locate and recognize objects within an image. Classification involves determining the object type within the bounding box [5]. Object identification can be challenging due to occlusion, clutter, item appearance, size, and orientation differences. Object recognition methods use image processing, machine learning, and computer vision techniques [6], [7].

Object detection has numerous industrial and field applications. Many surveillance systems use object detection to monitor the real-time movement of people, vehicles, and other objects. It is also crucial to autonomous driving systems, which must recognize and track other cars, pedestrians, and obstacles to ensure safe navigation [8].

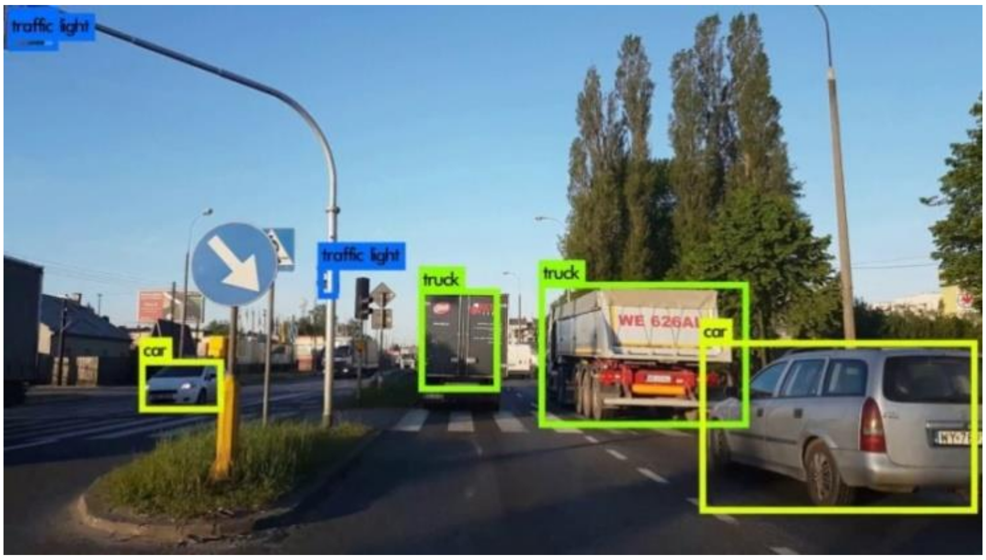


Figure 1. Object detection in an image

Object detection is also used in medical imaging to locate tumours, lesions, and other pathological signs, aiding diagnosis and therapy planning. In addition to these applications, object detection is utilized in agriculture to identify crops, monitor plant health, and track crop growth. It is also used in sports analytics to monitor player and ball movement throughout games, assisting coaches and analysts in creating better team strategies. Art and cultural heritage applications employ object detection to identify and evaluate objects within paintings, sculptures, and other artefacts [9]. This technology enables researchers and historians better to understand these objects' historical significance and cultural importance.

2. RELATED WORK

Sultana *et al*. [1] studied object identification models like region-based convolutional neural networks (RCNN) and RefineDet, which can be classified into one-stage and two-stage methods. Several new models have given researchers hope for quicker and more accurate real-time item detection. Haque *et al*. [2] proposed a visual geometry group (VGG)-residual network (ResNet) network object identification model that reduced the VGG network training error and improved tiny item detection. The model combined the VGG and ResNet network to decrease layer size and training error. The upgraded convolutional architecture of the model performed well, with an object detection accuracy of 85.8%. Radovic *et al.* [3] suggested a convolutional neural networks (CNN)-based object identification model for aerial images. The you only look once (YOLO) platform uses a CNN to follow, recognize, and classify unmanned aerial vehicles (UAV) video streams in real time. Despite having 84% accuracy in detection and classification, the YOLO platform is more efficient than typical machine learning methods. It is perfect for UAV autonomous flying. Galvez *et al.* [7] suggested a CNN model for object detection. This model used CNN to recognize environmental items. Single shot detector (SSD) with MobileNet offers great detection speed but poor accuracy, whereas faster-RCNN with InceptionV2 has low speed but higher accuracy. SSD with MobileNet is ideal for detecting objects quickly, particularly in real-time applications. For high-accuracy detection, one should use faster-RCNN with InceptionV2.

Kang *et al*. [8] proposed using tubelets with CNN for video object detection (VOD). This model introduced a deep learning (DL) framework for video object recognition using temporal information. This framework won the VOD challenge with given data and performed well on ImageNet object identification from video. Yu *et al.* [9] presented an advanced object detection network unit box. This study examined the unique bounding box prediction loss, intersection over union (IoU). Instead of four independent variables, the IoU loss layer regresses an object candidate's bounding box, speeding computation and improving object localization. They also presented the UnitBox, a sophisticated object identification network for face recognition that performs world-class based on IoU loss. In their research on image classification, Tammina [10] utilized transfer learning along with VGG16 and deep CNN. The pre-trained model information was used in this study to achieve Gao *et al*. [11] introduced deep spatial feature transformation (DSFT) for oriented aerial object detection. They used DSFT networks to identify objects in aerial photos. They found that the aerial detector requires a lot of rotating data since ordinary CNNs cannot represent direction change. They also presented the polarization function to construct essential feature selection and detection features.

Mohsin *et al*. [12] surveyed deep neural networks and algorithms for defect detection. They wanted to understand the functioning of the latest industrial Technologies for detecting faults in quality control and how they can be applied in more complex ways. They also discussed improving inference performance, model correctness for tiny errors, network design for deployment on low-resource devices, and deep model training with less processing power. Shankar *et al*. [13] worked on images to retrieve the data by using a genetic algorithm. Their model reduces the time interval and increases precision using the content-based image retrieval (CBIR) model. Alharthi *et al*. [14] presented 3-dimensional convolutional networks (C3D)-based on massive crowd abnormal behavior recognition. They evaluated ways to improve aberrant behaviour detection using the Hajj crowd dataset. They explored DL algorithms to categorize Hajj dataset videos as anomalous. The C3D model achieved excellent F1-score, precision, and accuracy on the test set, with scores of 0.90, 0.89, and 0.88. Kahler *et al*. [15] proposed an AI-based approach to detect surface defects in aviation components for rectification. They presented a defect monitoring system for landing gear components using AI to identify the endpoint of the process. Song *et al*. [16] used an objective and synthetic data-trained YOLOv4 object detection network to detect corrosion, monitor grinding, and offer feedback when the endpoint is achieved. To manipulate the video database to detect duplicate videos using frames and their identification [17].

Ahmed *et al*. [18] presented an internet of things (IoT)-enabled DL framework for multiple object recognition in aerial remote-sensing photos. The study used the pyramid scene parsing network (PSPNet) DL-based segmentation model to separate objects using AI and achieved high accuracy rates using VGG16, ResNet-0, and MobileNet models. Loraksa *et al*. [19] proposed an SSD-VGG16-based model to identify cancer-causing osteosarcoma lung nodules. The study aimed to build an AI system that can recognize lung nodules using a straightforward model with high accuracy rates. Ma *et al*. [20] suggested an end-to-end scale-aware split-merge-enhancement network for remote sensing object recognition using various modules that improve object saliency, object segmentation, and feature alignment. Wang *et al*. [21] proposed a hybrid feature-aligned network (HFANet) to recognize salient objects in optical remote sensing images. The study used an adjacent feature-aligned module (AFAM) to integrate nearby features using unparameterized alignment and devised a new interactive guidance loss (IGLoss) to blend edge detection and saliency. To recognize objects like flower species using DL models, the authors achieved excellent results [22]. Loganathan *et al*. [23] suggested using upgraded global pooling (GP)-RCNN to identify and recognize multiple objects. The study showed that GP-RCNN outperformed RCNN by 21% in detecting tiny and large objects.

Shankar *et al*. [24] classified the voice using various machine learning models. Their work obtained good results compared with the existing models. Yasaswini *et al*. [25] identified the model to avoid road accidents using DL models. Abdu and Noor [26] presented a VGG16-based DL model for domestic trash classification to classify trash items, achieving over 96% precision appropriately. Shyam *et al*. [27] suggested an unsupervised domain adaptation technique that leverages a simulator-created synthetic dataset to ensure performance consistency of multi-object-tracking (MOT) algorithms over various manually annotated real situations. Li *et al*. [28] proposed a self-knowledge distillation-based few-shot object identification approach to reduce overfitting in few-shot object recognition. They demonstrated their method's applicability and practicality using benchmark datasets. Pravallika and Baskar [29] studied using VGG16 and support vector machine (SVM) algorithms for brain tumour magnetic resonance imaging (MRI) image classification. VGG16 outperformed SVM in accuracy and precision, improving the overall performance. Sajjad and Hemavathi [30] explored object-detecting methods and how to build models for accurate results using DL techniques. Yang *et al*. [31] proposed a method for identifying hidden objects in passive millimeter-wave images. This method is a one-stage anchor-free detector that is fast and accurate. It utilizes convolution techniques and transformer analysis. After conducting research, it was found that this approach is superior to the most advanced techniques in speed and accuracy.

3. METHODOLOGY

Machines must be able to recognize objects in images to understand and interpret the visual world, just like humans do. Computer vision is a field that relies on machines recognizing and identifying things in pictures and videos. Object detection is used in various disciplines, including robotics, to help robots understand and interpret visual information. Significantly improve productivity, safety, and efficiency across many sectors. To find objects in an input picture, the deep convolutional neural network (DCNN) VGG16 can be used. The goal is to detect visual objects and create bounding boxes accurately. To train the model, tagged photos and object annotations should be used. The ultimate objective is to achieve high object detection accuracy with minimal false positives and negatives. Applications such as autonomous driving, picture segmentation, and object identification are mainly used.

3.1. Objectives

The main goal is to recognize objects in photos and sort them into specific categories. The primary objectives are to detect objects from a greater distance and in the shortest possible time. Compared to earlier models, the aim is to categorize objects into more categories. The objective is to identify objects in low-quality images without converting them into high-quality photos.

3.2. Dataset

The Canadian Institute For Advanced Research (CIFAR-10) provides images commonly used to train machine learning and computer vision algorithms. This dataset is one of the most popular choices for this purpose. Machine learning algorithms that identify objects in images learn by example, and CIFAR-10 pictures are beneficial for this type of training. The dataset is designed to help computers distinguish between different objects in images, and it is an excellent resource for researchers who want to test multiple algorithms. The pictures in CIFAR-10 are relatively low-resolution (32×32), which makes them particularly useful for testing different CNN.

3.2.1. Dataset description

CIFAR-10 contains 60,000 32×32 color pictures in 10 categories. Figure 2 shows that airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks each have 6,000 photos. The dataset contains 50,000 training and 10,000 test photos. Five batches of 10,000 training pictures exist. One set of 1,000 test photographs per category is used. The training photos are randomly sorted, with some batches having more of one type. Every training batch has 5,000 photos from each category. CIFAR-10 contains 60,000 32×32 color pictures in 10 categories. Figure 2 shows that airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks each have 6,000 photos. The dataset contains 50,000 training and 10,000 test photos. Five batches of 10,000 training pictures exist. One set of 1,000 test photographs per category is used. The training photos are randomly sorted, with some batches having more of one type. Every training batch has 5,000 photos from each category.

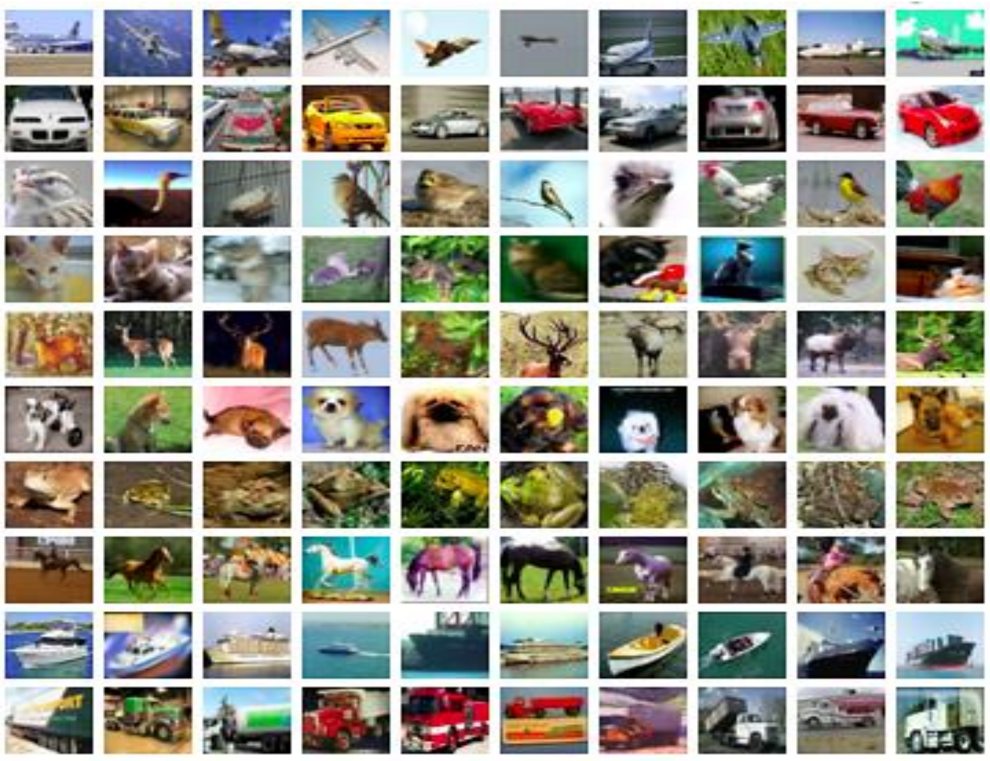


Figure 2. Sample dataset for object detection

3.3. Models used

To reach the objective of this work, we used VGG16, YOLO-v3, YOLO-v4, and the proposed model, which consists of simplenet-random forest models. For these models, we have evaluated various performance metrics. Those are accuracy, precision, recall, and F1 score. This section describes each model clearly to help the reader understand this work.

3.3.1. VGG16

VGG16 is a DCNN architecture developed by the Oxford visual geometry group. The network was trained using ImageNet, a collection of millions of tagged pictures organized into 1,000 categories. VGG16 consists of 3 fully connected and 13 convolutional layers, and it utilizes max-pooling layers with 2×2 filters and a stride of 2 pixels. The convolutional layers use small 3×3 filters with 1. The network has more filters, ranging from 64 in the first layer to 512 in the final layer. You can find the model's architecture in Figure 3.

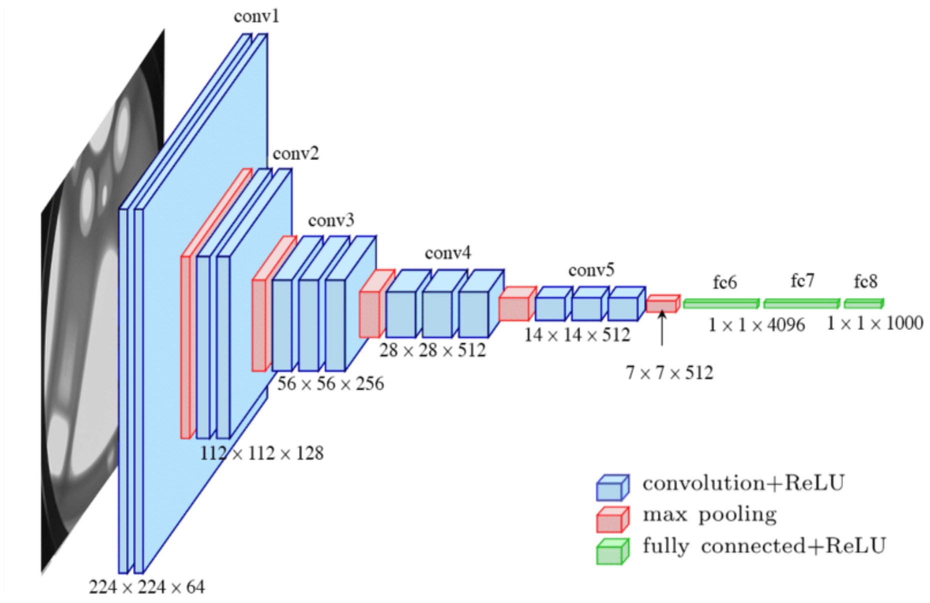


Figure 3. VGG architecture

```
Algorithm 1: VGG16
Step 1:   Define the transformations for training and testing data
Step 2:   Define a function for loading training data
          a. `train_transform` is used to define the transformation.
          b. Load the CIFAR10 dataset for training using `torch-vision Datasets.'
          c. CIFAR10`, and then use the `DataLoader` class from `torch.utils.data` to load the data.
          d. Return the data loader.
Step 3:   Create a function for loading testing data, using `test_transform` to define the transformation.
          a. Load the CIFAR10 dataset for testing using `torchvision.datasets.CIFAR10`, then use the
             `DataLoader` class to load the data.
          b. Return the data loader.
Step 4:   Use the functions created in step 2 and step 3 to get the train and test data loaders,
          respectively.
Step 5:   Define a dictionary class.
Step 6:   Create the Bottleneck class,
          a. Initialize the bottleneck block layers using the `init` function.
          b. Define the forward function to pass input through the bottleneck.
Step 7:   Create the Transition class
          a. Initialize the transition block layers using the `init` function
          b. Set the forward function to pass input via the transition block.
Step 8:   Create the VGG16network by initializing the VGG16network layers using the `init` function.
```

```
          a. Set the inner_channels and conv1 layers, and create dense and transition blocks using
             `_make_dense_layers.
          b. Add the final batch norm and rectified linear unit (ReLU) activation layer, and classify
             using average pooling and fully linked layers.
          c. Define the VGG16network input forward function with the appropriate parameters.
Step 9:   Create a VGG16network with the appropriate parameters.
```

For the explanation for Algorithm 1, we need to import several libraries, including numpy, pandas, os, time, matplotlib, and PyTorch. We must also import related modules for data processing, model development, and training, such as torch and torchvision. Next, we perform data preprocessing to prepare the data for the model. This involves using train_transform and test_transform to perform data augmentation methods such as random horizontal flip, rotation, and normalization. We then use get_training_dataloader and get_testing_dataloader to generate data loaders for loading training and testing data batches. These functions preprocess the data using train_transform and test_transform transformers.

Once the data is prepared, we move on to data visualization using Matplotlib. Here, we plot the first 25 training data pictures, displaying images with their class names in a 5×5 grid. The next step is model architecture, where we implement DenseNet using PyTorch modules. The design includes a bottleneck layer, a transition layer, and a dense block. The bottleneck layer consists of three layers: batch normalization, rectified linear unit (ReLU) activation, and convolutional. After applying these layers to the input feature map, the output feature map is concatenated with it along the channel axis to form the final output. The transition layer consists of batch normalization and convolutional layers. After applying these layers to the input feature map, the output feature map is sent through an average pooling layer to downsample its spatial dimensions by 2. The bottleneck layers and the input feature map form the dense block along the channel axis bottleneck layers output feature maps from concatenated feature maps. Concatenating the output feature maps from each bottleneck layer with the input feature map along the channel axis produces the final output.

We also use the VGG16model, which uses bottleneck, transition, and dense block layers. After one convolutional layer, dense blocks and transition layers make up the model. The final dense block output undergoes batch normalization and ReLU activation before a global average pooling layer. Final output class probabilities are produced by passing the global average pooling layer output via a fully connected layer. Finally, we train the model using the stochastic gradient descent (SGD) optimizer and learning rate scheduler. The training procedure involves iterating over batches of training data, calculating the forward pass, cross-entropy loss function, gradients, and optimizer model weight updates. After a predetermined number of training epochs, the model's performance is assessed using testing data. After training, the model's accuracy and loss of testing data are displayed.

3.3.2. Hybrid model (simpleNet-random forest)

The simpleNet-random forest hybrid model is an machine learning approach that improves the categorization of picture datasets. The model combines two techniques: SimpleNet, a CNN that extracts relevant features from raw picture data, and a random forest classifier, which handles high-dimensional input data. This hybrid model uses the strengths of both techniques to achieve better performance in picture classification tasks. SimpleNet effectively extracts the features from raw picture data. At the same time, the random forest classifier generalizes well to new cases and handles high-dimensional input data—the architecture of this hybrid model is presented in Figure 4.

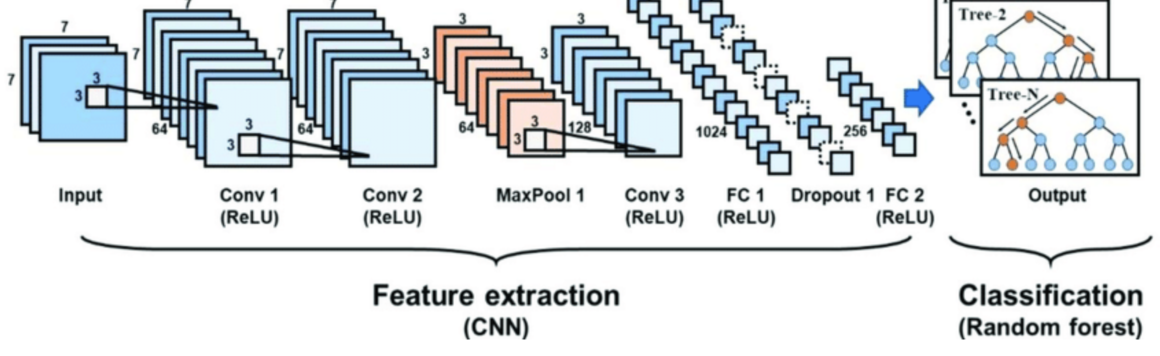


Figure 4. Architecture of hybrid model (simpleNet-RF)

```
Algorithm 2: Hybrid model (simplenet-random forest)
Step 1:  Import numpy, matplotlib.pyplot, RandomForestClassifier, classification_report, confusion_matrix,
         cifar10, Model, Input, Conv2D, Dense, Flatten, np_utils, EarlyStopping, and ModelCheckpoint.
Step 2:  Divide the CIFAR-10 dataset into training and testing sets using cifar10.load_data(). Divide each
         pixel value by 255 to normalize data. One-hot encode labels with np_utils.to_categorical().
Step 3:  Use Keras' functional API to define SimpleNet—model compilation using
         categorical_crossentropy loss, Adam optimizer, and accuracy metric. Save the best model
         throughout training using early stopping and model checkpoint callbacks.
Step 4:  Train SimpleNet using the model fit (), batch size 128, 20 epochs, and 10% validation data. Load
         the stored best SimpleNet model using model.load_weights().
Step 5:  Use best_model.predict() to extract training and test SimpleNet features. Reshape retrieved
         features to a 2D array using reshape,
Step 6:  Train a Random Forests model using RandomForestClassifier(), 100 estimators, entropy criteria,
         and 42 random states. Test the Random Forests model using classification_report() and
         confusion_matrix().
Step 7:  Chart loss and accuracy using matplotlib. Pyplot.
Step 8:  Use matplotlib. Python uses the Random Forests model to show the accurate and predicted labels
         for the first 10 test photos.
```

– Algorithm explanation

The explanation for Algorithm 2 is that the process involves using the CIFAR-10 dataset to develop a DL model known as simpleNet. This model is then used to extract features from the dataset, after which the images in the dataset are classified using a random forest classifier based on the retrieved features. The NumPy, matplotlib, and Keras modules are imported for the CIFAR-10 dataset, model architecture components, and callback methods to begin the process. The scikit-learn's random forest classifier, classification report, and confusion matrix (CFM) functions are also imported. The CIFAR-10 dataset is loaded and normalized, and the labels are one-hot encoded using np_utils.to_categorical. The simpleNet model comprises six convolutional layers and a dense output layer with ten units per CIFAR-10 class. The model uses categorical cross-entropy as the loss function, Adam optimizer, and accuracy metric.

The simpleNet model is trained on CIFAR-10 for 20 epochs, with a batch size of 128 and a validation split of 0.1. If the validation loss does not improve after five epochs, training is stopped, and the best model is saved using the ModelCheckpoint callback method. After loading the best simpleNet model, the reshape method is used to flatten the training and test sets into one-dimensional arrays for feature extraction. These features then train a random forest classifier with 100 trees and entropy for node splitting. The classification report and CFM functions are used to output the test set picture labels the trained random forest classifier predicted. The approach also displays simpleNet model loss and accuracy curves during training. Finally, the algorithm runs over the first 10 test photos, indicating the actual and predicted labels and showing the image using matplotlib.

We initially imported the dataset from Keras and loaded it using dataloaders. Data preparation involved using train_transform and test_transform after data collection. These transformers employ random horizontal flips, random rotation, and normalization to prepare data for the model. Finally, we used get_training_dataloader and get_testing_dataloader to construct the training and testing data loaders. After splitting the data, we used DL techniques like VGG16 and a hybrid model (simpleNet-random forest) to recognize objects in images. We evaluated the outcomes using accuracy metrics, which showed that the hybrid model had an accuracy of 89.6%.

4. RESULTS

The dataset is preprocessed after collection. Data preparation involves using train_transform and test_transform to train and test the model by loading batches of preprocessed data. The best simpleNet model is loaded, and the reshape method is used to flatten the training and test sets into one-dimensional arrays to extract features. These features are then utilized to train a random forest classifier with 100 trees and entropy for node splitting. The classification report and CFM functions output the test set picture labels predicted by the trained random forest classifier. We discovered that VGG16 has an accuracy of 84.7%, while the hybrid model (simpleNet-random forest) has an accuracy of 89.6%. Therefore, the hybrid model is 89.6% more accurate.

4.1. Confusion matrix for VGG16

The CFM for the proposed system VGG16 is shown in Figure 5. The validation curves for VGG16 are shown in Figures 6 and 7. Using this CFM makes it easy to evaluate the performance of metrics to predict accurate results. For that purpose, we have calculated the CFM. Accuracy and loss for VGG16 are also shown in Figures 6 and 7.

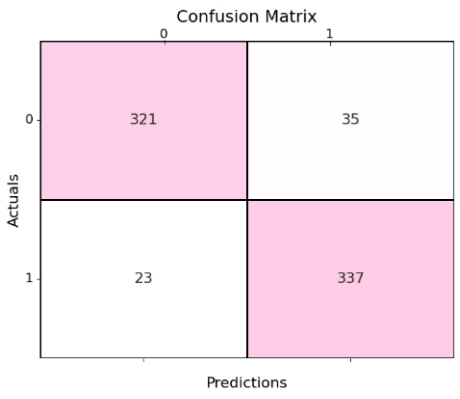


Figure 5. CFM for VGG16

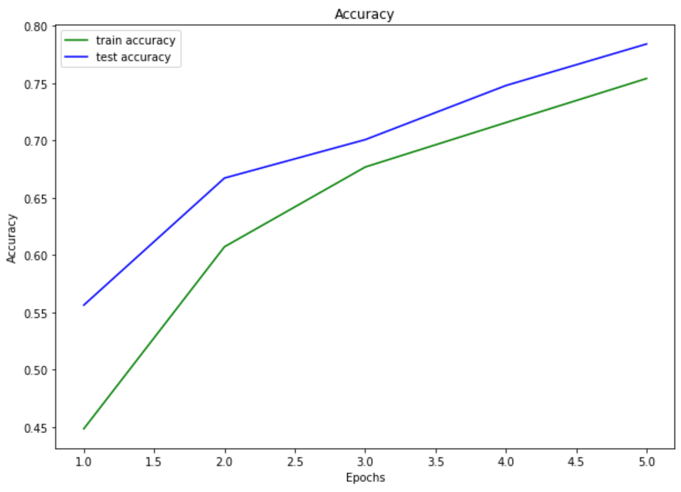


Figure 6. Validation curve of accuracy for VGG16

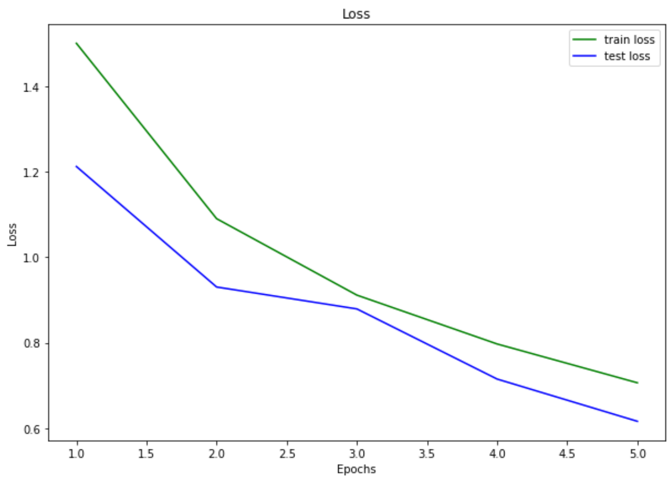


Figure 7. Validation curve of loss for VGG16

4.2. Confusion matrix for hybrid model (simpleNet-random forest)

The CFM for the hybrid model is shown in Figure 8. The validation curve for the hybrid model is shown in Figure 9. The performance metrics used in this work are shown in Table 1. Using this CFM makes it easy to evaluate the performance of metrics to predict accurate results. For that purpose, we have calculated the CFM. Accuracy and loss for hybrid models are also shown in Figure 9.

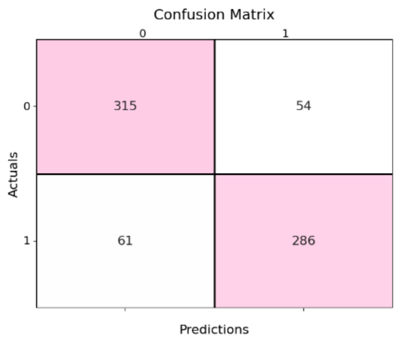


Figure 8. CFM for hybrid model

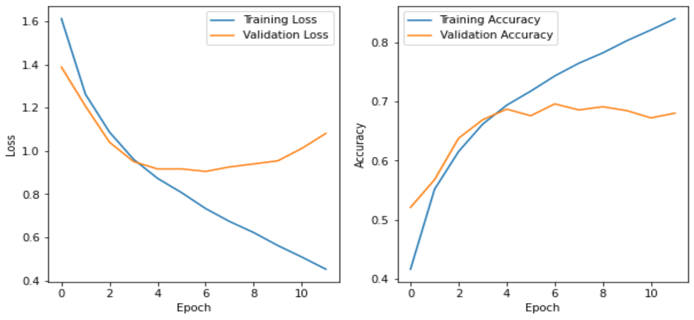


Figure 9. Validation curves for the hybrid model

Table 1 observed that accuracy was 89.6%, precision was 76%, recall was 78.5%, F1-score was 81%, and Matthews correlation coefficient (MCC) got 0.82% for the proposed model by comparing it with existing models. From Table 1, the obtained values for accuracy are shown in Figure 10, the performance metrics like precision, recall, and F1-score graph are shown in Figure 11, and the graph for MCC is shown in Figure 12. The outcomes are evaluated based on accuracy, precision, and recall when testing data to make predictions. The best model is recommended after comparing the predictions made by different algorithms. Among the suggested techniques, VGG16 achieved a classification accuracy of 84.7%, while the hybrid model (simpleNet-random forest) attained an accuracy of 89.6%. Based on the results, the hybrid model performed better between the two methods.

Table 1. Performance of the model

S. No	Model	Accuracy	Precision	Recall	F1-score	MCC
1	VGG16	84.7	72.3	73.5	75	0.68
2	YOLO-v3	82.8	71	69.8	73.4	0.71
3	YOLO-v4	85.7	74.2	75.6	79	0.74
4	Proposed model (SimpleNet-random forest)	89.6	76	78.5	81	0.82

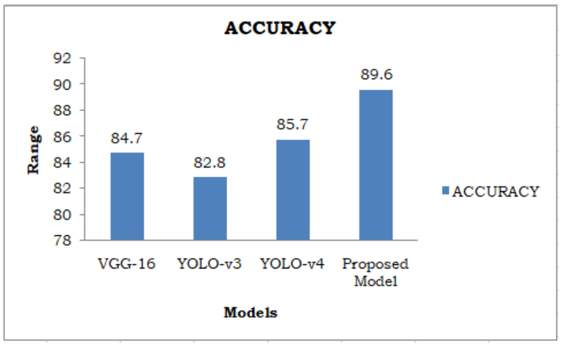


Figure 10. Graph for accuracy with various models

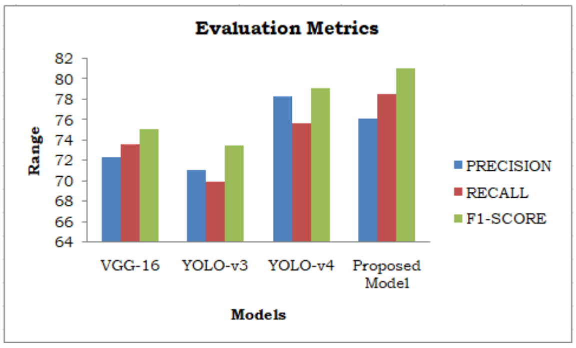


Figure 11. Graph for various performance metrics

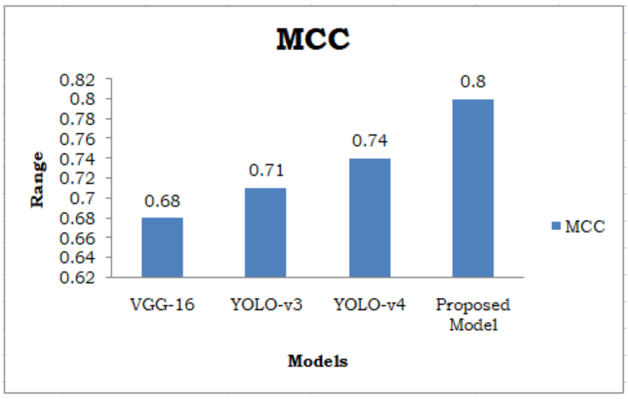
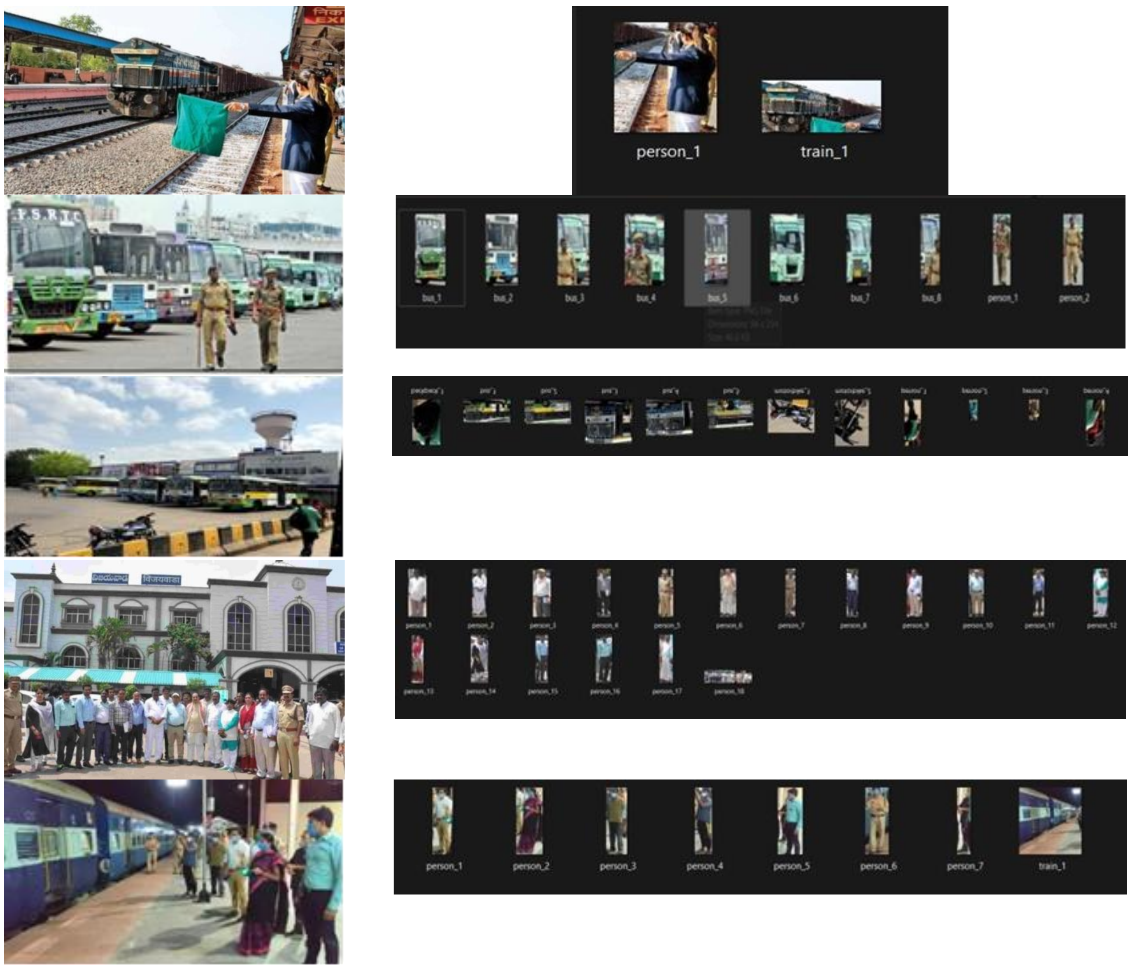


Figure 12. Graph for MCC with various models

The input photos are depicted in Figure 13(a), while the cropped ones are shown in Figure 13(b). We excluded blurry photos by trimming them. The cropped pictures are displayed in Figure 13. To identify and count items, we developed custom YOLOv4 functions. A high confidence score implies a high probability that the detected item is a member of the class that YOLOv4 predicted. Utilizing a threshold allowed us to exclude detections with a low confidence level.



(a) (b)

Figure 13. Output screens for the images as given in (a) input images and from that (b) output images

5. DISCUSSION

When it came to picture classification, the prior technique was used, and just a few training examples were required for each category. The purpose of classifying the pictures using CNN depended on the process of passing the input images through many different layers, including convolutional, pooling, flattening, and fully connected layers. On the other hand, the system had a low level of accuracy when it came to identifying things in photographs and was ineffective when assigning objects to the appropriate categories. Utilizing the VGG16 model and the CIFAR-10 datasets, which are comprised of ten different classes, is the recommendation that has been made to enhance the object identification system. A pre-trained model known as VGG16 can adequately distinguish between a wide variety of items based on a substantial collection of photographs. The network incorporates convolutional filters into its operations to extract features of varying sizes and resolutions from an input picture. These characteristics are subsequently processed via ultimately linked layers to determine whether or not the items in the picture are present and where they are located. Transfer learning is another prominent method that may be used to decrease the amount of time and data required for training. This method includes employing a model that has already been trained to train on fresh datasets. Generally, the suggested object detection system uses VGG16 to perform feature extraction and object prediction. The advanced deep neural network known as VGG16 exhibits exceptional accuracy when identifying objects in photos of varying sizes and degrees of orientation.

6. CONCLUSION

We proposed a method that utilizes DL techniques such as VGG16, implement, and random forest classifier to identify objects in CIFAR-10 images. These models can handle input sizes effectively. We tested various methods on a Keras dataset and compared the classification accuracy of VGG16 and a hybrid model (simpleNet-random forest). The VGG16 method achieved 84.7% classification accuracy, while the hybrid model achieved 89.6%, indicating it was more accurate. Based on our results, we can say that our system can detect objects in CIFAR-10 dataset images with 89.6% accuracy.

REFERENCES

[1] F. Sultana, A. Sufian, and P. Dutta, "A review of object detection models based on convolutional neural network," *Advances in Intelligent Systems and Computing*, vol. 1157, pp. 1–16, 2020, doi: 10.1007/978-981-15-4288-6_1.

[2] M. F. Haque, H. Y. Lim, and D. S. Kang, "Object detection based on VGG with ResNet network," *2019 International Conference on Electronics, Information, and Communication (ICEIC)*, Auckland, New Zealand, 2019, pp. 1-3, doi: 10.23919/ELINFOCOM.2019.8706476.

[3] M. Radovic, O. Adarkwa, and Q. Wang, "Object recognition in aerial images using convolutional neural networks," *Journal of Imaging*, vol. 3, no. 2, 2017, doi: 10.3390/jimaging3020021.

[4] S. Zhao, Z. Wen, Q. Qi, K.-M. Lam, and J. Shen, "Learning fine-grained information with capsule-wise attention for salient object detection," *IEEE Transactions on Multimedia*, pp. 1–14, 2023, doi: 10.1109/tmm.2023.3234436.

[5] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, vol. 9, no. 2, pp. 85–112, 2020, doi: 10.1007/s13748-019-00203-0.

[6] X. Xie, G. Cheng, J. Wang, X. Yao, and J. Han, "Oriented R-CNN for object detection," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Montreal, QC, Canada, 2021, pp. 3500-3509, doi: 10.1109/ICCV48922.2021.00350.

[7] R. L. Galvez, A. A. Bandala, E. P. Dadios, R. R. P. Vicerra, and J. M. Z. Maningo, "Object detection using convolutional neural networks," *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, Jeju, South Korea, 2018, pp. 2023-2027, doi: 10.1109/TENCON.2018.8650517.

[8] K. Kang *et al.*, "T-CNN: Tubelets with convolutional neural networks for object detection from videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2896–2907, 2018, doi: 10.1109/TCSVT.2017.2736553.

[9] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "UnitBox: An advanced object detection network," *MM 2016 - Proceedings of the 2016 ACM Multimedia Conference*, pp. 516–520, 2016, doi: 10.1145/2964284.2967274.

[10] S. Tammina, "Transfer learning using VGG16 with deep convolutional neural network for classifying images," *International Journal of Scientific and Research Publications (IJSRP)*, vol. 9, no. 10, pp. 143-150, Oct. 2019, doi: 10.29322/ijsrp.9.10.2019.p9420.

[11] Y. Gao, Z. Che, L. Li, J. Gao, and F. Bi, "Deep spatial feature transformation for oriented aerial object detection," *IEEE Journal on Miniaturization for Air and Space Systems*, vol. 4, no. 2, pp. 93–99, 2023, doi: 10.1109/JMASS.2023.3234076.

[12] M. Mohsin, O. S. Balogun, and K. Haataja, "Defect detection using deep neural networks and algorithms: a survey," *Solid State Technology*, vol. 66, no. 1, pp. 23–29, 2023.

[13] R. S. Shankar, K. Sravani, L. V. Srinivas, and D. R. Babu, "An approach for retrieving an image using genetic algorithm," *International Journal of Latest Trends in Engineering and Technology (IJLTET)*, vol. 9, no. 2, pp. 57-64, Nov. 2017, doi: 10.21172/1.92.10.

[14] R. Alharthi, A. Alhothali, B. Alzahrani, and S. Aldhaheri, "Massive crowd abnormal behaviors recognition using C3D," *2023 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, 2023, pp. 01-06, doi: 10.1109/ICCE56470.2023.10043437.

[15] F. Kahler, A. C. K. Shetty, and T. Schuppstuhl, "AI-based endpoint detection for surface defect removal on aircraft components," *2023 IEEE/SICE International Symposium on System Integration (SII)*, Atlanta, GA, USA, 2023, pp. 1-6, doi: 10.1109/SII55687.2023.10039239.

[16] K. Song, Y. Bao, H. Wang, L. Huang, and Y. Yan, "A potential vision-based measurements technology: information flow fusion detection method using RGB-thermal infrared images," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, 2023,

doi: 10.1109/TIM.2023.3236346.

[17] R. S. Shankar, A. B. Krishna, J. Rajanikanth, and C. H. S. Rao, “Implementation of object oriented approach to query processing for video subsequence identification,” *2012 National Conference on Computing and Communication Systems, NCCCS 2012*, pp. 126–130, 2012, doi: 10.1109/NCCCS.2012.6412979.

[18] I. Ahmed, M. Ahmad, A. Chehri, M. M. Hassan, and G. Jeon, “IoT enabled deep learning-based framework for multiple object detection in remote sensing images,” *Remote Sensing*, vol. 14, no. 16, 2022, doi: 10.3390/rs14164107.

[19] C. Loraksa, S. Mongkolsomlit, N. Nimsuk, M. Uscharapong, and P. Kiatisevi, “Development of the osteosarcoma lung nodules detection model based on SSD-VGG16 and competency comparing with traditional method,” *IEEE Access*, vol. 10, pp. 65496–65506, 2022, doi: 10.1109/ACCESS.2022.3183604.

[20] W. Ma *et al.*, “Feature split-merge-enhancement network for remote sensing object detection,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, 2022, doi: 10.1109/TGRS.2022.3140856.

[21] Q. Wang, Y. Liu, Z. Xiong, and Y. Yuan, “Hybrid feature aligned network for salient object detection in optical remote sensing imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, 2022, doi: 10.1109/TGRS.2022.3181062.

[22] R. S. Shankar, L. V. Srinivas, V. V. S. Raju, and K. V. S. S. Murthy, “A comprehensive analysis of deep learning techniques for recognition of flower species,” *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, Tirunelveli, India, 2021, pp. 1172-1179, doi: 10.1109/ICICV50876.2021.9388503.

[23] G. B. Loganathan, T. H. Fatah, E. T. Yasin, and N. I. Hamadamen, “To develop multi-object detection and recognition using improved GP-FRCNN method,” *8th International Conference on Smart Structures and Systems, ICSSS 2022*, 2022, doi: 10.1109/ICSSS54381.2022.9782296.

[24] R. S. Shankar, J. Raghaveni, P. Rudraraju, and Y. V. Sravya, “Classification of gender by voice recognition using machine learning algorithms,” *Journal of Critical Reviews*, vol. 7, no. 9, pp. 1217–1229, 2020, doi: 10.31838/jcr.07.09.222.

[25] L. Yasaswini, G. Mahesh, R. S. Shankar, and L. V. Srinivas, “Identifying road accidents severity using convolutional neural networks,” *International Journal of Computer Sciences and Engineering*, vol. 6, no. 7, pp. 354–360, 2018, doi: 10.26438/ijcse/v6i7.354360.

[26] H. Abdu and M. H. M. Noor, “Domestic trash classification with transfer learning using VGG16,” *ICCSCE 2022 - Proceedings: 2022 12th IEEE International Conference on Control System, Computing and Engineering*, pp. 137–141, 2022, doi: 10.1109/ICCSCE54767.2022.9935653.

[27] P. Shyam, S. Mishra, K. J. Yoon, and K. S. Kim, “Infra sim-to-real: an efficient baseline and dataset for infrastructure based online object detection and tracking using domain adaptation,” *2022 IEEE Intelligent Vehicles Symposium (IV)*, Aachen, Germany, 2022, pp. 1393-1399, doi: 10.1109/IV51971.2022.9827395.

[28] Y. Li, Y. Gong, and Z. Zhang, “Few-shot object detection based on self-knowledge distillation,” *IEEE Intelligent Systems*, pp. 1-8, 2022, doi: 10.1109/MIS.2022.3205686.

[29] C. R. Pravallika and R. Baskar, “Image processing-based brain tumor classification using VGG16compared with SVM to improve accuracy,” *Proceedings of the 2022 11th International Conference on System Modeling and Advancement in Research Trends, SMART 2022*, pp. 1398–1401, 2022, doi: 10.1109/SMART55829.2022.10047671.

[30] M. Sajjad and D. Hemavathi, “Comparative analysis of different approaches to object detection: a survey,” *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)*, Trichy, India, 2022, pp. 1571-1574, doi: 10.1109/ICOSEC54921.2022.9952000.

[31] H. Yang, Z. Yang, A. Hu, C. Liu, T. J. Cui, and J. Miao, “Unifying convolution and transformer for efficient concealed object detection in passive millimeter-wave images,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 33, no. 8, pp. 3872–3887, 2023, doi: 10.1109/TCSVT.2023.3234311.

BIOGRAPHIES OF AUTHORS

Dr. Nilambar Sethi is an Associate Professor in GIET University, Odisha, India. He completed his Ph.D. degree from Berhambpur University on 2013. Under his guidance 02 Ph.D. scholars got awarded. Life member of IE and ISTE. He has published nearly 30+ papers in various international journals and conferences. His research area interests are data mining, machine learning, deep learning, and image processing. He is the reviewer for various peer reviewed international journals and conferences. He can be contacted at email: nilambar@giet.edu.

Dr. Vetukuri Venkata Siva Rama Raju is an Associate Professor in computer science engineering at KL University, Vijayawada. He completed his Ph.D. at Biju Patnaik University of Technology, Rourkela, Odisha. He has been in the teaching profession for more than 13 years. He has published 22 papers in national and international journals, conferences, and symposiums. His main areas of interest include big data analysis, machine learning, and computer networks. He can be contacted at email: sivaramaraju.vetukuri@gmail.com or sivaramarajuvetukuri@kluniversity.in.

Venkata Srinivas Lokavarapu is an Assistant Professor in the Department of Computer Science and Engineering in Sagi Rama Krishnam Raju Engineering College, Bhimavaram, Andhra Pradesh, India. He Received B.Tech. and M.Tech. degrees from Andhra University, Visakhapatnam. Currently he is pursuing Ph.D. in computer science and engineering at Andhra University. His research areas are machine learning, edge computing, natural language processing, and image processing. He can be contacted at email: srinivas.srkrcse@gmail.com.

Dr. Ravi Babu Devareddi is an Sr. Assistant Professor working at GITAM University, Hyderabad, Telangana. He received his B.Tech. (computer science & engineering) degree in 2005 from Andhra University, obtained M.Tech. (computer science& technology) degree from Andhra University in 2009 and completed his Ph.D. in computer science and engineering in Acharya Nagarjuna University, Guntur, India. His current research interests are in internet of things, image processing, computer vision, and artificial intelligence. He can be contacted at email: ravibabu.devareddi@gmail.com.

Dr. Shiva Shankar Reddy is an Assistant Professor at the Department of Computer Science and Engineering in Sagi RamaKrishnam Raju Engineering College, Bhimavaram, Andhrapradesh, India. He was awarded Ph.D. degree in computer science and engineering with a specialization in medical mining and machine learning at Biju Patnaik University of Technology (BPUT), Odisha, India. His research areas are image processing, medical mining, machine learning, deep learning, edge computing, IoT, and pattern recognition. He published 100+ papers in international journals and conferences which were indexed in SCIE/ESCI/Scopus. He has 5 patents and 1 design patent was granted. He can be contacted at email: shiva.csesrkr@gmail.com.

Dr. Silpa Nrusimhadri is working as an Assistant Professor in the Department of Computer Science & Engineering at Shri Vishnu Engineering College for Women (A), Andhra Pradesh, India. She was awarded her Ph.D. in Computer Science and Engineering at Centurion University of Technology and Management (CUTM), Odisha, India. She has 13 years of teaching experience and 8 years of research experience. Her research interests include data mining, web mining, big data analytics, text mining, data science, artificial intelligence, and machine learning. She is actively involved and successfully implemented two projects funded by DST. She has 20 research Scopus-indexed papers. She can be contacted at email: nrusimhadri.silpa@gmail.com.