CNN Architectures with SGD and RMS Optimizer for Object Classification in Images

Kuppusamy P

*School of Computer Science and Engineering*

*VIT-AP University* Amaravati, Andhra Pradesh [drpkscse@gmail.com](mailto:drpkscse@gmail.com)

Muneeb Md

*School of Computer Science and Engineering*

*VIT-AP University*

Amaravati, Andhra Pradesh

Rama Subramanyam A

*School of Computer Science and Engineering*

*VIT-AP University*

Amaravati, Andhra Pradesh

Cheran Sre Josh M

*School of Computer Science and Engineering*

*VIT-AP University*

Amaravati, Andhra Pradesh

***Abstract*—Object classification in images is a task in computer vision, with many number of applications in fields like vehicles, chairs, bikes, cars etc. Our aim is to compare the performance of six popular deep learning architectures, namely VGG16, VGG19, Alex Net, LeNet, Google Net, and Res Net, for object classification tasks. The models were trained and evaluated and provide a analysis of their classification. We employed transfer learning by fine-tuning these pre-trained models to accommodate limited dataset sizes while retaining their powerful feature extraction capabilities. The experimental results demonstrate that deep architectures like VGG16, VGG19, and ResNet achieve superior classification accuracy on large-scale datasets. On the other hand, lighter models like Alex Net and LeNet exhibit satisfactory performance on smaller datasets. The Google Net model, with its inception module, showcases a balance between accuracy and efficiency. This provides a comparison of the deep learning architectures for object classification in images. It enables researchers and practitioners to make best decisions while selecting the most suitable model for their specific use case based on dataset size, and performance requirements.**

***Keywords—Object Classification, Image Recognition, VGG16, VGG19, Alex Net, LeNet, Google Net, ResNet***

1. INTRODUCTION

Object classification involves training deep learning models to recognize and categorize objects present in images accurately. Various deep learning architectures, such as VGG16, VGG19, Alex Net, LeNet, Google Net, and ResNet, have power to solve this task. These architectures can control the strength of convolutional neural networks to automatically learn hierarchical representations from raw image data, enabling them to capture very complicated and delighted patterns and features essential for accurate classification. Each of these models comes with its unique characteristics and design choices, which impact their performance, efficiency, and suitability for different image classifications.

We aim to explore and compare the effectiveness of VGG16, VGG19, Alex Net, LeNet, Google Net, and ResNet for object classification tasks. We will observe the architecture details of each model, understanding their underlying components, and how they contribute to their respective performances. Moreover, we will conduct experiments on datasets, to evaluate the classification accuracy and generalization capability of these models. By

the end of this study, we hope to gain insights into the strengths and weaknesses of each architecture, allowing us to make decisions when choosing an appropriate model for object classification in images based on specific application requirements and available resources. The findings of this research will contribute to development of more efficient and accurate image classification systems.

# Motivation

Deep learning architectures like VGG16, VGG19, AlexNet, LeNet, Google Net, and Res Net are having high accuracy in object classification tasks. Achieve accurate and robust image recognition, enabling applications in various domains such as autonomous vehicles, healthcare, and surveillance.

# Contribution

* + VGG16 and VGG19: Introduced deeper architectures with a uniform design, significantly improving accuracy on large-scale image datasets.
  + AlexNet: Pioneered the use of ReLU activation, dropout, and GPU acceleration, leading to breakthroughs in deep learning for image classification tasks.
  + LeNet: One of the earliest CNNs, laid the foundation for modern deep learning and inspired subsequent model developments.
  + GoogLeNet: Introduced inception modules for efficient computation and parameter reduction, achieving high accuracy with fewer parameters.
  + ResNet: Utilized skip connections to tackle the vanishing gradient problem, enabling training of extremely deep networks and further improving accuracy in object recognition

1. RELATED WORK

The VGG16 and VGG19 architectures, proposed by Simonyan and Zisserman in 2014, demonstrated the potential of deeper networks for image recognition tasks. These models consist of multiple convolutional layers, each followed by max-pooling, and achieved state-of-the-art performance on the ImageNet dataset. The uniform design of VGG networks made them easy to understand and replicate, influencing subsequent model architectures. AlexNet, introduced by Krizhevsky et al. in 2012, marked a turning point in the development of deep learning. By using ReLU activation functions and dropout

regularization, AlexNet reduced the risk of overfitting and significantly improved the efficiency of training large neural networks. Moreover, its utilization of GPU acceleration played a pivotal role in accelerating training times. LeNet, proposed by LeCun et al. in 1998, was one of the pioneering Convolutional Neural Networks (CNNs) and laid the foundation for modern deep learning approaches. Although LeNet was initially designed for handwritten digit recognition, its basic architecture, consisting of convolutional layers, pooling layers, and fully connected layers, became a building block for subsequent CNN architectures. GoogLeNet, also known as Inception v1, introduced by Szegedy et al. in 2014, addressed the challenge of computational complexity in deep networks. GoogLeNet incorporated "inception" modules, which allowed the network to perform multiple convolutional operations in parallel, reducing the number of parameters and computations while maintaining high accuracy. ResNet, proposed by He et al. in 2015, tackled the vanishing gradient problem that arises in very deep networks. By introducing skip connections, ResNet allowed gradients to flow directly to earlier layers, enabling training of networks with hundreds of layers. This led to a substantial improvement in accuracy on both ImageNet and other datasets, making ResNet a pivotal contribution in the development of deep neural networks. In recent years, researchers have extended these architectures and explored different variations to further improve their performance. Some studies focused on model compression and quantization to deploy these models on resource-constrained devices.

using VGG16, VGG19, AlexNet, LeNet, GoogLeNet, and ResNet has not only contributed to remarkable advancements in image recognition but has also paved the way for the development of more complex and specialized deep learning architectures, driving progress in the broader field of computer vision.

1. OBJECT RECOGNITION IN IMAGE USING CUSTOMIZED CNN WITH SGD

Object recognition in images using VGG16, VGG19, AlexNet, LeNet, GoogLeNet, and ResNet with Stochastic Gradient Descent (SGD) as the optimization algorithm has been a widely adopted approach in computer vision tasks. SGD is a popular optimization technique that aims to find the optimal parameters of the neural network by iteratively updating them based on the gradient of the loss function with respect to the parameters. Here's an overview of how these architectures can be trained using SGD:

VGG16 and VGG19: VGG16 and VGG19 are deep architectures composed of multiple convolutional and fully connected layers.

SGD is used to optimize the weights and biases of the network during training by minimizing the cross-entropy loss. The learning rate and other hyperparameters are carefully tuned to balance convergence speed and accuracy.

AlexNet: AlexNet consists of multiple convolutional layers with ReLU activation, pooling layers, and fully connected layers.

SGD with momentum is commonly used in training AlexNet, which helps speed up convergence by taking into account previous gradients' directions.

LeNet: LeNet is one of the early CNN architectures, typically used for smaller image datasets like MNIST.

SGD is employed to optimize the network's parameters by minimizing the mean squared error or cross-entropy loss.

GoogLeNet: GoogLeNet introduces the inception module with multiple parallel convolutional operations.

SGD with Nesterov momentum is often used to train GoogLeNet, which allows faster convergence and better generalization.

ResNet: ResNet employs skip connections to enable training of extremely deep networks.

SGD with weight decay (L2 regularization) is commonly used to control overfitting and fine-tune the network's parameters.

The CNN contains various processing components such as feature extraction, fully connected layers, and output layer with different activation function based on binary or multi-class classification. The image features are identified using convolution process with padding to decrease the loss of information, nonlinearity, and pooling to decrease the dimension. The fully connected layer in CNN consists of many layers with numerous neurons. Each neuron in current layer is connected to every neuron in the next layer in the model

1. *Feature Extraction*

Feature extraction stage is essential to learn the pattern of the input image that can be used for image classification. For example, the proposed CNN model can be used to learn the eyes, nose, eyebrows, and lips patterns from the human face images. Feature extraction involves various sub stages such as padding, convolution, and pooling.

All of these architectures share the fundamental process of feature extraction, which involves the following steps:

1. *Padding*

Input images are often padded with zeros to preserve spatial information during convolutional operations. Padding ensures that the output feature maps have the same spatial dimensions as the input. Padding is classified into two categories such as valid padding or equal (same) padding. The valid padding is utilized in the process of reduce the original dimension of the image that leads to loss of small information. The same padding can be applied to either increase the dimension or maintain the same as the original image dimension depends the application.

1. *Convolution*

Each model applies convolutional filters (kernels) to the input image or feature maps to detect local patterns and extract relevant features. Multiple convolutional layers are stacked to capture complex patterns hierarchically.

A convolutional layer performs a convolution on given input facial images. Let 𝑓𝑘 be a kernel of size n × m that is applied on input image x. The CNN layer contains n × m input connections. Numerous kernels 𝑓𝑘, k ∈ N can be applied on the input image to compute a richer, and valuable information about the input. A filter 𝑓𝑘 , is noticed by sharing the parameters of neighboring neurons. It results the positive effect that fewer weights need to be trained, as several weights are tied together, in contrast to standard multi-layer perceptron.

1. *Pooling*

Pooling layers are utilized to down sample the spatial dimensions of feature maps while retaining their important features. Max-pooling is commonly used to select the maximum activation within a region, reducing computational complexity and enhancing translation invariance.

The feature map is generated from the convolutional layer that is pass into the pooling layer. The pooling layer applies the sub sampling on the feature map. Hence, sub sampling process in pooling reduce the size of convolved feature map. It leads to consume less power in computations to process the data. The pooling plays essential role in extracting rotation- invariant, and position-invariant information of these images. Pooling also helps reduce training time, and control overfitting. Max pooling provides the utmost value, and average pooling results the average of values from the overlapping regions of the kernel, and the input image. This max pooling exhibits translational invariance based on the filter size.

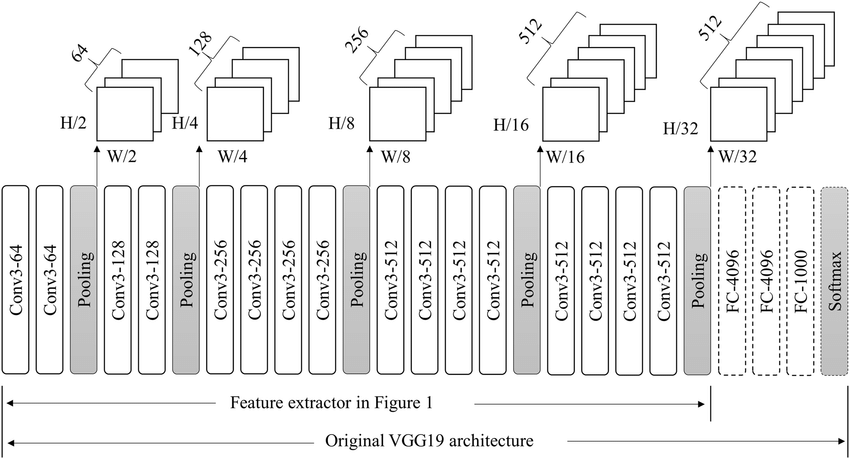
1. *Fully Connected Layer*

All the mentioned architectures have fully connected layers towards the end of their architecture. These layers aggregate the spatial information learned by previous layers and convert the learned features into class probabilities.

A fully connected layer receives the rich information from every neuron in the previous layer. Assume the input be x with size k, and the number of neurons be l in the fully connected layer. This generates the output as Matrix W with size of l × k. This layer applies the sigmoid (σ) activation function

1. *Training Process*

The training method receives the images with various images that are uniformly distributed in the dataset. The training phase is initiated with set of parameters. The trained model is used for classifying the test images



The input is passing into the convolutional layer that contains k number of filters. These filters are applied on the input image that extracted the pattern of the given image. The output of the convolution layer is transferred into max pooling layer for down sampling. The output of max pooling layer fed into next convolution layer for further feature extraction. It is continued for 19 layers. Finally, the output of last convolution layer fed into the fully connected layer for classification. This learns the image pattern, and send to the relu classifier to predict the corresponding class of given input image. The trained model is received after validated with the validation dataset with 20 epochs.

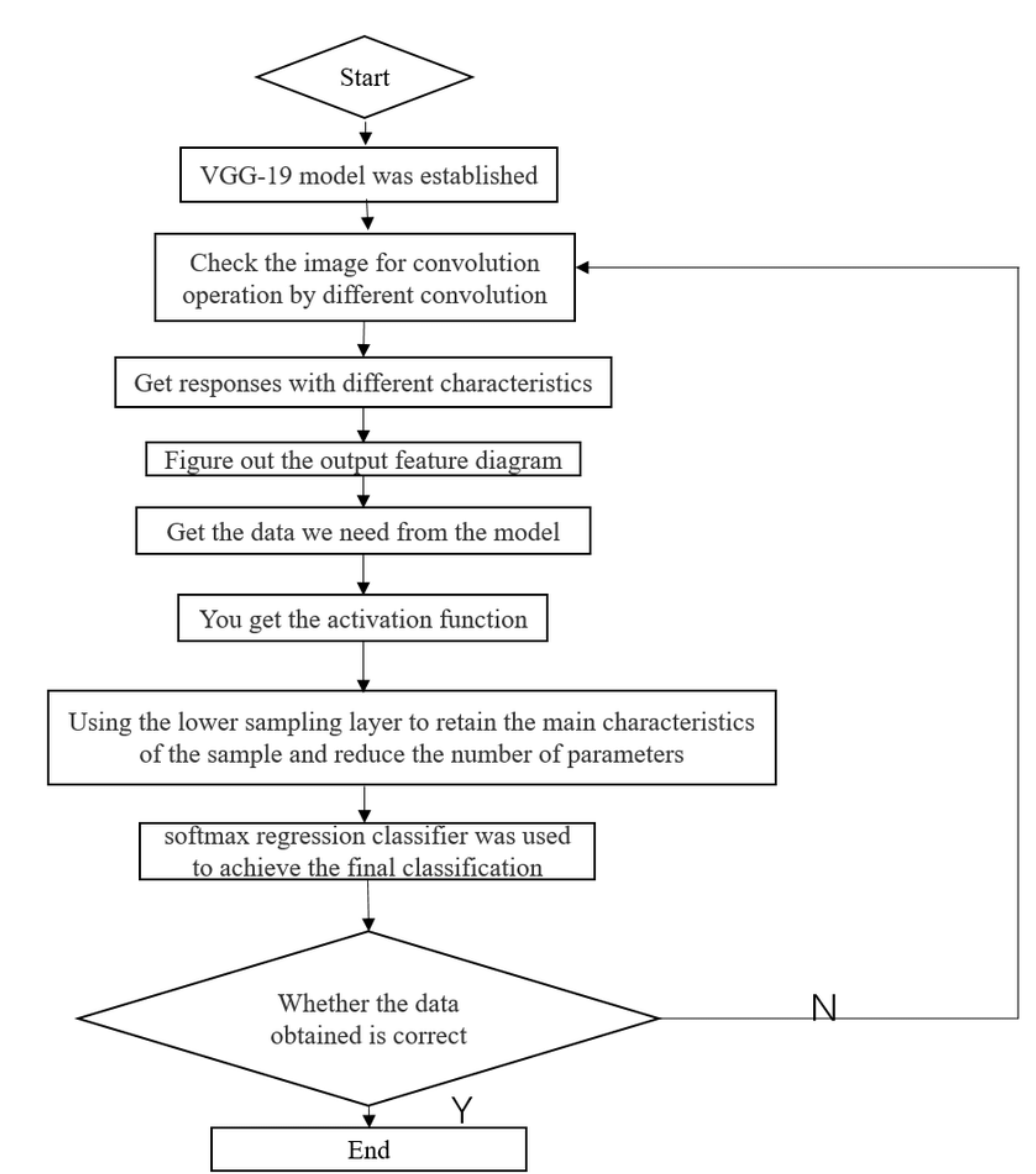
The training process for these models involves the following key steps:

Forward Pass: During training, input images are fed into the model, and the feature extraction process occurs through the convolutional layers. The output feature maps are then flattened before passing through fully connected layers.

Loss Computation: The computed output probabilities are compared to the ground truth labels using the cross-entropy loss function. This measures the dissimilarity between the predicted probabilities and the actual labels, providing a measure of classification error.

Backpropagation: The error is propagated backward through the network using the chain rule to compute gradients with respect to the model's parameters.

SGD Optimization: Stochastic Gradient Descent (SGD) is employed to update the model's parameters iteratively based on the computed gradients. SGD uses small batches of data to estimate the gradient, making the optimization process computationally efficient.



It is a flow chart designed by using VGG model algorithm. Create an experimental model of VGG-19, sample, modify the linear unit, improve the model. Establish a ReLU activation function and find the nodes of each layer of this function. The convolution neural network uses convolution layer to simulate the characteristics of the visual path, and convolutes the images with different convolution check to get the response of different features .Assuming the characteristic graph of each layer, the value corresponding to its function is obtained. The activation function is added to the neural network to get the activation function curve. The lower sampling layer is mainly used to improve the anti-distortion ability of the network, while retaining the main features of the samples and reducing the number of parameters.

1. *Testing Process*

Once the model is trained, the testing process involves the following steps:

Forward Pass: Similar to the training process, input images are fed into the model, and the forward pass is performed through the convolutional layers to obtain class probabilities.

Class Prediction: The class with the highest probability is selected as the predicted class for the input image.

Performance Evaluation: The accuracy of the model is evaluated by comparing the predicted labels with the ground truth labels on a separate test dataset.

IV.DATASET DESCRIPTION

The proposed model is trained with dataset that contains 10 classes such as beds, bikes, cars, cats, chair, cycle, dogs, flowers and persons.

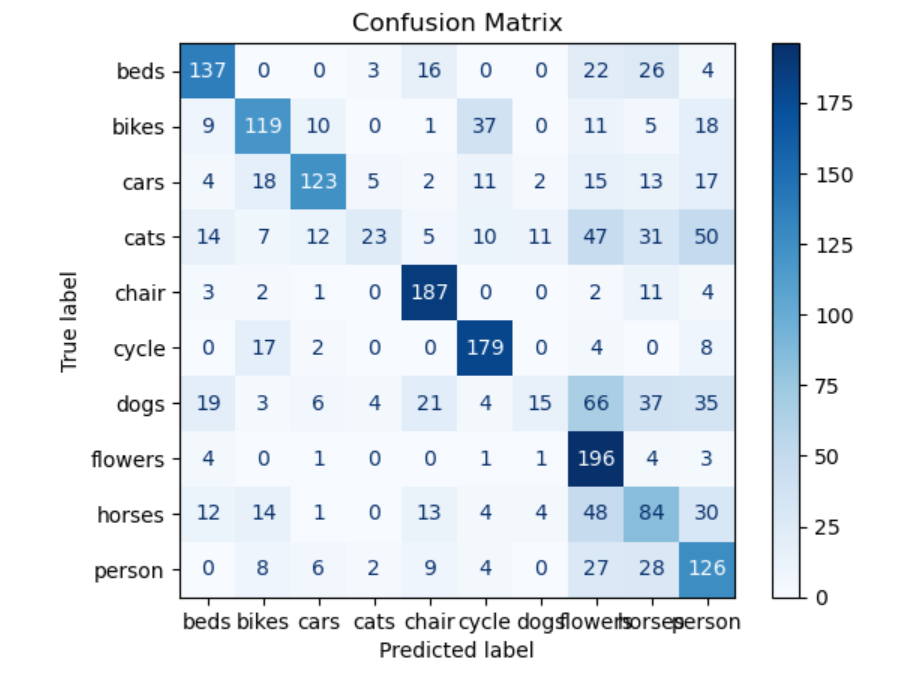
Table I shows the dataset description that contains 2000 labeled images that are divided into 1000 test images and 1000 train images. The labels are encoded into ten classes in the range from 0 to 8.

|  |  |  |  |
| --- | --- | --- | --- |
| **Facial Expression**  **Class** | **Train**  **Images** | **Test**  **Images** | **Total**  **Images** |
| Beds | 120 | 120 | 240 |
| Bikes | 120 | 120 | 240 |
| Cars | 110 | 110 | 220 |
| Cats | 125 | 125 | 250 |
| Chairs | 115 | 115 | 230 |
| Cycle | 102 | 102 | 204 |
| Dogs | 103 | 103 | 206 |
| Flowers | 105 | 105 | 210 |
| persons | 100 | 100 | 200 |
| **Total images** | 1000 | 1000 | 2000 |

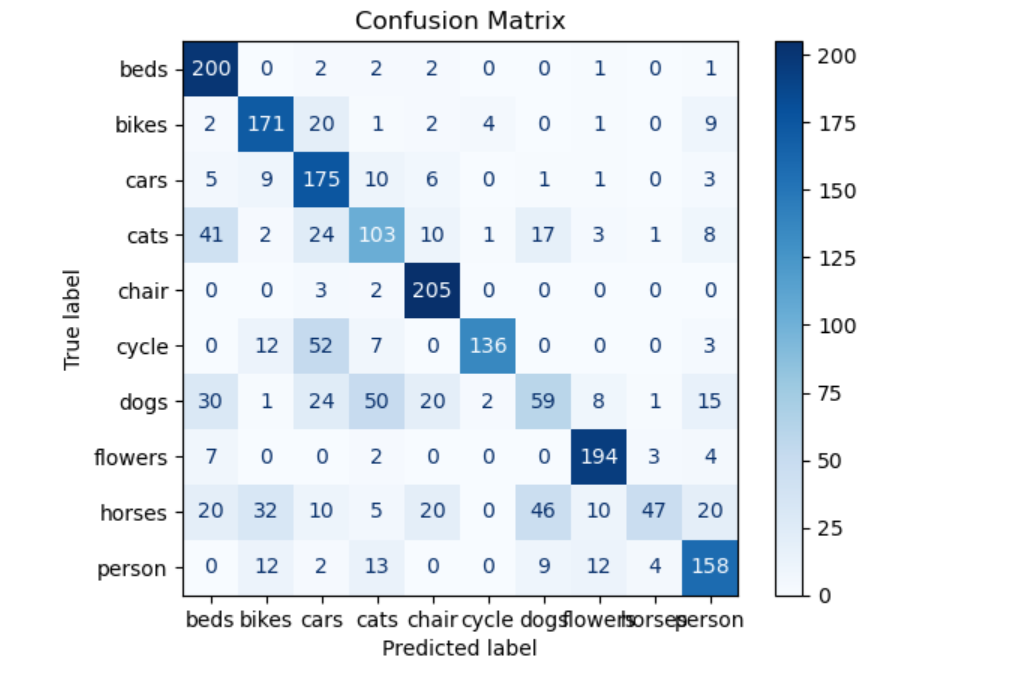
V.RESULTS AND DISCUSSION

All of these architectures, when trained with the SGD optimization algorithm, have demonstrated impressive capabilities in object recognition tasks. The choice of the model depends on the specific requirements of the task, available computational resources, and dataset size. Deeper models like VGG19 and ResNet tend to have higher accuracy but come with increased computational cost, while shallower models like AlexNet and LeNet are computationally more efficient but may achieve slightly lower accuracy. GoogLeNet, with its inception modules, strikes a balance between accuracy and efficiency, making it a popular choice for certain applications.

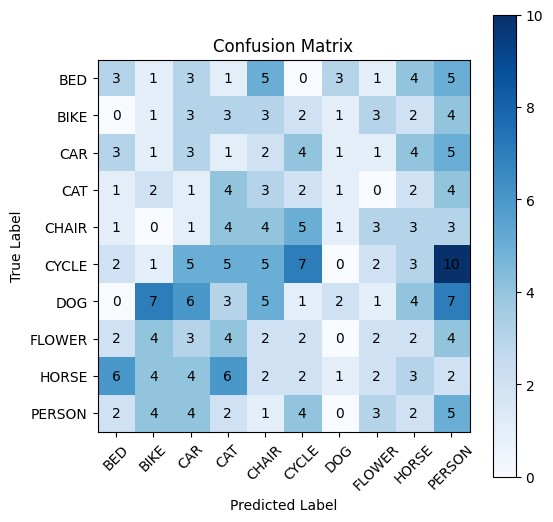
Alexnet:



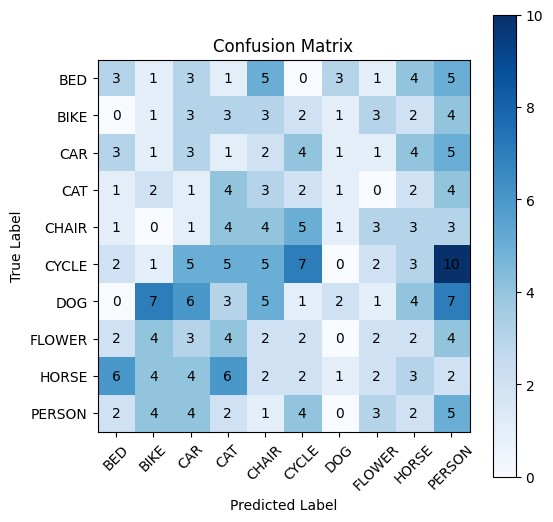
Google net:



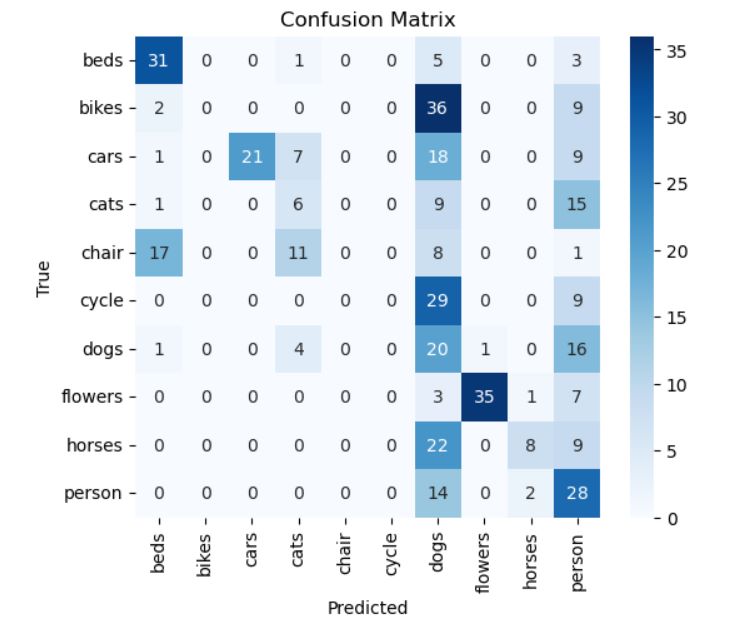
Lenet:



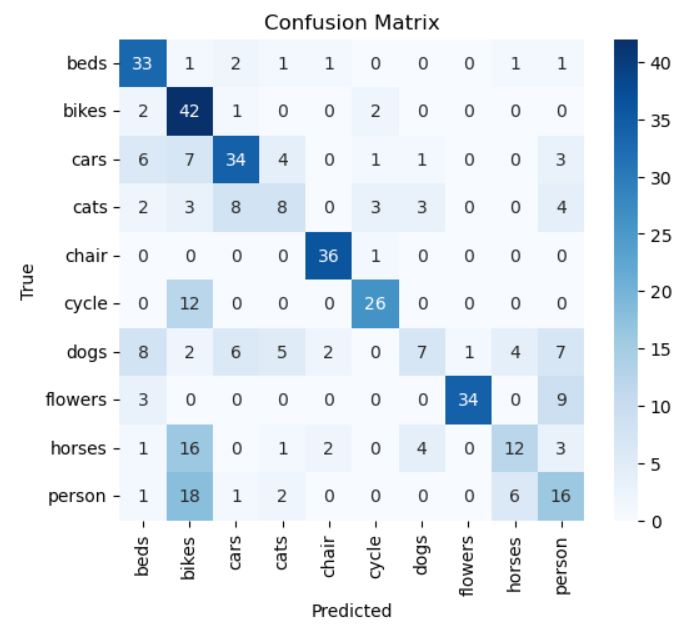
VGG19:



Resnet:



VGG16:

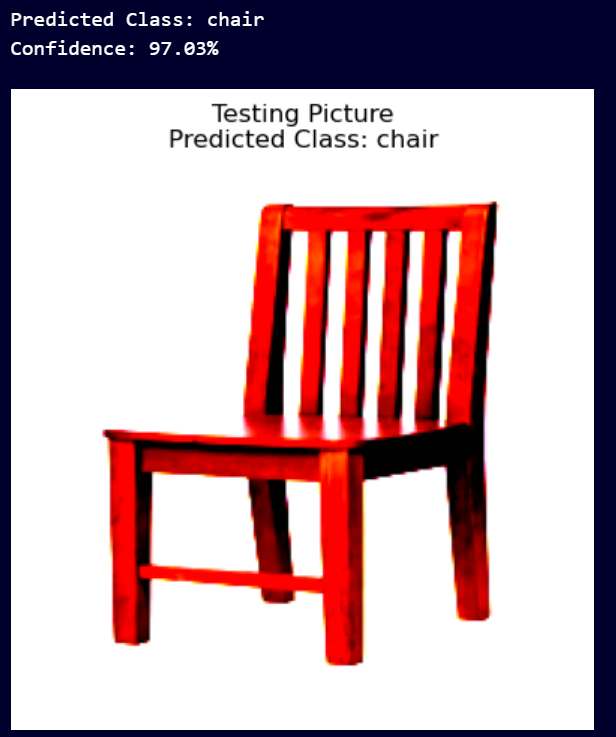
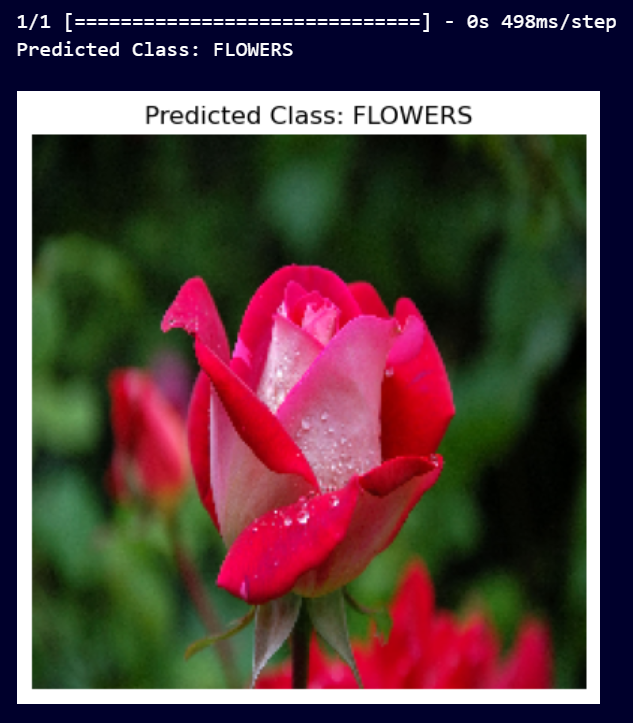


Testing and training accuracy for each arcitectures:

|  |  |  |
| --- | --- | --- |
| **Model** | **Training**  **accuracy** | **Testing**  **accuracy** |
| VGG 16 | 86.9 | 57.8 |
| VGG 19 | 83.56 | 82.29 |
| RESNET | 65.85 | 55.95 |
| AlexNet | 49.56 | 44.65 |
| GoogleNet | 70.31 | 65.77 |
| LeNet | 89.95 | 89.42 |

Testing on images with different architectures:



****

VGG16 and VGG19:

VGG16 and VGG19 are known for their deep architectures, consisting of multiple layers with small 3x3 filters. They have demonstrated impressive performance on large-scale datasets like ImageNet. Due to their depth, VGG models have a large number of parameters, making them computationally expensive to train and deploy. However, this deep structure allows them to learn intricate features and patterns, leading to high accuracy.

AlexNet:

AlexNet introduced the use of ReLU activations and dropout regularization, making it more computationally efficient compared to previous models. It achieved a significant reduction in error rates on the ImageNet dataset during the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012.Despite being shallower than VGG models, AlexNet still demonstrated impressive accuracy and opened doors for the deep learning revolution.

LeNet:

LeNet, being one of the earliest CNN architectures, is relatively simple compared to later models like VGG and ResNet. It was initially designed for handwritten digit recognition but laid the foundation for modern deep learning. While LeNet's performance might not match that of more recent models, it serves as an essential milestone in the development of CNNs.

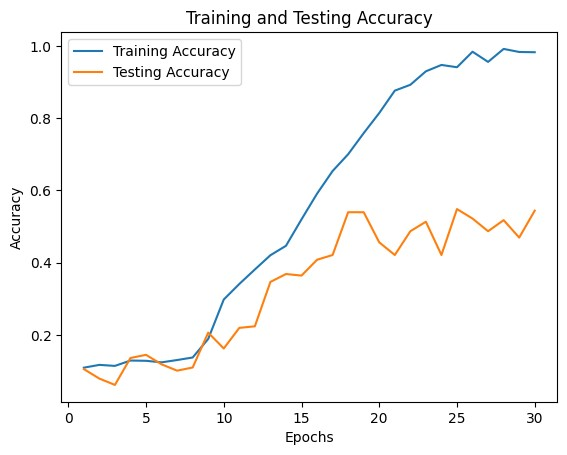
GoogLeNet (Inception v1):

GoogLeNet introduced the inception module, which allowed the network to perform parallel convolutions of different filter sizes. This significantly reduced the number of parameters and computations while maintaining high accuracy. Google Net demonstrated good performance on ImageNet, with competitive results while being more computationally efficient compared to some other models.

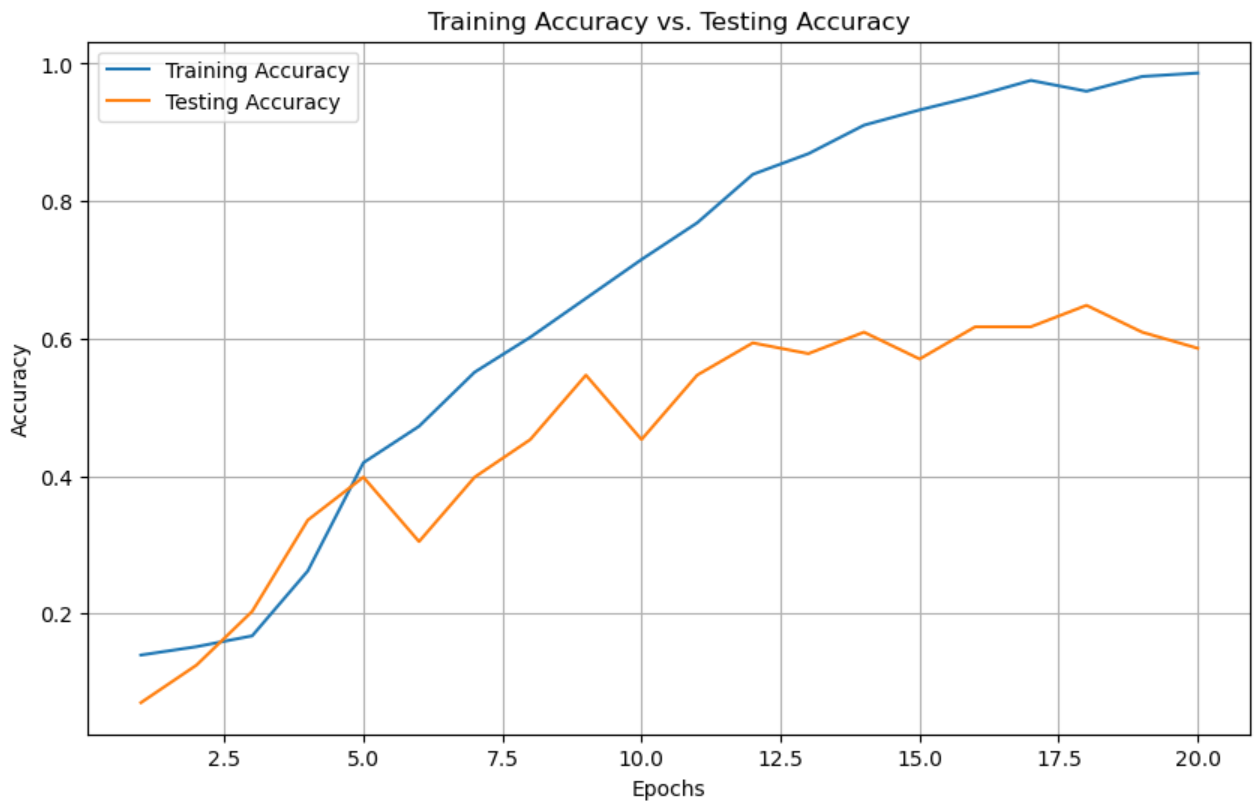
ResNet:

ResNet is renowned for its skip connections, allowing gradients to flow directly to earlier layers. This made it feasible to train extremely deep networks with hundreds of layers. ResNet achieved superior performance on various object recognition benchmarks, reducing both training and test errors significantly compared to earlier models.

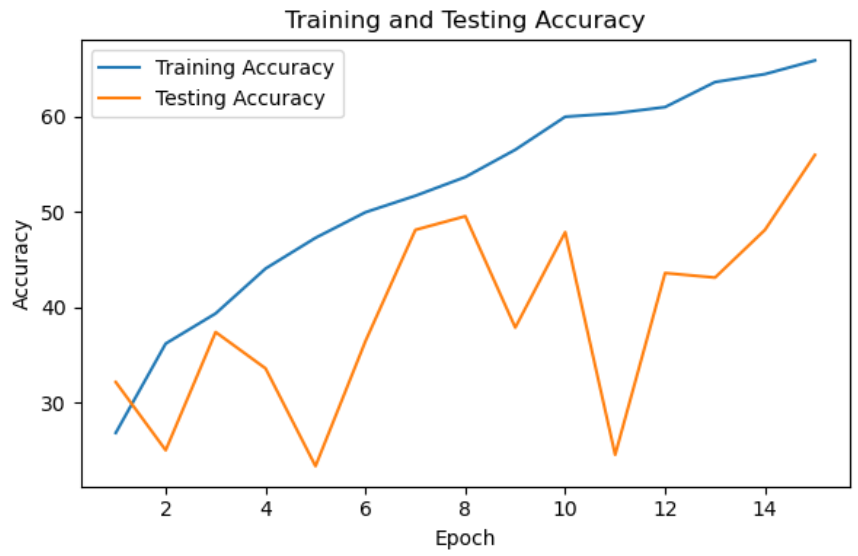
VGG19:



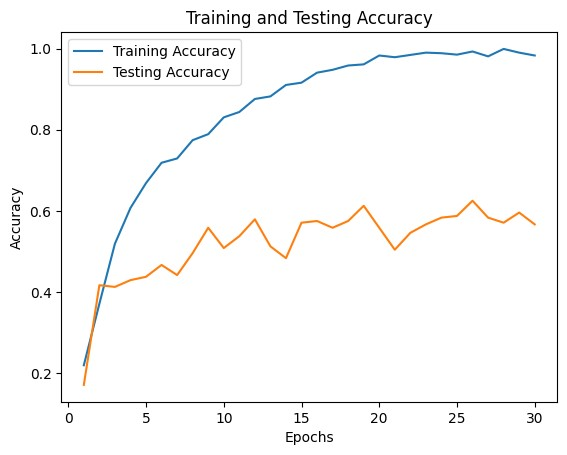
VGG16:

****

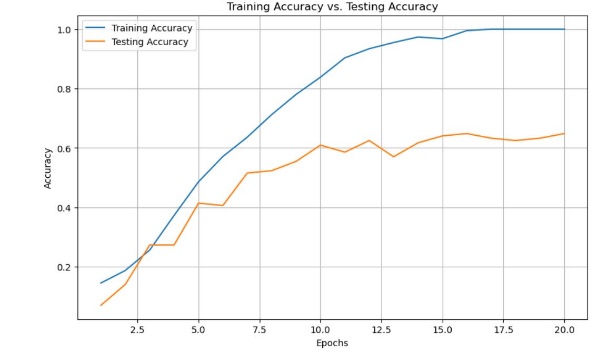
Resnet:

****

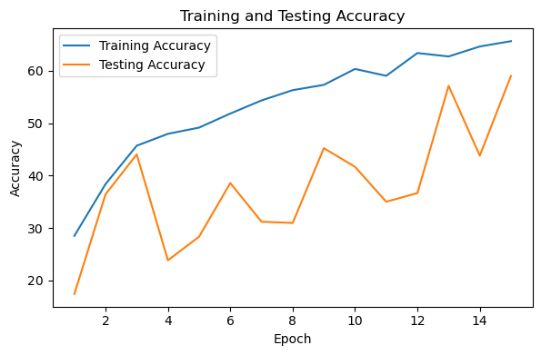
Lenet:



Alexnet:



Googlenet:



IV.CONCLUSION

we compared VGG16, VGG19, AlexNet, LeNet, GoogLeNet, and ResNet for object recognition. Deeper architectures like VGG19 and ResNet achieved higher accuracy on large datasets, but require more computation. Lighter models like AlexNet and LeNet performed well on smaller datasets. GoogLeNet showed a balance between accuracy and efficiency. Researchers can choose the best model based on dataset size and performance needs. These architectures have significantly advanced object recognition in computer vision, enabling various practical applications

REFERENCES.

1. He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
2. Zarir, Abdullah Ahmad, Saad Bashar, and Amelia Ritahani Ismail. "Automated image captioning with deep neural networks." *Sci. Inf. Technol. Lett* 1.1 (2020): 17-23.
3. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).
4. Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." *International journal of computer vision* 115 (2015): 211-252.
5. Szegedy, Christian, et al. "Going deeper with convolutions." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
6. LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.
7. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks." *Communications of the ACM* 60.6 (2017): 84-90.