Comparison of Pretrained Networks for Image Classification

Presenter

Songjalern Kamonpurn (王佩佩) P86107061

Luigi Gan (蔡嘉煌) P86107053

Outline

INTRODUCTION

METHODS

RESULTS

DISCUSSION

Introduction

What is a convolutional neural network (CNN)?

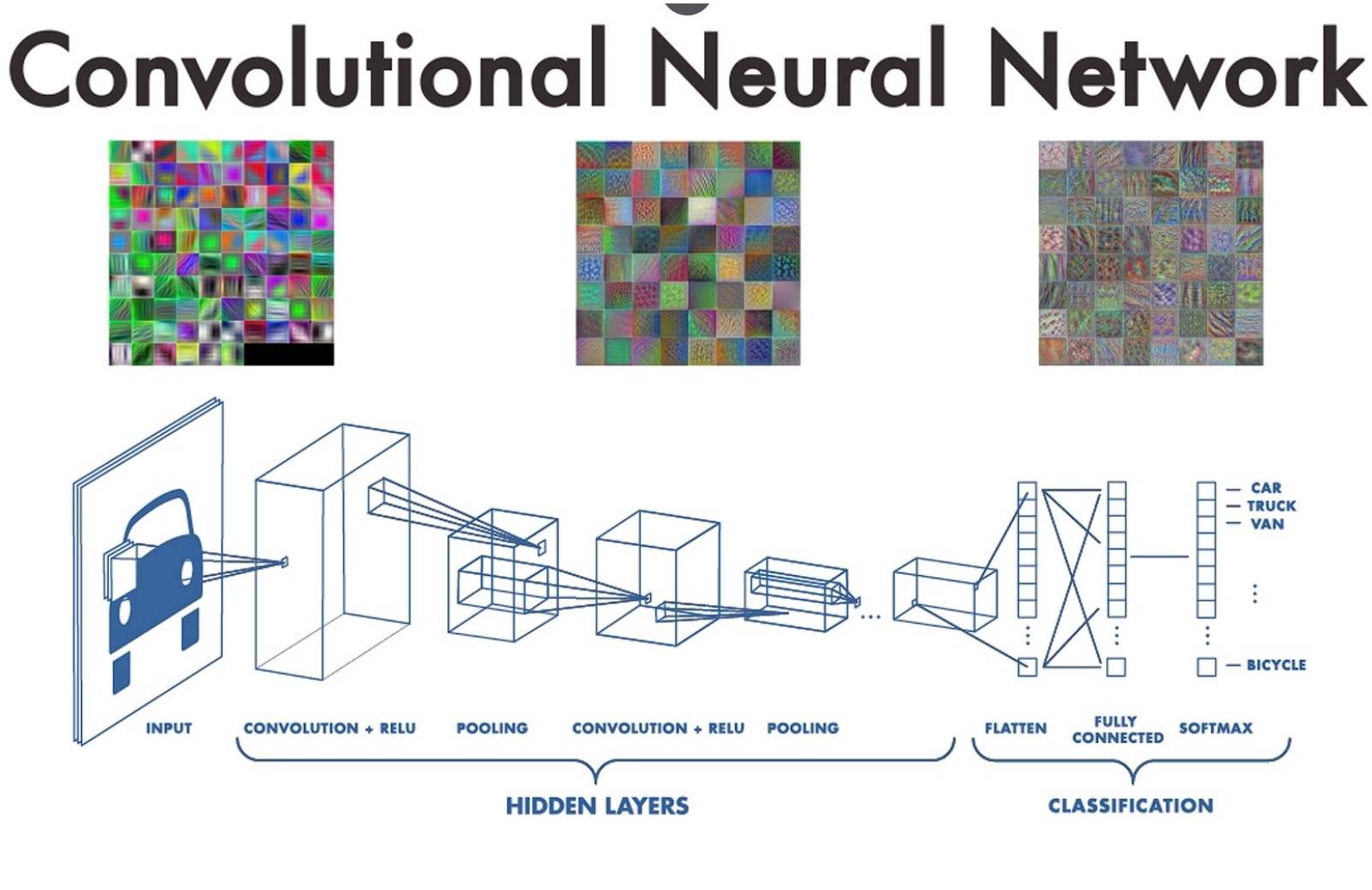
-A network architecture for deep learning that learns directly from data specifically designed for image recognition and processing to recognize objects, classes, and categories.

-CNNs are composed of a series of layers that extract features from the input image these features are used to classify the image into one of several predefined categories.

-CNNs are trained using large datasets of labeled images, and they can learn and improve their performance over time.

-CNN also includes one or more fully connected layers, which are responsible for making the final classification decision based on the features learned by the convolutional layers.

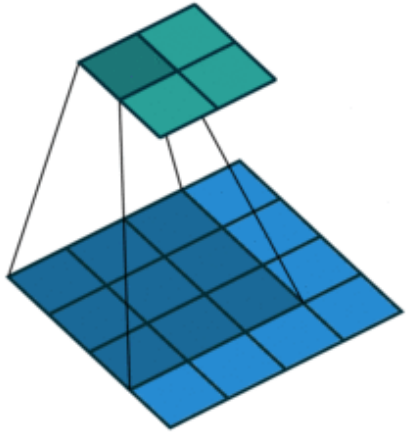
Introduction



Introduction

Convolution

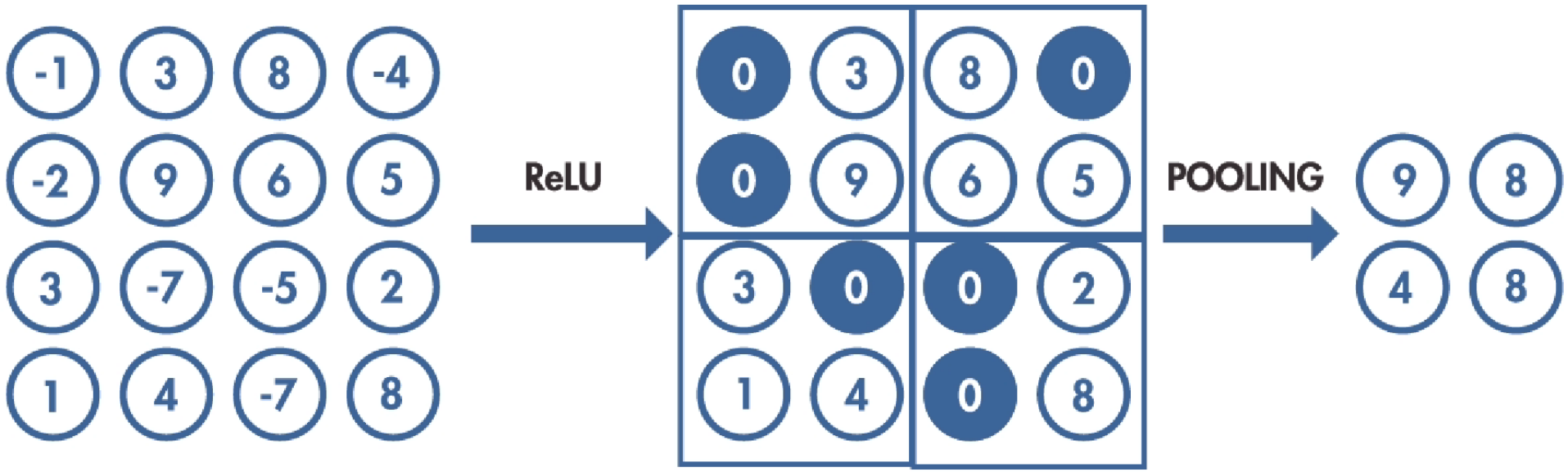
-Images goes through sets of convolutional filters, each of which activates certain features from the images.



-1	-1	-1
-1	8	-1
-1	-1	-1

Introduction

ReLU and Pooling



Introduction

Final layers before classification

Fully connected layer – Combines features learned from previous layers

Softmax layer- Normalizes output of fully connected layer

Classification layer- Uses probabilities returned by softmax layer to assign classes

Introduction

Our project

The goal of this project is to develop an app for classifying images into three categories: dogs, cats, and neither dogs nor cats. The app will use pretrained networks as the basis for the classification model and compare the performance of different pretrained networks models.

Cat

Dog

NotCatorDog

Methods

Dateset : We used a dataset of 2400 images, 800 images for each class.

800 Cat images

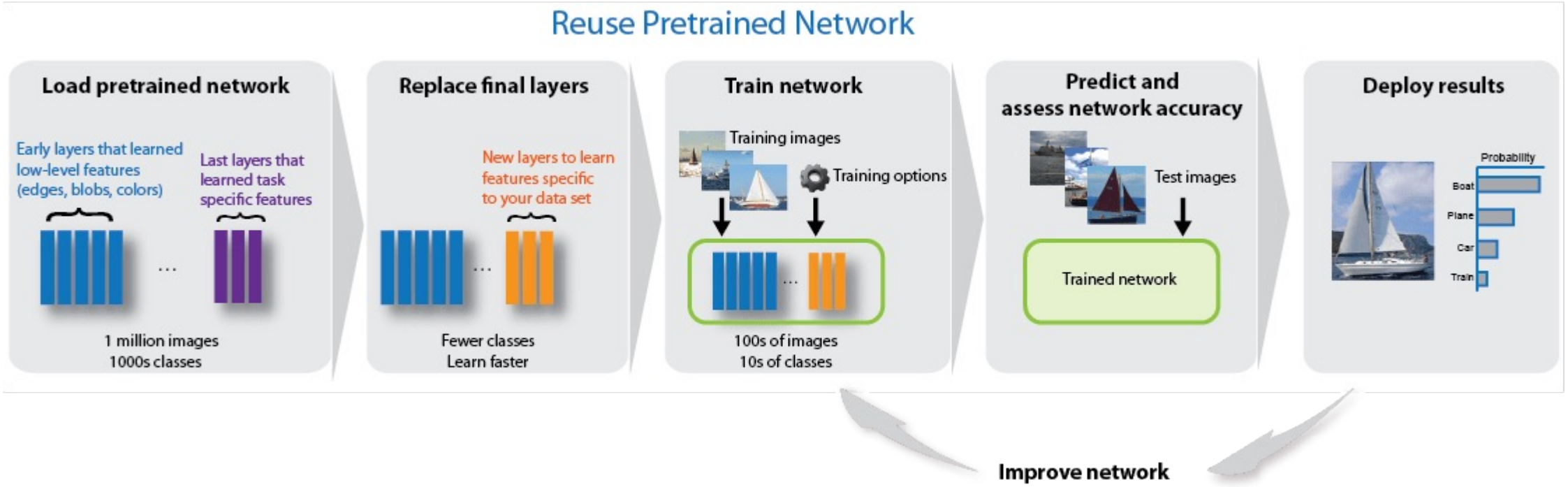
Cat and Dog images retrieved from Kaggle Dog vs Cat dataset https://www.kaggle.com/c/dogs-vs-cats

800 Dog images

NotCatorDog images retrieved from Kaggle Caltech 256 Image Dataset https://www.kaggle.com/datasets/jessicali9530/caltech256.

800 NotCatorDog images(5 categories)

Methods


Reuse Pretrained Network
Load pretrained network
Early layers that learned low-level features (edges, blobs, colors)
Last layers that learned task specific features
1 million images
1000s classes
Replace final layers
New layers to learn features specific to your data set
Fewer classes
Learn faster
Train network
Training images
Training options
100s of images
10s of classes
Predict and assess network accuracy
Test images
Trained network
Deploy results
Probability
Boat
Plane
Car
Train
Improve network

Methods

we followed the steps outlined below

Data preparation

. The data was split into a training set (1920 images) and a validation set (480 images).

Model selection

- ResNet50
- GoogleNet
- AlexNet networks

Data augmentation

scaling to the training dataand pre-processing using the **imageDataAugmenter** object.

Network modification

- modified the final layers of the networks to fit the number of classes in our dataset
- replaced the fully connected and classification layers with new layers and froze the weights of the first 10 layers of the networks.

Training

- trained the modified ResNet50, GoogleNet, and AlexNet networks and the categorical cross-entropy loss function. The network was trained for 10 epochs with a mini-batch size of 10.

APP

- used results networks of ResNet50, GoogleNet, and AlexNet networks to develop an app in MATLAB APP designer

Methods

Data preparation

- Load data into image data stores and split data into training validation and testing sets
- Load the pretrained network

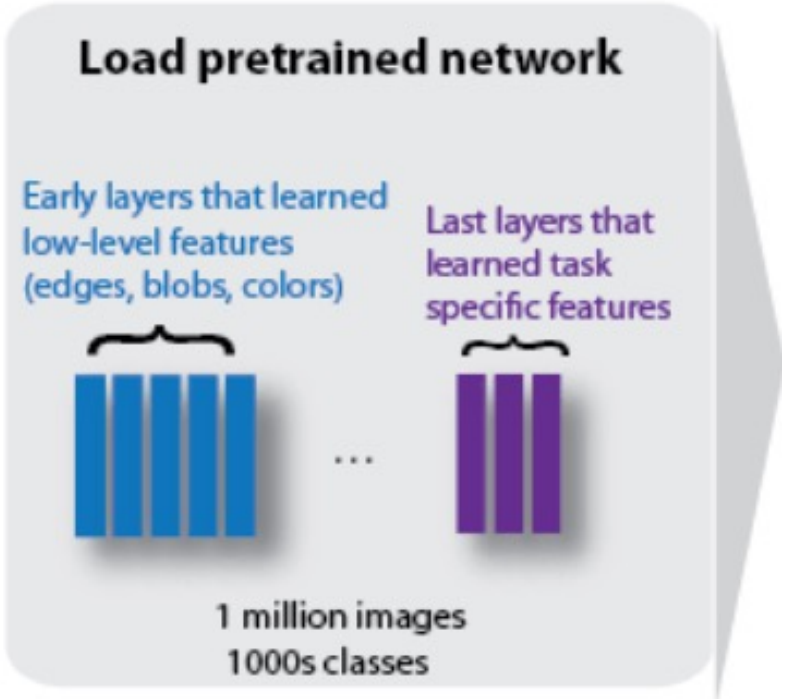
```
% Store the output
imageFolder = fullfile(path,'Dataset');
imds = imageDatastore(imageFolder, 'LabelSource', 'foldernames',
'IncludeSubfolders',true);
```

```
[imdsTrain,imdsValidation] = splitEachLabel(imds,0.8,'randomized');
```

Load Pretrained Network

resnet50

```
net1=resnet50;
net1.Layers(1)
```

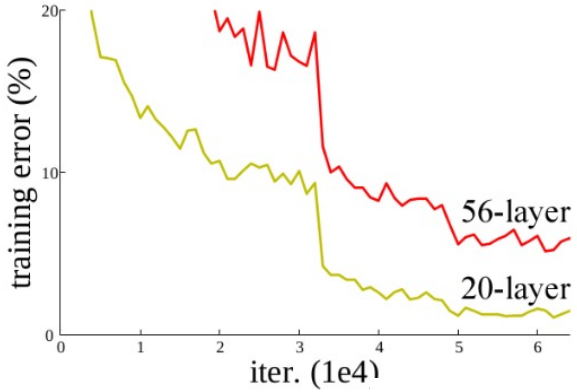


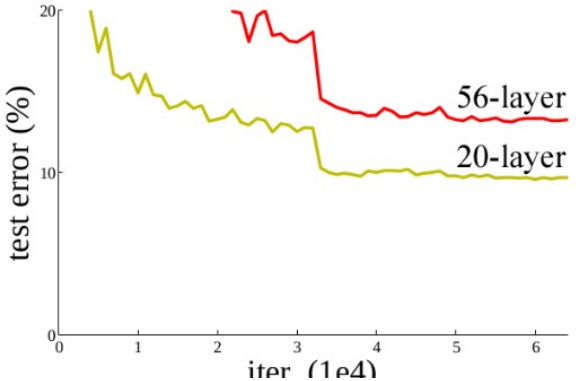
Methods

ResNet-50

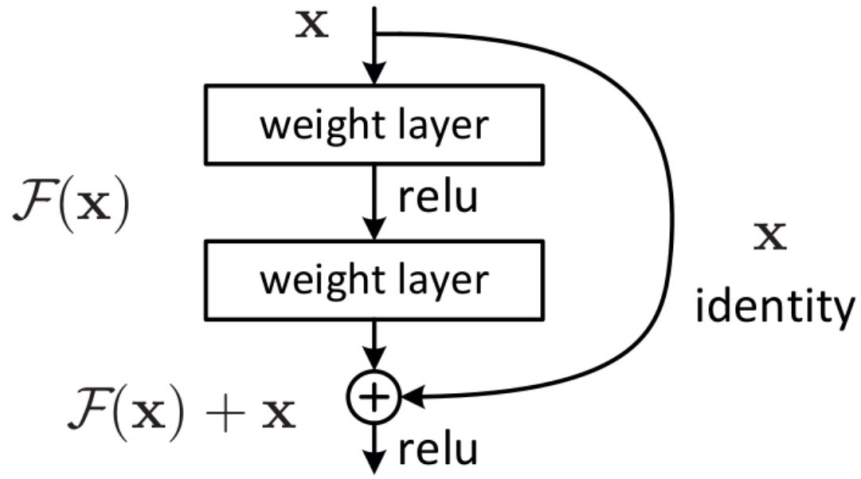
a convolutional neural network that is 50 layers deep

- 48 Convolutional layers
- 2 Pool layers





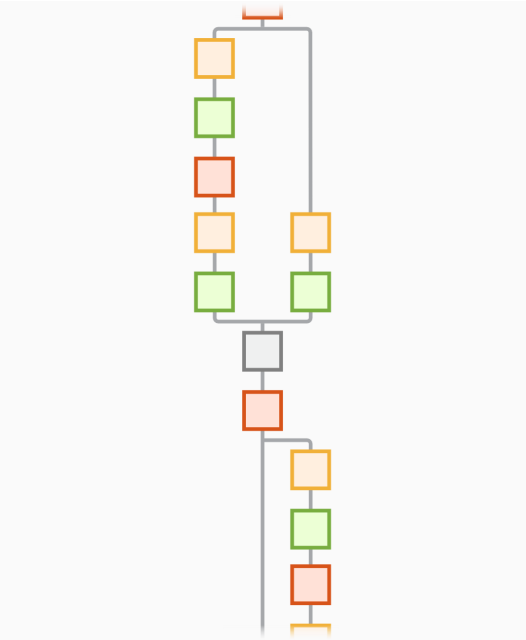
Addresses the problem of “vanishing gradient” in deeper networks which can affect accuracy by skipping connections



Methods

ResNet-50

- Backpropagation- Update parameters of the network to minimize loss
- Changes to be made in the network calculated by the gradient of the loss function
- To go back to earlier layers multiplication of gradients is performed
- Skip connections backpropagate through the identity function (1) thus preserving the gradient



Methods

Alexnet

5 Convolutional layers

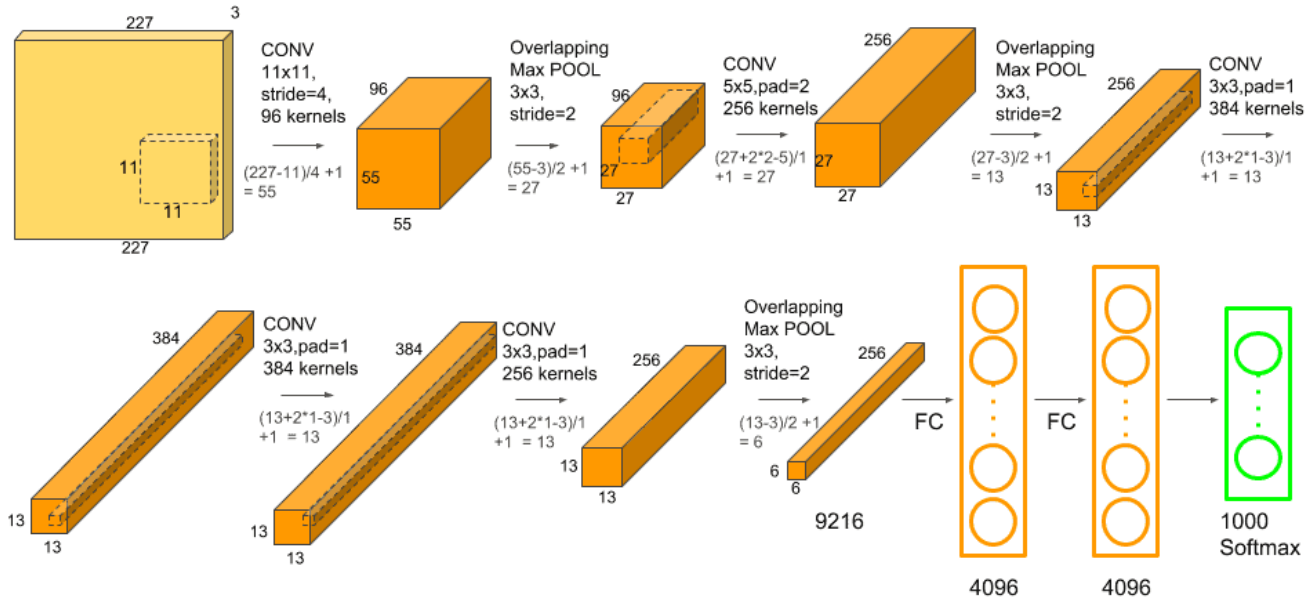
3 Fully connected layers

Introduced the ff.

ReLU

Data augmentation

Dropout

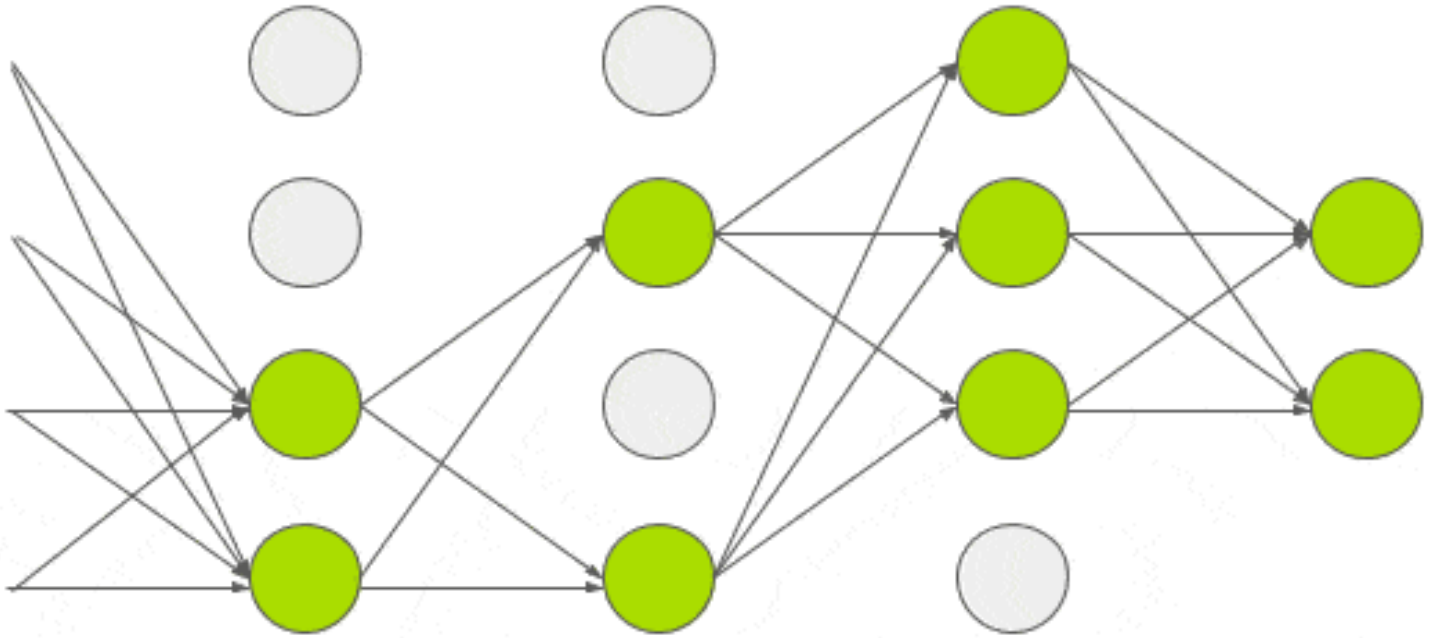


Methods

Data augmentation

Methods

Dropout

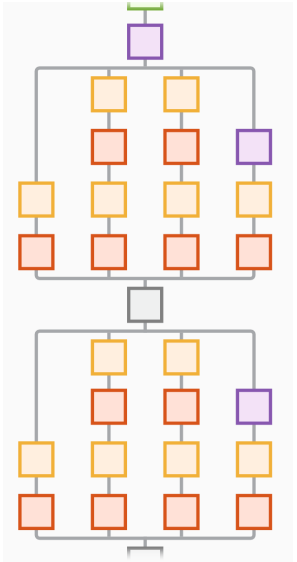


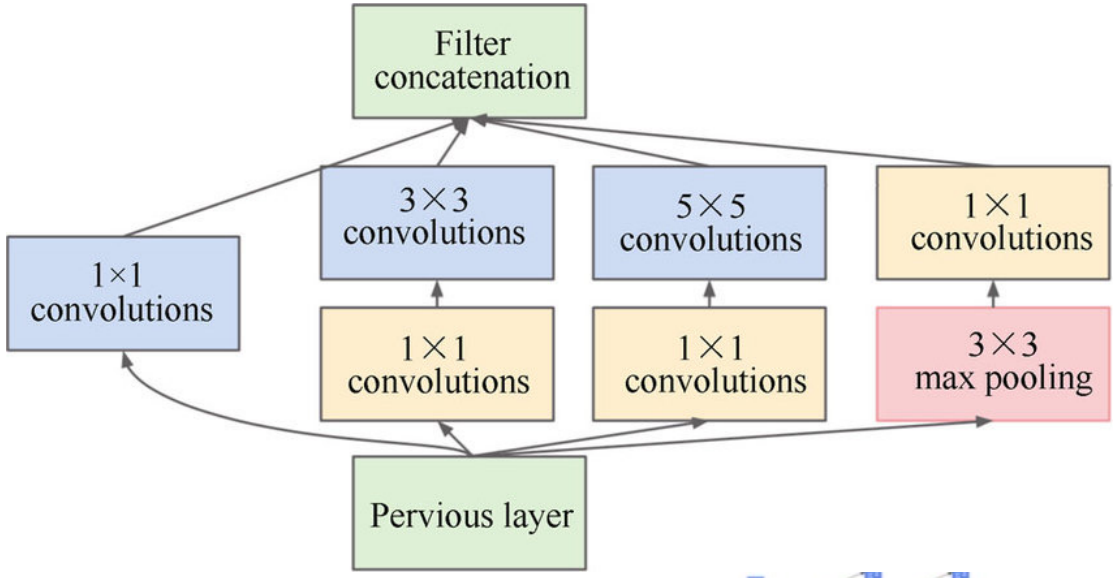
Methods

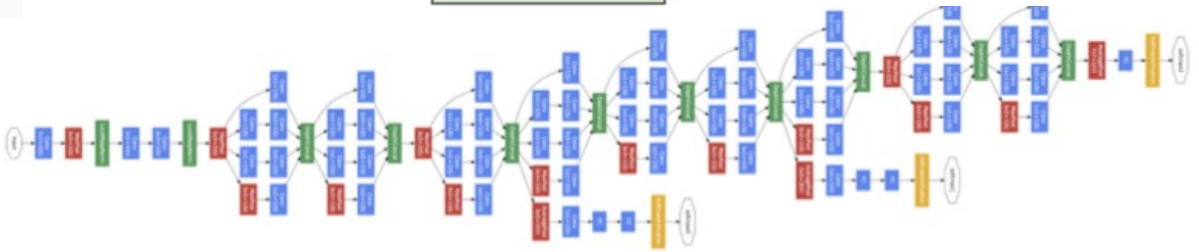
GoogLeNet

22 layers

Multiple convolutions are performed at each level







Methods

Data augmentation

Resize images depending on network

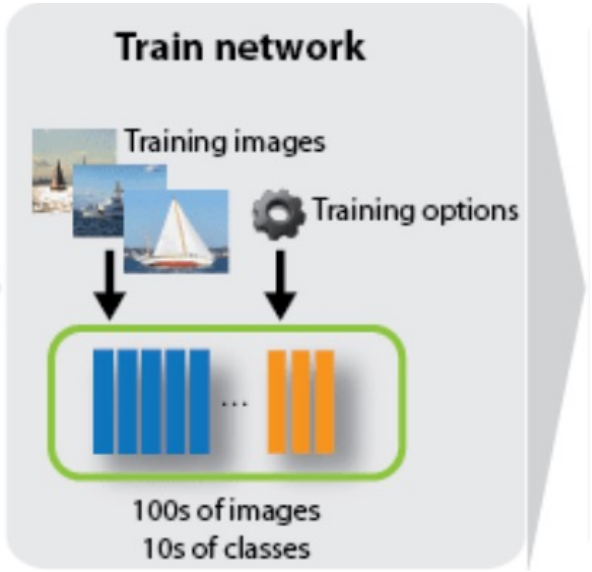
AlexNet 227x227x3

ResNet50 224x224x3

GoogLeNet 224x224x3

```
inputSize = net1.Layers(1).InputSize
```

```
pixelRange = [-30 30];
scaleRange = [0.9 1.1];
imageAugmenter = imageDataAugmenter( ...
    'RandXReflection',true, ...
    'RandXTranslation',pixelRange, ...
    'RandYTranslation',pixelRange, ...
    'RandXScale',scaleRange, ...
    'RandYScale',scaleRange);
augimdsTrain = augmentedImageDatastore(inputSize,imdsTrain, ...
    'DataAugmentation',
imageAugmenter,'ColorPreprocessing','gray2rgb');
augimdsValidation =
augmentedImageDatastore(inputSize,imdsValidation,"ColorPreprocessing",
"gray2rgb");
```



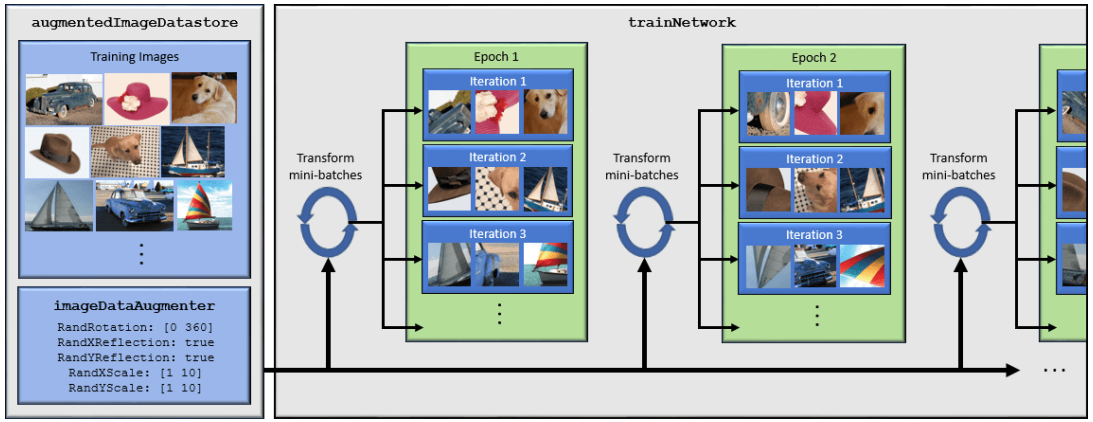
Training options the same for three networks

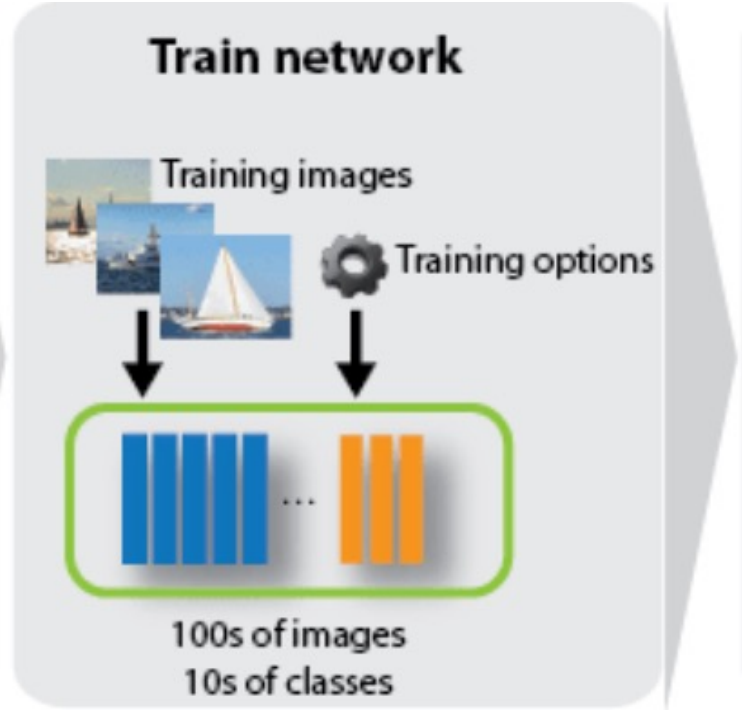
to make results comparable

Methods

Training option

```
miniBatchSize = 10;
valFrequency = floor(numel(augimdsTrain.Files)/miniBatchSize);
options = trainingOptions('sgdm', ...
    'MiniBatchSize',miniBatchSize, ...
    'MaxEpochs',6, ...
    'InitialLearnRate',3e-4, ...
    'Shuffle','every-epoch', ...
    'ValidationData',augimdsValidation, ...
    'ValidationFrequency',valFrequency, ...
    'Verbose',false, ...
    'Plots','training-progress');
```



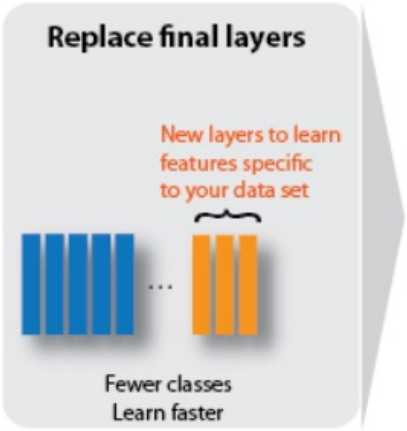


Methods

Modified Network

Replace final layers

1000 image categories → 3 image categories



Used a custom Matlab function to replace layers

“findLayersToReplace”

```
lgraph1 =replaceLayer(lgraph1,learnableLayer.Name,newLearnableLayer);
newClassLayer = classificationLayer('Name','new_classoutput');
lgraph1 = replaceLayer(lgraph1,classLayer.Name,newClassLayer);
layers = lgraph1.Layers;
connections = lgraph1.Connections;
layers(1:10) = freezeWeights(layers(1:10));
lgraph1 = createLgraphUsingConnections(layers,connections);
```

net1 | net1.Layers | net1.Layers(177, 1)

net1.Layers(177, 1)

roperty	Value
Name	'ClassificationLayer_fc1000'
Classes	*1000x1 categorical*

nettrain | nettrain.Layers | nettrain.Layers(177, 1)

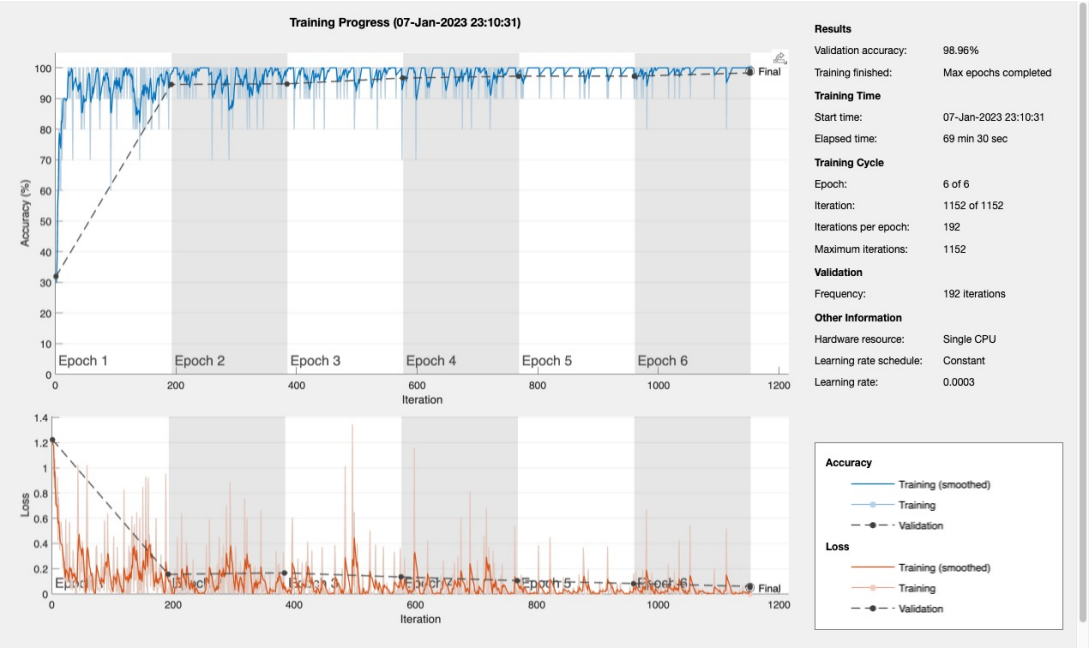
nettrain.Layers(177, 1)

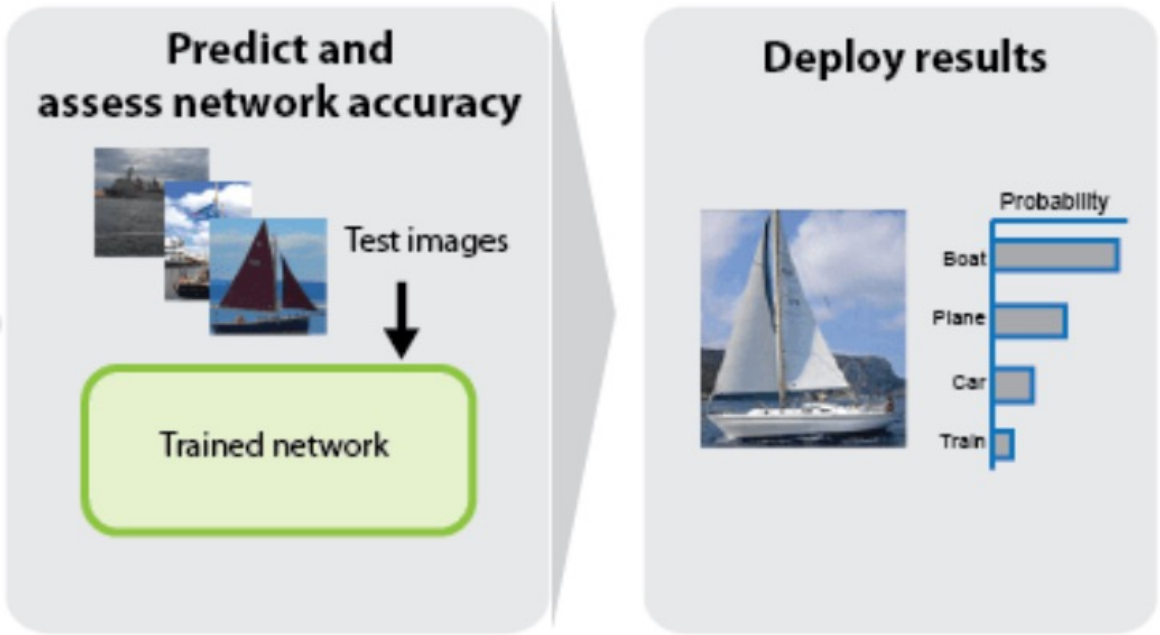
Property	Value
Name	'new_classoutput'
Classes	*3x1 categorical*

Methods

Training

Use testing data set to assess network performance





Results

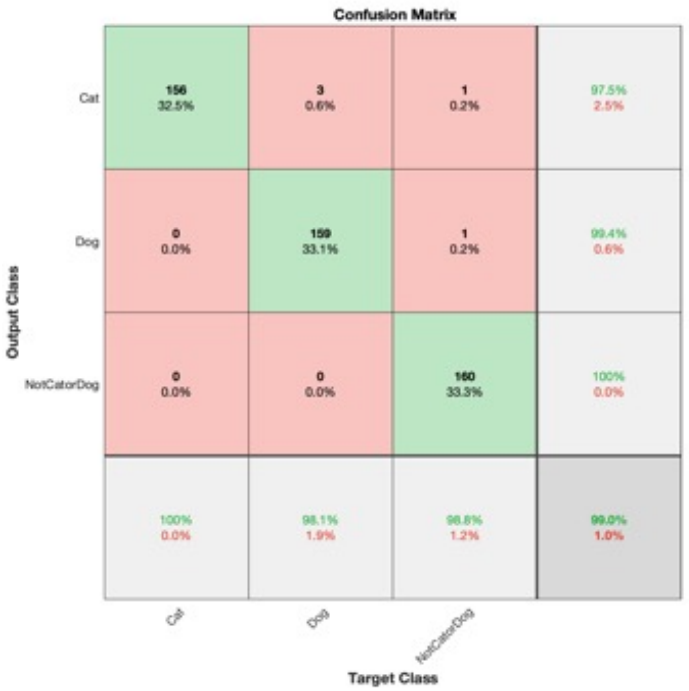
Table 1: Classification Accuracy for each network.

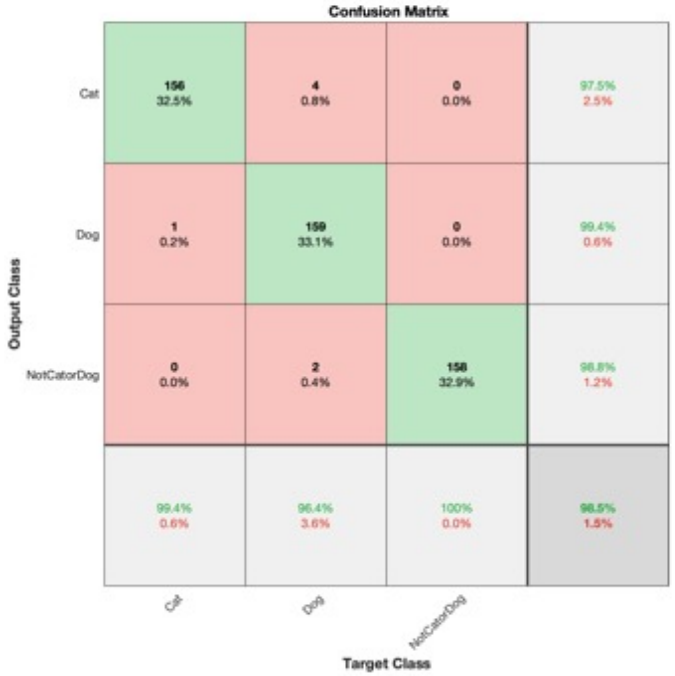
Network	Accuracy	Time
ResNet50	0.9896	69 min 30 sec
GoogLeNet	0.9854	26 min 30 sec
AlexNet	0.9521	12 min 35 sec

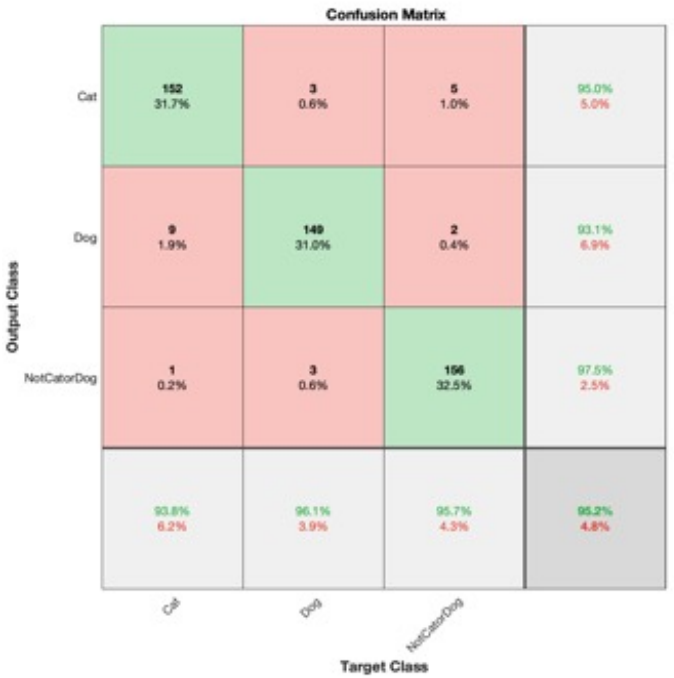
The picture can't be displayed.

Results

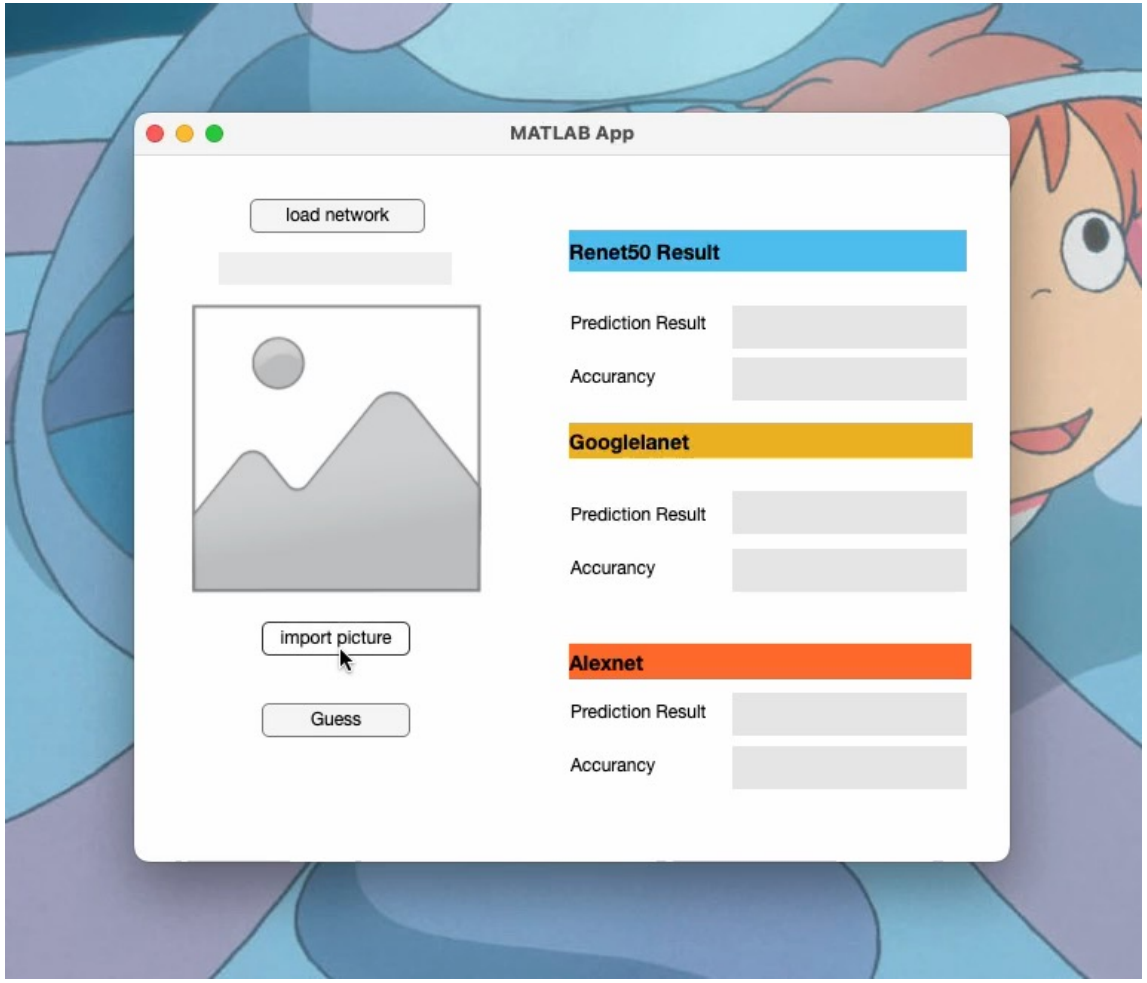
ResNet50	GoogLeNet	AlexNet



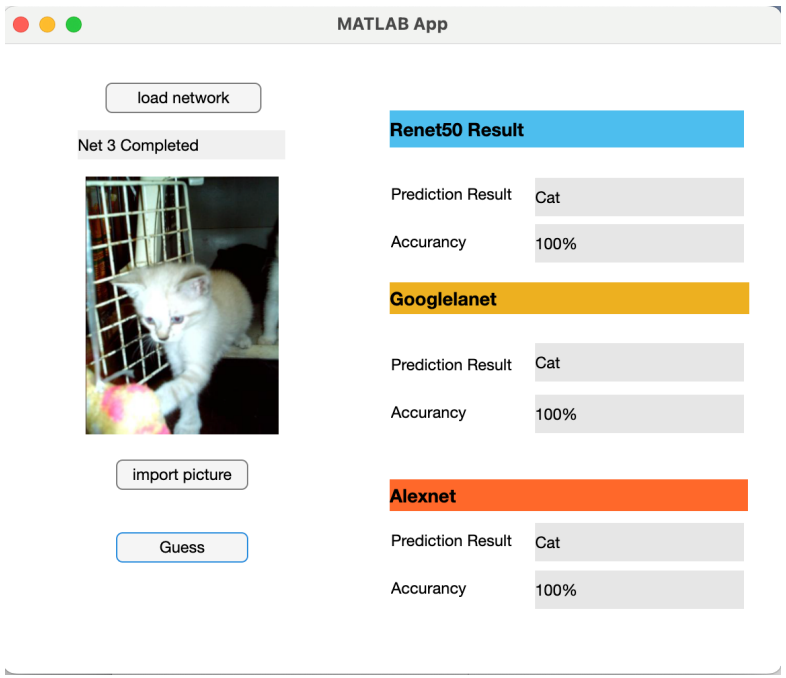


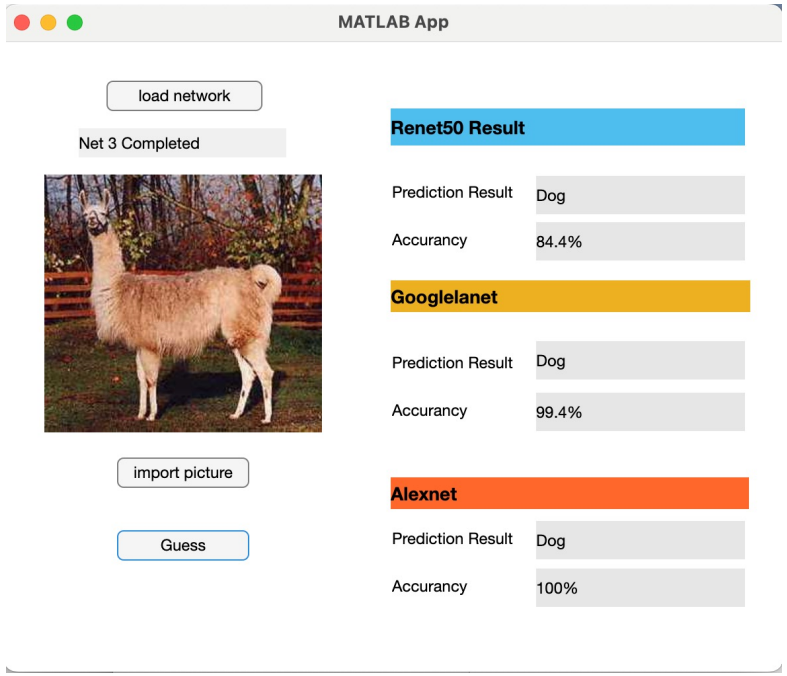


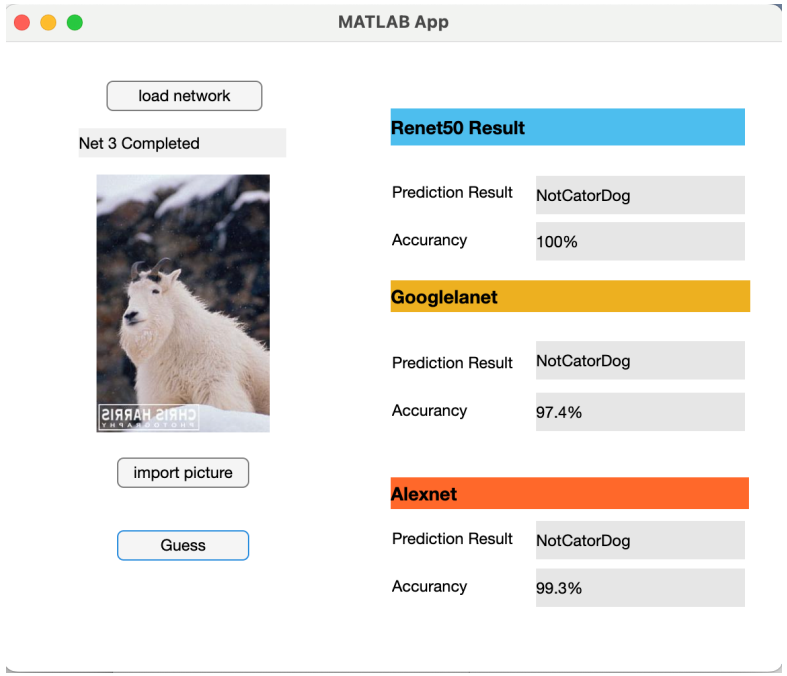
Results


MATLAB App
load network
import picture
Guess
Renet50 Result
Prediction Result
Accurancy
Googlelanet
Prediction Result
Accurancy
Alexnet
Prediction Result
Accurancy

Results


MATLAB App
load network
Net 3 Completed
import picture
Guess
Renet50 Result
Prediction Result
Cat
Accurancy
100%
Googlelanet
Prediction Result
Cat
Accurancy
100%
Alexnet
Prediction Result
Cat
Accurancy
100%


MATLAB App
load network
Net 3 Completed
import picture
Guess
Renet50 Result
Prediction Result
Dog
Accurancy
84.4%
Googlelanet
Prediction Result
Dog
Accurancy
99.4%
Alexnet
Prediction Result
Dog
Accurancy
100%


MATLAB App
load network
Net 3 Completed
import picture
Guess
Renet50 Result
Prediction Result
NotCatorDog
Accurancy
100%
Googlelanet
Prediction Result
NotCatorDog
Accurancy
97.4%
Alexnet
Prediction Result
NotCatorDog
Accurancy
99.3%

Discussion

Results in terms of accuracy are consistent with a study comparing different CNN's for classification of plant diseases

Table 6. Performance measures (%) for every pre-trained model.

Performance Measures	AlexNet	GoogleNet	Inception V3	ResNet 18	ResNet 50
Accuracy	98.93	99.39	98.65	99.06	99.15

AlexNet may have a worst performance as it is the oldest network among the three and its architecture is very similar to that of LeNet which was created on 1998

Discussion

Longer training times on ResNet-50 and GoogLeNet due to their deeper layers and more convolutions as compared to AlexNet

In this case, the ResNet50 and GoogLeNet networks were able to effectively learn and classify images of cats and dogs, These results suggest that both the ResNet50 and GoogLeNet networks are effective at classifying images of cats and dogs, with the GoogLeNet network performing slightly better for high accuracy and time-consuming

On the other hand, the AlexNet network performed not much better but well as decrease training time

Limitations :

- Only one training option is used
- Amount of data for training is relatively small
- Require high performance computer if increasing dataset to train network

References

- *Inception Module*. (n.d.). Retrieved from Deep AI: https://deepai.org/machine-learning-glossary-and-terms/inception-module
- Adaloglou, N. (n.d.). *Intuitive Explanation of Skip Connections in Deep Learning*. Retrieved from AI Summer: https://theaisummer.com/skip-connections/
- Kim, S. (n.d.). *A Beginner's Guide to Convolutional Neural Networks (CNNs)*. Retrieved from Medium: https://towardsdatascience.com/a-beginners-guide-to-convolutional-neural-networks-cnns-14649dbddce8
- Maeda-Gutiérrez, V.; Galván-Tejada, C.E.; Zanella-Calzada, L.A.; Celaya-Padilla, J.M.; Galván-Tejada, J.I.; Gamboa-Rosales, H.; Luna-García, H.; Magallanes-Quintanar, R.; Guerrero Méndez, C.A.; Olvera-Olvera, C.A. Comparison of Convolutional Neural Network Architectures for Classification of Tomato Plant Diseases. *Appl. Sci.* **2020**, *10*, 1245. https://doi.org/10.3390/app10041245
- Sachan, A. (n.d.). *Detailed Guide to Understand and Implement ResNets*. Retrieved from CV-Tricks: https://cv-tricks.com/keras/understand-implement-resnets/#:~:text=Key%20Features%20of%20ResNet%3A,network%20from%20vanishing%20gradient%20problem.
- Shafkat, I. (n.d.). *Intuitively Understanding Convolutions for Deep Learning*. Retrieved from Medium: https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1
- Siddharth, D. (n.d.). *CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more…*. Retrieved from Medium: https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5
- *What Is a Convolutional Neural Network?* (n.d.). Retrieved from Mathworks: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html#:~:text=Tutorials%20and%20examples-,How%20CNNs%20Work,input%20to%20the%20next%20layer.

Thank you for attention