

Table of Contents

Introduction.....	3
Background	3
AI Methods and Tools	3
Evaluation Method	6
Results	7
Conclusion	10
References	11
Appendix	12
User Guide	13

Image Classification using Convolutional Neural Network

Introduction

Image classification is an important subject in artificial intelligence systems and has attracted a considerable amount of interest over the last decades. This speciality aims to classify an input image based on visual content. This project deals with image classification of a benchmark dataset called CIFAR-10 using Convolutional neural network, also known as CNN or ConvNets. Although, there are many pretrained models available that are ready to be used, but one of the aim behind doing this project is to understand the architecture of ConvNets by implementing and training them.

Goal: Training and testing a Convolutional Neural Network on CIFAR-10 dataset. It's learning curve will be plotted which depicts the final score of the network, and its performance will be demonstrated using random test samples.

Background

CNN image classification technique is about taking an input image, processing it and classifying it under certain categories (E.g., Automobile, Horse, Cat, Dog). CNNs are being used for interpretation and diagnosis of medical images, classification of galaxies, street sign recognition and so on. The University of Technology Sydney has recently developed a drone-based device that can perform real-time crocodile detection by implementing a type of CNN technique [1].

Traditionally, most CNNs used for image classification were built on the principle of alternating convolution and max-pooling layers followed by few fully connected layers. This method was re-evaluated by a research team from the University of Freiburg in Germany. They carried out their experiments using CIFAR-10 and CIFAR-100 datasets and found that max-pooling could simply be replaced by a convolution layer with increased stride without any loss in accuracy [2]. The CNN used in my project is inspired from this experiment and is implemented using CIFAR-10 dataset.

AI Method and Tools

A Convolutional Neural Network is an artificial neural network that is popularly used for analysing images and classification problems. ConvNets can be found in the core of almost everything today from social media to self-driving cars, from healthcare to security etc.

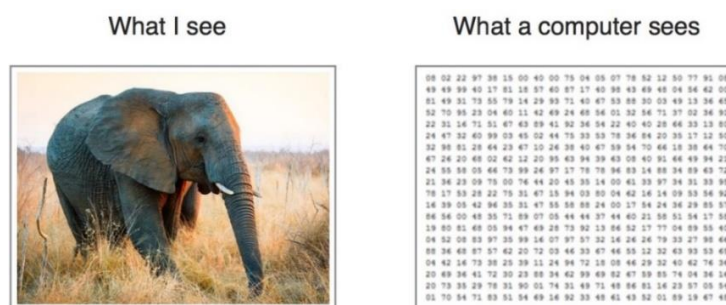


Figure 1: Image as a matrix [3]

Computers read an input image as a matrix of pixels and based on the image resolution, the matrix has certain dimensions. For example, if the image is of size 32 x 32. Then, in this case, the size of the matrix will be 32 x 32 x 3. Where 32 is height, next 32 is width and 3 is the RGB channel of the image.

Filter:

We convolve or slide a filter over the image spatially and compute dot products at every spatial location, for example, in the below image the filter is 5 x 5 x 3.

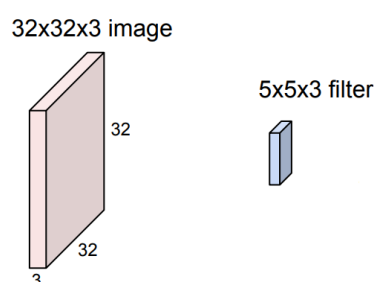


Figure 2: Filters in CNN see [4]

The filters always extend the full depth of our input volume of the image. The filter starts sliding from the upper left-hand corner and centres on top of every pixel in the input volume. At every position, the dot product between the filter and a small 5 x 5 x 3 chunk of the image is computed, and this will produce one value in our output activation map.

```
convolution2dLayer([5 5], 96, 'Padding', 2, 'Stride', 2, 'Name', 'conv1')
reluLayer('Name', 'relu')
dropoutLayer(0.5, 'Name', 'dropout1');
```

Figure 3: Convolving 96 times

In our model, we convolved 96 times over the first layer to produce 96 output layers. Then we convolve 192 times over the layer of depth 96 to produce another output layer of depth 192, and so on.

```
convolution2dLayer([5 5], 192, 'Padding', 2, 'Stride', 2, 'Name', 'conv2')
reluLayer('Name', 'relu2')
dropoutLayer(0.5, 'Name', 'dropout2');
```

Figure 4: Convolving 192 times

Convolutional Layers:

In Convolutional neural networks, we use a set of multiple filters that are called Convolutional layers. Basically, ConvNet is a sequence of Convolutional Layers, interspersed with activation functions [4], for example, a ReLU activation function. The output of each convolutional layers becomes the input to the next convolutional layer.

So, each of these layers have multiple filters and each of the filter produces an activation map. The filters in the earlier layers usually represent low-level features like edges, the middle layers look for more complex features like corners and blobs, and then the final layers look for higher-level features that are starting to resemble concepts.

Activation Function (ReLU Layer):

Activation functions are the functions used in neural networks to compute the weighted sum of input and biases, which is used to determine if a neuron can be fired or not [5].

The ReLU is a faster learning Activation function which is the most successful and widely used function [6].

It computes $f(x) = \max(0, x)$

If the input is negative, then it sets it to zero and if the input is positive then its identity is maintained.

Pooling Layer:

Pooling layer down samples the image, that is, making the representations smaller and more manageable. For example, a 128 x 128 sized image is down sampled to 32 x 32. It is important to note here that the depth of the input volume does not change, because we are only pooling spatially.

Common ways to do this are Max Pooling and Average Pooling. In Max Pooling, the pooling layer also has a filter size which is the region where we pool over. We slide the filter along our input volume in the same way we do it on convolutional layers. But here instead of finding the dot products we just take the maximum value of the pixels in that region of the input volume, avoiding any overlaps. Whereas in Average Pooling, the average value of all the pixels in a region is selected.

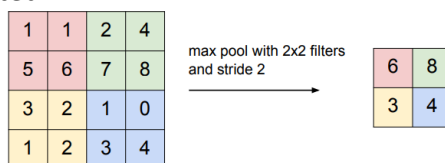


Figure 5: Max Pooling [4]

Softmax Layer:

The softmax activation function normalizes the output of the network. The output of the softmax layer consists of positive values that are restricted to one, which are then used by the classification layer for classification purpose [7].

Classification Layer:

The final layer is the classification layer. It assigns a classification score to the classes, the class with the highest score should ideally be the correctly classified class [7].

Using MATLAB with CNN:

The ConvNet used in this project was built and trained the Deep Learning Toolbox available in MATLAB. The Deep Learning Toolbox provides a framework to design and implement deep neural networks like ConvNets.

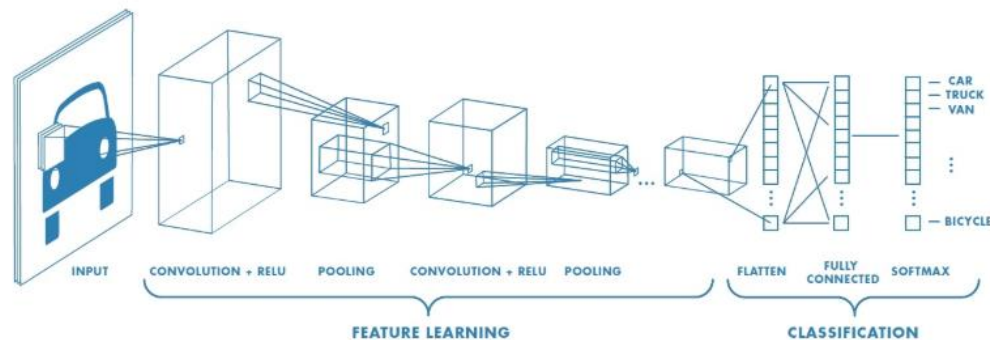


Figure 6: Layers of a CNN in MATLAB see [8]

MATLAB offers all the different layers that are required in building a Convolutional neural network, like the Convolution layer, the ReLU layer, the Pooling layer, the Softmax, and the Classification layer.

Evaluation Method

Dataset:

The CIFAR-10 dataset (Canadian Institute For Advanced Research) was used in this project, which is a commonly used dataset in machine learning. The CIFAR-10 dataset gives us 10 different classes, with 50,000 training images and 10,000 testing images evenly distributed among the 10 categories.

Here are the 10 classes present in the dataset, along with 10 random images from each class:

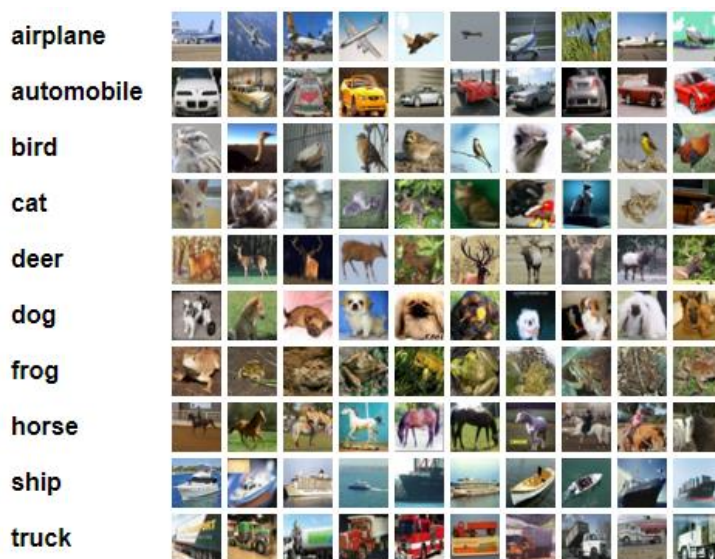


Figure 7: CIFAR-10 dataset see [9]

Steps taken to set up the experiment:

- Selecting a suitable tool: MATLAB's Deep Learning Toolbox was chosen in this case
- Selecting the dataset: CIFAR-10 was selected
- Finding a useful guide: MATLAB's documentation, see [7]
- Finding a scholarly research paper which would guide me in setting up a tried and tested method to carry out the experiment using CNN, especially with CIFAR 10 dataset, see [2].
- Finalising the hyper-parameters that works best for our solution: hyper-parameters were implemented according to those indicated in the scholarly paper [2]. For example, filter size, stride, dropouts, number of layers, epoch etc.

Key variables of interest:

Key value of interest is the classification accuracy of our CNN. It tells us the accuracy up to which our CNN will successfully be able to classify a given test image.

$$\text{Accuracy} = \frac{\text{No. of images correctly classified}}{\text{Total no. of images}}$$

Figure 8: Key variable formula

Results

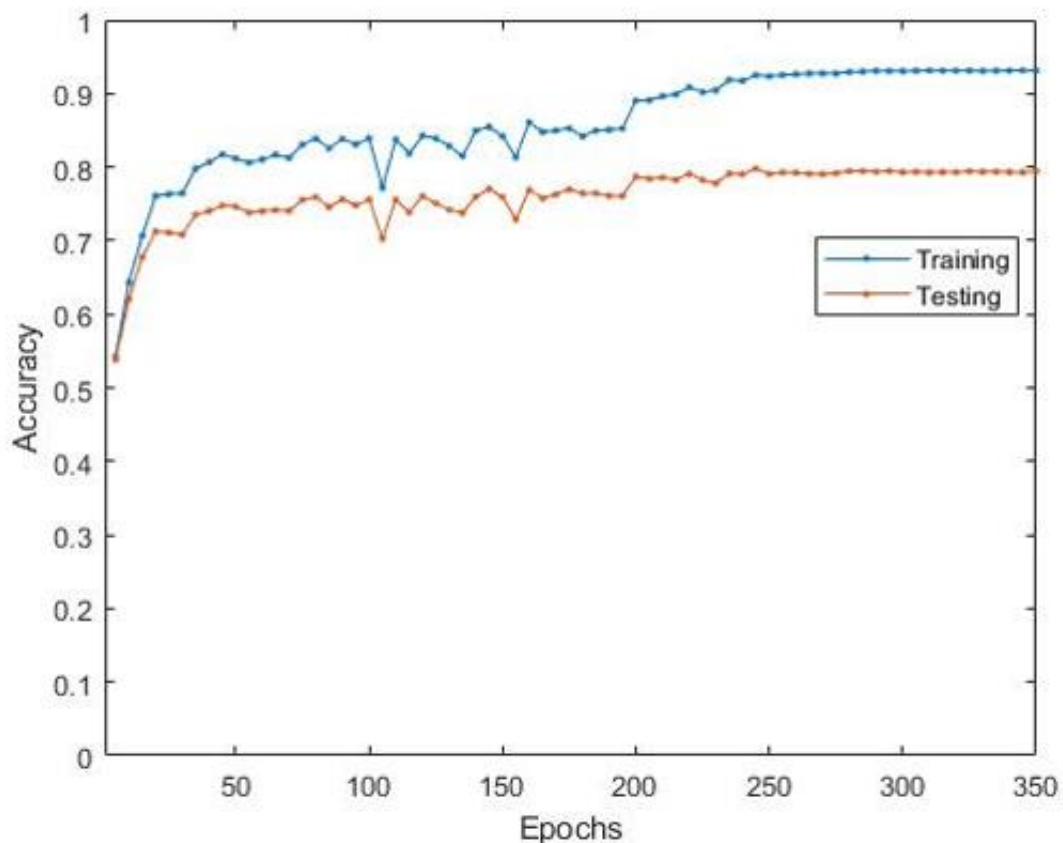
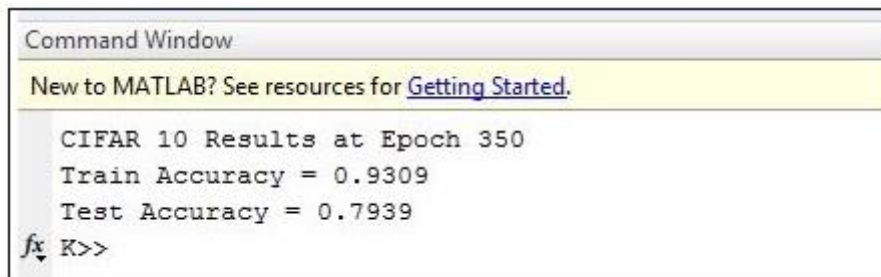


Figure 9: Result Graph

```
% Accuracy at Epoch 350 %
fname = sprintf('TestAccuracy_350.mat');
load(fname,'accuracy_train','accuracy_test');

fprintf('CIFAR 10 Results at Epoch 350\n')
fprintf('Train Accuracy = %2.4f\n', accuracy_train);
fprintf('Test Accuracy = %2.4f\n', accuracy_test);
```

Figure 10: Code to get the accuracy at 350 epochs



Command Window

New to MATLAB? See resources for [Getting Started](#).

```
CIFAR 10 Results at Epoch 350
Train Accuracy = 0.9309
Test Accuracy = 0.7939
fx K>>
```

Figure 11: Accuracy

Training:

- The network undergoes supervised training. In supervised training we tell the network exactly about input-output pairs, i.e., what should be the output of each input. In CIFAR 10 dataset all the image classes are already known.
- Each connection between neurons has a weight. During training these weights are constantly updating in order to reach their optimum value.
- Forward propagation: the image is passed through the network layer by layer to generate an output
- Error: output generated from the network is compared with the expected output, and based on the difference error is calculated.
- Backward propagation: The error is back propagated through the network from the last layer to the first layer and the weights are updated accordingly.
- For fine tuning the network parameters are changed, and the network is further trained.

Hyper-parameters:

a) Iterations:

One Epoch represents an entire dataset being passed forward and backward through the network only once, and Batch size represents the number of training samples in one forward/backward pass. In this project we have 50,000 training images in total, with a batch size of 256, and Epoch as 100.

No. of iterations to complete 1 epoch = $50,000/256 = 195$ iterations

Total no. of iterations in complete training = $195 \times 350 = 68,250$ iterations

b) Dropout:

Dropout is a regularization technique in neural networks where randomly selected neurons are ignored during training [10]. This results in the neurons being temporarily removed during the forward propagation, and the weights are also not updated to these neurons on the backward propagation.

For our method, dropout was applied to the input image as well as after each pooling layer. The probability for dropping out inputs was 20%, and 50% otherwise [2].

c) Epoch:

The network was trained up to 350 epochs with a batch size of 256.

d) Learning rate:

The rate at which a neural network model learns is referred to as learning rate. In our network, the learning rate is different in different range of epochs. Below is a table depicting the learning rates for corresponding epoch range.

Epoch range	Learning rate
Below 200	0.01
200 to 250	0.001
250 to 300	0.0001
Above 300	0.00001

Training Accuracy and Error:

During training, it was observed that the learning rate of our ConvNet steadied after 250 epochs, and achieved an accuracy of 79.39% after 350 epochs. Which means that our network can classify images with an accuracy of **79.39%**.

The error value signifies the performance of the network, having low error is desirable. Basically, error tells us the difference in accuracy in what the network is classifying for a given image, and what the image actually is.

We know that,

$$\text{Error} = 1 - \text{Accuracy}$$

$$\text{Error} = 1 - 0.7939 = 0.2061$$

$$\text{i.e. Error} = 20.61\%$$

Conclusion

Outcomes of the experiment:

A competent performance and ability of image classification was expected from our solution with an accuracy of around 80%. The resulting accuracy (**79.39%**) meets the goals of this experiment.

Learning Outcomes:

- Understanding the learning behaviour of neural networks.
- Understanding the architecture of Convolutional neural networks.
- Understanding how Convolutional Neural Networks are trained and tested.
- Appreciating the previous research carried out in this field.
- Appreciating the wide applications of CNN as a solution in various technologies to solve different real-life problems.
- Identifying various datasets, pretrained models and tools available for ConvNets.

Scope of Improvement:

- The error could be further minimized by performing additional fine-tuning experiments.
- Presently, own test images need to be in JPG format only, the MATLAB program can be improved to accept other image formats as well.
- The network could be fine-tuned by applying changes and using different combination of hyper-parameters like learning rate, layers, filter size, pooling, batch size, stride, padding, dropout etc.
- The network at this stage is unable to classify test images that are provided in a different orientation to our dataset, so this issue needs addressing in future.

References

- [1] U. o. T. Sydney. "AI tech applied to real-time crocodile detection."
<https://www.uts.edu.au/about/faculty-engineering-and-information-technology/news/ai-tech-applied-real-time-crocodile> (accessed 2019).
- [2] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for Simplicity: The All Convolutional Net," 2014. [Online]. Available:
<http://murdoch.summon.serialssolutions.com/2.0.0/link/0/eLvHCXMwY2AwNtlz0EUrE8wNE4Etg9SUtFRLgzRgG8TSyDTFGfjZWxpYJhbgNdmlpapMjHlw7bGJBZVZJBjgtOKtY3NDE00jOzAJ2wzWgEbgZFAtuUYDVIIYKblAM tDWn4AgJfiEGptQ8EQaj4JKiTFA XQHYIFQIzgSv2QY2dq0UgHGi4JiTo-Ccn1cGiXKgbr UEIEGOTfXEGcPXbAt8QWQlyDiQS6JB7nEWlyBBdhPT5VgUDA3NUIOsUwzTjSySDRJMzdOSjQ0Mwk2clzNUk3SUhJNJRnEcRgihVNGmoELWEODTxY0MpRhYCKpKk2VhV91JAcOAIp7h8BAI75Zcc>.
- [3] K. Sorokina. "Image Classification with Convolutional Neural Networks." medium.com.
<https://medium.com/@ksusorokina/image-classification-with-convolutional-neural-networks-496815db12a8> (accessed 2019).
- [4] F.-F. L. J. S. Yeung, "Convolutional Neural Networks," ed: Stanford University School of Engineering, 2017.
- [5] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation Functions: Comparison of trends in Practice and Research for Deep Learning," 2018. [Online]. Available:
<http://murdoch.summon.serialssolutions.com/2.0.0/link/0/eLvHCXMwdV1Na8JAEB3UUy-itMX6OX9Am2ajrt5EDb21hxbaU0h2d4oXFavSn-9kkmgRPO-yPGZg3xuYmQeg IHXv oTiLQxWnk2TgU7U75IWRvYCVeyGluxkLy0qZYAi9GYePe3OmbgrpPfZ2ajl4Gn1FiXoez74tTw9sWYUq7944WwBtVc0OEsy0AdSm59D58zU9iGYcjMlcmd4vxs-ocbwr20o-Jqje 5qBJyWY9FLxyynMSFc1vMd6D-PEAvXH7MX sCJNpmiyKiFG8keNUjVLicdw1Az1jIYm18PaKADCUTM0xM4MiJBSc9QePWK83bRy24YyrXMiWn21DZ7w6uc ZE6qaS8bsr4ToBn291qA>.
- [6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015, doi: 10.1038/nature14539.
- [7] Mathworks. "Create Simple Deep Learning Network for Classification."
https://au.mathworks.com/help/deeplearning/examples/create-simple-deep-learning-network-for-classification.html#responsive_offcanvas (accessed 2019).
- [8] Mathworks. "Convolutional Neural Network." <https://au.mathworks.com/solutions/deep-learning/convolutional-neural-network.html> (accessed 2019).
- [9] A. Krizhevsky. "The CIFAR-10 dataset." Alex Krizhevsky.
<https://www.cs.toronto.edu/~kriz/index.html> (accessed 2019).
- [10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929-1958, 2014.

Appendix

Test Input –

Testing the network with 40 randomly selected images –



Figure 12: Test images

Output –

result							
		40x2 cell					
1	2						
1	airplane1.jpg	airplane		12	bird4.jpg	bird	
2	airplane2.jpg	airplane		13	cat1.jpg	cat	
3	airplane3.jpg	airplane		14	cat2.jpg	cat	
4	airplane4.jpg	ship		15	cat3.jpg	dog	
5	automobile...	automobile		16	cat4.jpg	bird	
6	automobile...	automobile		17	deer1.jpg	deer	
7	automobile...	automobile		18	deer2.jpg	deer	
8	automobile...	truck		19	deer3.jpg	deer	
9	bird1.jpg	bird		20	deer4.jpg	deer	
10	bird2.jpg	bird		21	dog1.jpg	dog	
11	bird3.jpg	airplane		22	dog2.jpg	dog	
				23	dog3.jpg	bird	
				24	dog4.jpg	dog	
				25	frog1.jpg	dog	
				26	frog2.jpg	frog	
				27	frog3.jpg	frog	
				28	frog4.jpg	frog	
				29	horse1.jpg	horse	
				30	horse2.jpg	horse	
				31	horse3.jpg	bird	
				32	horse4.jpg	horse	
				33	ship1.jpg	ship	
				34	ship2.jpg	ship	
				35	ship3.jpg	ship	
				36	ship4.jpg	ship	
				37	truck1.jpg	truck	
				38	truck2.jpg	truck	
				39	truck3.jpg	truck	
				40	truck4.jpg	truck	

Figure 13: Output of testing

Accuracy –

In total 7 images out of 40 were wrongly classified. Therefore, the classification test accuracy of our test images was 82.5%.

How down-sampled images looked like –



Figure 14: Down sampling of images

User Guide

1. Place the 'Image Classification Project' folder in C directory, or change the path accordingly:

```

18
19 -   path = 'C:\Image Classification Project\ConvNet\Test images';
20

```

2. Load the program file 'CIFAR10C_net_Alayers_training' in MATLAB R2017b or R2019b:

This PC > Windows (C:) > Image Classification Project > ConvNet

Name	Date modified	Type	Size
Test images	3/10/2019 11:16 AM	File folder	
Allresults	3/10/2019 11:26 AM	Microsoft Access ...	1 KB
CIFAR10C_net_Alayers_training	3/10/2019 12:17 PM	GNU Octave Script	6 KB

3. The test images (JPG format) are placed in 'Test images' folder, keep all your test images in this folder. This folder currently contains 40 images to test each class:

This PC > Windows (C:) > Image Classification Project > ConvNet

Name	Date modified	Type	Size
Test images	3/10/2019 11:16 AM	File folder	
Allresults	3/10/2019 11:26 AM	Microsoft Access ...	1 KB
CIFAR10C_net_Alayers_training	3/10/2019 12:17 PM	GNU Octave Script	6 KB

4. The network has already been trained up to 350 epochs, and the training has been saved. The program runs on the saved training file

```

155 -   fname = sprintf('TrainedNet_350.mat');
156 -   load(fname, 'cifar10Net');

```

5. Run the program
6. After the program has successfully run, the classification results get stored in the 'Allresults' file:

This PC > Windows (C:) > Image Classification Project > ConvNet

Name	Date modified	Type	Size
Test images	3/10/2019 11:16 AM	File folder	
Allresults	3/10/2019 11:26 AM	Microsoft Access ...	1 KB
CIFAR10C_net_Alayers_training	3/10/2019 12:17 PM	GNU Octave Script	6 KB

7. Load the 'Allresults' file into MATLAB by dragging and dropping into the console:

```
Elapsed time is 0.050028 seconds.
>> load('Allresults.mat')
fx >>
```

8. Double click on the loaded results file in the Workspace section to view the classification output:

result		
40x2 cell		
	1	2
1	airplane1.jpg	airplane
2	airplane2.jpg	airplane
3	airplane3.jpg	airplane
4	airplane4.jpg	ship
5	automobile...	automobile
6	automobile...	automobile
7	automobile...	automobile
8	automobile...	truck
9	bird1.jpg	bird
10	bird2.jpg	bird
11	bird3.jpg	airplane

Grading Sheet for Assignment 2

1. Identification, Presentation

Item	Your Check	Tutor's Use Only
Submitted on time		
Coversheet and Grading sheets filled in		
References cited in IEEE style		
Clear and understandable writing		
Marks	/ 4	

2. Problem, Background Research and Goal

Item	Check	Tutor's Use Only
Clear overview of the problem, diagrams	(Overview)	
Clearly describe and cite prior work	(Shortcomings)	
At least one clearly-explained goal specified	(Reading)	
Marks	/ 6	

3. Solution and Implementation

Item	Check	Tutor's Use Only
Good solution chosen	(Description)	
Good use of tools and info. sources	(Diagram)	
Marks	/5	

4. Performance

Item	Check	Tutor's Use Only
Solution runs according to test	(Benefits)	
Goal(s) were reached	(Risks)	
A User Guide was included	(Timescale)	
Marks	/ 10	

5.Evaluation and Conclusion

Item	Check	Tutor's Use Only
How the solution was evaluated	(Summary)	
What issues arose; what was learned	(Conclusion)	
Marks	/ 5	

Total Marks (Sections 1-5)	/ 30	
-----------------------------------	-------------	--

Tutor Comments: