Sergio Munguia Team (1) Member

Final Project: Cats vs Dogs

By using computer vision, I can take advantage of machine learning techniques to detect objects of interest in images and classify or identify categories of objects.
In the project I extract features, and then use them to train a model to classify or learn patterns in the image data.
I use local detectors for locally "interesting points" in the image.
These image features: are collections of locally interesting points

- Combined to build classifiers

Standard image classification approach, Extract features
Using the features of a pre-trained network, I can achieve 90% accuracy in a minute-wise performance.

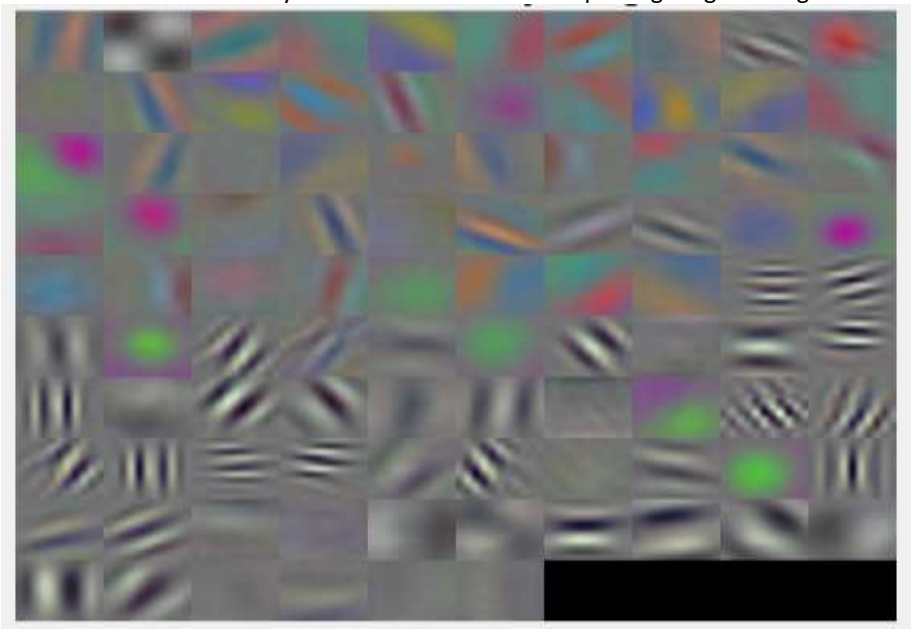
In the code I instantiate the convolutional part of the model, and everything up to the fully-connected layers.
If training a large network, can use regularization to defeat overfitting.

1st I take in a trained model of data, either a CNN pretrained or CNN trained from scratch on 25,000 cats & dogs
-Locating the pre-trained "AlexNet" in file location of the folder cats_dogs_starter\networks
(Loading the MatConvNet data into ConvNet, a series network object from NN toolbox, using helperImportMavConvNet in Computer Vision System Toolbox.
I will use the series network object to inspect the network architecture, classify new data, and extract network activations from specific layers.

2nd Inspect the layers of the convnet.Layers
-The convolutional layers, interspersed with rectified linear units (ReLU) and max-pooling layers
-Following layers are the 3 Fully-connected layers
-the last layer is a classification layer
Convet.Layers(end) # inspect the last layer)

3rd Inspect the network weights for the second convolutional layer

- The first layer has learned filters for capturing blog and edge features.



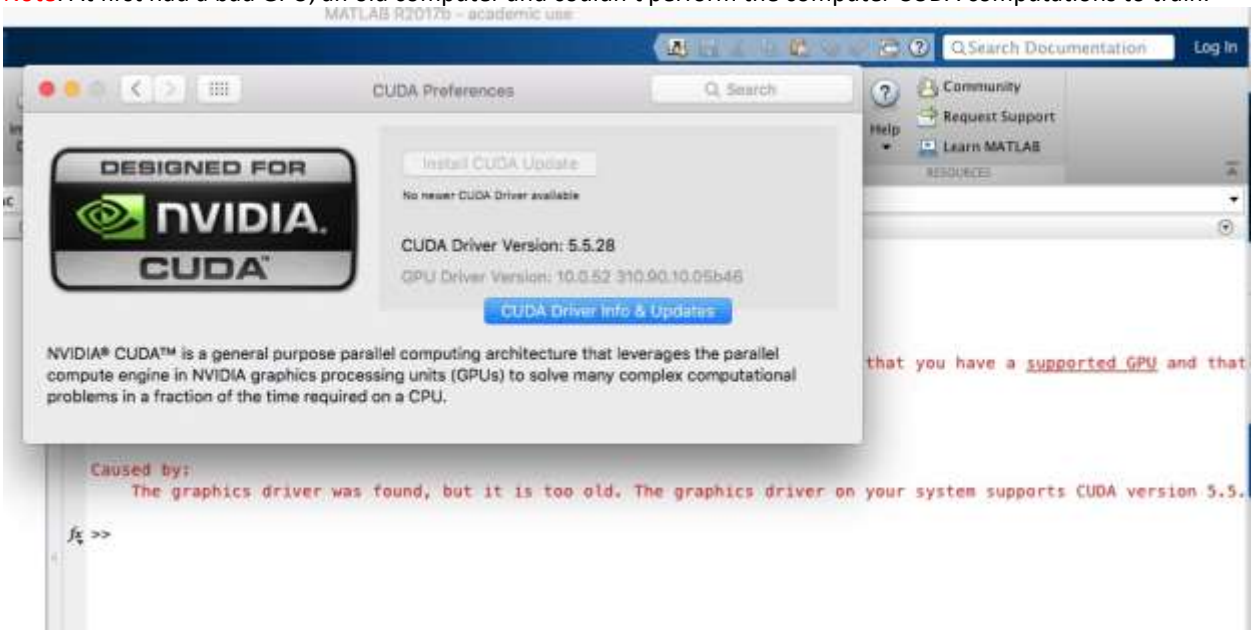
4th Inspect and extract features from one of the deeper layers using activations method.
5th Classify the training Features, training Labels

- CNN trained from scratch

- CNN using pre-trained data
- SVM Classifier
- TREE Classifier
- Naive Bayes Classifier
- K-Nearest Classifier

6th Show predications of 4 different classifiers % accuracy

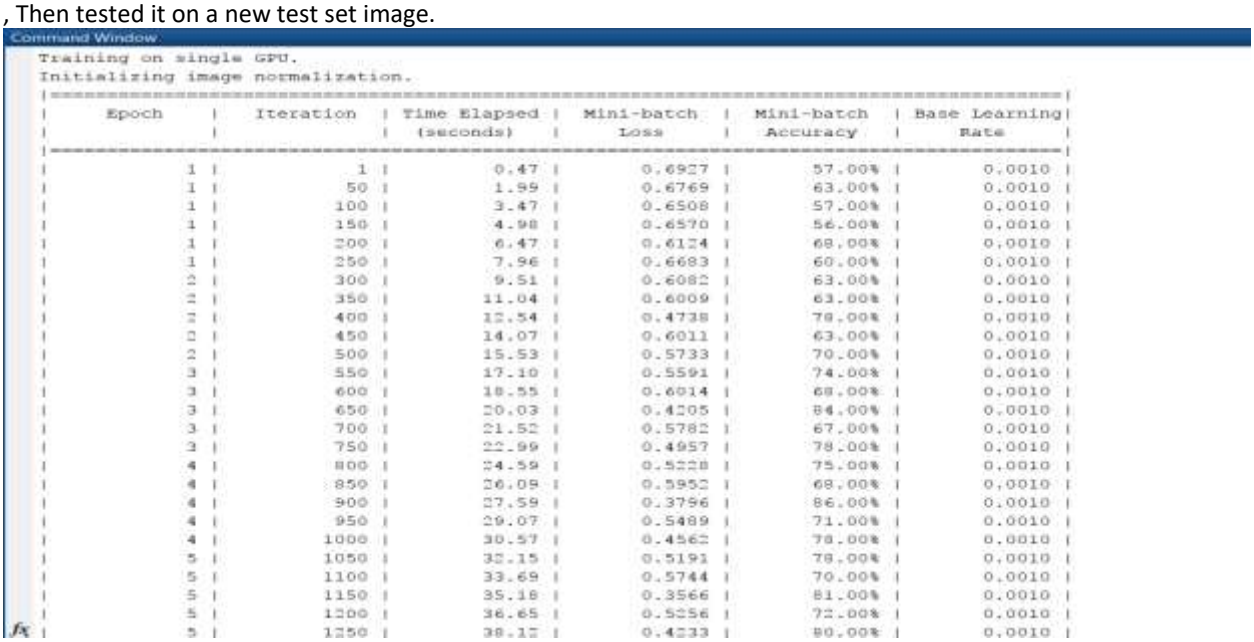
Note: At first had a bad GPU, an old computer and couldn't perform the computer CUDA computations to train.



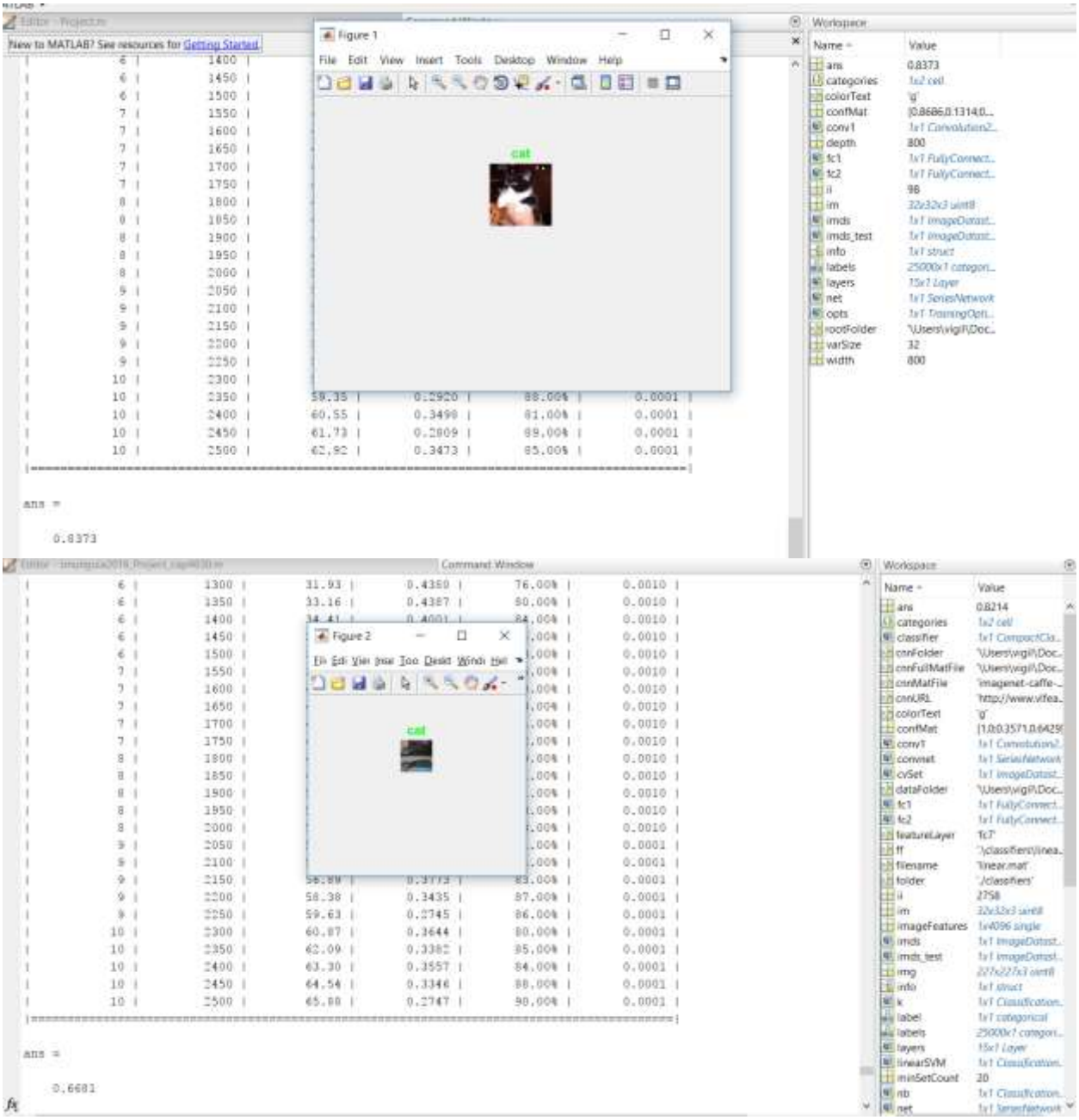
So, I went to Best Buy and purchased a new computer…

Here I used an NVIDIA GeForce GTX 1060 on an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 2801 Mhz, 4 Core(s), 8 Logical Processor(s) to train a CNN from of the 25,000 dogs and cats images data set from Kaggle.

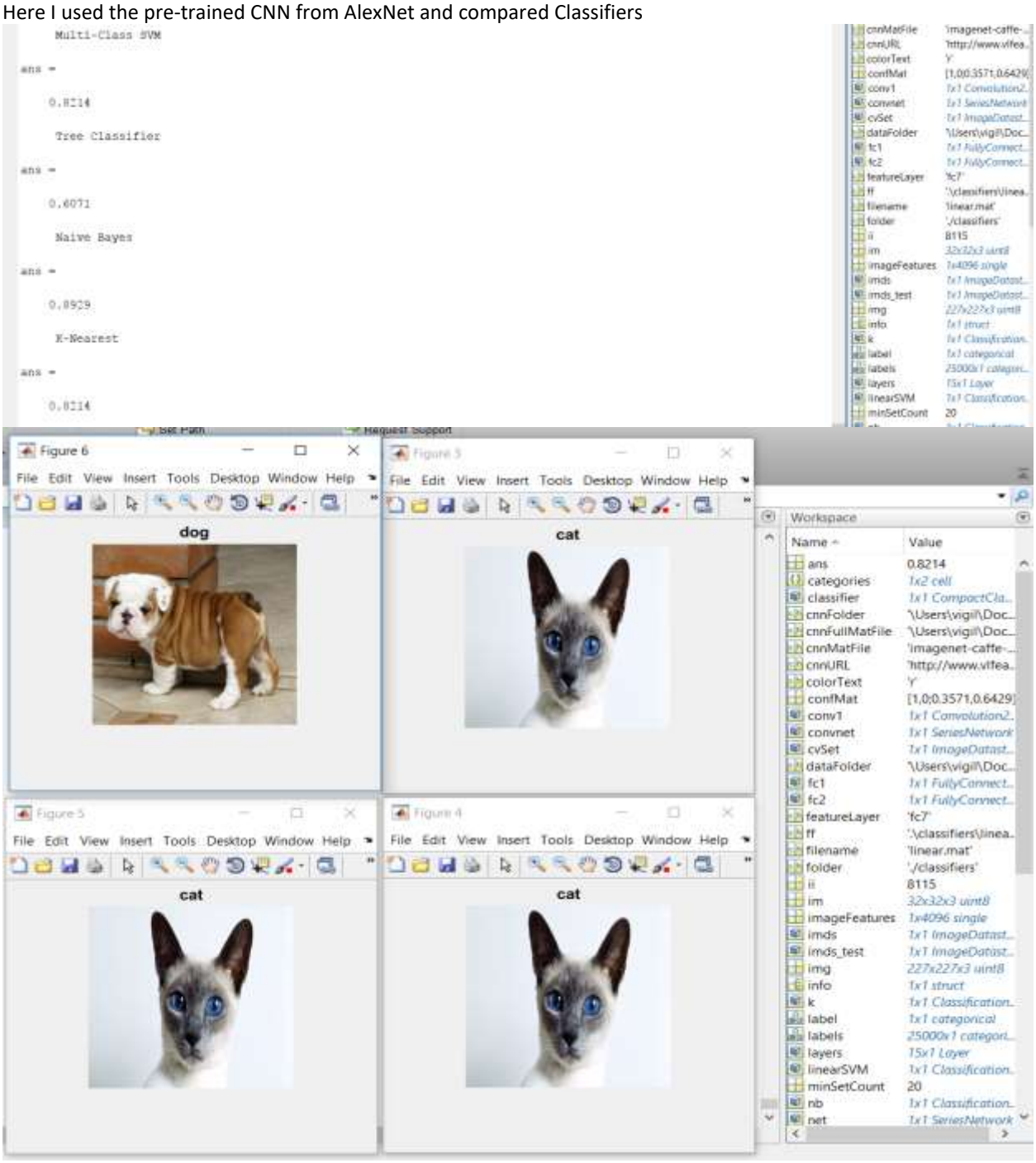
, Then tested it on a new test set image.

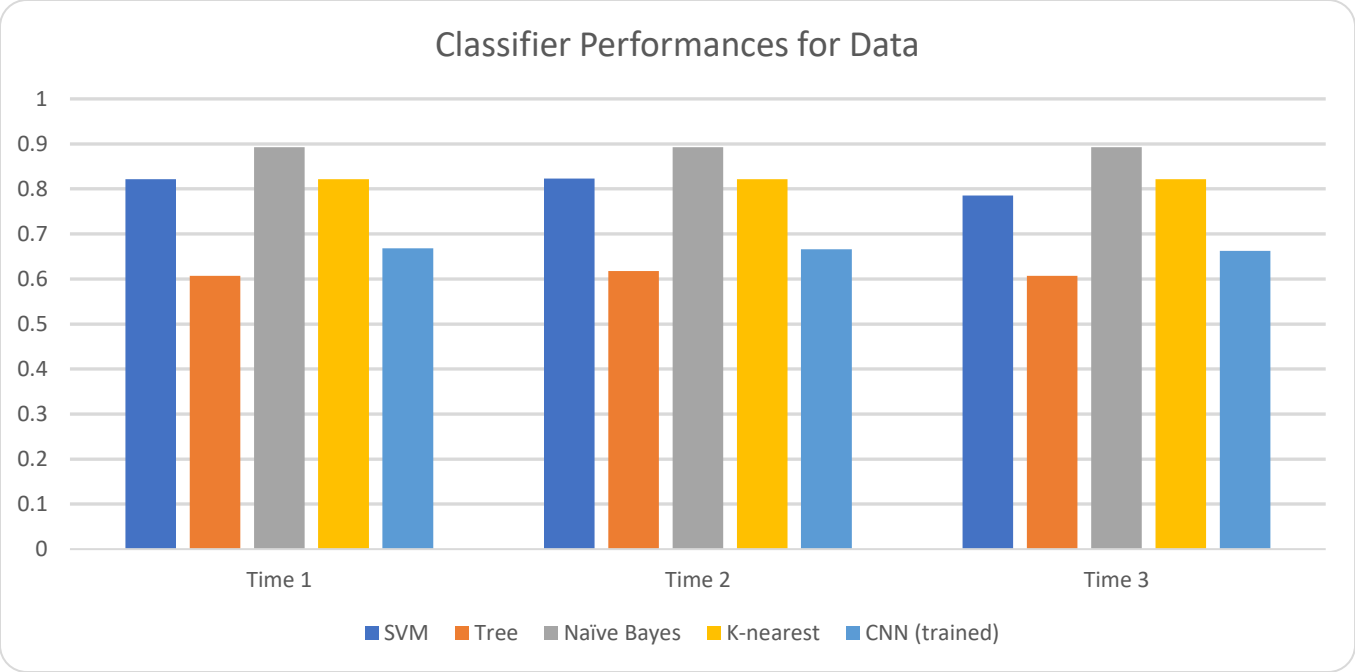


Epoch	Iteration	Time Elapsed (seconds)	Mini-batch Loss	Mini-batch Accuracy	Base Learning Rate
1	1	0.47	0.6927	57.00%	0.0010
1	50	1.99	0.6769	63.00%	0.0010
1	100	3.47	0.6508	57.00%	0.0010
1	150	4.98	0.6570	56.00%	0.0010
1	200	6.47	0.6124	68.00%	0.0010
1	250	7.96	0.6683	60.00%	0.0010
2	300	9.51	0.6082	63.00%	0.0010
2	350	11.04	0.6009	63.00%	0.0010
2	400	12.54	0.4738	78.00%	0.0010
2	450	14.07	0.6011	63.00%	0.0010
2	500	15.53	0.5733	70.00%	0.0010
3	550	17.10	0.5591	74.00%	0.0010
3	600	18.55	0.6014	68.00%	0.0010
3	650	20.03	0.4205	84.00%	0.0010
3	700	21.52	0.5782	67.00%	0.0010
3	750	22.99	0.4957	78.00%	0.0010
4	800	24.59	0.5228	75.00%	0.0010
4	850	26.09	0.5952	68.00%	0.0010
4	900	27.59	0.3796	86.00%	0.0010
4	950	29.07	0.5489	71.00%	0.0010
4	1000	30.57	0.4562	78.00%	0.0010
5	1050	32.15	0.5191	78.00%	0.0010
5	1100	33.69	0.5744	70.00%	0.0010
5	1150	35.18	0.3566	81.00%	0.0010
5	1200	36.65	0.5256	72.00%	0.0010
5	1250	38.12	0.4233	80.00%	0.0010


New to MATLAB? See resources for Getting Started
Figure 1
File Edit View Insert Tools Desktop Window Help
cat
Workspace
Name
Value
ans 0.8373
categories 1x2 cell
colorText 'g'
confMat [0.8686,0.1314,0...
conv1 1x1 Convolution2...
depth 800
fc1 1x1 FullyConnect...
fc2 1x1 FullyConnect...
ii 98
im 32x32x3 uint8
imds 1x1 ImageDatast...
imds_test 1x1 ImageDatast...
info 1x1 struct
labels 25000x1 categori...
layers 15x1 Layer
net 1x1 SeriesNetwork
opts 1x1 TrainingOpti...
rootFolder '\Users\vigil\Doc...
varSize 32
width 800
6	1400
6	1450
6	1500
7	1550
7	1600
7	1650
7	1700
7	1750
8	1800
8	1850
8	1900
8	1950
8	2000
9	2050
9	2100
9	2150
9	2200
9	2250
10	2300
10	2350	59.35	0.2920	88.00%	0.0001
10	2400	60.55	0.3490	81.00%	0.0001
10	2450	61.73	0.2809	89.00%	0.0001
10	2500	62.92	0.3473	85.00%	0.0001
ans =
0.8373
Command Window
Figure 2
cat
Workspace
Name
Value
ans 0.8214
categories 1x2 cell
classifier 1x1 CompactCla...
cnnFolder '\Users\vigil\Doc...
cnnFullMatFile '\Users\vigil\Doc...
cnnMatFile 'imagenet-caffe-...
cnnURL 'http://www.vlfea...
colorText 'g'
confMat [1.0,0.3571,0.6429
conv1 1x1 Convolution2...
convnet 1x1 SeriesNetwork
cvSet 1x1 ImageDatast...
dataFolder '\Users\vigil\Doc...
fc1 1x1 FullyConnect...
fc2 1x1 FullyConnect...
featureLayer 'fc7'
ff '\classifiers\linea...
filename 'linear.mat'
folder '.\classifiers'
ii 2758
im 32x32x3 uint8
imageFeatures 1x4096 single
imds 1x1 ImageDatast...
imds_test 1x1 ImageDatast...
img 227x227x3 uint8
info 1x1 struct
k 1x1 Classification...
label 1x1 categorical
labels 25000x1 categori...
layers 15x1 Layer
linearSVM 1x1 Classification...
minSetCount 20
nb 1x1 Classification...
net 1x1 SeriesNetwork
6	1300	31.93	0.4359	76.00%	0.0010
6	1350	33.16	0.4387	80.00%	0.0010
6	1400	34.41	0.4001	84.00%	0.0010
6	1450	.00%	0.0010
6	1500	.00%	0.0010
7	1550	.00%	0.0010
7	1600	.00%	0.0010
7	1650	.00%	0.0010
7	1700	.00%	0.0010
7	1750	.00%	0.0010
8	1800	.00%	0.0010
8	1850	.00%	0.0010
8	1900	.00%	0.0010
8	1950	.00%	0.0010
8	2000	.00%	0.0010
9	2050	.00%	0.0001
9	2100	.00%	0.0001
9	2150	58.89	0.3773	83.00%	0.0001
9	2200	58.38	0.3435	87.00%	0.0001
9	2250	59.63	0.2745	86.00%	0.0001
10	2300	60.87	0.3644	80.00%	0.0001
10	2350	62.09	0.3382	85.00%	0.0001
10	2400	63.30	0.3557	84.00%	0.0001
10	2450	64.54	0.3346	88.00%	0.0001
10	2500	65.80	0.2747	90.00%	0.0001
ans =
0.6681

Here I used the pre-trained CNN from AlexNet and compared Classifiers




Classifier Performances for Data
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Time 1
Time 2
Time 3
SVM
Tree
Naïve Bayes
K-nearest
CNN (trained)