
Robotic Inference Project

Mohamed Sayed Antar

Robotics Software Engineer

mohamedsayed99.ms@gmail.com

March 15, 2019

ABSTRACT

working on a supplied data from a camera fixed above a conveyor belt and a collected data using a mobile camera, first using the supplied data with Nvidia DIGITS workflow to train a model by tuning the hyperparameters and choosing the best network to achieve a specific inference time and accuracy, second using the collected data for an inference idea and training a model using Nvidia DIGITS to deploy this model on an embedded system like Jetson TX2 board.

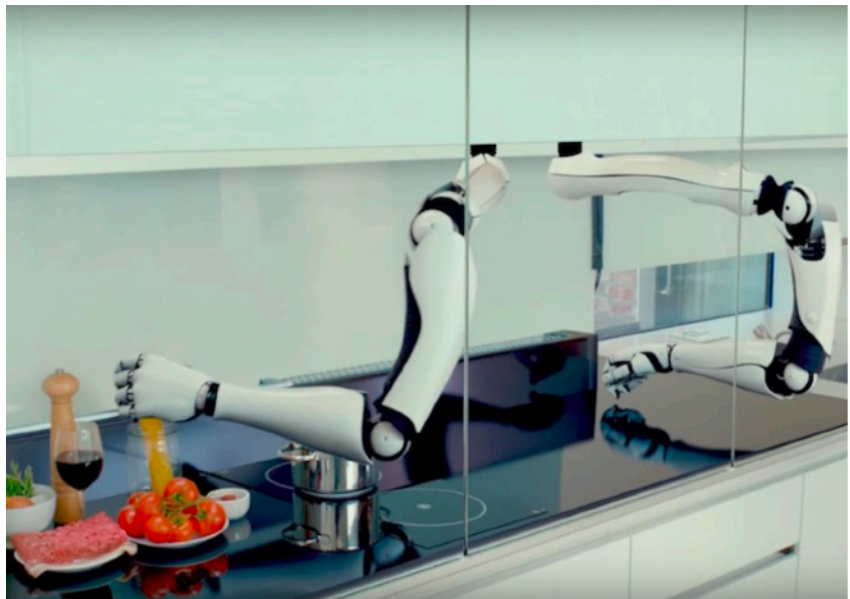
Keywords Deep Learning . Neural Networks . Object recognition . Nvidia DIGITS.

1 Introduction

the robotic kitchen is now an interesting part of robotics which full of challenges, due to the routine of life nowadays we are very busy because of the tight schedules, so it's difficult to take care of our food, people tend to approach fast-food instead of preparing healthy meals at home which leads to severe diseases.

nowadays there are some cooking robots which enable us to cook the food.

they are fully automated robots, so they have to capture the world around them using several kinds of sensors and cameras. in turn classifying images around the robot and determining each kitchenware is a core element of the robot perception process, using Nvidia DIGITS workflow with a collected photos of spoons or forks or even no thing to train a network for the classification processes.



2 Background / Formulation

several types of DNN's have been developed on the ImageNet benchmark dataset like AlexNet, VGGNet, ResNet, Inception, GoogleNet and their many variations.

The increased accuracy is the result of breakthroughs in design and optimization, but comes at a cost when computation resources are considered.

The following table provides a sampling of the results (values are approximated from graphs in the paper), including a derived metric called information density. The information density is a measure of the efficiency of the network, or how much accuracy is provided for every one million parameters that the network requires.

DNN	Top1 Accuracy	Operations	Parameter	Inference	Power	Memory	Info Density
Network name	%	G-Ops	M	fps	W	MB	%accuracy/Mparams
AlexNet-BN	57	2	60	50	10.9	310	1.0
GoogleNet	68	3	7	33	10.7	200	9.7
VGG-16	71	31	135	6	11.8	850	0.5
ResNet-101	76	16	45	10	12.9	300	1.7
Inception-v3	78	12	25	12	11.6	200	3.1

Note that only the results based on a batch size of one are included. In most cases, the batch size provides a speedup in inference time but maintains the same relative performance among architectures. However, an exception is AlexNet, which sees a 3x speedup when going from 1 to 64 images per batch due to weak optimization of its fully connected layer.

for the supplied data collected using Jetson TX2 camera above a conveyor belt AlexNet gave a good results in this case however the other DNN's may give more accurate results.

for the collected data, using about 350 image per class GoogleNet gave more accurate results than AlexNet using 0.001 learning rate.

3 Data Acquisition

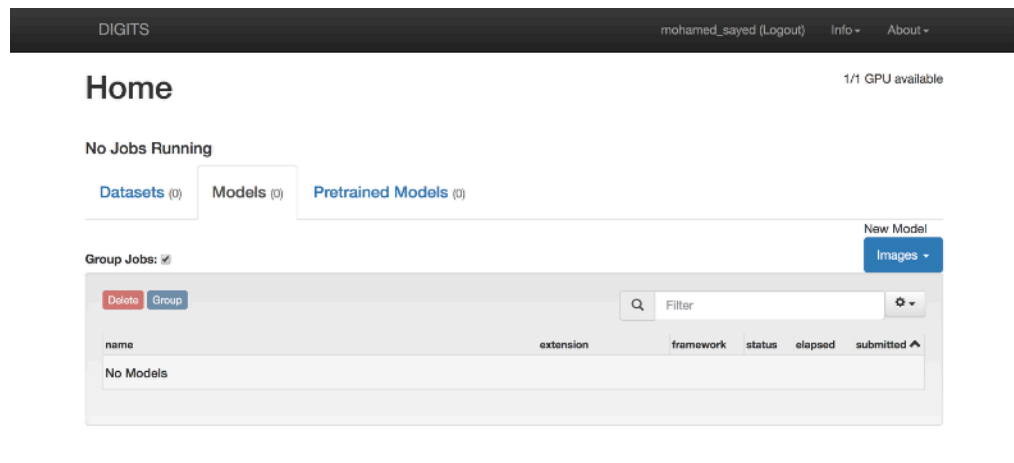
for the supplied data there are 7570 images for the following 3 classes:

1. Bottle
2. Candy Box
3. Nothing

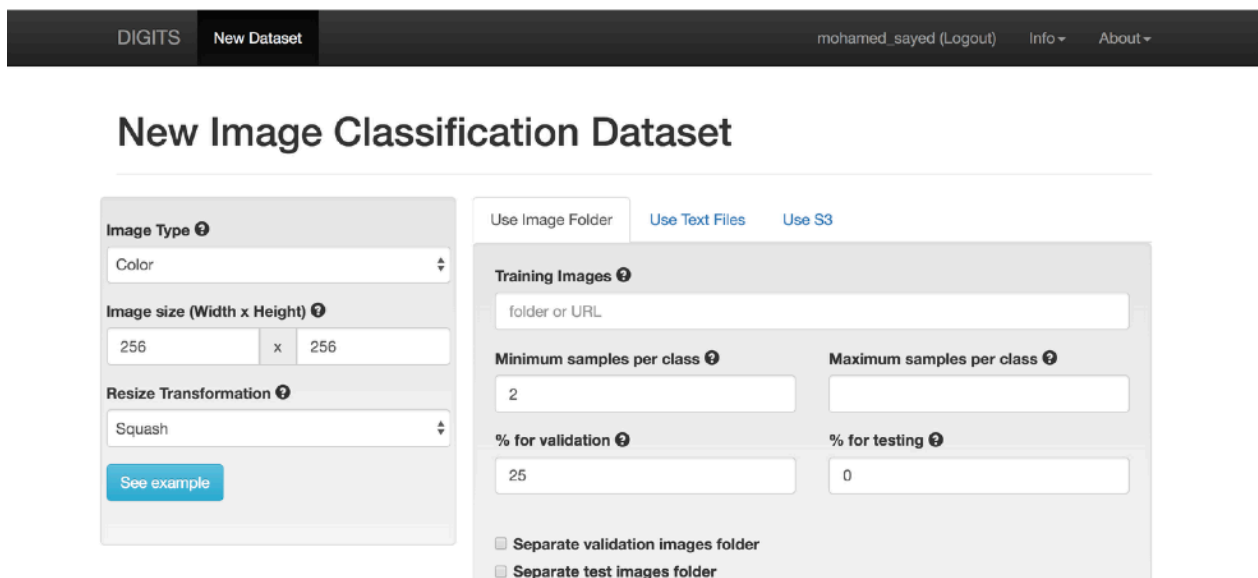
the data looks like this photo —>



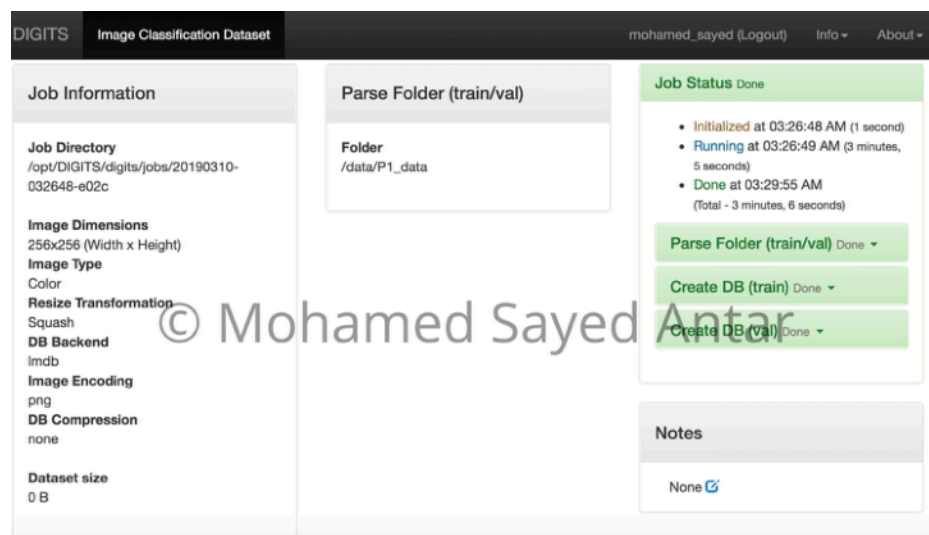
first by opening the DIGITS workspace we should see something like this



- by choosing images then classification we will see something like so

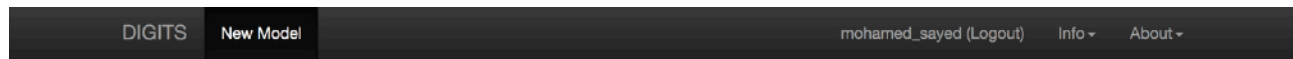


- adding our dataset url in this case using udacity workspace it will be `/data/P1_data`
- more setting can be done here like Minimum samples per class or maximum samples per class or even the percentage of validation photos and testing photos
- then by giving the dataset a name and choosing create :



the model

- by choosing `images` then `classification`



New Image Classification Model

The configuration form for a new image classification model. It is divided into three main sections: 'Select Dataset' (with a text input), 'Python Layers' (with a 'Server-side file' input and a 'Use client-side file' checkbox), and 'Solver Options' (with inputs for 'Training epochs' (30), 'Snapshot interval (in epochs)' (1), 'Validation interval (in epochs)' (1), and 'Random seed' ([none])). To the right is a 'Data Transformations' section with 'Subtract Mean' (Image) and 'Crop Size' (none).

- now we can set the number of epochs, Snapshot interval, Base Learning Rate, Validation interval
- by choosing one of the 3 networks provided LeNet, AlexNet and GoogLeNet in this case AlexNet gave a good results.
- giving the model a name then create.

The job status page for the 'Image Classification Model'. It shows the 'Job Directory' path, 'Disk Size' (0 B), and links for 'Network (train/val)', 'Network (deploy)', 'Network (original)', 'Solver', and 'Raw caffe output'. The 'Dataset' section shows 'inference1' is 'Done' at 03:29:55 AM, with 'Image Size' 256x256, 'Image Type' COLOR, and 'DB backend' lmdb. It also shows 'Create DB (train)' with 7570 images and 'Create DB (val)' with 2524 images. The 'Job Status' section shows a timeline: 'Initialized' at 03:40:09 AM (1 second), 'Running' at 03:40:10 AM (17 minutes, 33 seconds), and 'Done' at 03:57:44 AM (Total - 17 minutes, 35 seconds). A 'Train Caffe Model' button is shown as 'Done'. A 'Related jobs' section at the bottom shows 'Image Classification Dataset'.

for the Collected Data there are 1070 images for the following 3 classes:

1. spoon
 2. fork
 3. no-thing
- the collected data using mobile camera then by resizing the collected data to be 256*256
 - by choosing images then classification.
 - adding our dataset url in this case using udacity workspace it will be /data/kitchen
 - as previous more setting can be done like Minimum samples per class or maximum samples per class or even the percentage of validation photos and testing photos
 - then by giving the dataset a name and choosing create :



DIGITS Image Classification Dataset mohamed_sayed (Logout) Info About

inference4
Owner: mohamed_sayed

Clone Job Delete Job

Job Information
Job Directory
/opt/DIGITS/digits/jobs/20190310-025702-5988
Image Dimensions
256x256 (Width x Height)
Image Type
Color
Resize Transformation
Squash
DB Backend
Imdb

Parse Folder (train/val)
Folder
/data/kitchen
Number of Categories
3
Training Images
1070
Validation Images
356 (25.0%)

Job Status Done

- **Initialized** at 02:57:02 AM (1 second)
- **Running** at 02:57:04 AM (15 seconds)
- **Done** at 02:57:19 AM (Total - 16 seconds)

Parse Folder (train/val) Done

Create DB (train) Done

Create DB (val) Done

the model

- by choosing `images` then `classification`
- now we can set the number of epochs, Snapshot interval, Base Learning Rate, Validation interval
- by choosing one of the 3 networks provided LeNet, AlexNet and GoogLeNet in this case GoogLeNet gave a better results than AlexNet.
- giving the model a name then create.

DIGITS Image Classification Model mohamed_sayed (Logout) Info About

fourth_model
Owner: mohamed_sayed
[Clone Job](#) [Abort Job](#) [Delete Job](#)

Job Directory
/opt/DIGITS/digits/jobs/20190310-025832-r4ab
Disk Size
32.7 KB
Network (train/val)
[train_val.prototxt](#)
Network (deploy)
[deploy.prototxt](#)
Network (original)
[original.prototxt](#)
Solver
[solver.prototxt](#)
Raw caffe output
[caffe_output.log](#)

Dataset
[inference4](#)
Done 02:57:19 AM
Image Size
256x256
Image Type
COLOR
DB backend
lmdb
Create DB (train)
1070 images
Create DB (val)
356 images

Job Status Initialized
Initialized at 02:58:32 AM
[Train Caffe Model](#) Initialized

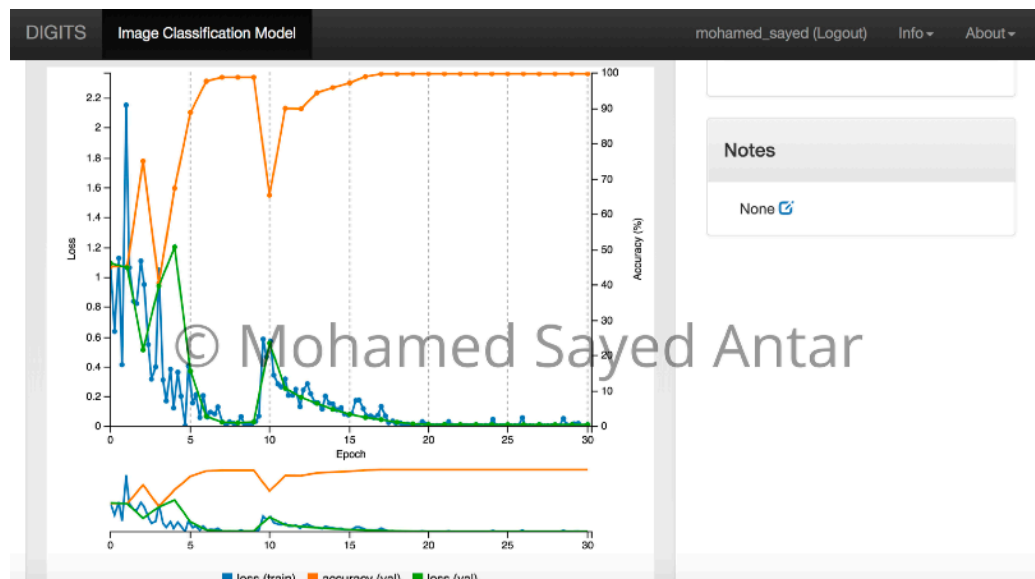
Hardware
Tesla K80 (#0)
Memory
3.28 GB / 11.2 GB (29.3%)
GPU Utilization
96%

4 Results

first the Supplied Data

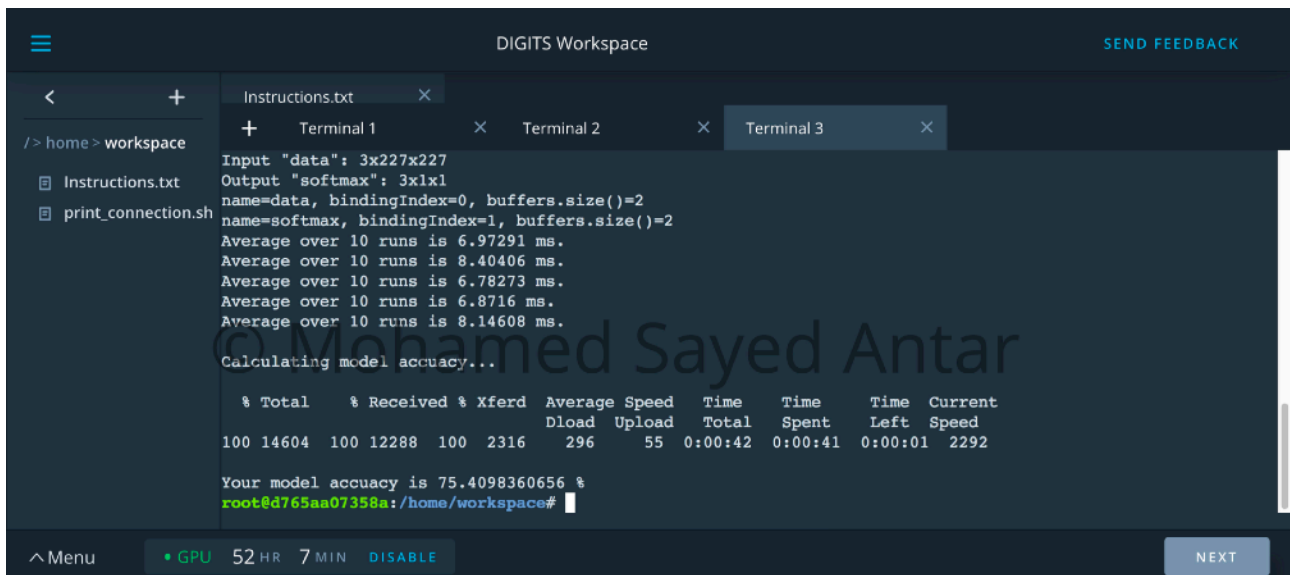
the output model

graph for the
supplied data
using AlexNet



two ways to see the model results as below:

- by running the `evaluate` command in a new terminal in udacity workspace the results will be like so



The screenshot shows the DIGITS Workspace interface with a terminal window open. The terminal displays the output of the `evaluate` command, which includes the input data shape, output softmax shape, and a table of performance metrics. The model accuracy is reported as 75.4098360656%.

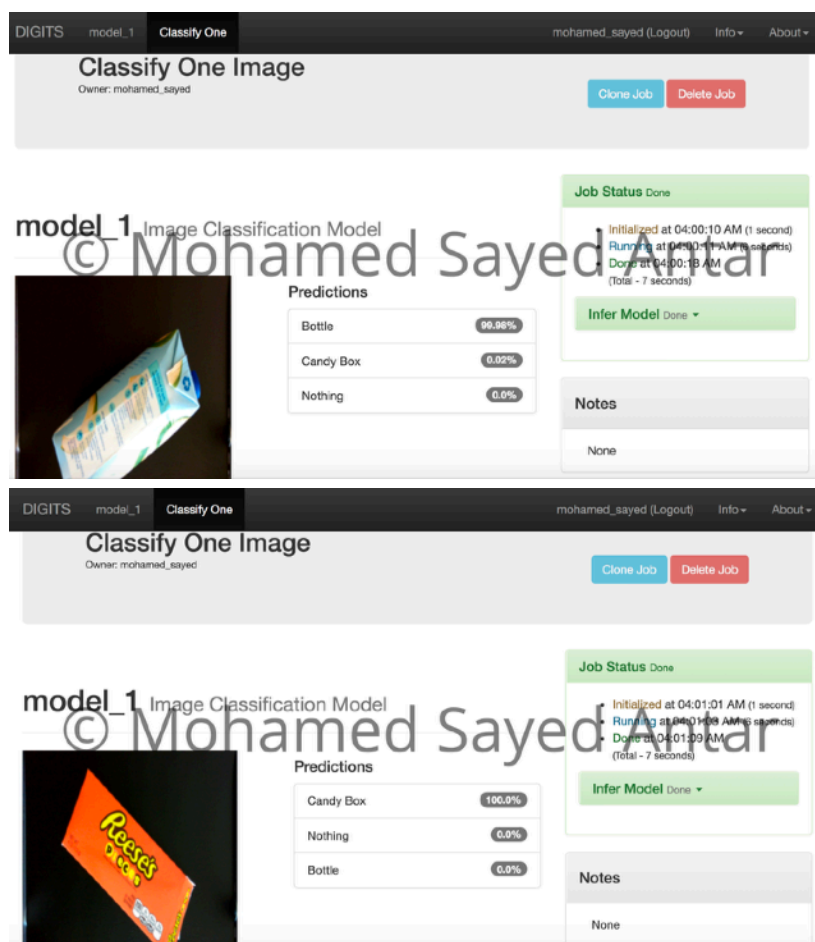
```
Input "data": 3x227x227
Output "softmax": 3x1x1
name=data, bindingIndex=0, buffers.size()=2
name=softmax, bindingIndex=1, buffers.size()=2
Average over 10 runs is 6.97291 ms.
Average over 10 runs is 8.40406 ms.
Average over 10 runs is 6.78273 ms.
Average over 10 runs is 6.8716 ms.
Average over 10 runs is 8.14608 ms.
Calculating model accuracy...

% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
   Dload  Upload   Total             Spent    Left     Speed

100 14604  100 12288  100  2316    296    55  0:00:42  0:00:41  0:00:01  2292

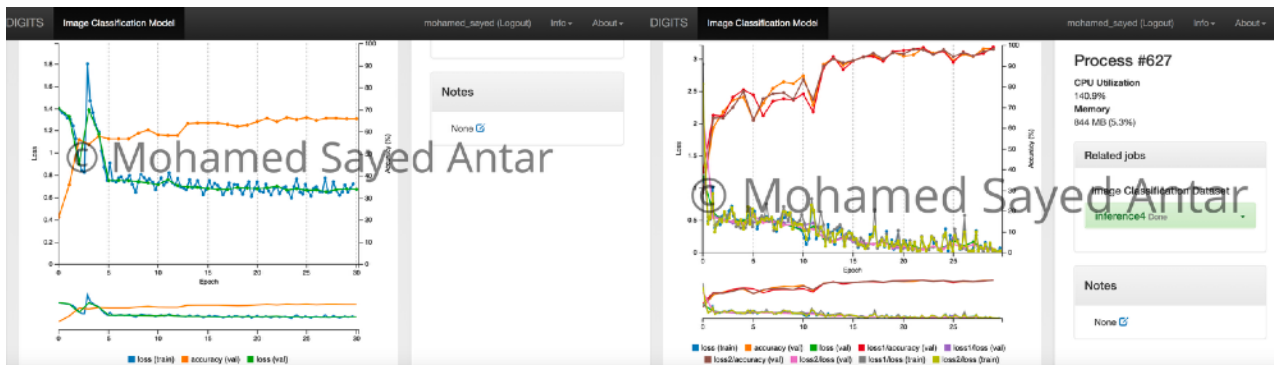
Your model accuracy is 75.4098360656 %
root@d765aa07358a:/home/workspace#
```

- by choosing images for test



second the Collected Data

the 2 models results AlexNet and GoogLeNet



- to see this model result by choosing images for test

DIGITS fourth_model **Classify One** mohamed_sayed (Logout) Info About

Classify One Image

Owner: mohamed_sayed

Clone Job Delete Job

fourth_model Image Classification Model

Job Status Done

- Initialized at 03:07:25 AM (1 second)
- Running at 03:07:26 AM (12 seconds)
- Done at 03:07:38 AM (Total - 13 seconds)

Infer Model Done

Predictions

fork	92.85%
spoon	6.3%
no thing	0.85%

Notes

DIGITS fourth_model **Classify One** mohamed_sayed (Logout) Info About

Classify One Image

Owner: mohamed_sayed

Clone Job Delete Job

fourth_model Image Classification Model

Job Status Done

- Initialized at 03:08:50 AM (1 second)
- Running at 03:08:51 AM (11 seconds)
- Done at 03:09:03 AM (Total - 12 seconds)

Infer Model Done

Predictions

spoon	99.73%
fork	0.27%
no thing	0.0%

Notes

5 Discussion

huge difference in accuracy between AlexNet and GoogLeNet as shown previously and this difference appear more with small number of data given to the model like the previous case about 1070 images only for classification, while this difference become smaller with increasing the number of data, after many attempts the inference time for GoogLeNet is greater than AlexNet.

6 Conclusion / Future Work

for the inference time AlexNet is good, but developers in many cases look for the accuracy so GoogLeNet is the best choice, increasing the number of collected data will lead to increasing accuracy but in case too much data the change will be very small almost no thing.

for the robotic kitchen increasing the number of classes with almost every kitchenware known till now, will lead to more intelligent robots due to variety in classifying objects ability

7 references

<https://www.asme.org/engineering-topics/articles/robotics/the-robotic-kitchen-is-cooking>

<https://www.iflscience.com/technology/robot-chef-home-could-arrive-2017/>

<https://meeee-services.com/how-to-use-cooking-robots-in-your-kitchen/>