Project: Robotic Inference

Abstract:

This is a report for Project: Robotic Inference. This project contains two parts: the first part is a practising of the NVIDIA DIGITS workflow, whose purpose is to gain familiarity with the workflow, and as a preparation for the second part of the project. A data set was provided and a network model was trained on that data set. The model was evaluated and has achieved an inference time of 5.5 ms (on average), and has an accuracy of 75.41%, which satisfied the requirement. For the second part of the project, an idea of training a classification network to classify cats is proposed. A customized data set consists of 2785 images of cats was collected, and a network model was trained on that data set. The model has an overall accuracy of 82.67%, and was evaluated manually by performing single and multiple images classification tests.

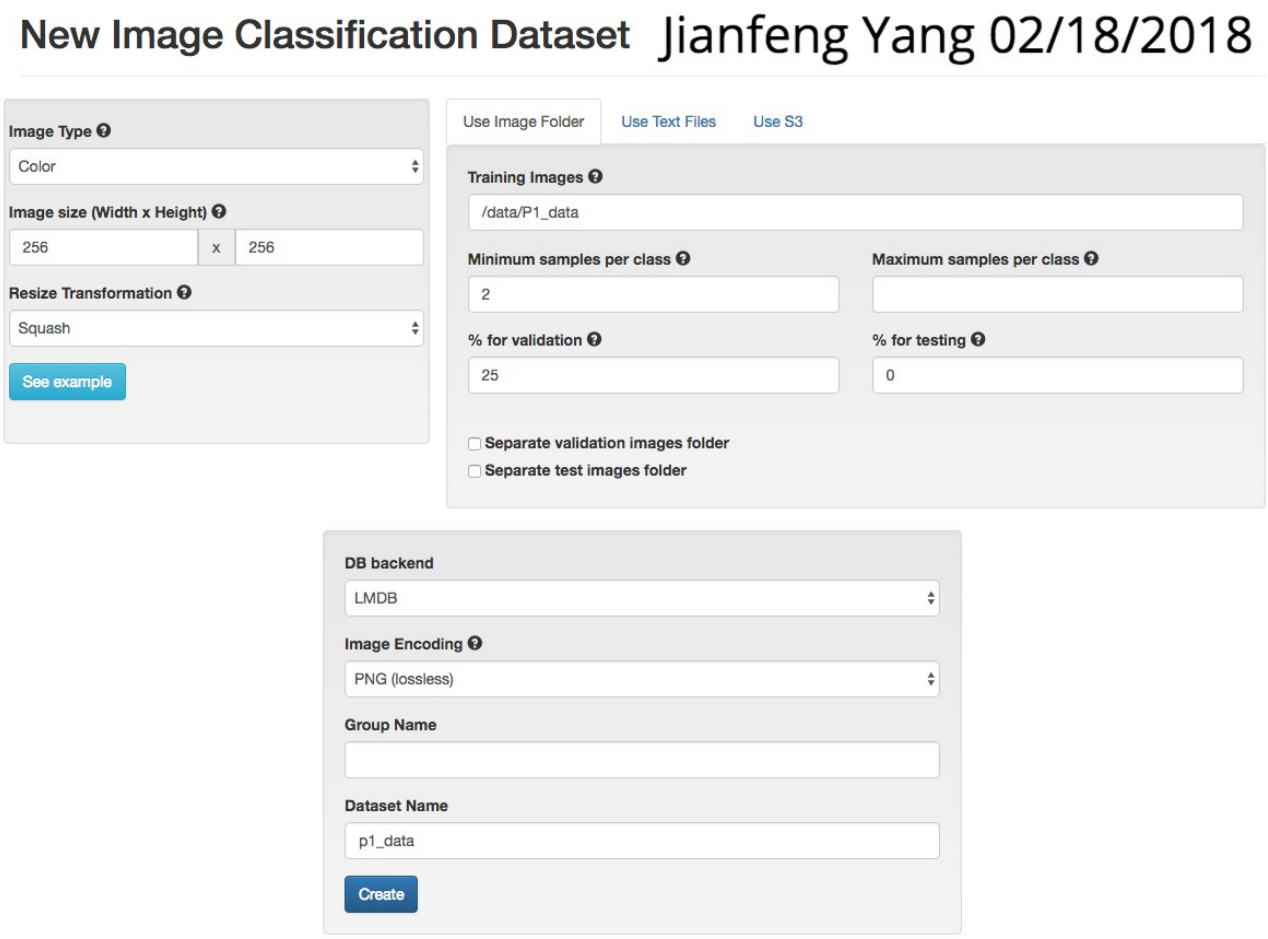
Part A: Practicing with the DIGITS Workflow

Introduction:

This is a practice of the NVIDIA DIGITS workflow. A detailed guidance is provided in the course along with the data set.

Data Acquisition

- Data set is provided on DIGITS server
- Create data set on DIGITS as instructed. Keep everything as default, and name it p1_data.



- Observe the data set

It can be seen that this data set has 3 categories: bottle, candy box, and nothing. After the split for training set and validation set, the numbers of images in each set are:

1. training set:
    - bottle: 3426
    - nothing: 2273
    - candy box: 1871
2. validation set:
    - bottle: 1142
    - nothing: 758
    - candy box: 624

So in total this set has 10094 images, and they are all color images sized 256 x 256.

Background / Formulation:

- Choose the network

Available network on DIGITS:

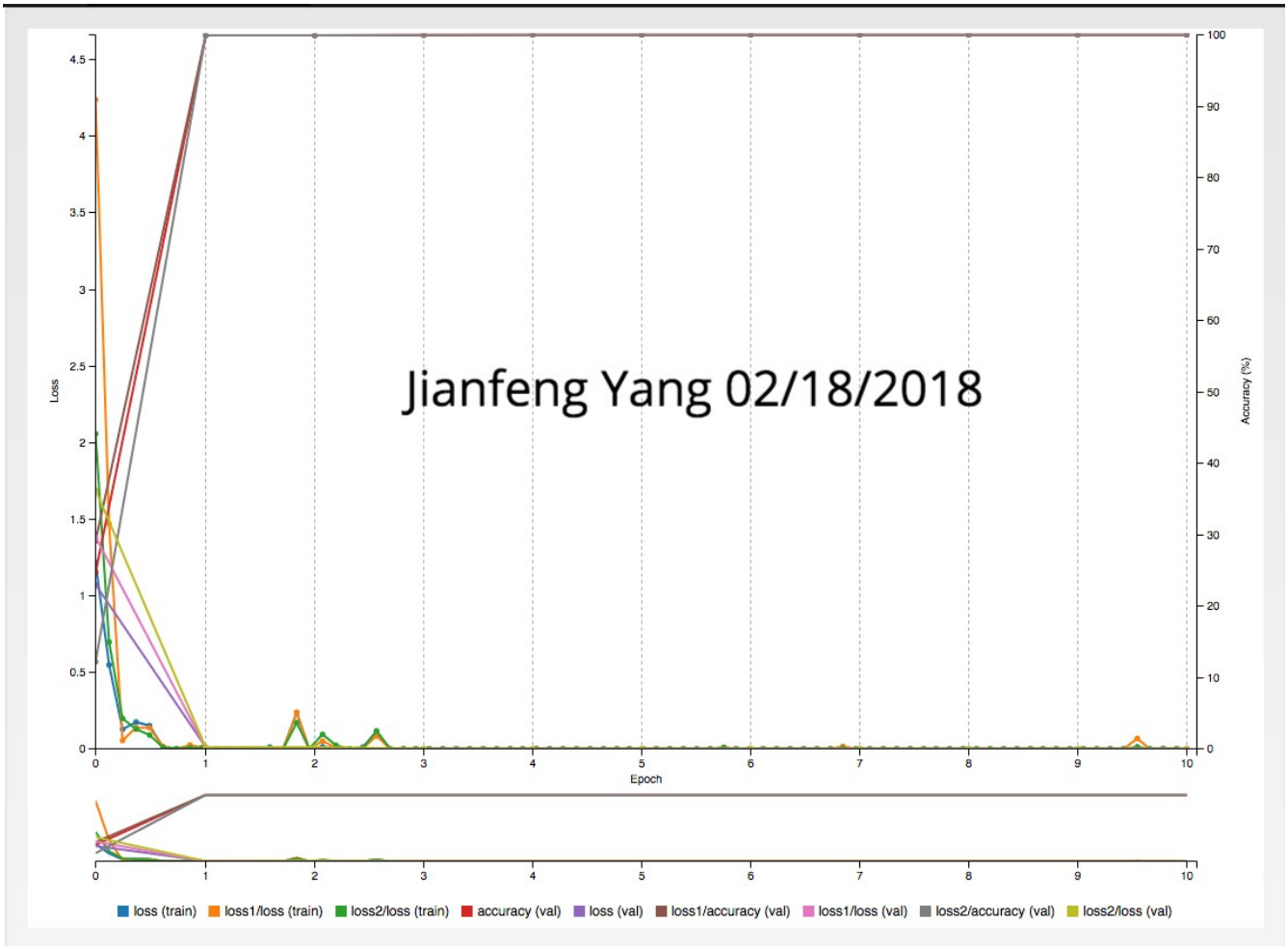
Network	Intended image size
LeNet	28x28 (gray)
AlexNet	256x256
GoogLeNet	256x256

Based on the observation, all the images in the data set are color images sized 256 x 256. This essentially removes LeNet since it was intended for 28 x 28 grayscaled images. Between the rest two networks, namely GoogLeNet and AlexNet, GoogLeNet is chosen as the target network. According to this article (https://medium.com/@siddharthdas_32104/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5)[1], the major difference between AlexNet and GoogLeNet is the reduction of parameters: from 60 million in AlexNet to a drastically reduced 4 million. It's reasonable to believe that this reduction can help with the training time, and hence it is chosen over AlexNet as the target network. For a quick start, the epoch number is set to 10. Eeverything else was kept default. The model was named googLeNet_1, in which the 1 indicates first run for this network.

Result

First run

- Epoch is set to 10



A interesting observation: after 1 epoch, the accuracy reaches 99%, after 3 epochs it reaches 100%, which seems too good to be true. After examining the data set more carefully, several reasons can be draw as to why the accuracy was so high:

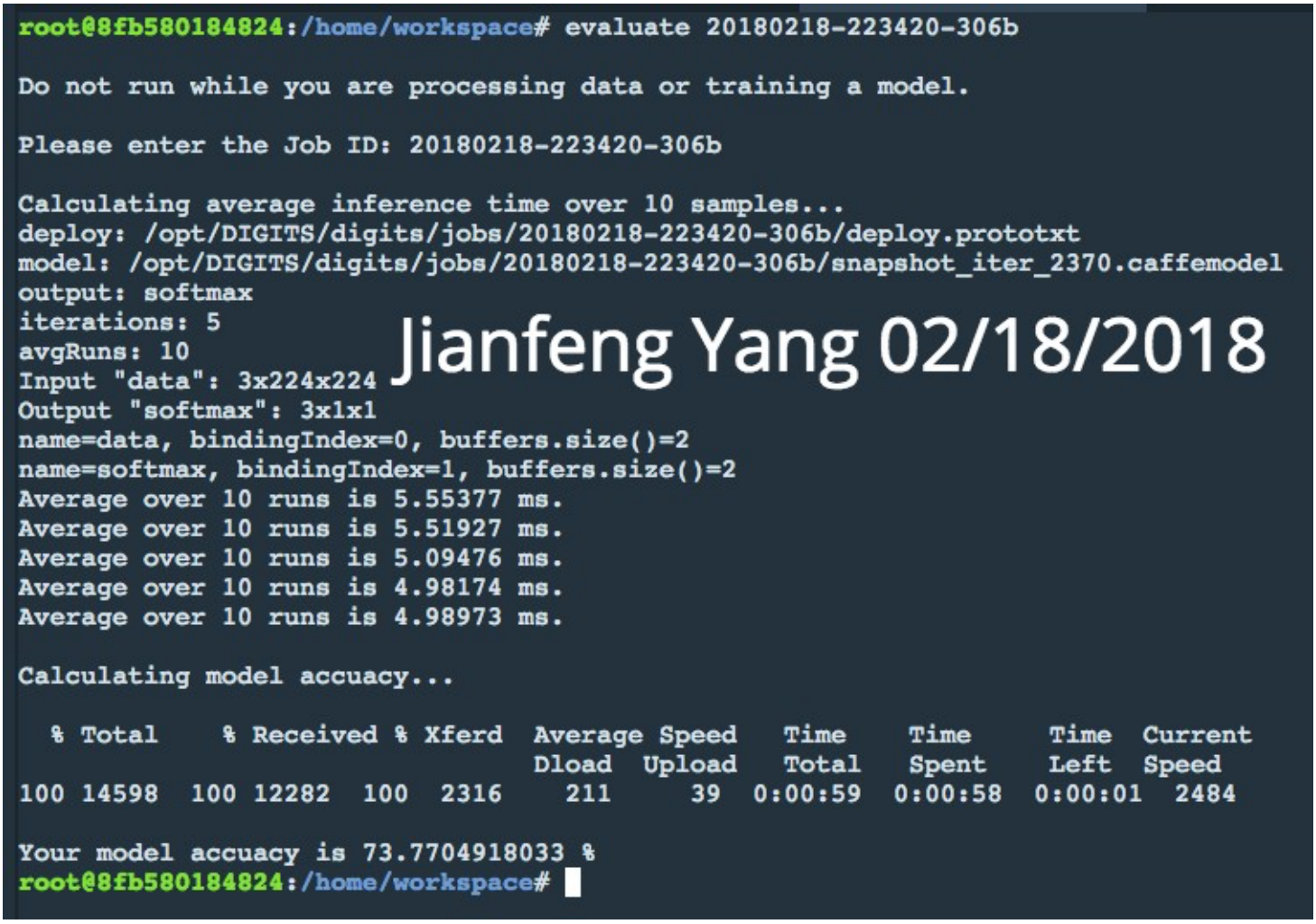
1. A comparatively large data set. For a data set that contains only 3 categories, it has over 10000 images, and this helps the network to learn more about the objects;

2. All the imges have a unified background. Being captured in the same environment really helps the network to learn better.

3. Simple objects. The bottle and candy box are all relatively simple, and each have a regular shape.

Evaluation

Here is the evalution for the network after first training:

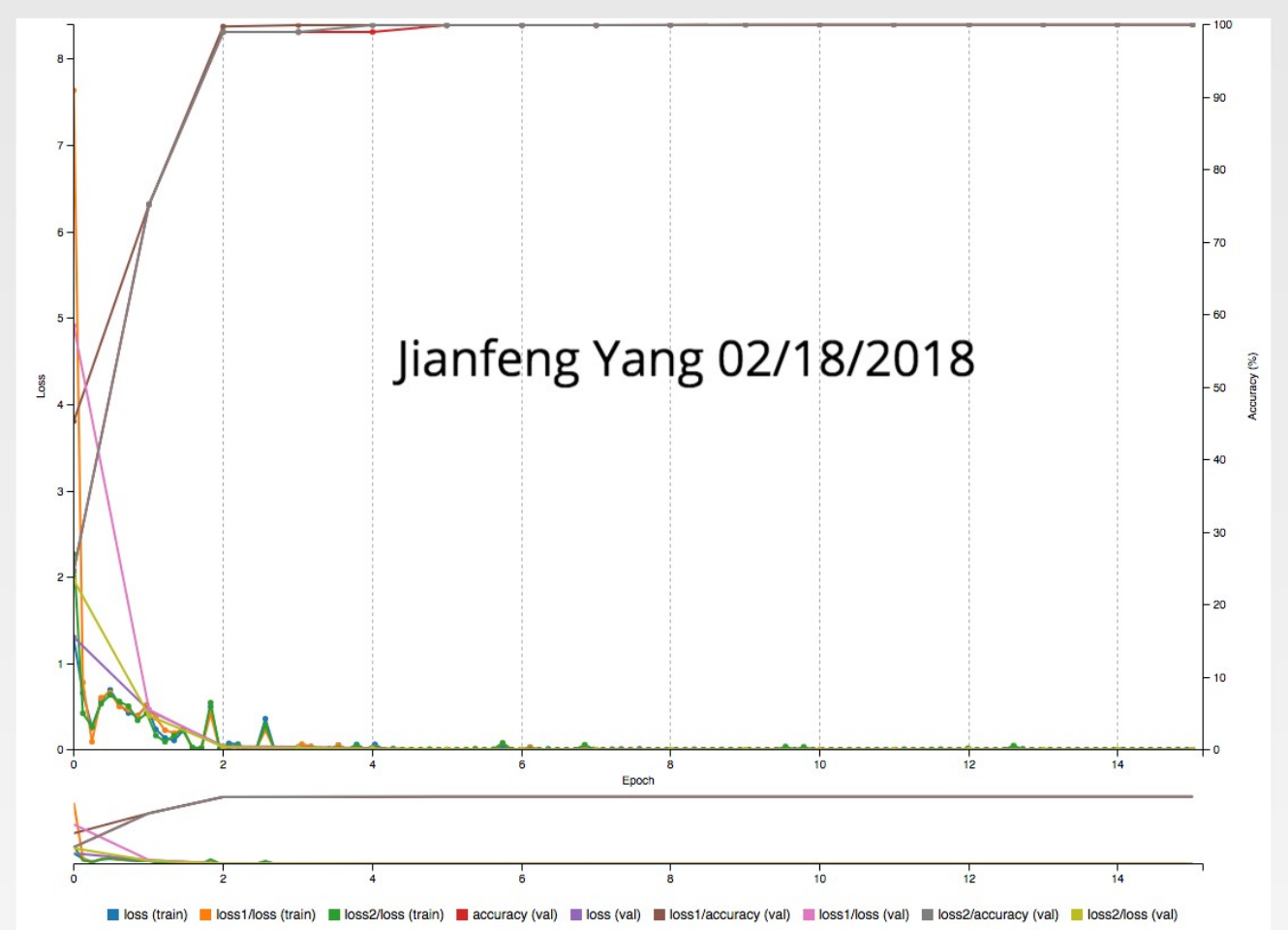


As seen in the screenshot, the run time is around 5 ms to 5.5 ms, and the model accuacy is 73.7704918033%, only slightly off the target of 75%. Based on previous deductions, a second run with more epoch should be able to promote the accuracy above requirement.

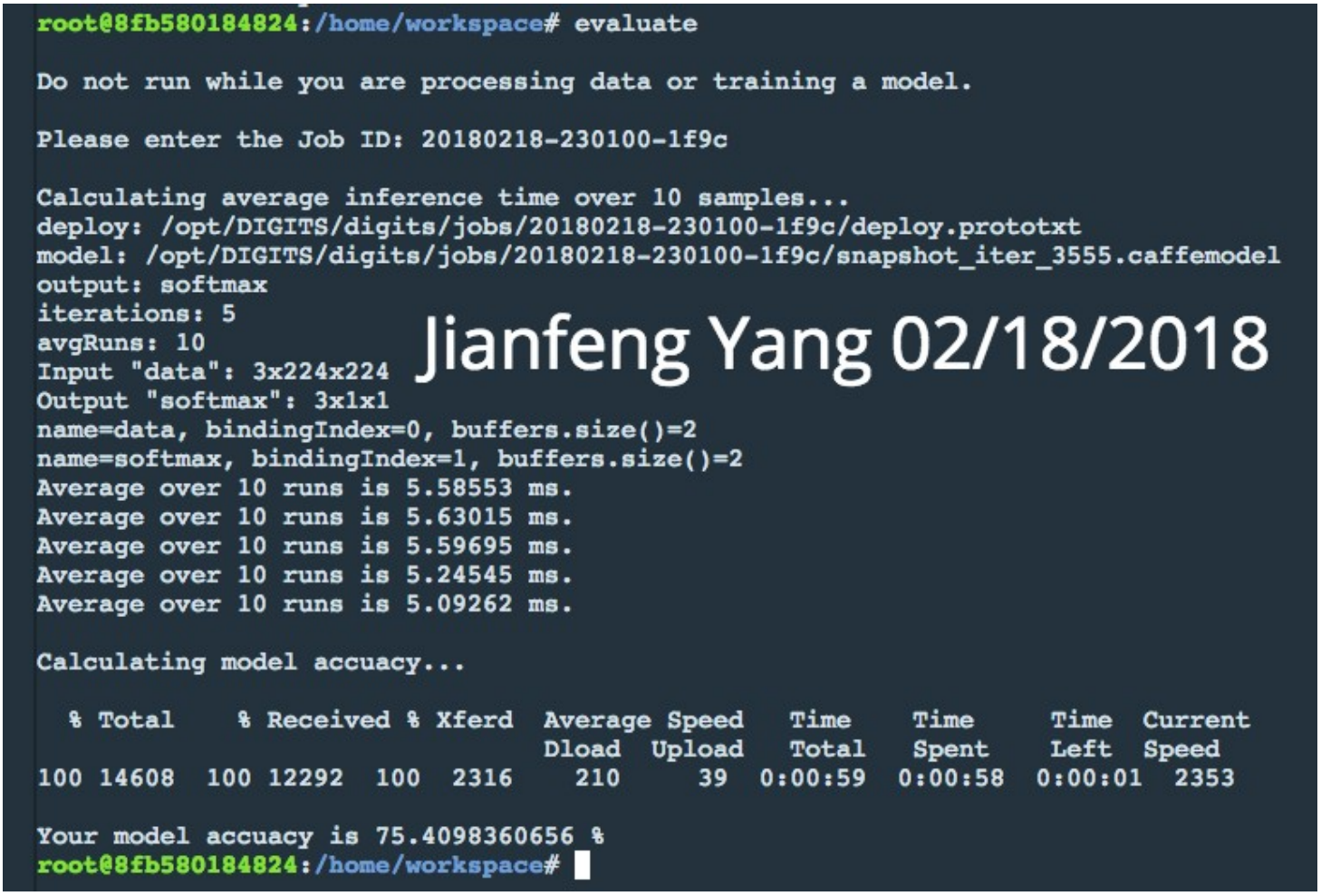
The model is saved at model\20180218-223420-306b_epoch_10.0\

Second run

- Epoch is increased to 15



This time, the overall trend is almost the same but do notice that the accuracy increases at a lower rate before epoch 2. After 3 epochs, the accuracy reaches over 99%, and after 6 epochs it reaches 100% again.



The run time is actually increased a bit to around 5.5 ms, and the model accuacy is 75.4098360656%, successfully achived the target of 75%.

The model is saved at model\20180218-230100-1f9c_epoch_15.0\

Discussion

Overall this network performs very good, achiving requirments in only two runs with 15 epochs. The reasons, as discussed above, could be due to the comparatively large data set, standardized images, unified backgrounds and environments, and simple objects. And this gives an inspiration on how to organize the images when collecting customized data set for Part B.

Part B: Classification Network For Cats

Introduction:

For this part in the inference project, a classification network to classify cats is proposed. More specifically:

There are two cats in my house, with distinctive features.

As you can see from the picture, one of them is a shorthair with black, gray and white color, and is a male named Cucumber. The other is a shorthair with orange color, and is a female named Ginger.

The intention of this project is to propose a classification network that can serve as a trigger for the automatic food dispenser. The main reason for building such a system is that one of the cats, namely Cucumber, is going on a diet while the other one is not. Thus, their food needs to be differentiated and dispensed on a different schedule: two meals a day for Cucumber and three meals for Ginger.

At first, this job was purely manual, and someone must feed them according to their unique food types and schedules. Counting in all the differences, that means someone has to manually fill their food tank at most 5 times a day. After a while this becomes too tedious to persist. Then an automatic feeder was set up to dispense the food on a timer, which effectively eliminated the labor of manual dispensing. Still, it cannot satisfy the need to differentiate their food since it can only hold one type of food at a time. A second feeder holding the other type of food would be a logical extension to the setup, but a person would still have to be present to prevent one cat from consuming another one's food. A fully automatic feeding system based on a classification network should be able to replace the human work completely.

Ideally, this classification network would need to distinct the two cats from each other, and trigger the dispenser only when it's the scheduled time, AND the right cat is approaching. For this purpose, there should be essentially 3 categories for the classification network: Cucumber, Ginger, and other cats or

animals that are neither of them.

Data Acquisition

- Data set is collected from my own photo collection of the cats, and the internet.

- A tatal of 2785 images are collected. Among them, a total of 1349 ges are of Cucumber, and a total of 501 images are of Ginger. These images all comes from the photo collection I took for them and are of different angles, poses, and are taken at different times and life stages. The rest are all images of other cats or animals that I randomly saved from the internet.

- The selection of the photos are intentional so that no photos are included with the presence of both of the cats in frame at the same time. And also the photos are excluded when they include faces of people or the cats are not in the center of the frame.

- There are repeatitions in the photo collection that may only differ slightly from each other, due to the nature of the high speed continuous capturing in order to get a good picture of the cats. Since they are used for training, however, which means that they will be used in a bunch anyway, a seperation and selection among them are not performed.

- Normalize the pictures

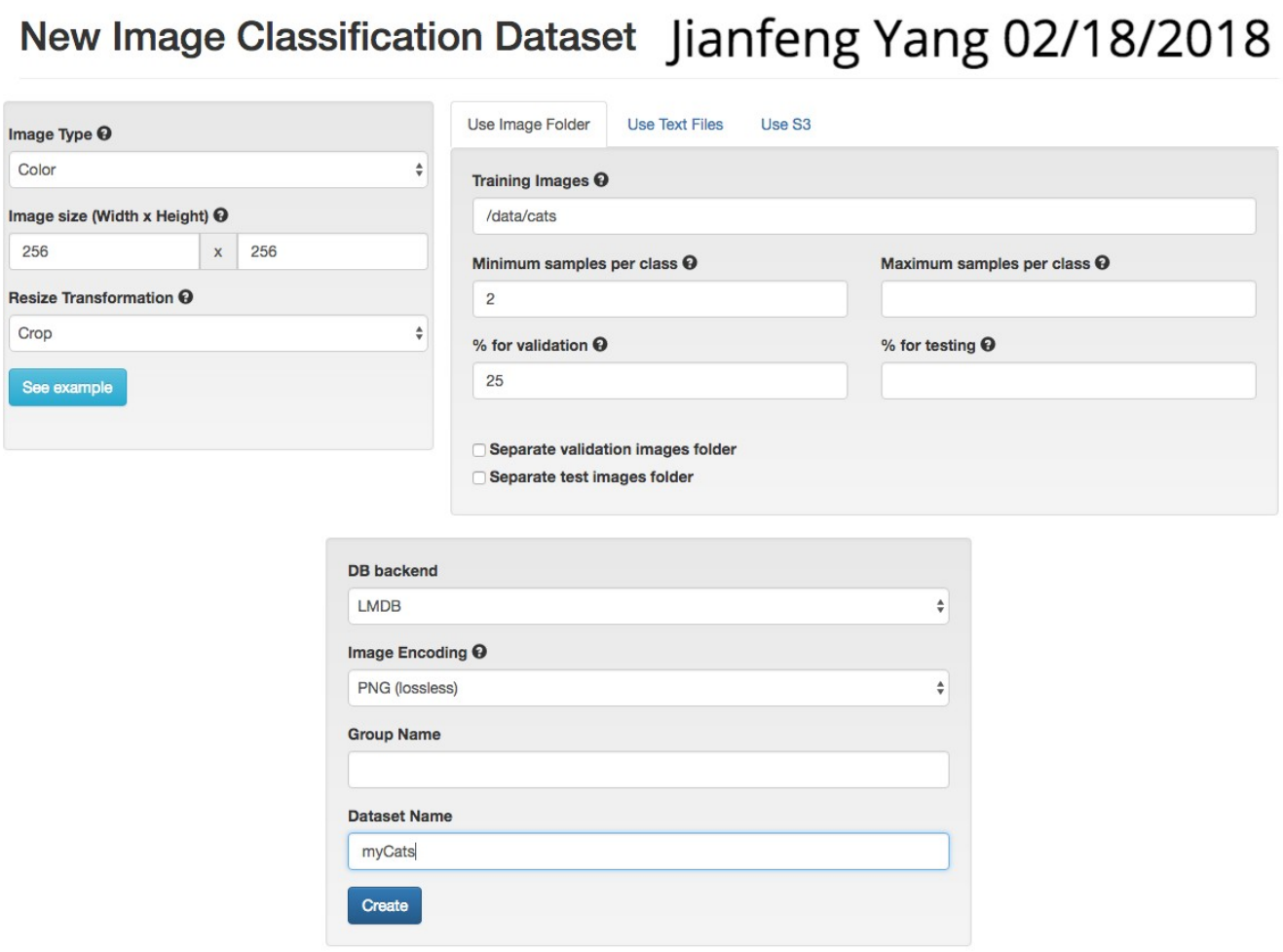
  As observed in Part A, a uniformed, well organized data set can help improve the training process. Thus before importing into DIGITS, besides grouping into labeled folder, a series of normalization procedures are also performed on the images:

  1. Batch rename
     All the images are named as "label" + "_x" where the x is a sequential number, like "cucumber_1", "cucumber_2", etc.

  2. Batch crop and resize
     All the images are cropped around the corner and resized to a 256 x 256 square image.

  3. Batch convert to .jpg file
     All the images are converted to the .jpg format.

The data set is then uploaded to the DIGITS server under the /data folder.



- Observe the data set

It can be seen here that this data set has 3 categories: cucumber, ginger, and others. After the split for training set and validation set, the numbers of images in each set are:

- training set:
  - cucumber: 1012
  - others: 701
  - ginger: 376
- validation set:
  - cucumber: 337
  - others: 234
  - ginger: 125

So in total this set has 2785 images, and they are all color images sized 256 x 256. This also confirms that all the images are uploaded and processed correctly.

Background / Formulation:

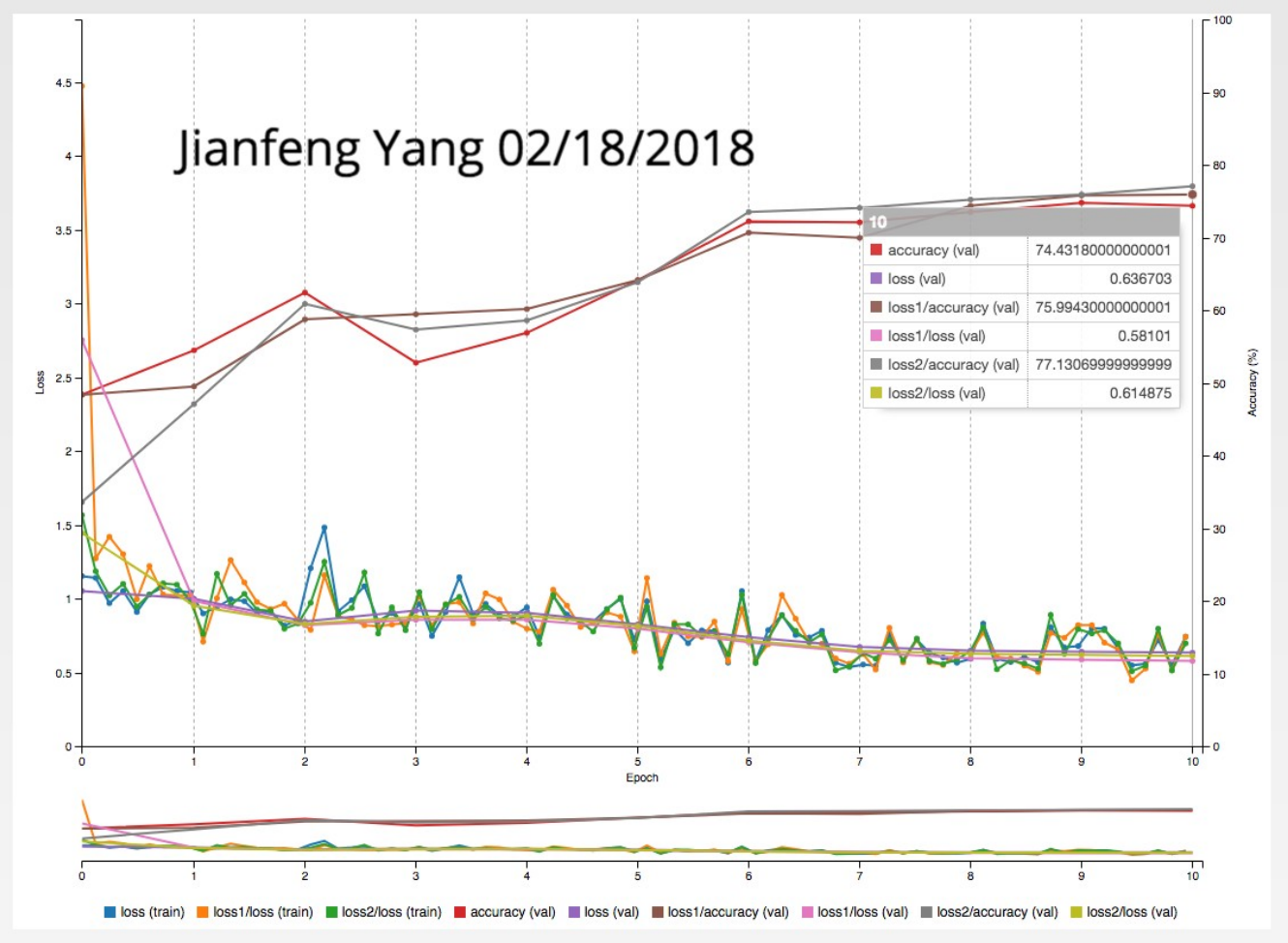
- Choose the network

Since the data set is normalized like the one used in Part A, it is reasonable to follow the same network selection. So googLeNet is chosen as the target network, and the epoch number is set to 10 just for a quick test run. Eeverything else was kept default. The model was named googLeNet_cats_1.

Result

First run

- Epoch is set to 10



Unlike in Part A, this time it took 6 epochs for the network to gain a mediocre accuracy, and even after the training is done, the accuracy is only 74.43%. A comparison between the data sets used in the two parts reveals several possible causes:

1. Comparing with the data set used in Part A which contains over 10000 images, this one with less than 3000 images is very small. This lack of abundant training materials is one of the reasons why the accuracy is way lower.

2. The images, being taken from a daily photo collection, have various background, lighting, and environments. This creats extra burdens on the network and contributes largely to the low accuracy.

3. The objects are much more complex. Comparing with bottles and boxes that are mostly regularly shaped, the objects here are cats, who are well known for their ability to reshape themselves into many different forms. This variation in forms creats another layer of difficulty for the network and thus further lowers the accuracy.

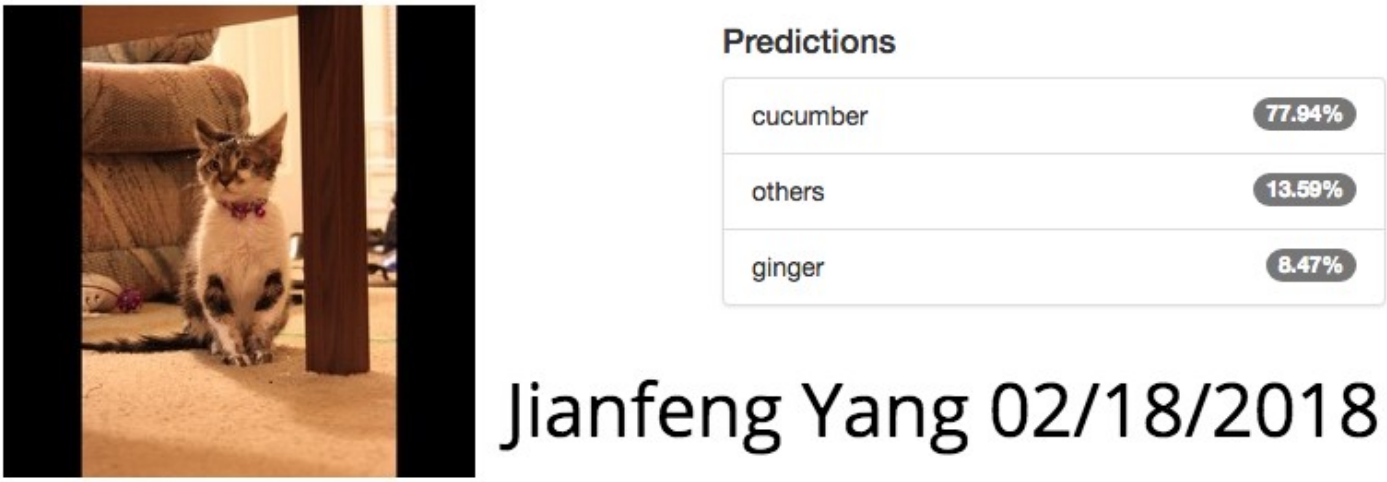
4. Comparing to the static objects like bottles and boxes, cats are alive and growing. The second data set contains photos of the cats that span from when they are kittens until they grow into adults, during which time their size, shape and many other features changed drastically. This essentially raised the level of difficulty even further.

Evaluation

Since there's no evaluation script for the customized data set, a manual inspection is performed.

- Single image classification

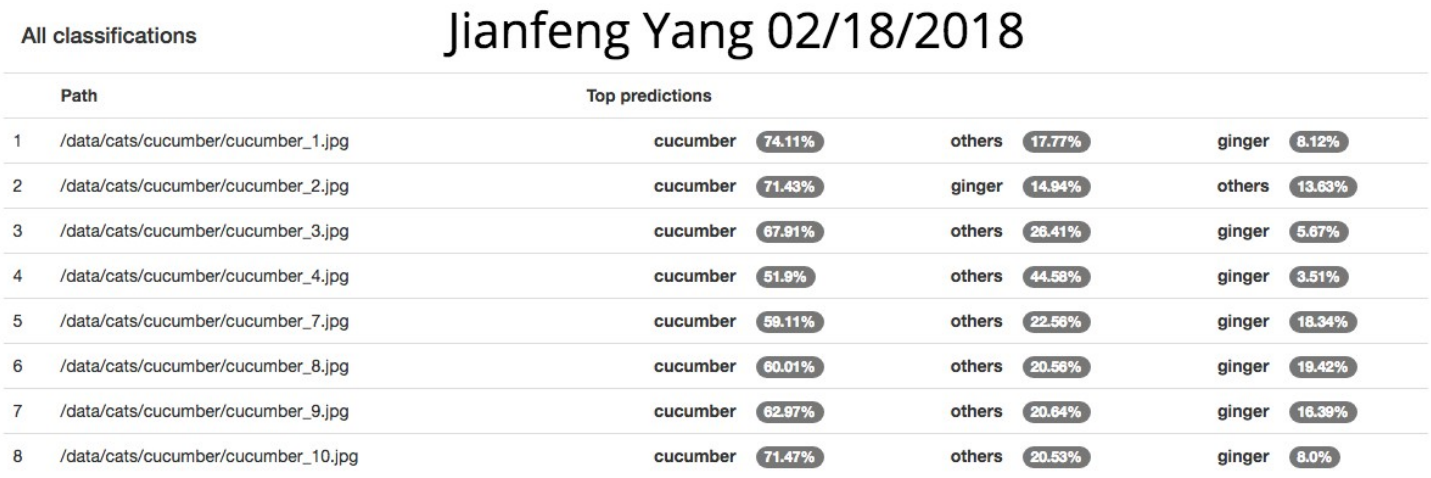
  Here are two classification tests performed on Cucumber's images.



The network classified them correctly with an accuracy of about 77%, which is higher than the overall accuracy. That means there must be some other images that the network performed poorly on.

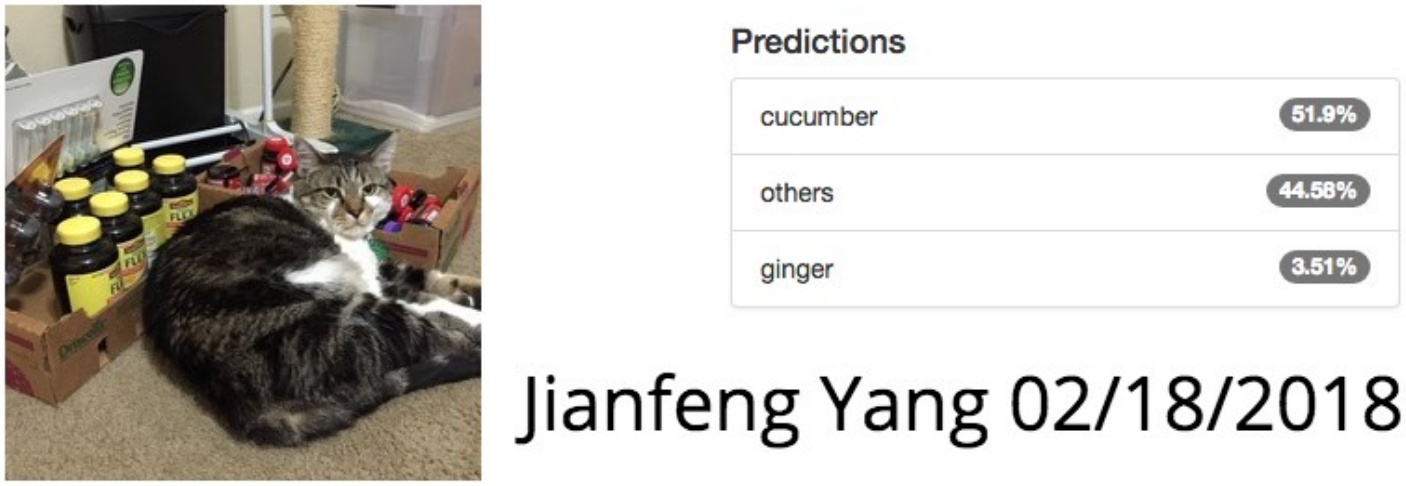
- Multiple image classification

  Use the images for Cucumber again, a multiple image classification test is performed and here is the result:



	Path	Top predictions		
1	/data/cats/cucumber/cucumber_1.jpg	cucumber 74.11%	others 17.77%	ginger 8.12%
2	/data/cats/cucumber/cucumber_2.jpg	cucumber 71.43%	ginger 14.94%	others 13.63%
3	/data/cats/cucumber/cucumber_3.jpg	cucumber 67.91%	others 26.41%	ginger 5.67%
4	/data/cats/cucumber/cucumber_4.jpg	cucumber 51.9%	others 44.58%	ginger 3.51%
5	/data/cats/cucumber/cucumber_7.jpg	cucumber 59.11%	others 22.56%	ginger 18.34%
6	/data/cats/cucumber/cucumber_8.jpg	cucumber 60.01%	others 20.56%	ginger 19.42%
7	/data/cats/cucumber/cucumber_9.jpg	cucumber 62.97%	others 20.64%	ginger 16.39%
8	/data/cats/cucumber/cucumber_10.jpg	cucumber 71.47%	others 20.53%	ginger 8.0%

All images were classified correctly, but the accuracy are not all that high. More specifically, the image cucumber_4.jpg only scored an accuracy of 51.9%. On that image, the second possible label, others, comes very close at 44.58%. Perform a single image classification on that image:



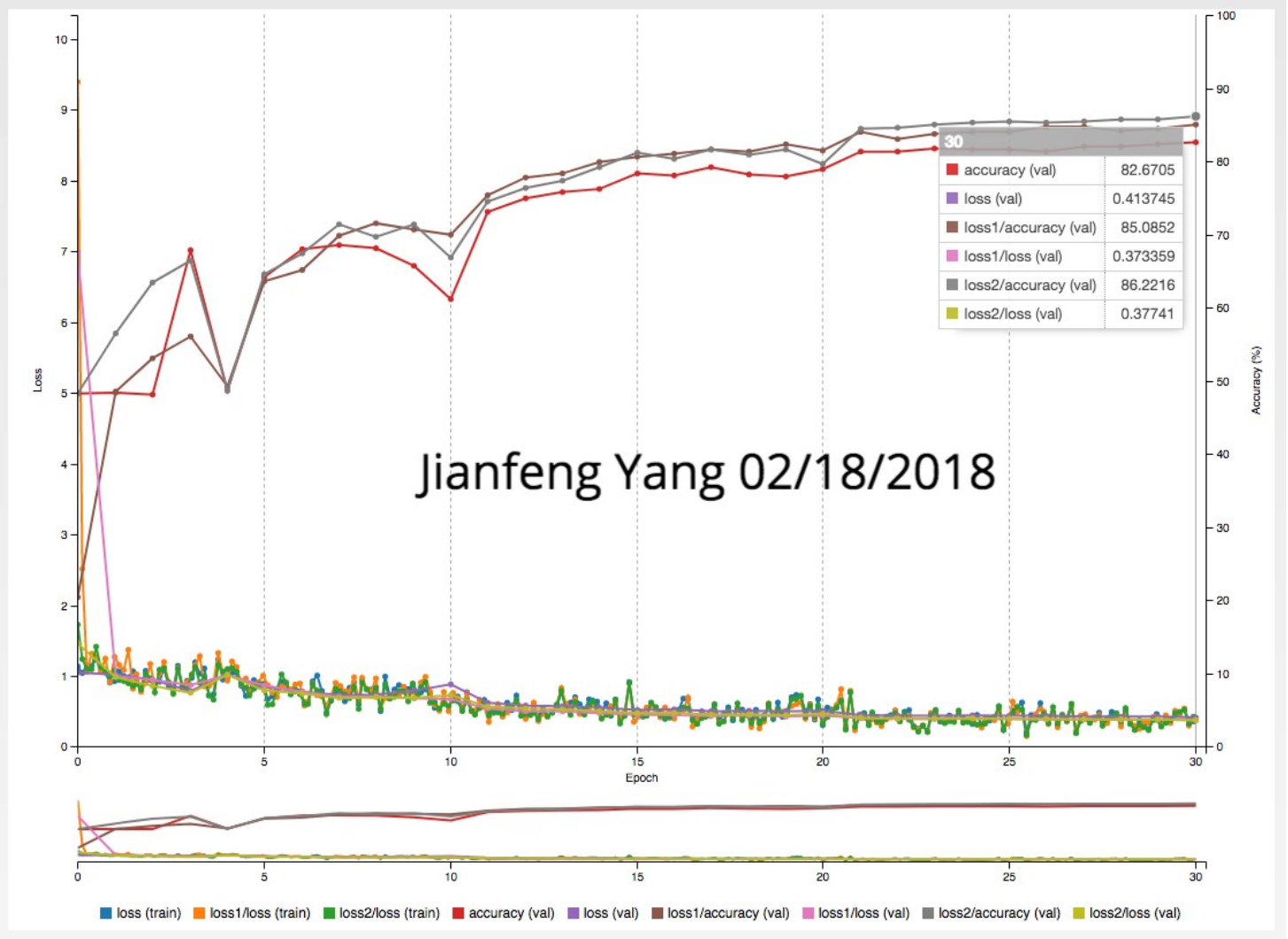
It's obvious that this image is hard to classify. There are a lot of other irrelevant objects in it, and the distinction between the cat and the background is not very clear.

It is very clear that the network needs more training.

The model is saved at model\catsModel\20180218-235336-45fe_epoch_10.0\

Second run

- Epoch is increased to 30



This time, the accuracy was increased to 82.67%.

Evaluation

- Multiple image classification

  Use the images for Cucumber.

All classifications

Jianfeng Yang 02/18/2018

	Path	Top predictions		
1	/data/cats/cucumber/cucumber_1.jpg	cucumber 92.53%	others 5.68%	ginger 1.79%
2	/data/cats/cucumber/cucumber_2.jpg	cucumber 81.43%	others 10.81%	ginger 7.75%
3	/data/cats/cucumber/cucumber_3.jpg	cucumber 82.36%	others 15.99%	ginger 1.65%
4	/data/cats/cucumber/cucumber_4.jpg	cucumber 61.97%	others 36.73%	ginger 1.3%
5	/data/cats/cucumber/cucumber_7.jpg	cucumber 85.14%	others 13.14%	ginger 1.72%
6	/data/cats/cucumber/cucumber_8.jpg	cucumber 79.49%	others 17.49%	ginger 3.02%
7	/data/cats/cucumber/cucumber_9.jpg	cucumber 43.76%	others 38.64%	ginger 17.6%
8	/data/cats/cucumber/cucumber_10.jpg	cucumber 67.28%	others 21.54%	ginger 11.18%

All images were classified correctly, and the accuracy were essentially promoted, except cucumber_9.jpg, which actually drops from 62.97% to 43.76%. Perform a single image classification on that image:

googLeNet_cats_2 Image Classification Model

Predictions

cucumber	43.76%
others	38.64%
ginger	17.6%

Jianfeng Yang 02/18/2018

It's not so obvious why this image is hard to classify. A speculation would be that there is a wide area of white in the background (the plastic bag) that might have confused the network when mixed with Cucumber's white fur.

A multiple image classification test was also done on Ginger's images:

All classifications Jianfeng Yang 02/18/2018

	Path	Top predictions		
1	/data/cats/ginger/ginger_1.jpg	others 94.9%	cucumber 4.27%	ginger 0.83%
2	/data/cats/ginger/ginger_2.jpg	others 60.22%	ginger 33.58%	cucumber 6.2%
3	/data/cats/ginger/ginger_3.jpg	ginger 92.02%	others 6.4%	cucumber 1.57%
4	/data/cats/ginger/ginger_4.jpg	cucumber 50.39%	ginger 35.99%	others 13.63%
5	/data/cats/ginger/ginger_7.jpg	ginger 96.32%	others 3.2%	cucumber 0.48%
6	/data/cats/ginger/ginger_8.jpg	ginger 96.4%	others 3.19%	cucumber 0.41%
7	/data/cats/ginger/ginger_9.jpg	ginger 94.82%	others 4.59%	cucumber 0.59%
8	/data/cats/ginger/ginger_10.jpg	ginger 84.38%	others 10.58%	cucumber 5.05%

Here are several notable observations:

1. The network can identify Cucumber, but failed to label all of Ginger's images correctly. Could it be the difference of their fur color?

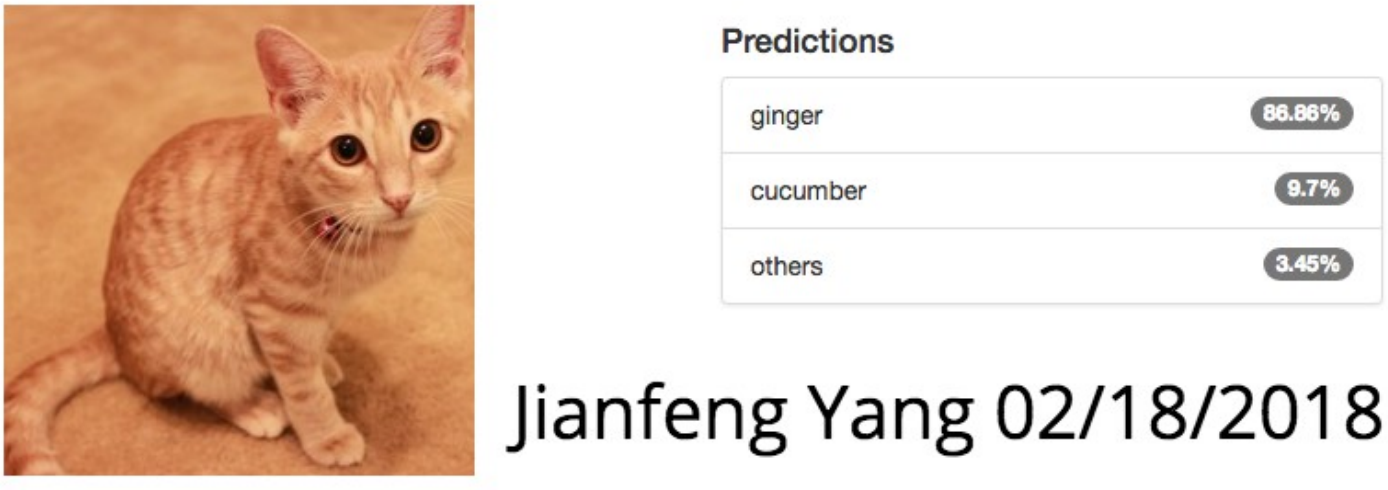
2. Though the network couldn't get all correct on Ginger's images, on the correct ones however, its accuracy was very high (80% ~ 90%); on the contrary, all the accuracy on Cucumber's images are mediocre (just over half for most of the time)

Note that 3 images are incorrectly labeled. Single image classification tests are performed on each one of them to further examine the reason.

- Single image classification

The above two images that were classified as Others. Just by looking, it's very hard to tell why these images were misclassified. But do notice that they persent a similar pattern: there's a large white, furry area in the image surrounding the cat. A reasonable speculation would be that in the Others category there is a similar looking cat that has orange and white fur, and the network was confused by the carpet or pad, thinking they are parts of the cat.

The image above was classified as Cucumber instead of Ginger. One possible reason could be that from this angle, most of the facial features are lost, and the white fur on her chest is confusing because Cucumber also has white fur on his chest.



This is an image that was correctly classified as Ginger. As stated before, for the correctly classified images of Ginger, the network can achieve a relatively high accuracy.

The model is saved at model\catsModel\20180219-010041-24ee_epoch_30.0\

Discussion

One major flaw when designing the category is that images of other animals, mostly cats, are used as the third category instead of a "nothing" category. Comparing to an empty background, this adds unnecessary complexity to the difficulty for the network, since there might be another cat in the Other category that resembles similar features with Cucumber or Ginger. The evaluation process above also indicates this possibility.

Regarding this specific project, accuracy should weight over inference time since it was to be integrated into a monitor network, under which circumstance the speed of the inference is less important since there's no instant reaction required.

Conclusion / Future Work

Overall, the network has achieved the goal. Still, there are many speculations that require confirmation. Should more time be granted, the network can be improved even further. Several improvments include:

1. Capture more images in the same environment
2. Use empty background as the third category
3. Train multiple runs, and eliminate images that are not suitable for the task
4. Try other networks, and tweak the parameters

- Possible usage
  As stated above, if this network is part of a monitor network, then the accuracy should be emphasized over inference time; on the other hand, if the project requires instant reaction, for instance, a "house defense system" against rabbits and squirrels, then the inference time should weight more than accuracy, since in that setup, the classification network can be trained to only deal with as few as two categories: friend, or enemy.

Reference

[1] "CNNs Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more …", Siddharth Das, Nov 16, 2017, Medium.com (https://medium.com/@siddharthdas_32104/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5)