

## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

### Introduction:

This lab allows students an entry point into running models from the command line. The model being created has the purpose of identifying fresh vs. rotten fruit. Specifically, and emphasis is placed on running models within a singularity container via the command line. Both interactive and batch modes are used to run and train models within a singularity container, this is so that comparisons can be made between the approaches and so students can begin to understand the positives and benefits between both ways of running models. In addition, one of the main challenges of the lab is to adjust hyperparameters in order to find the best possible combination of parameters for this given problem space.

---

### Dataset:

The data used to train this fresh vs. rotten fruit identifier is sourced from Kaggle.com and was used for a model creation competition. The data consists of 6 different categories of color images, fresh apples, fresh oranges, fresh bananas, rotten apples, rotten oranges, and rotten bananas. The data is 1.82 GB in size, and is divided into testing and training subsets, where the training set contains **10,901 images**, and the testing set contains **2,698 images**. The number of images in each image category varies, with the training set having a 700 image difference between the category with the lowest number of images to the category with the highest number of images, while the testing set has a variance of about 200 images per category.

---

### Part 1:

The models created throughout this lab all utilize the python file **Lab9.py**, written by Dr. Riley. One the highlights of this python file are all of the parameters that can be passed into this python file when it is called from the command line. The available parameters for Lab9.py is as follows:

- Data, the subdirectory of images for training
- Batch size, the batch size used for training
- Epochs, the number of epochs to use for training
- Main dir, the location that produced models are placed
- Augment data, Boolean indication for whether to use data augmentation
- Fine tune, Boolean indication for whether to use fine tuning

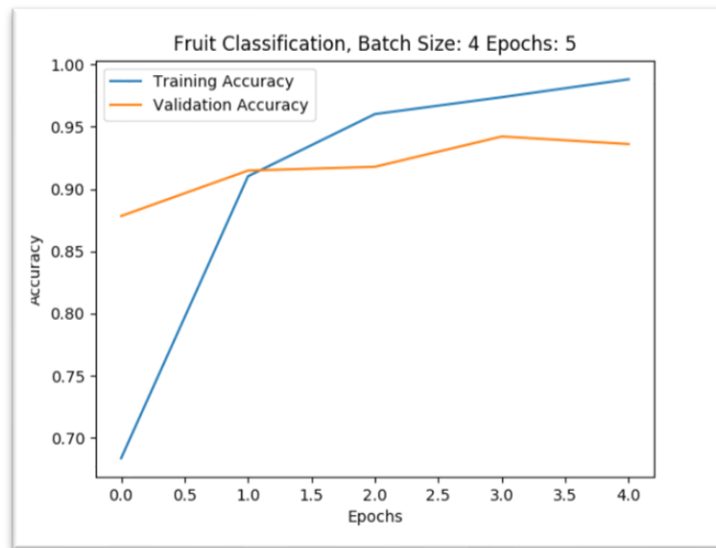
## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

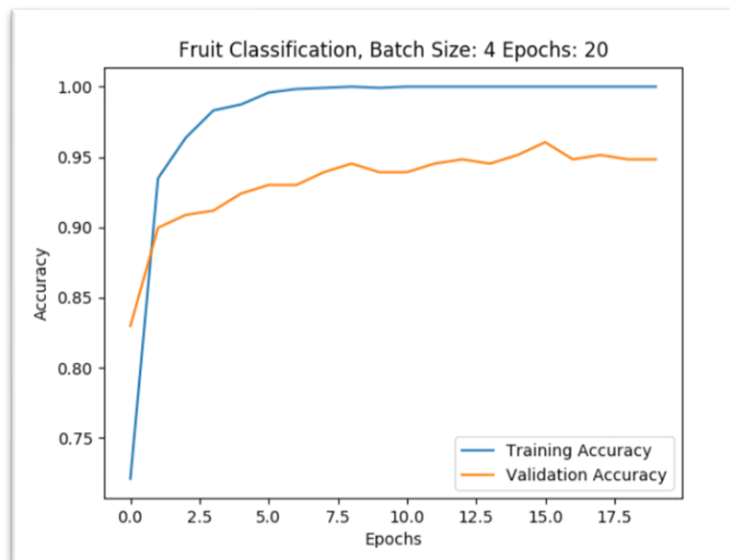
Date: 2/12/20

After running the Lab9.py as it was provided, a model was produced using a batch size of 4, without fine tuning or data augmentation, and trained for 5 epochs.



*Model 1, accuracy plot*

Following this initial run, a subsequent model was created using identical hyper parameters, except that this model was trained over 20 epochs.



## Lab 9: Containers, Grid Search

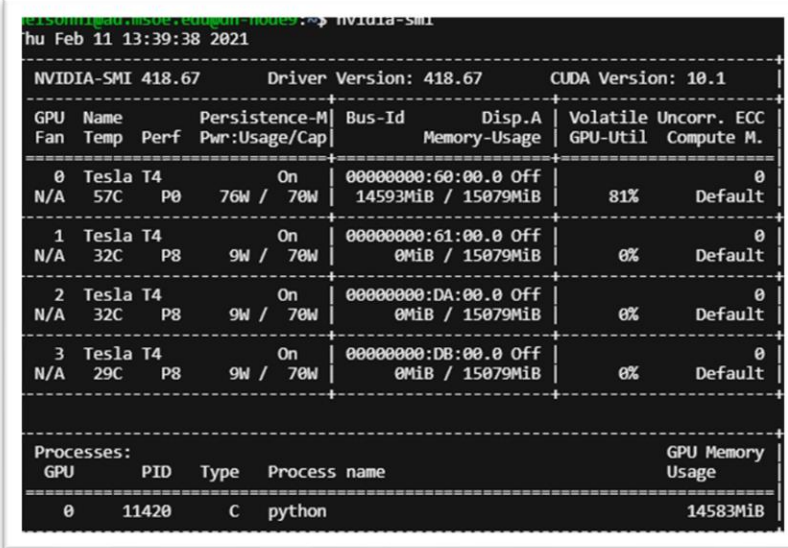
Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

### *Model 2, accuracy plot*

During the training of *Model 2*, the command “nvidia-smi” was issued on the node responsible for training this model in order to gain information on the GPUs that were being used during this process.



```
teslamigda-m208-edgen-nodes:~$ nvidia-smi
Thu Feb 11 13:39:38 2021

+-----+
| NVIDIA-SMI 418.67              Driver Version: 418.67          CUDA Version: 10.1         |
+-----+-----+
| GPU   Name           Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|     Memory-Usage | GPU-Util  Compute M. |
+-----+-----+
|  0   Tesla T4              On      | 00000000:60:00.0 Off  |      0          0     |
| N/A   57C   P0      76W /  70W    | 14593MiB / 15079MiB |    81%    Default    |
+-----+-----+
|  1   Tesla T4              On      | 00000000:61:00.0 Off  |      0          0     |
| N/A   32C   P8       9W /  70W    |    0MiB / 15079MiB |     0%    Default    |
+-----+-----+
|  2   Tesla T4              On      | 00000000:DA:00.0 Off  |      0          0     |
| N/A   32C   P8       9W /  70W    |    0MiB / 15079MiB |     0%    Default    |
+-----+-----+
|  3   Tesla T4              On      | 00000000:DB:00.0 Off  |      0          0     |
| N/A   29C   P8       9W /  70W    |    0MiB / 15079MiB |     0%    Default    |
+-----+-----+

Processes:
+-----+-----+
| GPU    PID    Type   Process name          | GPU Memory Usage |
+-----+-----+
|  0     11420   C      python                | 14583MiB          |
+-----+-----+
```

### *Model 2, GPU usage*

Next, to further practice creating models from the command line, a third model was created that had a batch size of 8, no data augmentation or fine turning, and was trained for 8 epochs. To demonstrate this execution, the command line output from this model is displayed below.

## Lab 9: Containers, Grid Search

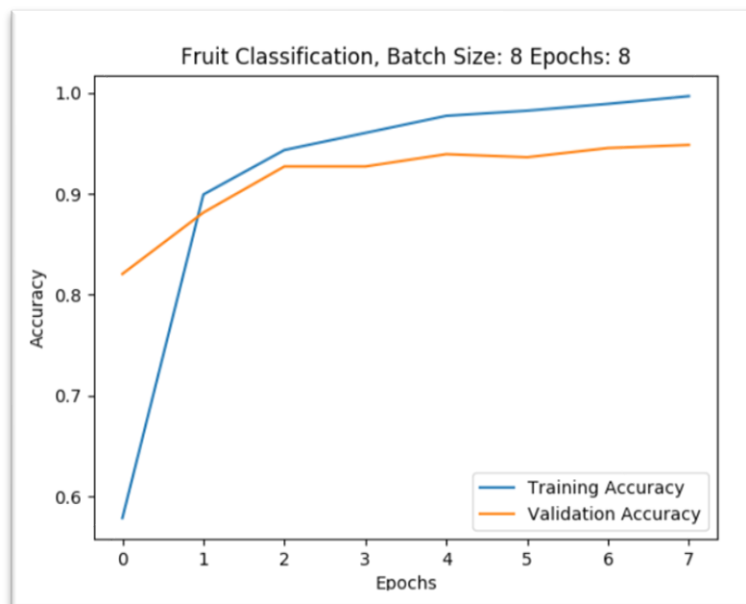
Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

```
Found 1182 images belonging to 6 classes.
Found 329 images belonging to 6 classes.
Epoch 1/8
147/147 [=====>.] - ETA: 0s - loss: 2.0684 - acc: 0.5775Epoch 1/8
148/147 [=====] - 21s 141ms/step - loss: 2.0621 - acc: 0.5787 - val_loss: 0.5904 - val_acc: 0.8207
Epoch 2/8
147/147 [=====>.] - ETA: 0s - loss: 0.3170 - acc: 0.8986Epoch 1/8
148/147 [=====] - 10s 68ms/step - loss: 0.3150 - acc: 0.8993 - val_loss: 0.3492 - val_acc: 0.8815
Epoch 3/8
147/147 [=====>.] - ETA: 0s - loss: 0.1825 - acc: 0.9446Epoch 1/8
148/147 [=====] - 10s 69ms/step - loss: 0.1847 - acc: 0.9433 - val_loss: 0.2427 - val_acc: 0.9271
Epoch 4/8
147/147 [=====>.] - ETA: 0s - loss: 0.1164 - acc: 0.9600Epoch 1/8
148/147 [=====] - 10s 69ms/step - loss: 0.1157 - acc: 0.9602 - val_loss: 0.2533 - val_acc: 0.9271
Epoch 5/8
147/147 [=====>.] - ETA: 0s - loss: 0.0745 - acc: 0.9770Epoch 1/8
148/147 [=====] - 10s 70ms/step - loss: 0.0741 - acc: 0.9772 - val_loss: 0.2210 - val_acc: 0.9392
Epoch 6/8
147/147 [=====>.] - ETA: 0s - loss: 0.0513 - acc: 0.9821Epoch 1/8
148/147 [=====] - 10s 71ms/step - loss: 0.0512 - acc: 0.9822 - val_loss: 0.2075 - val_acc: 0.9362
Epoch 7/8
147/147 [=====>.] - ETA: 0s - loss: 0.0360 - acc: 0.9889Epoch 1/8
148/147 [=====] - 10s 70ms/step - loss: 0.0359 - acc: 0.9890 - val_loss: 0.2109 - val_acc: 0.9453
Epoch 8/8
147/147 [=====>.] - ETA: 0s - loss: 0.0234 - acc: 0.9966Epoch 1/8
148/147 [=====] - 10s 70ms/step - loss: 0.0233 - acc: 0.9966 - val_loss: 0.1977 - val_acc: 0.9483
nelsonni@ad.msos.edu@dh-mgmt2:~$
```

*Model 3, command line output*



*Model 3, accuracy plot*

### Part 2:

For the second part of the lab, which involved testing a model with validation images, **Model 2** was used for this portion of the lab. The testing was conducted using the supplied

## Lab 9: Containers, Grid Search

Author: Nigel Nelson

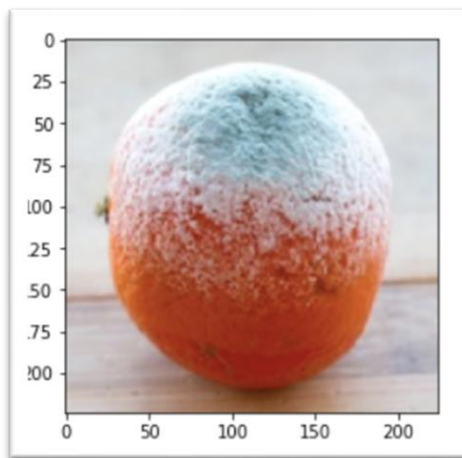
Course: CS2300

Date: 2/12/20

“Lab9Predictions.ipynb” Jupyter notebook, written by Dr. Riley. However, the notebook had to be edited in order to get meaningful testing results from each category of fruit. As such, the proper changes were made and the testing results for each respective category are shown below.

### Validation Testing: Rotten Oranges

- Accuracy: **48.9%**
- Number incorrectly identified images: **23**
- Total images tested: **45**
- Example of incorrectly identified image:



*Incorrectly identified rotten orange*

### Validation Testing: Rotten Bananas

- Accuracy: **100.0%**
- Number incorrectly identified images: **0**
- Total images tested: **68**

### Validation Testing: Rotten Apples

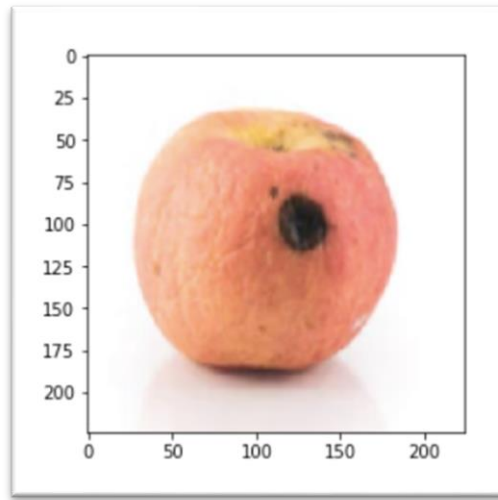
- Accuracy: **77.0%**
- Number incorrectly identified images: **17**
- Total images tested: **74**
- Example of incorrectly identified image:

## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

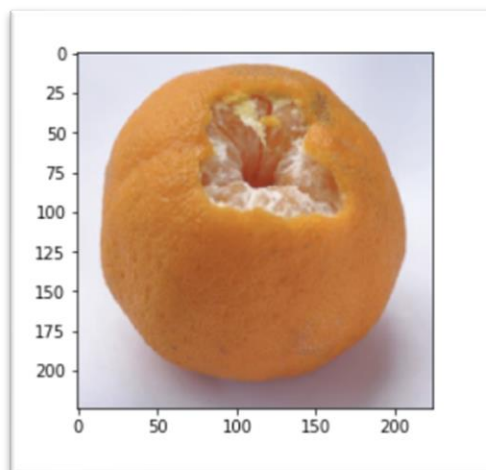
Date: 2/12/20



*Incorrectly identified rotten apple*

### Validation Testing: Fresh Oranges

- Accuracy: **97.7%**
- Number incorrectly identified images: **1**
- Total images tested: **43**
- Example of incorrectly identified image:



*Incorrectly identified fresh orange*

### Validation Testing: Fresh Bananas

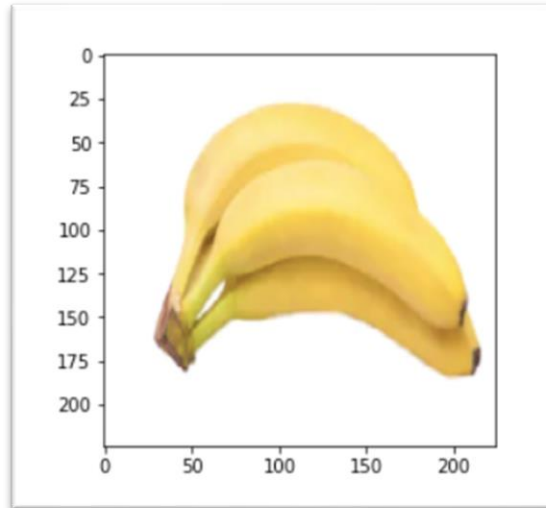
## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

- Accuracy: **79.6%**
- Number incorrectly identified images: **10**
- Total images tested: **49**
- Example of incorrectly identified image:



*Incorrectly identified fresh banana*

### Validation Testing: Fresh Apples

- Accuracy: **100.0%**
- Number incorrectly identified images: **0**
- Total images tested: **50**

---

### Part 3:

This portion of the lab focused on creating models in batch mode, within containers. This was done using the supplied “Lab9.sh” file, which acted as single callable location that took care of requesting a container, and running the Lab9.py file within that container. The advantage of this approach is that by compiling multiple commands into a single file, it is much easier to make multiple models because only a single command, “sbatch Lab9.sh”, must be issued, as opposed to multiple lengthy commands. In addition, this approach allows for greater reproducibility because by having all of your specific commands and arguments in a single place, it becomes much easier for a third party to simply run the supplied file as opposed to having to comb through lines of commands and run the model manually. However, a downside to this approach is that if a single job needs to be ran, creating an entirely separate file may be less efficient than simply typing out the commands directly into the command line. While this approach may lead to better documentation of the job, it is hard to justify the

## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

extra time needed when running a single job. In addition, while the level of abstraction can be seen as a benefit to this approach, it is also a negative. The reason for this is that if unexpected output is encountered, it may be more difficult to pinpoint where the error is stemming from.

```
3932 65/73 [=====>...] - ETA: 2:04 - loss: 5.2348e-06 - acc:
1.0000
3933 66/73 [=====>...] - ETA: 1:50 - loss: 5.2240e-06 - acc:
1.0000
3934 67/73 [=====>...] - ETA: 1:36 - loss: 5.1574e-06 - acc:
1.0000
3935 68/73 [=====>...] - ETA: 1:22 - loss: 5.0819e-06 - acc:
1.0000
3936 69/73 [=====>...] - ETA: 1:08 - loss: 5.0932e-06 - acc:
1.0000
3937 70/73 [=====>...] - ETA: 54s - loss: 5.0213e-06 - acc: 1.0000
3938 71/73 [=====>...] - ETA: 40s - loss: 4.9542e-06 - acc:
1.0000
3939 72/73 [=====>...] - ETA: 26s - loss: 4.8863e-06 - acc:
1.0000
3940 73/73 [=====>...] - ETA: 12s - loss: 4.8217e-06 - acc: 1.0000Epoch
1/20
3941
3942 1/73 [.....] - ETA: 5:15 - loss: 2.9802e-08 - acc:
1.0000
3943 2/73 [.....] - ETA: 4:50 - loss: 0.1172 - acc: 0.9688
```

*Model 4, slurm.out file for model with batch size of 16, 20 epochs*

After running an initial job in batch mode on the almost completely unedited “Lab9.sh” file, we were directed to increase the number of CPU cores used to 18. This was accomplished by editing the “Lab9.sh” file so that the argument “cpus-per-gpu” was set to 18. This modification led to a significant speed up in the time to train a model. It is likely that this speedup is a result of the GPU allowing the less parallelizable tasks to be executed by the CPUs while the GPU works on the most computationally intensive parallelizable process of training a model. Specifically, it is likely that the additional CPUs are tasked with the forward pass phase on the model for each image in a batch, and all the while the GPU is responsible for backpropagating in response to the previous batch that was passed through the model. With this line of logic, it makes sense why more CPUs would increase the execution time to train a model, as more CPUs are added, forward passes are executed faster and faster as each CPU is responsible for a smaller and smaller amount of the total batch size in the forward pass phase. Evidence of the manipulation of the “cpus-per-gpu” argument are shown below, which is the result of using the command “scontrol show jobid -d <jobid>”.



## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

```
nelsonni@ad.msoe.edu@dh-mgmt2:~/Lab9$ scontrol show jobid -d 14875
JobId=14875 JobName=Lab9_B16_E20.sh
UserId=nelsonni@ad.msoe.edu(964428880) GroupId=nelsonni@ad.msoe.edu(964428880) MCS_label=N/A
Priority=4294897118 Nice=0 Account=students QOS=interactive
JobState=RUNNING Reason=Prolog Dependency=(null)
Requeue=1 Restarts=0 BatchFlag=1 Reboot=0 ExitCode=0:0
DerivedExitCode=0:0
RunTime=00:00:06 TimeLimit=01:00:00 TimeMin=N/A
SubmitTime=2021-02-14T19:29:00 EligibleTime=2021-02-14T19:29:00
AccrueTime=2021-02-14T19:29:00
StartTime=2021-02-14T19:29:00 EndTime=2021-02-14T20:29:00 Deadline=N/A
SuspendTime=None SecsPreSuspend=0 LastSchedEval=2021-02-14T19:29:00
Partition=teaching AllocNode:Sid=dh-mgmt2.hpc.msoe.edu:64769
ReqNodeList=(null) ExcNodeList=(null)
NodeList=dh-node8
BatchHost=dh-node8
NumNodes=1 NumCPUs=18 NumTasks=1 CPUs/Task=1 ReqB:S:C:T=0:0:*:*
TRES=cpu=18,node=1,billing=18,gres/gpu=1
Socks/Node=* NtasksPerN:B:S:C=0:0:*:1 CoreSpec=*
Nodes=dh-node8 CPU_IDS=0-17 Mem=0 GRES=gpu(IDX:0)
MinCPUsNode=1 MinMemoryCPU=0 MinTmpDiskNode=0
Features=(null) DelayBoot=00:00:00
OverSubscribe=OK Contiguous=0 Licenses=(null) Network=(null)
Command=/home/ad.msoe.edu/nelsonni/Lab9/Lab9_B16_E20.sh
WorkDir=/home/ad.msoe.edu/nelsonni/Lab9
StdErr=/home/ad.msoe.edu/nelsonni/Lab9/sbatcherrorfile.out
StdIn=/dev/null
StdOut=/home/ad.msoe.edu/nelsonni/Lab9/slurm-14875.out
Power=
CpusPerTres=gpu:18
TresPerJob=gpu:1
```

*Model 5, details of the job used to create model identical to model 4*

---

### Part 4:

In part 4 of the lab, students were tasked with utilizing running jobs in a batch mode in order to achieve 98% validation accuracy without significant overfitting. The benefit of being able to run multiple jobs at the same time in batch mode was taken advantage of in this portion of the lab, and many, many different combinations of hyperparameters were tested in order to try look for the trends in the resulting accuracy plots, so that the best possible combination of hyperparameters could be found. Through trial and error, the first primary discoveries made were that the use of data augmentation lowered the instances of extreme overfitting, and the use of fine tuning increased the validation accuracy of the resulting model. The data augmentation used involved randomly tilting the images from 0 to 5 degrees, and randomly shifting images horizontally and/or vertically from 0% to 10% of their total width/height. Also, considering that our model uses the transfer learning in order to build off of the more general layers of the ImageNet model, fine tuning allows the complete model to be unfrozen so that the entire model can be trained with a very low learning rate, allowing for a more precise final model as opposed to if those general ImageNet layers were never unfrozen in training. In the many

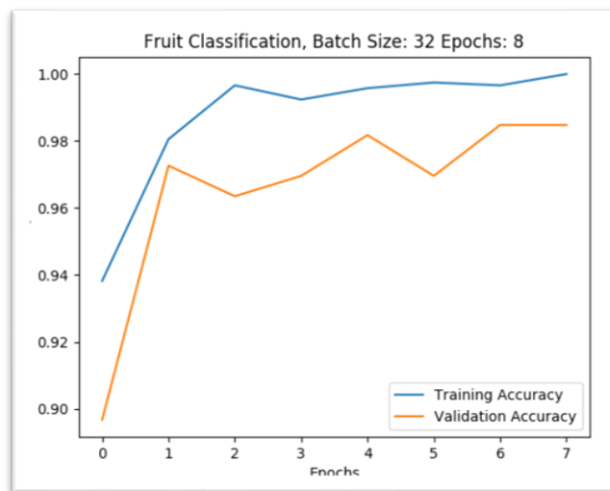
## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

iterations of creating models, it was also found that a batch size of 4 demonstrated precise learning, however, it's accuracy ceiling was near the 95% mark, no matter what number of epochs was used. Next a batch size of 16 was tried, and while these models could achieve 98% validation accuracy, they struggled with inconsistent levels of overfitting beyond the 2<sup>nd</sup> and 3<sup>rd</sup> epoch depending on the number of epochs used. Batch sizes of 64 and 128 were also attempted, however, these model also appeared to have a validation accuracy ceiling near the 98% mark and struggled with overfitting when they were trained for a larger number of epochs. As such, it was found that a **batch size of 32** seemed to demonstrate the best balance of being able to achieve a validation accuracy of 98%, while also not displaying a concerning amount of overfitting for the necessary number of epochs. To find the correct number of epochs, many values were attempted in order to find the best possible size. It was found that anything below 8 epochs struggled to reach the 98% validation mark, while any epoch beyond 14 struggled with overfitting. The accuracy plots of the models created in this interval are shown below.



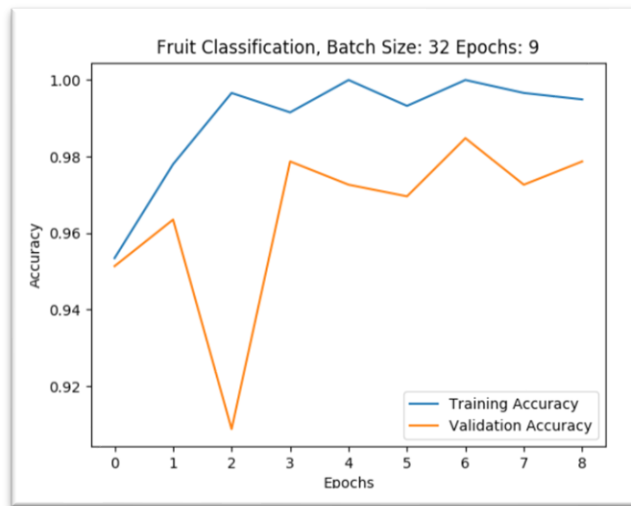
*Model 6, batch size of 32, 8 epochs*

## Lab 9: Containers, Grid Search

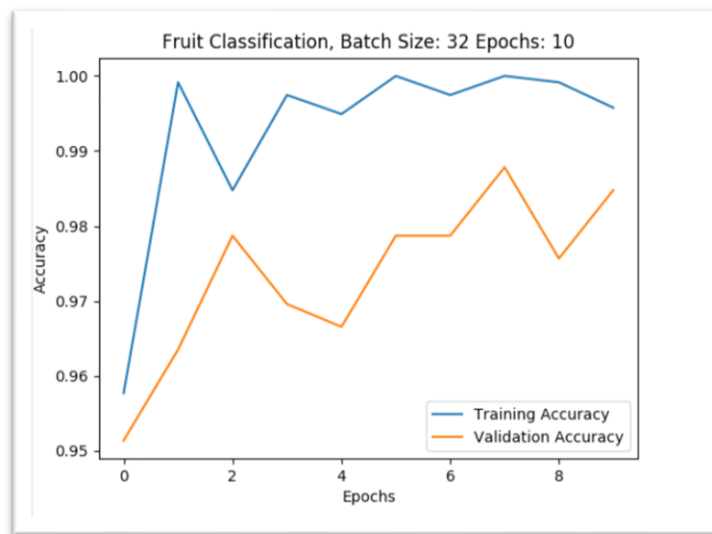
Author: Nigel Nelson

Course: CS2300

Date: 2/12/20



*Model 7, batch size of 32, 9 epochs*



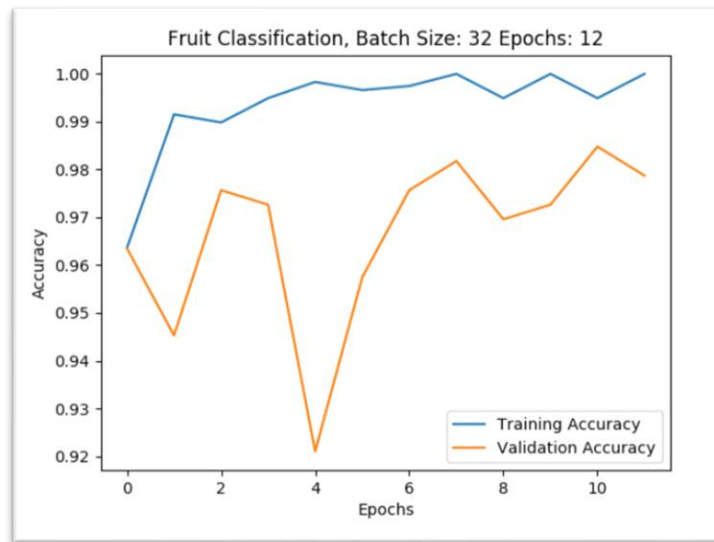
*Model 8, batch size of 32, 10 epochs*

## Lab 9: Containers, Grid Search

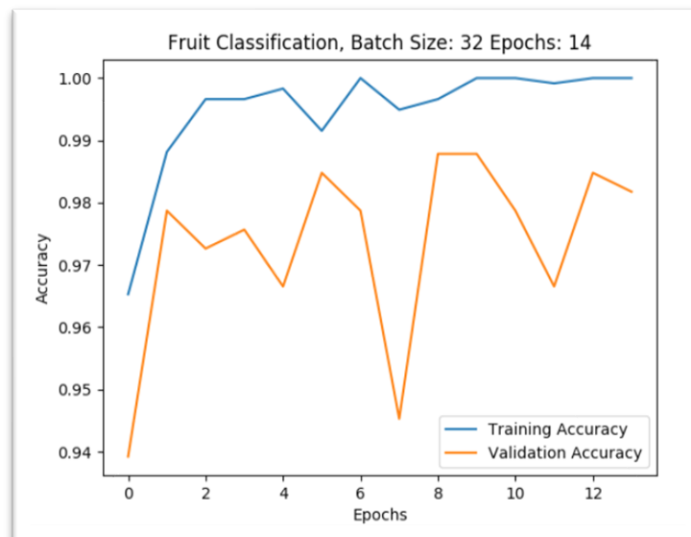
Author: Nigel Nelson

Course: CS2300

Date: 2/12/20



*Model 9, batch size of 32, 12 epochs*



*Model 10, batch size of 32, 14 epochs*

## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

With multiple runs at each interval in this range, it was ultimately concluded that training the model for **10 epochs** demonstrated the greatest balance of a lack of serious overfitting in the model's final form, and the highest validation accuracy. While concerning overfitting is experienced in earlier epochs, the final model ultimately trends towards the direction of a general solution that is not simply memorizing the training data set.

*Hyperparameters used to achieve final model:*

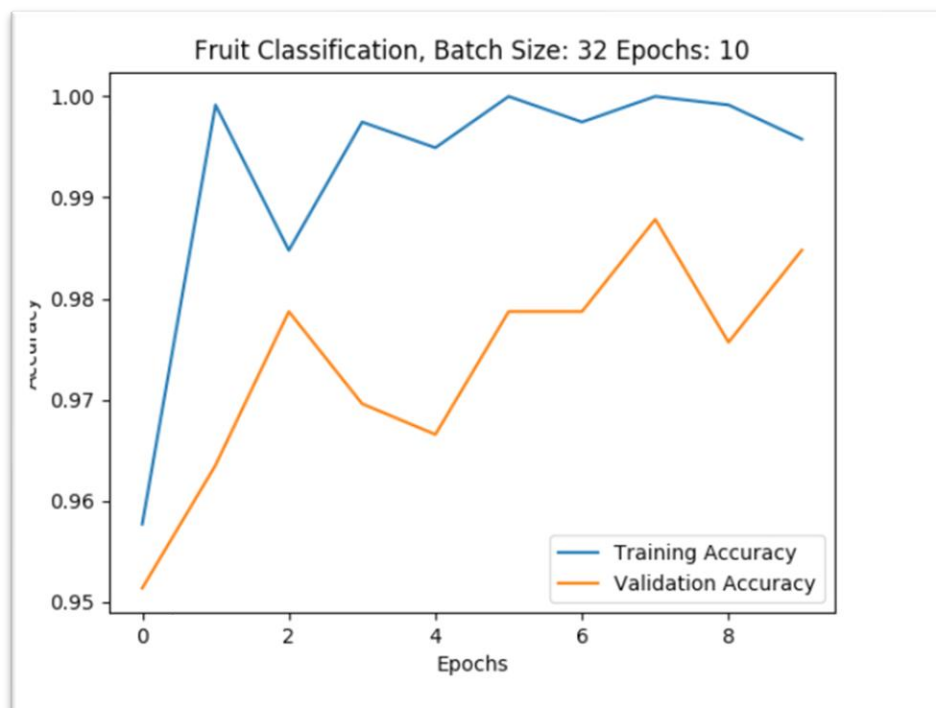
Epochs: **10**

Batch Size: **32**

*Arguments used to achieve final model:*

Data augmentation: **True**

Fine tuning: **True**



*Model 8, batch size of 32, 10 epochs*

## Lab 9: Containers, Grid Search

Author: Nigel Nelson

Course: CS2300

Date: 2/12/20

```
l040 0.9911 ..... - ETA: 13s - loss: 0.0145 - acc:
l041 8/36 [====>.....] - ETA: 13s - loss: 0.0145 - acc:
l042 0.9922 ..... - ETA: 13s - loss: 0.0145 - acc:
l043 9/36 [====>.....] - ETA: 13s - loss: 0.0145 - acc:
l044 0.9861 ..... - ETA: 12s - loss: 0.1008 - acc:
l045 10/36 [====>.....] - ETA: 12s - loss: 0.1008 - acc:
0.9844 ..... - ETA: 11s - loss: 0.0916 - acc:
11/36 [====>.....] - ETA: 11s - loss: 0.0916 - acc:
0.9848 ..... - 21s 570ms/step - loss: 0.0137 - acc: 0.9958 - val_loss: 0.0980 - val_acc: 0.9848
37/36 [=====] - 21s 570ms/step - loss: 0.0137 - acc: 0.9958 - val_loss: 0.0980 - val_acc: 0.9848
```

*Model 6, Output for final Epoch of Training*

---

### Conclusion:

This lab allowed students an entry point into running models from the command line. The model being created had the purpose of identifying fresh vs. rotten fruit. Specifically, and emphasis was placed on running models within a container via the command line. By running jobs via a singularity container, it was discovered that this allows a greater degree of control over what hardware is used. In this application, it was found that allocating a greater amount of CPUs allowed for increased performance. The hypothesis behind this speedup being that less parallelizable computations were allocated to these CPUs while the GPU focused on the parallelizable processes. Both interactive and batch modes were used to run and train models within a singularity container, and it was discovered that as the name implies, batch modes give an advantage when there is the intention to run multiple jobs, making it easier to run subsequent jobs. In addition, batch mode makes it easy to put hardware intensive jobs into a queue to be executed when the necessary resources become available. Also, in this lab's implementation batch mode allowed for greater reproducibility by third parties. Finally, it was discovered through extensive trial and error that the best set of hyperparameters to achieve a high validation accuracy with an acceptable level of overfitting on the provided data set, was training a model for 10 epochs with a batch size of 32.