

Colby Tobin
CS 6220

Programming Assignment 1: *Problem 2.1*

GitHub URL for Project

- Run this in Python 2.7
- Requirements
 - NumPy
 - OpenCV (cv2)
 - TensorFlow
 - Matplotlib
- Running each of the python files should work in Python 2.7
- Must run it from within Programming1 Folder

Input/Output/Outlier Analysis

- The data for comparing the different CNN's exists on the attached excel file
- There is a separate sheet for the input comparison, output comparison, and the outlier comparison

Screenshots of Environment

```

max_pooling2d (MaxPooling2D) (None, 13, 13, 32)      0
dropout (Dropout) (None, 13, 13, 32)      0
conv2d_1 (Conv2D) (None, 11, 11, 64)      18496
max_pooling2d_1 (MaxPooling2D) (None, 5, 5, 64)      0
dropout_1 (Dropout) (None, 5, 5, 64)      0
conv2d_2 (Conv2D) (None, 3, 3, 64)        36928
flatten (Flatten) (None, 576)            0
dense (Dense) (None, 64)                36928
dense_1 (Dense) (None, 10)              650
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0

None
Train on 48000 samples, validate on 12000 samples
Epoch 1/5
48000/48000 [=====] - 39s 814us/sample - loss: 0.1862 - accuracy: 0.9415 - val_loss: 0.858 - val_accuracy: 0.9816
Epoch 2/5
48000/48000 [=====] - 33s 678us/sample - loss: 0.0566 - accuracy: 0.9825 - val_loss: 0.8585 - val_accuracy: 0.9852
Epoch 3/5
48000/48000 [=====] - 34s 780us/sample - loss: 0.0407 - accuracy: 0.9876 - val_loss: 0.8436 - val_accuracy: 0.9875
Epoch 4/5
48000/48000 [=====] - 33s 694us/sample - loss: 0.0307 - accuracy: 0.9899 - val_loss: 0.8381 - val_accuracy: 0.9888
Epoch 5/5
48000/48000 [=====] - 48s 837us/sample - loss: 0.0254 - accuracy: 0.9918 - val_loss: 0.8366 - val_accuracy: 0.9906
12000/12000 [=====]

```

```
Programmiing — bash — 103x47
^C
(venv) verizon-ar@Programmiing colby$ python2 CpuVoxOps.py ["OM-HNST.py", "-DScore", "Catvoxops.py", "OM-HNST-fashion.py", "dog-vs-cat", "OM-Cifar10.py"]
[19-09-04 20:10:06.63398]: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
[19-09-04 20:10:06.64638]: I tensorflow/xla/service/device_compiler.cc:168] XLA service 0x1370515b0 executing computations on platform Host. Devices:
[19-09-04 20:10:06.64638]: I tensorflow/compiler/xla/service/service.cc:175] StreamExecutor device at 0; Host: Default Version
Train on 4800 samples
Epoch 1/10
=====
tensorflow/tensorflow/From /Users/colby/Library/Python/2.7/lib/python/site-packages/tensorflow/core/python/ops/numpy_impl.py:183: where (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
4800/4800 ===== - 38s Bms/sample - loss: 0.6691 - accuracy: 0.5375
Epoch 2/10
=====
4800/4800 ===== - 37s Bms/sample - loss: 0.6358 - accuracy: 0.6454
Epoch 3/10
=====
4800/4800 ===== - 37s Bms/sample - loss: 0.5747 - accuracy: 0.6994
Epoch 4/10
=====
4800/4800 ===== - 37s Bms/sample - loss: 0.5072 - accuracy: 0.7421
Epoch 5/10
=====
4800/4800 ===== - 37s Bms/sample - loss: 0.4581 - accuracy: 0.7875
Epoch 6/10
=====
4800/4800 ===== - 38s Bms/sample - loss: 0.3640 - accuracy: 0.8348
Epoch 7/10
=====
4800/4800 ===== - 38s Bms/sample - loss: 0.2866 - accuracy: 0.8886
Epoch 8/10
=====
4800/4800 ===== - 37s Bms/sample - loss: 0.2102 - accuracy: 0.9127
Epoch 9/10
=====
4800/4800 ===== - 38s Bms/sample - loss: 0.1307 - accuracy: 0.9550
Epoch 10/10
=====
4800/4800 ===== - 38s Bms/sample - loss: 0.0970 - accuracy: 0.9685
1200/1 =====
```

```
(env) verizon-ar:Programming1 -- bash -- 103x47
2019-09-04 20:02:24.831778: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX FMA
2019-09-04 20:02:24.849829: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x12e739830c executing computations on platform Host. Devices:
2019-09-04 20:02:24.849835: I tensorflow/compiler/xla/service/service.cc:175] StreamExecutor device (0): Host, Default Version
Train on 40000 samples, validate on 12000 samples
Epoch 1/5
40000/40000 [=====] - 36s 742us/sample - loss: 0.5433 - accuracy: 0.7984 - val_loss: 0.3926 - val_accuracy: 0.8592
Epoch 2/5
40000/40000 [=====] - 36s 747us/sample - loss: 0.3584 - accuracy: 0.8789 - val_loss: 0.3421 - val_accuracy: 0.8716
Epoch 3/5
40000/40000 [=====] - 34s 699us/sample - loss: 0.3847 - accuracy: 0.8876 - val_loss: 0.2976 - val_accuracy: 0.8918
Epoch 4/5
40000/40000 [=====] - 33s 695us/sample - loss: 0.2765 - accuracy: 0.8973 - val_loss: 0.2784 - val_accuracy: 0.8978
Epoch 5/5
40000/40000 [=====] - 34s 786us/sample - loss: 0.2572 - accuracy: 0.9040 - val_loss: 0.2691 - val_accuracy: 0.9015
10000/1 [=====]
```

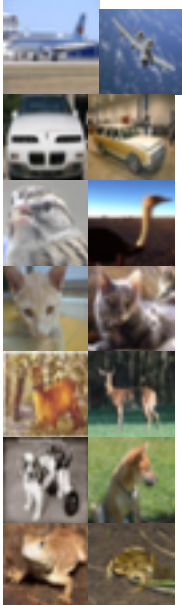
```
(venv) verizon-ar$Programming1 colabts python2 CNN-Cifar10.py
2019-09-04 20:28:07.273419 I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
2019-09-04 20:28:07.296633 I tensorflow/compiler/xla/service/service.cc:168] XLA service is_xla_backend_enabled:
Executing computations on platform Host. Devices:
0: Host, Default Version
StreamExecutor device 0: Host, Default Version
Train on 40000 samples, validate on 10000 samples
Epoch 1/10
40000/40000 [=====] - 31s 784us/sample - loss: 1.7020 - accuracy: 0.3826 - val_loss: 1.3757 - val_accuracy: 0.5145
Epoch 2/10
40000/40000 [=====] - 31s 770us/sample - loss: 1.2953 - accuracy: 0.5491 - val_loss: 1.1991 - val_accuracy: 0.5861
Epoch 3/10
40000/40000 [=====] - 31s 774us/sample - loss: 1.1393 - accuracy: 0.6036 - val_loss: 1.0949 - val_accuracy: 0.6194
Epoch 4/10
40000/40000 [=====] - 32s 803us/sample - loss: 1.0381 - accuracy: 0.6389 - val_loss: 1.0604 - val_accuracy: 0.6324
Epoch 5/10
40000/40000 [=====] - 32s 797us/sample - loss: 0.9510 - accuracy: 0.6693 - val_loss: 0.9897 - val_accuracy: 0.6583
Epoch 6/10
40000/40000 [=====] - 32s 798us/sample - loss: 0.9080 - accuracy: 0.6877 - val_loss: 0.9859 - val_accuracy: 0.6573
Epoch 7/10
40000/40000 [=====] - 31s 785us/sample - loss: 0.8412 - accuracy: 0.7084 - val_loss: 0.9376 - val_accuracy: 0.6819
Epoch 8/10
40000/40000 [=====] - 31s 784us/sample - loss: 0.7961 - accuracy: 0.7246 - val_loss: 0.9212 - val_accuracy: 0.6938
Epoch 9/10
40000/40000 [=====] - 32s 788us/sample - loss: 0.7541 - accuracy: 0.7357 - val_loss: 0.9311 - val_accuracy: 0.6822
Epoch 10/10
40000/40000 [=====] - 32s 793us/sample - loss: 0.7213 - accuracy: 0.7496 - val_loss: 0.9311 - val_accuracy: 0.6822
10000/1 [=====]
```

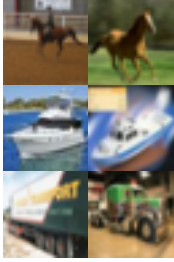
Sample Images

- Cats vs. Dogs



- Cifar-10





- MINST



- MINST-Fashion



Outlier Image Samples (used Wallets as the outlier images)



Observations of Experiments

- The cats vs. dogs dataset results in a classifier with lower classification accuracy than some of the other classifiers due to the data itself. The images themselves are fairly blurry, the images all have different resolutions, and the objects in the image are not just of the animals, but contain other objects with different orientations. Because of this, the images had to be resized to fit the classifier, which resulted in the changing of the image, reduced resolutions, and altered the data itself. This would result in worse accuracy for the classifier because of the training data itself.
- Additionally, a classifier is only as good as the data provided, and it can only learn the classifications we provide it. This is why it could not classify any of the outlier images because the classifier never learned about any of those labels and never saw the image before. Because of this reason, this is why all the classifiers do not consistently classify the outlier images as the same label because it doesn't know what to make of the random wallets provided. The only odd one is the cats vs. dogs classifier, which consistently classified the wallets as cats for an unknown reason.
- This may or may not be related, but the color image datasets result in classifiers that have a lower accuracy. This could be due to the fact that the color images result in blurrier edges when converted into grayscale, which makes the feature detection harder to perform. This poor resolution and difficulty to perform feature detection because of the colored images is one reason as to why the Cifar and Cats vs. Dogs classifier have low performance.