Finding Generalizations in Multi-class Image Classifiers

Claudia Richoux - CMSC35200 Final Project

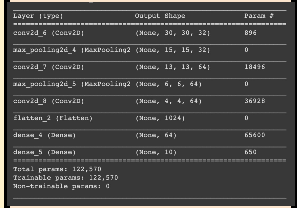
Introduction

Seeing how neural nets do things, and what they learn to look for while doing that, is both important for interpretability, and also really cool. People who study thought often study “different” brains in order to better understand the general case- for example, hippocampal lesions for studying memory, or Freud’s case studies, or Descartes and his dogs. I thought applying this to neural nets might yield something interesting, so I mistrained a bunch of CNNs and then did a bunch of statistics on their performance.

Methodology

I made a small CNN to train on the CIFAR-10 image dataset. Then, for each choice of two CIFAR-10 classes, I repeatedly randomly selected half the members of each class to mislabel as the second, and trained a net on them. This is so that the net would learn nothing about either of the confused classes (or at least, the information it DID pick up about them would cancel out in further computations!), and no other classes would be affected beyond the direct effect of not learning these two classes. I repeated this process 50 times per selection of classes to confuse, to ensure I could draw accurate assumptions about the loss/accuracy of models trained in this way- fortunately, these statistics tended to have small \sigma for each experiment, so averaging was more or less valid :)

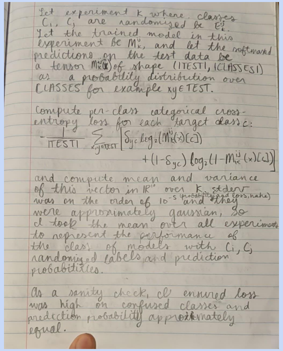
truck, car, frog



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_6 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 4, 4, 64)          36928
_________________________________________________________________
flatten_2 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                65600
_________________________________________________________________
dense_5 (Dense)              (None, 10)                650
=================================================================
Total params: 122,570
Trainable params: 122,570
Non-trainable params: 0
_________________________________________________________________
```

model architecture

Data Analysis (couldn’t typeset this, so I took pics)



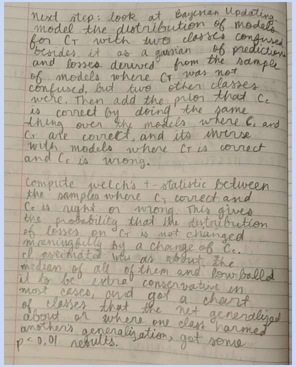
Let experiment k where classes C_i, C_j are randomized be E_k^{ij}. Let the trained model in this experiment be M_k^{ij}, and let the softmaxed predictions on the test data be a tensor $M_k^{ij}(x)$ of shape (|TEST|, |CLASSES|) as a probability distribution over CLASSES for example $xy \in$ TEST.

Compute per-class categorical cross-entropy loss for each target class c:

$$-\frac{1}{|TEST|}\sum_{x,y\in TEST}\Big[\delta_{yc}\log_2(M_k^{ij}(x)[c]) + (1-\delta_{yc})\log_2(1-M_k^{ij}(x)[c])\Big]$$

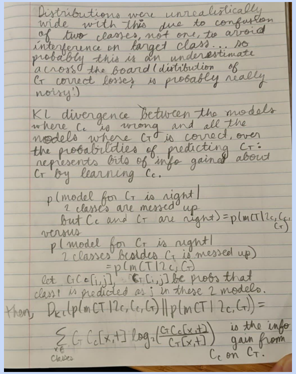
and compute mean and variance of this vector in $\mathbb{R}^{10}$ over k. stdev was on the order of 10^{-5} (probability and loss, haha) and they were approximately gaussian, so I took the mean over all experiments to represent the performance of the class of models with C_i, C_j randomized labels and prediction probabitilies.

As a sanity check, I ensured loss was high on confused classes and prediction probability approximately equal.



Next step: look at Bayesian Updating. model the distribution of models for C_T with two classes confused besides it as a gassian of predictions and losses derived from the sample of models where C_T was not confused, but two other classes were. Then add the prior that C_c is correct by doing the same thing over the models where C_c and C_T are correct, and its inverse with models where C_T is correct and C_c is wrong.

Compute welch's t-statistic between the samples where C_T correct and C_c is right or wrong. This gives the probability that the distribution of losses on C_T is not changed meaningfully by a change of C_c. I estimated ν as about the median of all of them and low balled it to be extra conservative in most cases, and got a chart of classes that the net generalized about or where one class harmed another's generalization, got some $p < 0.01$ results.



Distributions were unrealistically wide with this due to confusion of two classes, not one, to avoid interference on target class... so probably this is an underestimate across the board (distribution of C_T correct losses is probably really noisy!)

KL divergence between the models where C_c is wrong and all the models where C_T is correct, over the probabilities of predicting C_T: represents bits of info gained about C_T by learning C_c.

p(model for C_T is right | 2 classes are messed up but C_c and C_T are right) $= p(mCT \mid 2c, C_c, C_T)$

versus

p(model for C_T is right | 2 classes besides C_T is messed up) $= p(mCT \mid 2c, C_T)$

let $C_TC_c[i,j]$, $C_T[i,j]$ be probs that class i is predicted as j in these 2 models.

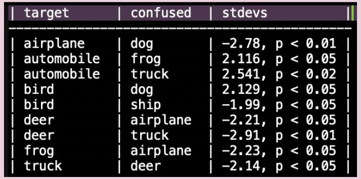
Then, $D_{KL}(p(mCT \mid 2c, C_c, C_T) \| p(mCT \mid 2c, C_T)) =$

$$\sum_{x\in \text{Classes}} C_TC_c[x,t]\log_2\left(\frac{C_TC_c[x,t]}{C_T[x,t]}\right)$$ is the info gain from C_c on C_T.

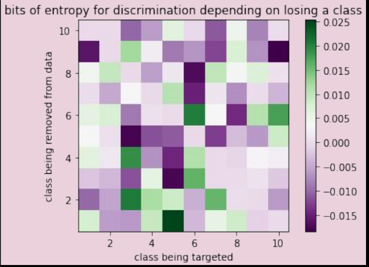
Results :)

Welch’s t-test shows that it generalized about cars and trucks (and also cars and frogs- look at headlights vs. frog eyes :P). Bird-dog significance was a statistical error with estimation of \nu. Learning about both animals and man-made things tended to confuse it- they look pretty different!

KL-entropy shows that classes whose accuracies correlate can gain up to 0.025 bits of information on classifying T by not confusing C, or lose almost the same amount by classes that clash. On the entropy plot, classes 2-7 are the animals- you see patches of green and purple because the net tends to correlate animals and vehicles separately :)



```
| target       | confused  | stdevs           |
==============================================
| airplane     | dog       | -2.78, p < 0.01  |
| automobile   | frog      | 2.116, p < 0.05  |
| automobile   | truck     | 2.541, p < 0.02  |
| bird         | dog       | 2.129, p < 0.05  |
| bird         | ship      | -1.99, p < 0.05  |
| deer         | airplane  | -2.21, p < 0.05  |
| deer         | truck     | -2.91, p < 0.01  |
| frog         | airplane  | -2.23, p < 0.05  |
| truck        | deer      | -2.14, p < 0.05  |
```



Future Ideas?

- try this with other types of models (text generation? sentiment analysis?)
- deeper models, more classes
- identify weird combos, identify important weights for those generalizations, target Lucid @ those…
- lots of other ways to mistrain...