Journal Report 20

3/2/20-3/6/20
Kevin Fu
Computer Systems Research Lab
Period 1, White

Daily Log

Monday March 2

Added another hundred labelled images to dataset. Adjusted data augmentations for new unsheared input data. Tried training on new dataset, noticed snowy was brutally slow. Emailed Dr. White.

Tuesday March 3

With snowy restarted, tried training again. Noticed the training accuracy was incredibly low while the validation accuracy was high (46%/84%). Also, running model outside of training loop said that every square was empty.

Wednesday, March 4

Turned off shuffle in ImageDataGenerator, found opposite accuracy for train/valid (99%/74%). Upgraded tf to 2.1. Confusion matrix generated from DataGenerator looked largely random. Turned shuffle back on. Found more reasonable train/valid accuracy agreement when not training off of imagenet weights.

Thursday March 5

Switched to manually generating confusion matrix. Found that ResNet_v1's preprocessing was different from ResNet_v2's, which is why running model outside of training loop returned all empty classifications. Decided to freeze first 2/3rds of ResNet_152 and revert to transfer learning off imagenet. Reached 99%/98% splits on dataset (with caveats, of course).

Timeline

Date	Goal	Met
Feb 17	Finish slanted square segmentation, return to piece labelling	Done
Feb 24	Finalize orthophoto tweaks, begin piece labelling (March 6th)	Done
Mar 2	Finish piece labelling, train new model for States (March 6th)	Not done, done
Mar 9	Gather more data, play with ResNet types	Not started
Mar 16	Add more to chess logic, try loading onto Google Coral	Not started

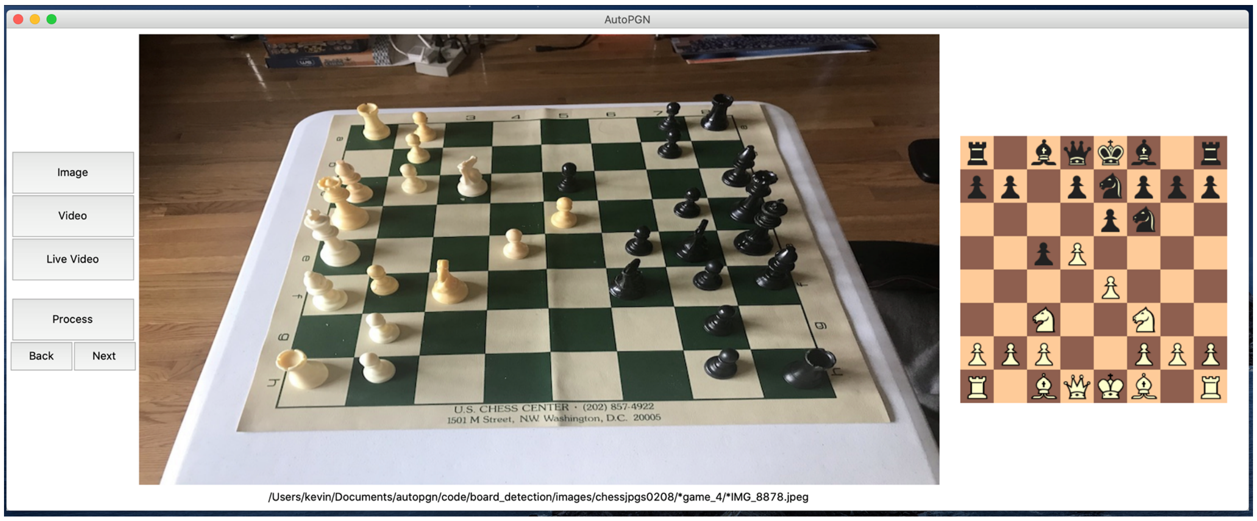
Reflection

This week was going poorly until I read the first couple pages of a Machine Learning Club lecture on transfer learning. It said that for image classification, it's sometimes unnecessary to retrain all the layers of a deep CNN, because the early layers already understand how to detect low-level features like edges. Instead, training only the later layers, which corresponds with higher-level feature detection, will often yield better results. This also partially explained why Kevin Chung's suggestion to train from scratch yielded better results than training off of pretrained weights.

Until Thursday, I stumbled through one confusing roadblock after another: not having access to snowy's GPUs, high validation accuracy with low training accuracy, the reverse of that, a mismatch between training accuracy and viewing the output of my CNN manually, and better results when training from scratch vs. pretrained weights. But by finding and fixing bugs, and freezing the first 101 layers of a 152-layer ResNet, then using pretrained weights, I was able to get this confusion matrix after 40 epochs:

```
Confusion Matrix
[[ 65   0   0   1   1   0   1   0   0   0   0   0   0]
 [  0  57   0   0   4   0   0   0   0   0   0   0   0]
 [  0   0  47   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0 320   0   0   0   0   0   0   0   0   0]
 [  0   1   0   0  26   0   0   0   0   0   0   0   0]
 [  0   1   1   0   0  70   0   0   0   0   0   0   0]
 [  0   0   1   1   0   0 315   0   1   0   2   0   0]
 [  0   0   0   0   0   0   0  56   0   0   1   0   1]
 [  0   0   0   0   0   0   0   1  61   0   0   0   1]
 [  0   0   0   0   0   0   0   0   0  51   0   0   0]
 [  0   0   0   0   0   0   0   0   0   1 278   0   0]
 [  0   0   0   0   0   0   1   0   0   0   0  33   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0  64]]
```

That accuracy translates in practice, too. On Kevin Chung's new UI, feeding an image similar to those in our dataset yields this perfectly-classified board:


AutoPGN
Image
Video
Live Video
Process
Back
Next
U.S. CHESS CENTER • (202) 857-4922
1501 M Street, NW Washington, D.C. 20005
/Users/kevin/Documents/autopgn/code/board_detection/images/chessjpgs0208/*game_4/*IMG_8878.jpeg

This high accuracy means my work with normalizing each segmented image paid off, and I'm willing to wager changing lighting conditions or minor alterations to the board and piece colors won't affect the accuracy, because I added significant channel-shift as a data augmentation. However, changing the camera angle likely will, as though I've normalized the side-to-side tilt of each piece before it goes into the CNN, I have no way of accounting for the steepness of the camera angle.

I'm going to try smaller ResNet models, as perhaps a 152-layer deep model is overkill for the normalized pieces, and add more data to bolster the model's invariance to camera angle. Kevin Chung tells me that on his GPU-enabled machine, though, piece detection on the full board runs in half a second, and board detection is the slowdown, so in the worst-case a 152-layer ResNet should still be fast enough for live video.

After that, at some point I'll need to help Kevin Chung with making our system run on live video. Originally, the hope was that we would have some sort of color-change detection for hand occlusion, and run the process after every detected hand occlusion, but currently it seems more likely we'll have to settle for running the process as often as possible, and restricting the speed of the chess game itself.