Deliverable 2 – Preliminary Results

Sheheryar Parvaz

October 19th 2020

1 Problem Statement

I'm writing a classifier for the American Sign Language alphabet. To do so, I'm using a *Convolutional Neural Network* (CNN) developed with PyTorch.

2 Data Preprocessing

The original dataset[1] I proposed has many samples, but they're all quite similar and so may lead to overfitting. So I merged a few more datasets[2][3]. The merging was the first part of preprocessing. I'll continue diversifying the dataset.

To train the model, the image must be converted to a tensor. Before doing so, PyTorch can apply given transformations to the image. For the initial implementation, I've applied no transformations other than rescaling the image to some arbitrary dimension and normalising the tensor values. However, to improve recognition I'll apply many more transformations such as cropping, rescaling, slightly rotating, adding noise, and so on. In doing so, the classifier will, in theory, generalise better.

3 Machine Learning Model

For preliminary testing, I used the CNN architecture that is suggested by the PyTorch tutorial for image classification[4]. For MAIS Hack 2020, I also worked on image classification for animals[5], so the current model is heavily based off the classification code I wrote for the hackathon. Currently, I'm not familiar enough with neural networks to tweak the model's hyperparameters. The next step is to familiarize myself with the internals of neural networks so I can tweak the model to generalise it better.

Currently, the data is split into into 80% for training and 20% for testing. Once the model is trained, the accuracy for the entire testing set and for each class are computed.

4 Preliminary Results

Since the data has little diversity, the accuracy over all classes is excessively high for a preliminary test, at 97.1%. As mentioned, adding more data and applying aggressive randomised transformations to it will help the model generalise better.

I've also tried the model with a few images of my own hand, with mixed results. Despite the data's lack of diversity, the model nonetheless recognised a few signed letters. I tested letters from A to G, and it successfully recognised the letters B, D, and E. It should be noted that my skin tone is similar to the one in the dataset, so it's worth testing with other people to get a better idea of the real accuracy. Additionally, some types of data augmentation should help with generalising to different skin tones and lighting conditions.

```
Accuracy: 97.1% on 18689
 a:  97.60% on  709
 b:  97.37% on  723
 c:  98.65% on  742
 d:  97.87% on  752
 e:  96.02% on  728
 f:  98.68% on  682
 g:  97.22% on  683
 h:  99.03% on  720
 i:  98.79% on  743
 j:  99.32% on  584
 k:  95.18% on  747
 l:  99.48% on  767
 m:  96.97% on  725
 n:  99.06% on  744
 o:  96.32% on  706
 p: 100.00% on  778
 q:  99.15% on  710
 r:  96.53% on  720
 s:  95.75% on  777
 t:  97.37% on  723
 u:  92.48% on  665
 v:  89.96% on  737
 w:  95.52% on  804
 x:  95.62% on  730
 y:  96.90% on  677
 z:  97.39% on  613
```

5 Next Steps

The data needs to be diversified with more datasets and more transformations such as scaling, cropping, rotating, and adding noise. These can be applied randomly using PyTorch transformations.

The model can be tweaked to use different CNN architectures and different hyper-parameters. I'll have to learn what the different architectures are and when they're useful.

Finally, for the project as a whole, since I want to show the sign in real time it might be useful to use YOLO[6] or some other detection algorithm to find the hand in the video stream and apply the classifier to a region, rather than to the entire stream. Otherwise, a sliding window might be sufficient, though the feasiblity needs to be tested.

Additionally, I also need to investigate the possibility of running model in the browser in JavaScript, and if so whether it can be hosted on a website somewhere for easy access.

References

[1] Akash. *ASL Alphabet*. Retrieved 2020–10–05.
`https://www.kaggle.com/grassknoted/asl-alphabet`

[2] Rasband, Dan. *ASL Alphabet Test*. Retrieved 2020–10–17.
`https://www.kaggle.com/danrasband/asl-alphabet-test`

[3] Geislinger, Victor. *ASL Fingerspelling Images (RGB & Depth)*. Retrieved 2020–10–18.
`https://www.kaggle.com/mrgeislinger/asl-rgb-depth-fingerspelling-spelling-it-out`

[4] PyTorch Tutorials. *Training a Classifier*. Retrieved 2020–10–17.
`https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html`

[5] Jiralerspong, Thomas and Parvaz, Sheheryar. *MAIS Hack 2020*.
`https://github.com/CherryMan/maishack2020`

[6] Redmon, Joseph and Farhadi, Ali. *YOLOv3: An Incremental Improvement*. arXiv:1804.02767, 2018.
`https://arxiv.org/pdf/1804.02767.pdf`