YOLO for American Sign Language

MSDS 462 section 55

Spring 2020

Steve Depp

13 June 2020

Introduction: A YOLO model provides speedy detection and classification of objects and thusly suits translating American sign language (ASL) into text. Possible use cases include subtitling hand signed conversations, confirming ASL translations of speech, and data collection from hearing-impaired dialogue. Nvidia Jetson Nano is chosen for inference due to cost, size, and power consumption which foster ubiquitous deployment and extensions of this model to battery operated devices.

Data set: The ASL set comprises 3000 still images per character extracted from video sequences of a thin white arm in short sleeves waving hand signals near to and far from the lens and mostly against sunlit background with periodic shadows. Missing are clothing, jewelry, different skin tones. Hand signals may or may not conform to American Sign Language; at the very least, hand signals differ from videos employed in the model test. For this paper, 200 and 20 images are selected with some degree of randomness for each of the 26 alphabet characters and the symbol for space to fill training and validation sets. A video collected from the internet and the author's own hand signals test the model.

Methods: 5400 training + 540 validation images are annotated with labels and bounding boxes using labelimg. A. Bochkovskiy's (2020) 4th version (YOLOv4) of J.P. Redmond's (2016) YOLO darknet framework is reconfigured and then trained in a Colab notebook with GPU runtime. After 20,000 iterations, training is stopped, and the model and its weights deployed so the Nano may infer classes of hands in videos. One test video is of a white arm against a white background hand signing 26 letters. Another is the author's hand signing letters A, B, C.

Results: After 4,000 iterations, mean average precision (mAP) climbed to 96% and average loss declined to 0.7 which seemed encouraging, but critical to actual tests were intersection over union (IOU) measures which indicated the model missing the hand in the image, (IOU ~40-60%). After 20,000 iterations, average loss continued to grind lower 0.4 (0.35 lowest) while mAP declined to 95% and IOU rose to 90%. Test results in both videos showed regular success only with the letters A, B, C, K, O, U, but this letter count

improved with more iterations. At varying iteration counts, the model exhibited a favorite letter when making mistakes: every poorly identified image was a U for example. After only a few minutes of inference, a fading monitor indicated a power-starved Nano (messages indicated low memory). Moreover, the Jetson output only 2-5 frames per second which does not suit the needs of this study.

Conclusion and recommendations: These results from an untuned darknet framework are impressive for a low variance dataset trained only 40% of the recommended iterations. The darknet framework is simple to tune with plenty of advice from the principal authors, Redmond and Bochkovskiy, and related forum contributors. Noisier data might come from another dataset or augmentation. Interactions between individuals should be part of training and tests as well. The Jetson Nano is however not suited to any of these activities due to low power and processing speed. Possibly, one of the Nano's more expensive siblings would suit better. I would fully recommend further work tuning this model and deploying it to a more powerful computer vision edge device. More afield applications of this model might be found in battery operated environments, such as model cars or drones. The author plans to deploy this on a drone to assist wildlife tracking and classification in Montana or Great Lakes region eventually.

Resources:

Colab notebook:

https://colab.research.google.com/drive/1O5hRmzLjUbuh-kksxZGPefBKohgdaBFv?usp=sharing

darknet.zip: drop this into Google drive and amend the path calling this file in Colab:

https://drive.google.com/file/d/13k7uWAEFmvjKV-0nXGuc4gFv-knwgInv/view?_usp=sharing

```
[ ]  1 from google.colab import drive
     2 drive.mount('/content/drive')
     3 !unzip "/content/drive/My Drive/462/nano/darknet.zip"
```

References:

Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

Lin, J., Gan, C., & Han, S. (2019). TSM: Temporal Shift Module for Efficient Video Understanding. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 7082-7092.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

Visée, R. J., Likitlersuang, J., & Zariffa, J. (2020). An effective and efficient method for detecting hands in egocentric videos for rehabilitation applications. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 28(3), 748-755.

Useful links for future enhancements:

A better data set

An Effective and Efficient Method for Detecting Hands in Egocentric Videos for Rehabilitation Applications https://github.com/victordibia/handtracking

Video recognition of arm gestures apply more widely to hearing impaired signals

Temporal Shift Module for Efficient Video Understanding https://github.com/mit-han-lab/temporal-shift-module/tree/master/online_demo

A better tuned model

YOLOv4: Optimal Speed and Accuracy of Object Detection

https://arxiv.org/pdf/2004.10934.pdf

A different application; I am a nut for wolf research!

Extreme inbreeding likely spells doom for Isle Royale wolves

https://www.sciencemag.org/news/2016/04/extreme-inbreeding-likely-spells-doom-isle-royale-wolves