Auto ML :

Sign Language recognition

Siddharth Sinha

JULY / AUGUST 2020

What is the problem that you are trying to solve?

In the current version of the world, most of the people don’t know sign language as we talk about specially abled or even normal people. The ratio of general people knowing sign language is miniscule. While sign language is easy to learn most people don’t want to put in the time to learn it or don't have enough reasons to do so and in my belief this isn’t even wrong in any way. Machine learning can help in this situation using image recognition and help people communicate even if they don’t know how to “wiggle their hands” in perfect manner :P

Why did you choose this problem?

It came at a glimpse because of our neighbor’s maiden servant who cannot speak but does know how to speak sign language, where me being a dumb peasant :| doesn’t know/ never felt the need to know sign language. Some interactions between us when my parents aren’t at home and kept the key at our neighbors when I went back home from school (lol when that used to be a thing before corona) were very rough, if I had an app which could recognise them would be amazing like what google translate does.

How does your machine learning application help solve this problem?

Machine learning is kind of perfect in these scenarios where image/video recognition is imminent. **While Auto ML isn’t the perfect tool for entire implementation** it could get the job done in a few really **important scenarios**.

The best custom implementation that I can think of would be either cloud processing of video or local GPU/NPU(in some mobile SOC’s case) accelerated models. The reason still images won't work for every case in this scenario is because of motion based sign language. While all letters, number in ASL

(American Sign Language) and some words like “baby” are possible to be identified just using still image contexts some do require previous frame reference in a video in order to be understood.
A model would need to be a hybrid of RNN and CNN with some extra layers, which would be required to do this properly and catch most edge cases. **In auto ML we focus on the CNN part where we can recognise most of the cases of ASL and help users understand most of the basic language reference.** In this way it would be able to create a client application which could process data on the server using REST api’s for communication or even gRPC.

What data did you use while building this project?

Good data was hard to find in the community about sign language, as we know data is the most crucial part of model preparation. Even bigger issue was which language to choose inside of sign language’s broader scope. There are about 500 different localized sign languages, this problem basically equals to what google translate does. BUT I AM DEFINITELY NOT A PERSON EQUIVALENT TO 1000’s OF PEOPLE who worked for years to make something like it. So I went for one of the most used, ASL (American Sign language) and collected my own dataset with the help of my friends and also took the help of kaggle datasets.

How did you collect this data?

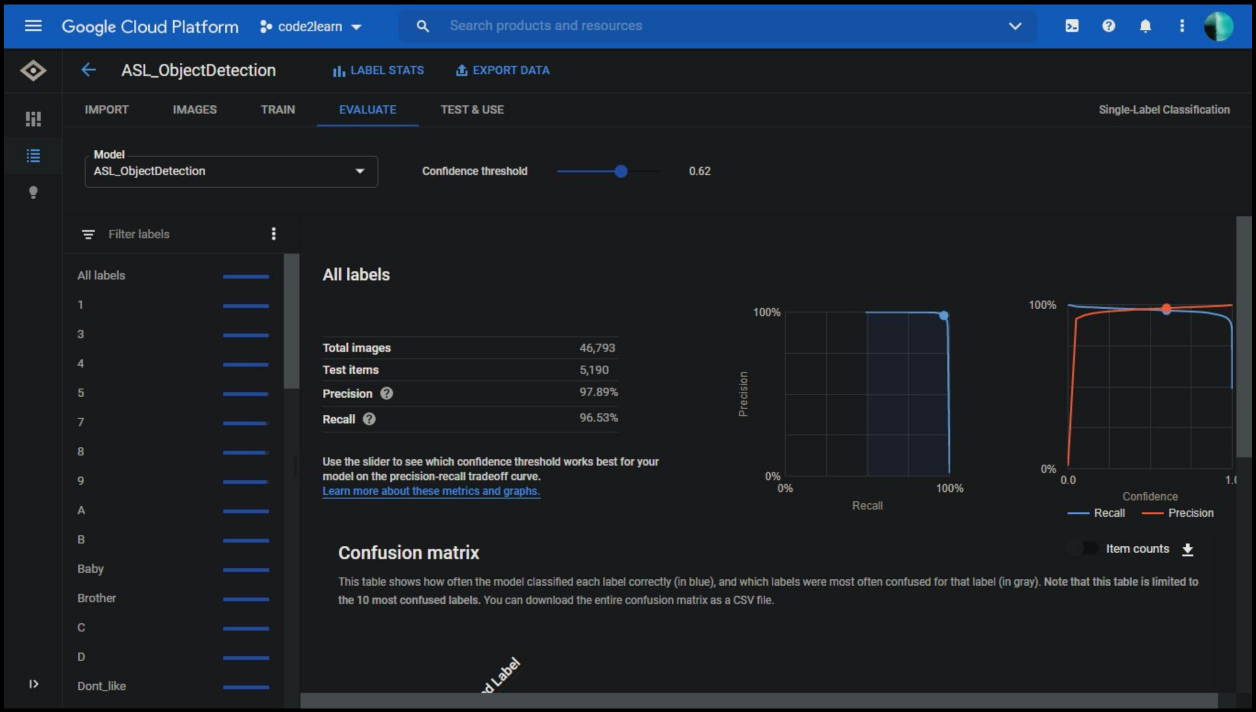
While some of them has been contributed by me and with help of my friends of course 51,983 data points aren’t all from me. Most data points were collected from kaggle asl-and-some_words, significant ASL dataset although there was a huge problem with these datasets, there were multiple repeating images *(almost, it was like a pixel different for 20 next images for example)* all over the data and that would create a huge chance of overfitting. To solve this issue I wrote a simple Powershell script, this kinda solved the repeating issue but also reduces the dataset size from 800,000 images and 8GB data to current number of 50,000ish images and 1 GB data compressed which helped uploading using my pesky slow upload speed internet :(

```
Get-ChildItem -Path F:\Data\1\dataset\* -Exclude *1* -Recurse | Remove-Item -Verbose
Get-ChildItem -Path F:\Data\1\dataset\* -Exclude *5* -Recurse | Remove-Item -Verbose
Get-ChildItem -Path F:\Data\3\ASL\* -Exclude *1* -Recurse | Remove-Item -Verbose
Get-ChildItem -Path F:\Data\3\ASL\* -Exclude *5* -Recurse | Remove-Item -Verbose
```

How many data points did you use to train and test your model respectively?

The total images used was about 51,983 while 46,793 were used to train, rest 5,190 were used to test.

Attach a clear screenshot with visible precision and recall scores from the Evaluate tab



Model: ASL_ObjectDetection

Confidence threshold 0.62

All labels

Total images	46,793
Test items	5,190
Precision	97.89%
Recall	96.53%

Use the slider to see which confidence threshold works best for your model on the precision-recall tradeoff curve.
Learn more about these metrics and graphs.

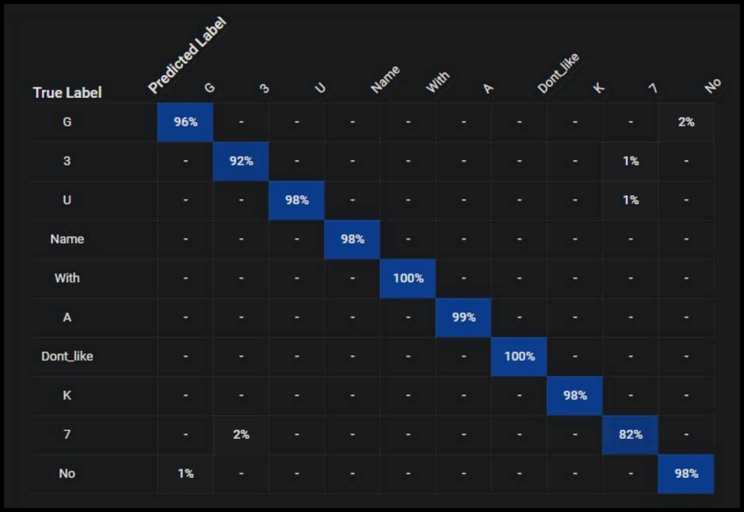
Confusion matrix

This table shows how often the model classified each label correctly (in blue), and which labels were most often confused for that label (in gray). Note that this table is limited to the 10 most confused labels. You can download the entire confusion matrix as a CSV file.

Fallbacks

Of course as mentioned many times before in this document that, this implementation is nowhere near perfect for real world use, there are areas like motion based words and many different types of sign languages which refer to 1 motion with different meanings.

Also 1 fluke 😅 in dataset processing as we can see in the confusion matrix, the 3 and 9 were somehow mixed in the 2 different datasets I used, which caused huge problems in detecting those specific numbers. But [redacted] didn't run the training again because I forgot to ask for credits before june 25, and it was running on my own money unfortunately.

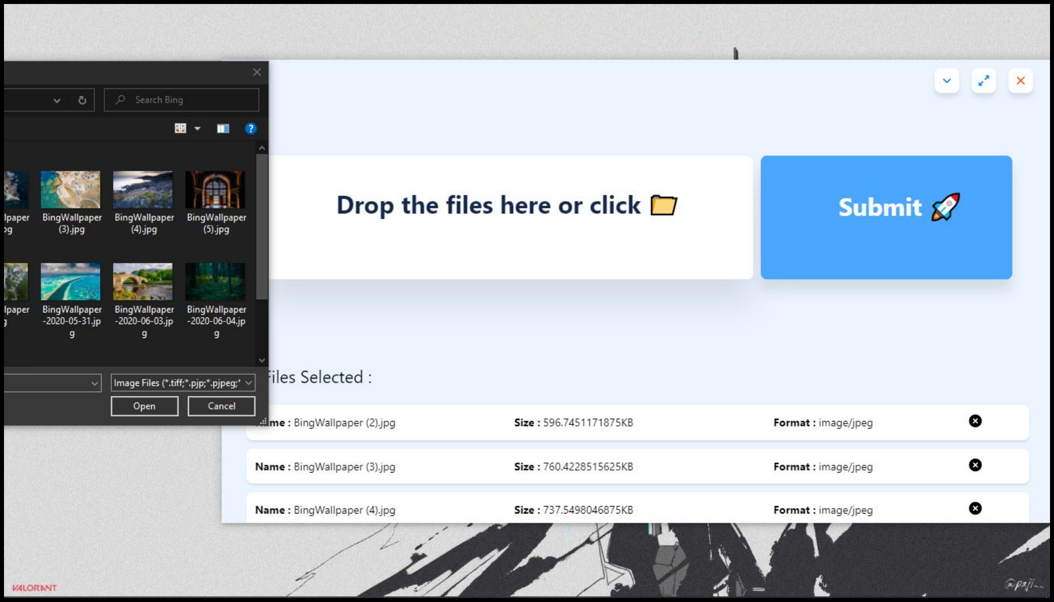


True Label \ Predicted Label	G	3	U	Name	With	A	Dont_like	K	7	No
G	96%	-	-	-	-	-	-	-	-	2%
3	-	92%	-	-	-	-	-	-	1%	-
U	-	-	98%	-	-	-	-	-	1%	-
Name	-	-	-	98%	-	-	-	-	-	-
With	-	-	-	-	100%	-	-	-	-	-
A	-	-	-	-	-	99%	-	-	-	-
Dont_like	-	-	-	-	-	-	100%	-	-	-
K	-	-	-	-	-	-	-	98%	-	-
7	-	2%	-	-	-	-	-	-	82%	-
No	1%	-	-	-	-	-	-	-	-	98%

Applications : Extra

I know the document was supposed to be 4 pages in content but as the deadline was increased *(plus I had space left on the page)*I wanted to try out something out of the scope of the competition but on my own 😇 and Thanks for reading.

To the right -> is the electron/node.js application I built as a prototype usage application which can allow users to select files and then send it to the server and the client receives the response and displays the result.


Search Bing
lpaper
pg
BingWallpaper (3).jpg
BingWallpaper (4).jpg
BingWallpaper (5).jpg
lpaper
BingWallpaper -2020-05-31.jpg
BingWallpaper -2020-06-03.jpg
BingWallpaper -2020-06-04.jpg
Image Files (*.tiff;*.pjp;*.pjpeg;
Open
Cancel
VALORANT

Drop the files here or click 📁

Submit 🚀

Files Selected :

me : BingWallpaper (2).jpg Size : 596.7451171875KB Format : image/jpeg

Name : BingWallpaper (3).jpg Size : 760.4228515625KB Format : image/jpeg

Name : BingWallpaper (4).jpg Size : 737.5498046875KB Format : image/jpeg

https://github.com/Sid911/pr