American Sign Language Fingerspelling Recognition

The Erdos Institute Data Camp

Steven Gubkin, Haidong Tian, Tong Qi, Owen Mireles

Overview - Problem

Voice-enabled assistants open the world of useful and sometimes life-changing features of modern devices. These revolutionary AI solutions include automated speech recognition (ASR) and machine translation.

Unfortunately, these technologies are often not accessible to the more than 70 million Deaf people around the world who use sign language to communicate, nor to the 1.5+ billion people affected by hearing loss globally.

Kaggle competition: https://www.kaggle.com/competitions/asl-fingerspelling/overview

Overview - Facts

Fingerspelling uses hand shapes that represent individual letters to convey words. While fingerspelling is only a part of ASL, it is often used for communicating names, addresses, phone numbers, and other information commonly entered on a mobile phone.

ASL fingerspelling can be substantially faster than typing on a smartphone’s virtual keyboard. But sign language recognition AI for text entry lags far behind voice-to-text or even gesture-based typing, as robust datasets didn't previously exist.

Kaggle competition: https://www.kaggle.com/competitions/asl-fingerspelling/overview

Overview - Goal

- ➢ Detect and translate American Sign Language (ASL) fingerspelling into text.
- ➢ There is potential for an app that can then translate text input using sign language-to-speech technology to speak the words. Such an app would enable the Deaf and Hard of Hearing community to communicate with hearing non-signers more quickly and smoothly.

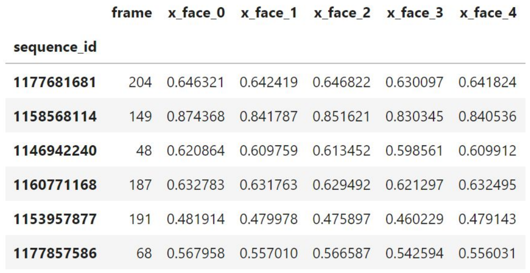
Kaggle competition: https://www.kaggle.com/competitions/asl-fingerspelling/overview

Data Description

More than three million fingerspelled characters produced by over 100 Deaf signers captured via the selfie camera of a smartphone with a variety of backgrounds and lighting conditions.

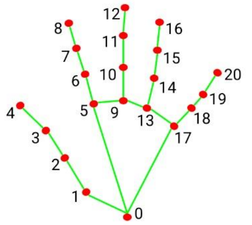
The videos were then processed with MediaPipe to yield the *landmark* files, consisting of the coordinates of several parts of the human body.

The train and test datasets contain randomly generated addresses, phone numbers, and URLs.



sequence_id	frame	x_face_0	x_face_1	x_face_2	x_face_3	x_face_4
1177681681	204	0.646321	0.642419	0.646822	0.630097	0.641824
1158568114	149	0.874368	0.841787	0.851621	0.830345	0.840536
1146942240	48	0.620864	0.609759	0.613452	0.598561	0.609912
1160771168	187	0.632783	0.631763	0.629492	0.621297	0.632495
1153957877	191	0.481914	0.479978	0.475897	0.460229	0.479143
1177857586	68	0.567958	0.557010	0.566587	0.542594	0.556031

Example of MediaPipe Holistic



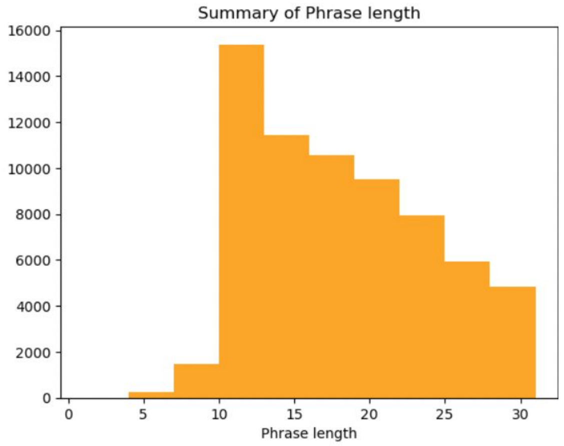
0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

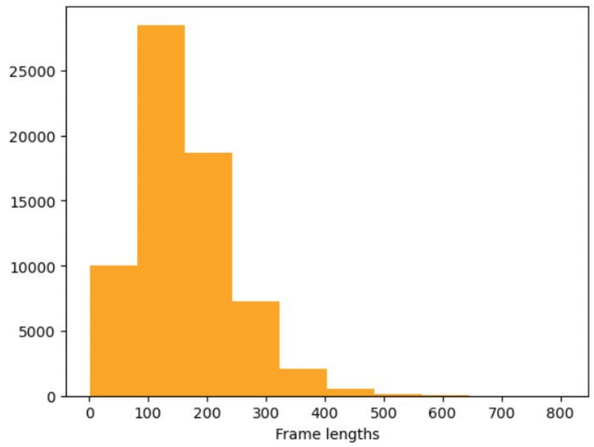
MediaPipe Landmarks for Hands

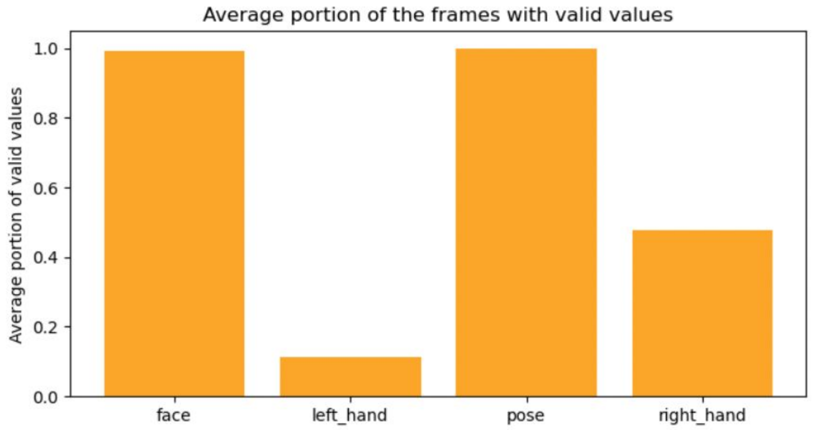
Dataset Challenges

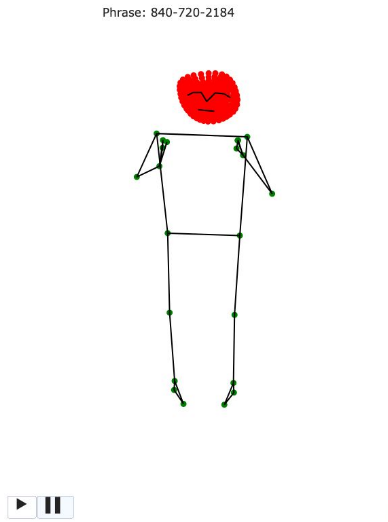
- The dataset is irregular.
  - There is an abundance of NaNs.
  - Most columns contain irrelevant information.
- Huge dataset.
  - The whole dataset occupies 170 GB.
- The base truth is hard to obtain.
  - While each *landmark* file does correspond to a phrase, the individual characters have to be guessed.
  - Some distinct characters have identical signs (like 6 vs. W or 0 vs. o)
  - Finding out where the transition between characters occur requires work.
    - Some letters, like I and Z, contain motion in their spelling.

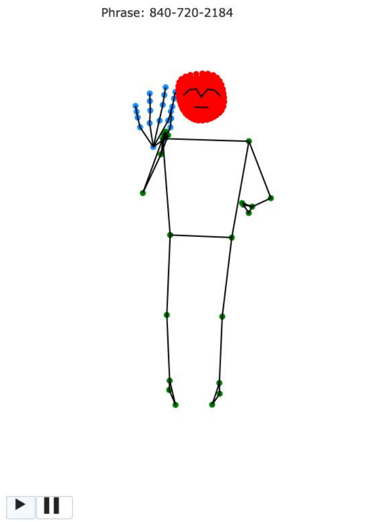
Data Visualization & EDA

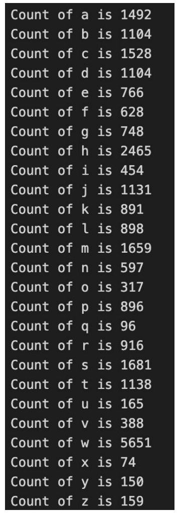












```
Count of a is 1492
Count of b is 1104
Count of c is 1528
Count of d is 1104
Count of e is 766
Count of f is 628
Count of g is 748
Count of h is 2465
Count of i is 454
Count of j is 1131
Count of k is 891
Count of l is 898
Count of m is 1659
Count of n is 597
Count of o is 317
Count of p is 896
Count of q is 96
Count of r is 916
Count of s is 1681
Count of t is 1138
Count of u is 165
Count of v is 388
Count of w is 5651
Count of x is 74
Count of y is 150
Count of z is 159
```

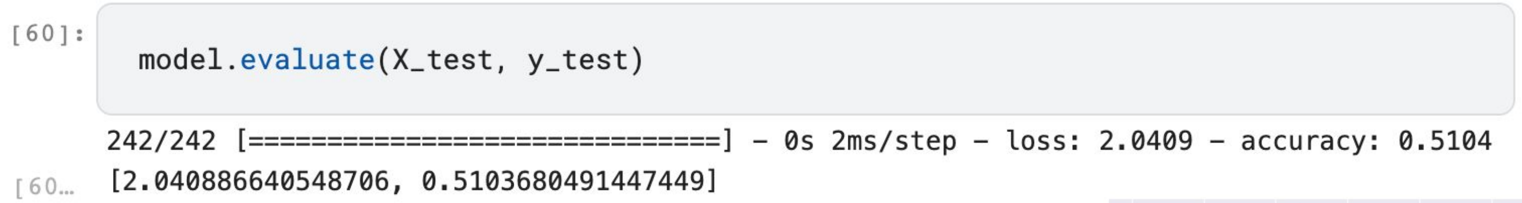
Insights about Data

- ➢ Keep only the columns relating to the left and right hands.
- ➢ Remove the rows with more than 20 NaNs, so as to remove the transitions.
- ➢ Center data around the coordinates x_0, y_0, which corresponds to the wrist.
- ➢ Left hand can be reflected to yield more samples for the right hand.
- ➢ Each finger is described by 8 features.

Method 1

- ➢ Use a 1D Convolutional NN to capture the fingers’ information.
- ➢ Use a feed-forward NN with 3 layers.
- ➢ Output one of the 59 possible characters with a softmax layer.
- ➢ Iterating through all frames, when a sequence of frames are all predicting the same character, add this character to our phrase prediction.

Results

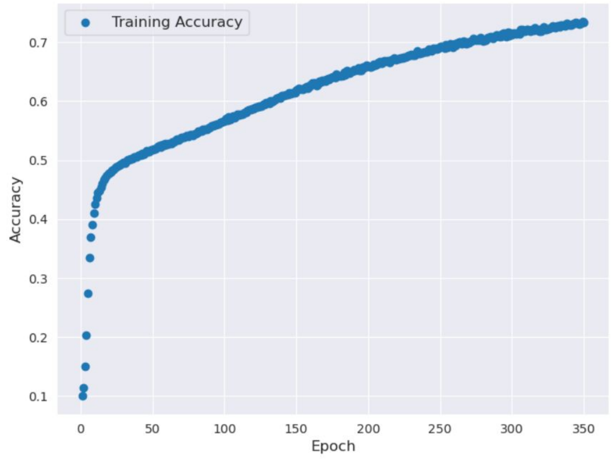


```
[60]:  model.evaluate(X_test, y_test)

       242/242 [==============================] - 0s 2ms/step - loss: 2.0409 - accuracy: 0.5104
[60…   [2.040886640548706, 0.5103680491447449]
```

Example prediction:

Prediction: 95wo+ano
Target: 9560 plano

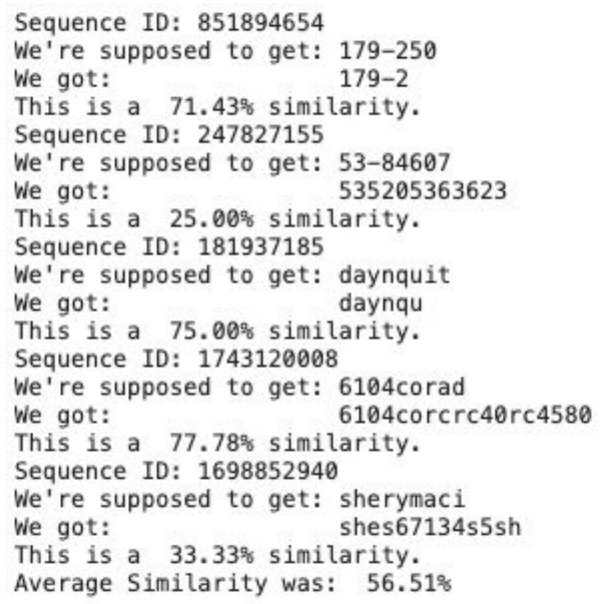


Method 2

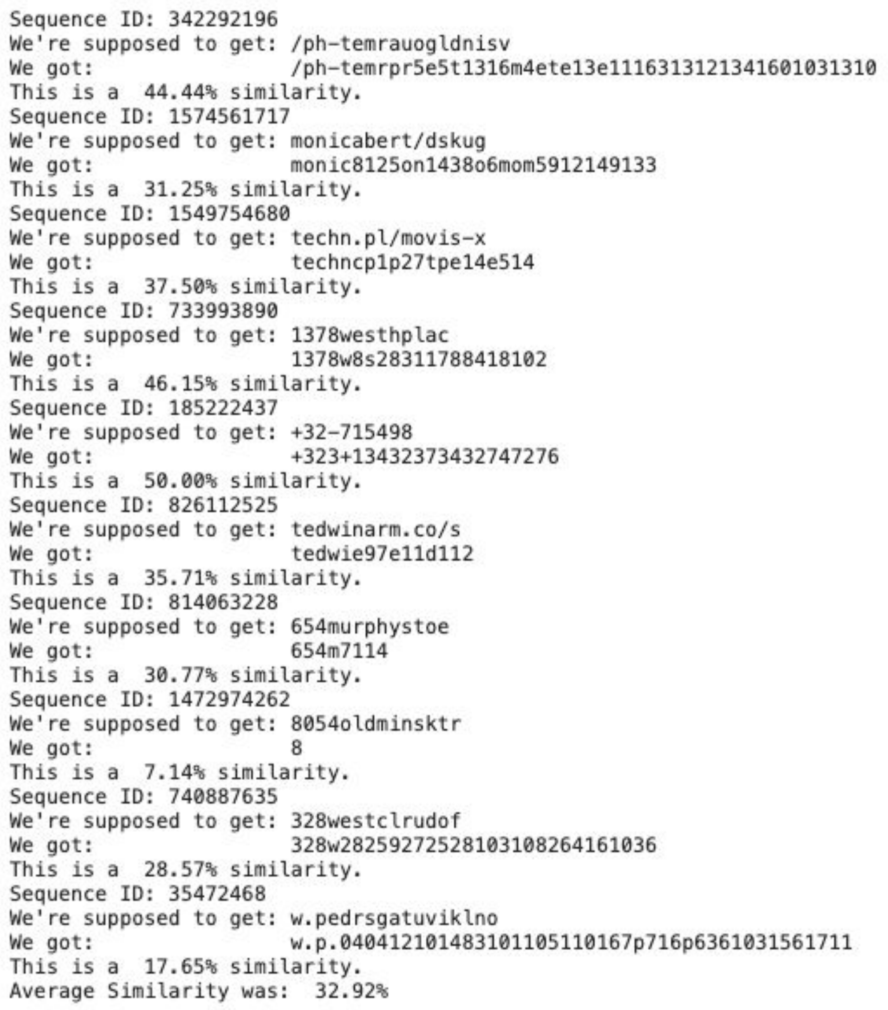
PCA → K-Means → Clusters → Translate

- ➢ Number of components: By observing the explained variance, we choose the dimension the contributes 1% information as much as the first one.
- ➢ Number of clusters: By removing the repeated letters/numbers/signs in the phrase, we set the number of unique letters as the number of clusters.
- ➢ Get cipher and do substitution translation to get the final text.

Results



```
Sequence ID: 851894654
We're supposed to get: 179-250
We got:                179-2
This is a  71.43% similarity.
Sequence ID: 247827155
We're supposed to get: 53-84607
We got:                535205363623
This is a  25.00% similarity.
Sequence ID: 181937185
We're supposed to get: daynquit
We got:                daynqu
This is a  75.00% similarity.
Sequence ID: 1743120008
We're supposed to get: 6104corad
We got:                6104corcrc40rc4580
This is a  77.78% similarity.
Sequence ID: 1698852940
We're supposed to get: sherymaci
We got:                shes67134s5sh
This is a  33.33% similarity.
Average Similarity was:  56.51%
```



```
Sequence ID: 342292196
We're supposed to get: /ph-temrauogldnisv
We got:                /ph-temrpr5e5t1316m4ete13e11163131213416010313 10
This is a  44.44% similarity.
Sequence ID: 1574561717
We're supposed to get: monicabert/dskug
We got:                monic8125on1438o6mom5912149133
This is a  31.25% similarity.
Sequence ID: 1549754680
We're supposed to get: techn.pl/movis-x
We got:                techncp1p27tpe14e514
This is a  37.50% similarity.
Sequence ID: 733993890
We're supposed to get: 1378westhplac
We got:                1378w8s28311788418102
This is a  46.15% similarity.
Sequence ID: 185222437
We're supposed to get: +32-715498
We got:                +323+13432373432747276
This is a  50.00% similarity.
Sequence ID: 826112525
We're supposed to get: tedwinarm.co/s
We got:                tedwie97e11d112
This is a  35.71% similarity.
Sequence ID: 814063228
We're supposed to get: 654murphystoe
We got:                654m7114
This is a  30.77% similarity.
Sequence ID: 1472974262
We're supposed to get: 8054oldminsktr
We got:                8
This is a  7.14% similarity.
Sequence ID: 740887635
We're supposed to get: 328westclrudof
We got:                328w28259272528103108264161036
This is a  28.57% similarity.
Sequence ID: 35472468
We're supposed to get: w.pedrsgatuviklno
We got:                w.p.04041210148310110511016 7p716p6361031561711
This is a  17.65% similarity.
Average Similarity was:  32.92%
```

What's Next?

- Downsize and smooth data
  - Identify which hand is used in the video and remove z-coordinate components.
  - For a specific sequence, group rows with no NaNs and keep only the groups whose length is larger than 10% of the total number of rows.
  - Extract the center of the hand, calculate the relative distances of landmarks to the center, map the right hand to the left hand by mirror symmetry and keep the five landmarks with the largest distances to the center to represent the outer shape of the hand.
- Clean data
  - Scale data so that the largest relative distance is one unit.
  - Cluster data for all the 59 symbols, filter out data whose distance to the center is larger than one standard deviation and find the correspondence between clusters and symbols.
- Try a recurrent NN with CTC loss function similar to DeepSpeech2.
- Train and test models