Capstone

Sprint 3

ANKIT DHEENDSA

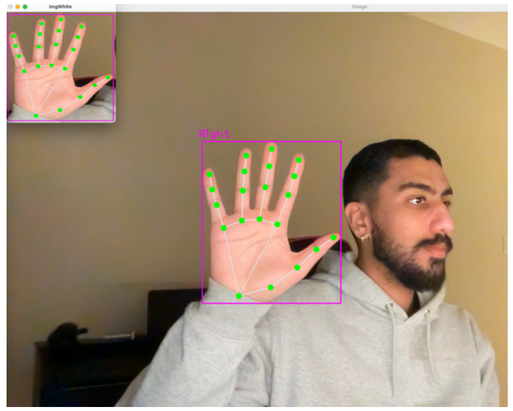
ASL Detection and Translation

Bridging the gap in communication caused by mutism

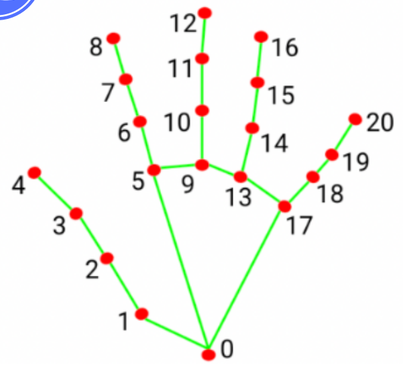
Looking to disrupt a market of 70M + people in need of support

Data Collection and Understanding

1



2



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

3

Image	Landmark 0_X	Landmark 1_X	Landmark 2_X	Landmark 3_X	Landmark 4_X	Landmark 5_X
Image_1	538	616	594	612	642	580
Image_2	528	616	581	609	628	574
Image_3	527	620	578	607	622	571
Image_4	528	619	572	609	619	569

Landmark 0_Y	Landmark 1_Y	Landmark 2_Y	Landmark 3_Y	Landmark 4_Y	Landmark 5_Y
492	643	511	630	531	610
483	609	515	608	530	596
475	600	507	599	521	588
474	589	515	599	522	586

A.csv I.csv K.csv N.csv T.csv

Logistic Regression

main.py

Classification Report:

	precision	recall	f1-score	support
A	0.97	0.99	0.98	389
B	0.96	0.88	0.92	527
C	0.87	0.95	0.91	401
HELLO	0.99	1.00	0.99	424
I	1.00	0.99	1.00	400
K	1.00	1.00	1.00	420
N	0.98	0.99	0.99	391
T	0.99	0.96	0.97	422
accuracy			0.97	3374
macro avg	0.97	0.97	0.97	3374
weighted avg	0.97	0.97	0.97	3374

CNN ON STILL IMAGES

```
Epoch 1/15
accuracy: 0.7326
Epoch 2/15
accuracy: 0.8820
Epoch 3/15
accuracy: 0.9148
Epoch 4/15
accuracy: 0.9320
Epoch 5/15
accuracy: 0.9404
Epoch 6/15
accuracy: 0.9511
......
Epoch 15/15
accuracy: 0.9741
```

Training

main.py

Class	Average Confidence	Accuracy
I	0.9999	1.0000
N	0.9977	0.9680
T	0.9984	0.9782
A	0.9994	0.9940
Hello	1.0000	1.0000
C	1.0000	1.0000
B	1.0000	1.0000
K	1.0000	1.0000

Testing

KEY INSIGHT ON PROBLEM SPACE

- **Predictions are made per frame (16ms) = too fast**
- **Minor changes in position or environment will change predictions**
- **Interpretation becomes <u>impossible</u> if predictions are made too fast**

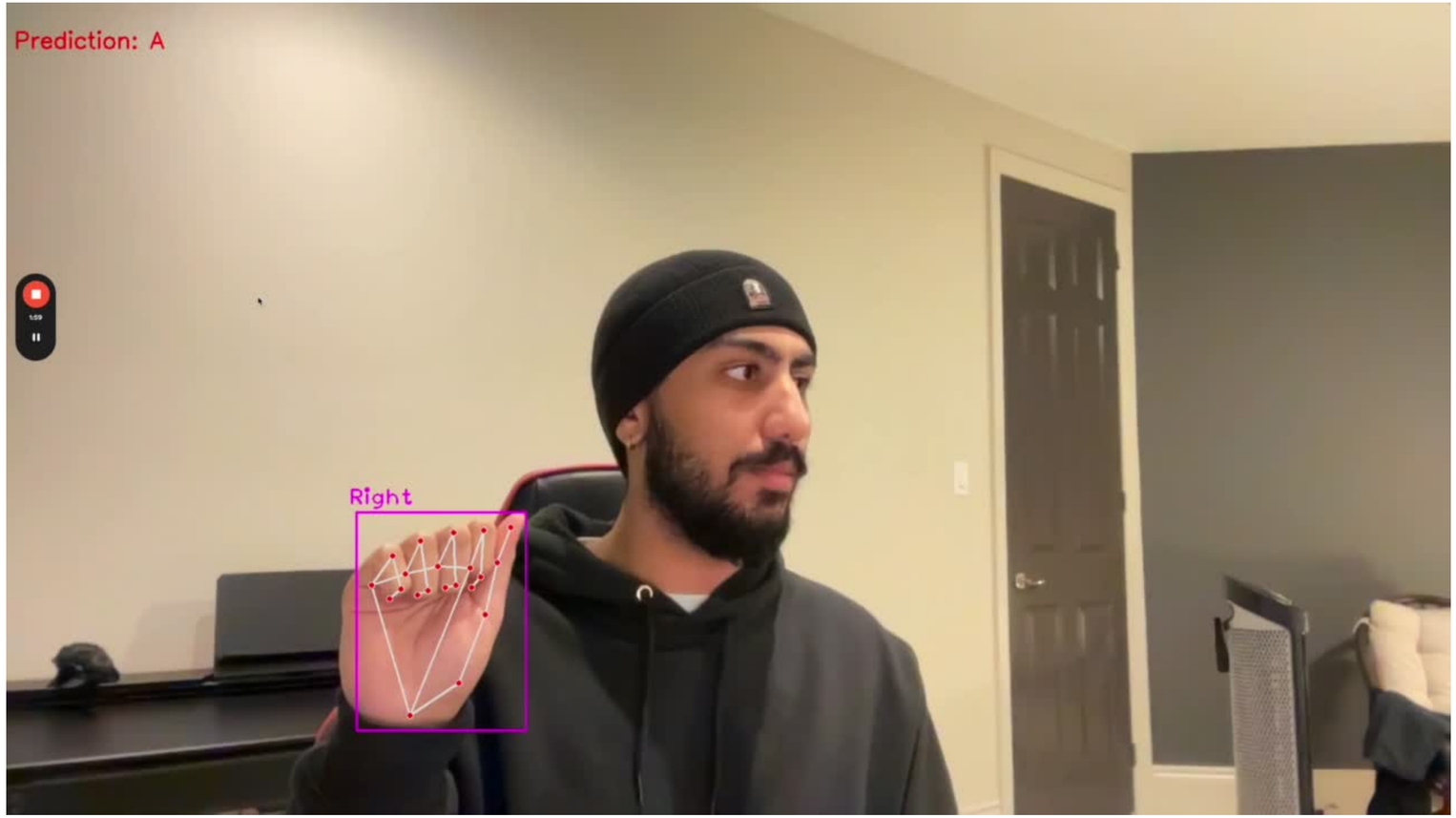
```
# Initialize variables for prediction smoothing
prediction_history = []  # Store a history of predictions
prediction_interval = 10  #Set interval to 10 frames

# Calculate the smoothed prediction as the mode
smoothed_index = max(set(prediction_history), key=prediction_history.count)
smoothed_prediction = labels[smoothed_index]
```

```
prediction_history = [A,A,A,B,A,K,C,C,B,Hello]
A = 4
B = 2
C = 2
K = 1
Hello = 1

smoothed_prediction = A
```

CNN Demo



NEXT STEPS

```
next_steps = [
    "Step 1: Increase data set for similar classes",
    "Step 2: Expand model to all letters",
    "Step 3: Train the model on video data of various people",
    "Step 4: Apply lighting alterations on video data"
]

print("My Next Steps:")
for step in next_steps:
    print(f"- {step}")
```

THANK YOU FOR LISTENING!