ASL Fingerspelling Project Mobile Computing

KUMAR SWAMY METTELA
Arizona State University, Tempe, USA
kmettela@asu.edu

VIJAY KUMAR PENUBOLU
Arizona State University, Tempe, USA
vpenubol@asu.edu

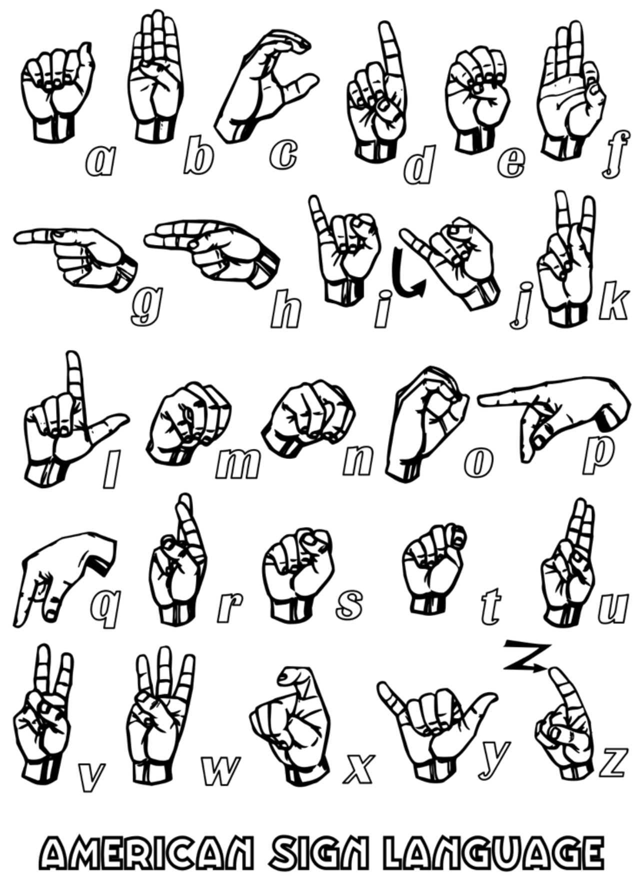
PAVAN KUMAR KONJETI
Arizona State University, Tempe, USA
pkonjeti@asu.edu

ESHWAR PALLEM
Arizona State University, Tempe, USA
epallem@asu.edu

ABSTRACT
People with impairments have long used sign language as a means of communicating. In this project, an image processing and supervised machine learning will be used to create a sign language fingerspelling alphabet recognition system. Specifically, 24 alphabetical symbols are conveyed via a variety of static motions (excluding 2 motion gestures J and Z). A palm detection algorithm is developed that uses PoseNet to gather hand contours and then trains a CNN model to recognize alphabets using the existing data. The next stage is to recognize a complete word using a series of segmentation and combination algorithms that extract alphabets from words in the video, recognize them, then combine them and show the entire word. This application has a high level of accuracy in recognizing letters and words.

Key Words: CNN (Convolutional neural network) model, PoseNet, ASL (American Sign Language), segmentation, Depth future recognition.

INTRODUCTION
The American Sign Language (ASL) is widely utilized by the deaf community to communicate. It is a full language with grammar and linguistic traits, like English. Hands and facial expressions are used to portray ASL alphabets. Because of the widespread use of ASL, we decided to create an application that can accept an ASL input and provide the matching meaning as an output.
The 26 alphabets of English are supported by American Sign Language by employing basic hand movements that otherwise would be utilized for Fingerspelling. It is kind of alphabet borrowing from one language to another. The two letters, 'J' and 'Z,' of all the 26 letters are depicted by static motions.



The enhanced Automated gesture recognition might improve computer-human interface and provide an alternate means of interacting with the system, particularly for the handicapped people. Human postures and faces might be used to aid in the understanding of human behavior.
Previously, Fingerspelling was done using several feature extractions approaches and machine learning models. The approach we chose involves extracting a frame from a video using image processing, which is then evaluated, and the picture's complex properties retrieved using a Convolutional Neural Networks (CNN) model. This information is used to train the model and predict ASL meaning.

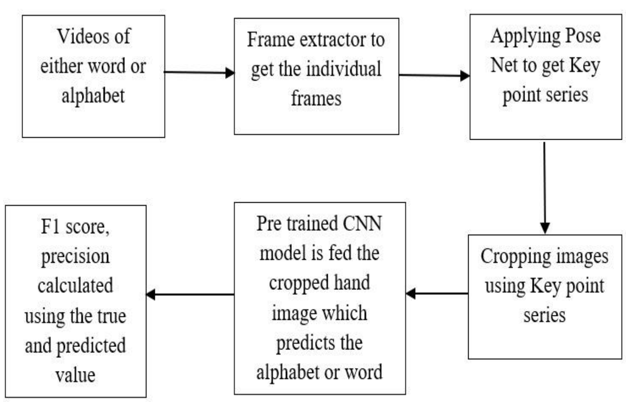
SOFTWARE/TOOLS USED

We have used the following software/tools to develop our project-

- ASL data set from Kaggle [https://www.kaggle.com/grassknoted /asl-alphabet](https://www.kaggle.com/grassknoted/asl-alphabet)
- Posenet
- Python3.7
- TensorFlow
- Keras
- Visual Studio

SYSTEM ARCHITECTURE

We created an application that employs American Sign Language and has been trained using alphabets to predict what the gesture in the video is about. A model is constructed and trained using ASL alphabet videos. Posenet, a deep learning network, is used by the palm recognition approach to obtain wrist points. The cropping method is used to isolate the area of the image that only contains the wrist. These images are then used to train the CNN model. Similarly, we build the model for word comprehension by training it with videos. Posenet facilitates in the creation of the key point series using images gathered from the videos. A different approach known as the segmentation algorithm is used to separate the alphabets in the video clipping. Another method is also implemented to link the individual alphabets to form the word.



IMPLEMENTATION

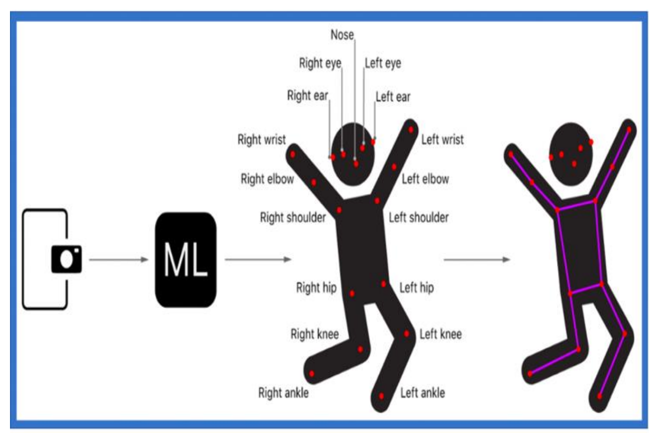
We have implemented various tasks for this project based on unique properties used for it uses at each stage, which are as follows.

1) **Acquiring individual frames from the videos of alphabets and words that are to be predicted.**

For instance, consider an alphabet A of the ASL alphabets. The ASL representation of the letter is present in the Input videos recorded. The wrist part of the video is isolated and individual frames are extracted from that part to identify the letter or word posed.

2) **Once the frames are extracted with the help of PoseNet Key points json file is obtained**

The resultant of the PoseNet could easily be comprehended using the Image below.



Consider the ASL alphabet "L" the key points are generated as follows.

[{"score":0.519720477216384,"keypoints":[{"score":0.999414324760437,"part":"nose","position":{"x":1422.0798220973784,"y":426.7812938904
{"score":0.9999403953552246,"part":"leftEye","position":{"x":1516.583274812734,"y":377.6434486189139}},{"score":0.998595714569092,
"part":"rightEye","position":{"x":1346.5971734550562,"y":374.73475538389516}},{"score":0.985547661781311,"part":"leftEar","position":{"
572272940075,"y":436.22198326310865}},{"score":0.9986810684204102,"part":"rightEar","position":{"x":1201.379988881086,"y":442.213102762
{"score":0.904606819152832,"part":"leftShoulder","position":{"x":1811.2350772471912,"y":780.5930331226591}},{"score":0.7430485486984253
"part":"rightShoulder","position":{"x":969.6611657303371,"y":812.722963483146}},{"score":0.49351778626441956,"part":"leftElbow","positi
{"x":1884.2282595973784,"y":1029.9906367041199}},{"score":0.19289246201515198,"part":"rightElbow","position":{"x":449.58801498127343,"y
0026919475656}},{"score":0.34777742624282837,"part":"leftWrist","position":{"x":721.6077803136704,"y":710.0941450140449}},{"score":0.
3791676461696625,"part":"rightWrist","position":{"x":721.5152446161048,"y":697.9464390215355}},{"score":0.18663758039474487,"part":"lef
"position":{"x":690.3635592228464,"y":952.6610779494383}},{"score":0.35957303643226624,"part":"rightHip","position":{"x":622.1483643492
3172109082398}},{"score":0.04349255561828613,"part":"leftKnee","position":{"x":812.2404611423221,"y":938.9583333333335}},{"score":0.
10141301155090332,"part":"rightKnee","position":{"x":794.6610925795881,"y":951.4015683520599}},{"score":0.04005280137062073,"part":"lef
"position":{"x":782.6479108146068,"y":1167.4741046348315}},{"score":0.05962541699409485,"part":"rightAnkle","position":{"x":576.6249341
"y":1170.090560627341}}]},{"score":0.503010078388102,"keypoints":[{"score":0.9994115829467773,"part":"nose","position":{"x":1441.425708
"y":428.03363471441946}},{"score":0.9999420046806335,"part":"leftEye","position":{"x":1515.9233087546818,"y":376.9851650280899}},{"scor
9998523592948914,"part":"rightEye","position":{"x":1345.3460030430713,"y":374.2878043071162}},{"score":0.9824978709220886,"part":"leftE
"position":{"x":1577.5956811797753,"y":426.70514396067415}},{"score":0.9986620545387268,"part":"rightEar","position":{"x":1200.07644253
"y":443.732663272472}},{"score":0.8925936222076416,"part":"leftShoulder","position":{"x":1810.0452071629213,"y":778.1224397237829}},{"s
7326787710189819,"part":"rightShoulder","position":{"x":971.4976152855805,"y":810.1321102528091}},{"score":0.3541119396686554,"part":"l
"position":{"x":1886.412979868914,"y":1043.599455758427}},{"score":0.17905688285827637,"part":"rightElbow","position":{"x":453.51039472
"y":1257.1286867977528}},{"score":0.4042171239852905,"part":"leftWrist","position":{"x":701.9216701779027,"y":701.509758309925}},{"scor

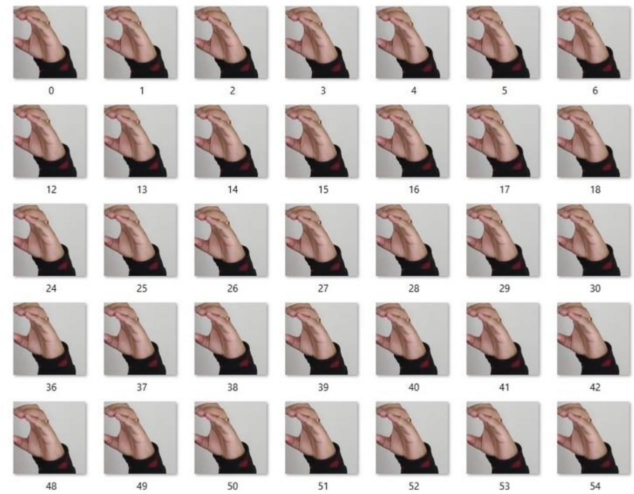
3) **Now, the json file is converted into the csv file**
So, the json file shown in the above task is converted into a csv file as shown-

Frames#	score_ove	nose_scor	nose_x	nose_y	leftEye_sc	leftEye_x	leftEye_y	rightEye_s	rightEye_x	rightEye_y	leftEar_sc	leftEar_x	leftEar_y	rightEar_s	rightEa
0	0.51972	0.999414	1422.08	426.7813	0.99994	1516.583	377.6434	0.99986	1346.597	374.7348	0.985548	1577.572	436.222	0.998681	1201
1	0.50301	0.999412	1441.426	428.0336	0.999942	1515.923	376.9852	0.999852	1345.346	374.2878	0.982498	1577.596	426.7051	0.998662	1200.
2	0.493245	0.99949	1440.608	427.8151	0.999944	1515.047	376.5825	0.999845	1344.664	373.5365	0.986264	1579.104	425.8227	0.998804	1200.
3	0.525515	0.999482	1441.097	428.1434	0.999954	1513.077	376.3665	0.999822	1344.527	373.2903	0.984829	1578.231	427.6848	0.998903	1200.
4	0.522988	0.999531	1440.218	428.869	0.99996	1511.786	375.9809	0.999811	1343.376	372.8996	0.985966	1580.62	424.0332	0.998811	1211.
5	0.524502	0.999558	1439.831	428.8631	0.999959	1510.335	375.829	0.999835	1342.289	373.6011	0.987401	1580.25	424.8618	0.998886	1211.
6	0.509065	0.999679	1438.203	428.9725	0.999962	1508.894	375.7793	0.999845	1340.929	374.0279	0.986982	1579.826	423.8256	0.998772	1210.
7	0.513841	0.999676	1438.23	429.1221	0.999958	1509.605	376.4591	0.999875	1340.265	374.7033	0.98572	1578.487	423.2605	0.998814	1209.
8	0.513527	0.99973	1436.618	428.971	0.999956	1508.545	376.5454	0.999899	1338.744	375.5094	0.983324	1576.872	421.4703	0.998884	1208.
9	0.539827	0.999771	1435.802	428.9776	0.999961	1506.518	376.9812	0.999904	1337.988	376.1033	0.973438	1571.552	426.7616	0.999103	1207.
10	0.535188	0.999769	1435.962	429.2654	0.999961	1506.265	377.5159	0.99991	1337.952	376.4283	0.972155	1572.193	426.1597	0.99917	1207.
11	0.527302	0.999763	1436.242	429.607	0.999961	1506.623	378.2173	0.999913	1337.293	376.9227	0.977414	1572.299	428.3082	0.999172	1206.
12	0.500547	0.999754	1435.644	431.2875	0.99996	1506.78	378.5229	0.999914	1336.002	377.0199	0.974603	1572.268	426.302	0.999304	1206.
13	0.51374	0.999768	1433.75	431.9033	0.999963	1506.224	379.0259	0.99991	1335.574	376.9636	0.974841	1569.97	430.8783	0.999282	1206.
14	0.499983	0.999777	1433.117	432.7102	0.999968	1505.308	379.4015	0.99991	1335.175	376.9938	0.975413	1569.773	432.033	0.999447	1206.
15	0.519663	0.999784	1432.462	432.7167	0.999969	1503.913	379.2976	0.99992	1334.666	377.3079	0.967543	1568.24	430.9564	0.999341	1206
16	0.506807	0.999772	1431.8	433.697	0.999967	1502.918	379.5607	0.99993	1334.286	377.6663	0.968795	1568.875	429.0543	0.999292	1205.
17	0.517639	0.999776	1431.243	434.6514	0.999964	1504.325	380.1108	0.999937	1334.146	378.0899	0.96345	1569.392	426.955	0.999019	1206.
18	0.513588	0.999768	1430.566	435.0885	0.999965	1503.808	380.3576	0.999935	1333.718	378.5175	0.967309	1568.817	430.9962	0.998906	1206.
19	0.509723	0.999754	1431.138	435.4653	0.999965	1504.211	380.5245	0.999931	1333.355	378.4304	0.965998	1567.844	435.1371	0.998824	1206.
20	0.506679	0.999781	1430.686	435.7938	0.999966	1504.508	380.5921	0.999934	1333.516	378.504	0.965175	1566.721	437.3786	0.999203	1206.
21	0.509912	0.999794	1431.313	436.1897	0.999968	1504.461	381.2971	0.999932	1333.719	378.7627	0.976103	1567.83	441.8414	0.999261	1205.
22	0.515815	0.99981	1430.144	436.3332	0.999971	1502.928	381.5182	0.999927	1332.668	379.1814	0.984488	1568	444.2311	0.999337	1205
23	0.524591	0.999808	1427.837	437.6146	0.999972	1502.101	381.458	0.999937	1332.063	379.5397	0.979261	1565.902	440.5679	0.999408	1203.
24	0.517954	0.999812	1426.571	438.1426	0.999969	1501.64	381.5631	0.999942	1332.129	379.7357	0.977234	1564.731	441.9092	0.999497	1203.

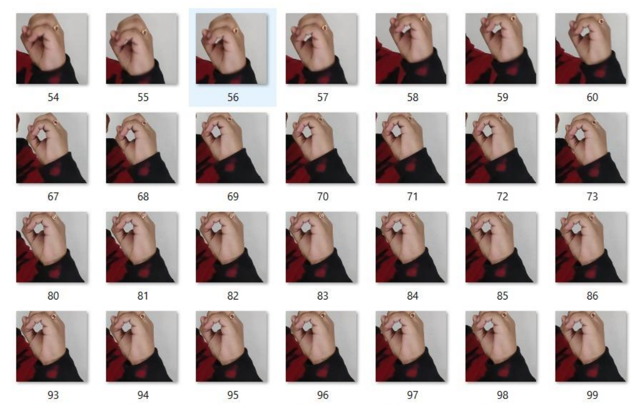
4) **Considering the coordinates from the right wrist and left wrist to the Key_points.csv file and the individual frames acquired from the first task only the part consisting of the hand is cut out.**
We'll obtain left_wrist_score, left_wrist x coordinate, left wrist y coordinate, right_wrist_score, right_wrist x coordinate, right_wrist y coordinate for each of the frames. We'll extract the x and y coordinates of that hand for that frame based on the leftwristscore and rightwristscore, whichever is larger.
With the X and Y coordinates we will be able to build a box with Y-d, Y+d, X-d, X+d. (where the variable d is a constant which is different based on the width and length of the video frame). As the box is built the frames are extracted in that manner in order to cut out the hand portion alone. Considering the alphabet "C" as we have the input videos, the video undergoes the above tasks to give out the output as the frame that only comprises of the hand and nothing else.Similarly, even for words frames are extracted throughout the video where the frames consist of the letters that form the word.

5) **With the cropped image in hand, the image is fed to the pre trained CNN model (trained priorly using the data from Kaggle) and it predicts the ASL word as the output.**
The extracted frames (hand part only) are fed to the CNN model, which predicts the alphabet, thanks to a python software.

The following is a snapshot of the output of the python application for ASL alphabet detection:



Similarly, the following is a snapshot of the output of the python application for ASL word detection:



ASL word detection algorithm:
Using the Posnet, we will obtain keypoints JSON of the ASL word video frames and convert the keypoints JSON to CSV. As a result, we'll end up with the essential keyframes from the ASL word video. From the keypoints csv file, this method will monitor the current and prior x and y coordinates of the Left or Right wrist.

If the absolute value of the difference between the current x coordinate and the previous x coordinate, or the absolute value of the difference between the current y coordinate and the previous y coordinate, in either hand, exceeds a threshold value, a transition of an alphabet occurs, and all frames from the current frame number to the transition frame number are fed to the pretrained CNN model to determine that alphabet. This method is repeated until the video's final frames.

6) **Considering the predicted value and the true value the precision, F1 score, recall has been displayed for both ASL Alphabet and Word detection.**

```
0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           A       0.00      0.00      0.00         2
           B       0.00      0.00      0.00         3
           C       0.00      0.00      0.00         0
           D       0.00      0.00      0.00         1
           E       0.00      0.00      0.00         0
           F       0.00      0.00      0.00         0
           G       0.00      0.00      0.00         1
           H       0.00      0.00      0.00         0
           I       0.00      0.00      0.00         4
           J       0.00      0.00      0.00         0
           K       0.00      0.00      0.00         1
           L       1.00      0.14      0.25         7
           M       0.00      0.00      0.00         0
           N       0.00      0.00      0.00         0
           O       0.00      0.00      0.00         0
           P       0.00      0.00      0.00         1
           Q       0.00      0.00      0.00         0
           R       0.00      0.00      0.00         0
           S       0.00      0.00      0.00         0
           T       0.00      0.00      0.00         0
           U       0.00      0.00      0.00         0
           V       0.00      0.00      0.00         0
           W       1.00      0.50      0.67         2
           X       0.00      0.00      0.00         0
           Y       1.00      0.25      0.40         4
           Z       0.00      0.00      0.00         0
```

```
    accuracy                           0.19        27
   macro avg       0.18      0.08      0.11        27
weighted avg       0.56      0.19      0.26        27
```

```
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

         ODD       0.00      0.00      0.00         0
         OLD       1.00      0.33      0.50         3
         WOW       0.00      0.00      0.00         0

    accuracy                           0.33         3
   macro avg       0.33      0.11      0.17         3
weighted avg       1.00      0.33      0.50         3
```

results

	pred	TRUE
0	P	A
1	G	B
2	A	C
3	L	D
4	I	E
5	L	F
6	J	G
7	K	H
8	I	I
9	I	J
10	Y	K
11	L	L
12	I	M
13	I	N
14	O	O
15	X	P
16	J	Q
17	B	R
18		r
19	I	S
20	Y	T
21	B	U
22	W	V
23	W	W
24	A	X
25	Y	Y
26	G	Z

7) Finally, a CSV file named result.csv is created. The csv file includes only two columns: projected value and true value.

```
-
Selection of Frame is Done


Predicting alphabets from frames extracted.
-
-
-
generating keypoint timeseries for the word from posenet.csv
-
-
-

True Value: OLD Prediction: OLD
Running for WOW.mp4
```

RELATED WORK

Frame Extraction-

Cv2 a python library is used for extraction the frames. Cv2 in full form OpenCV is a an open-source machine learning and computer vision python library. It establishes a standard framework for computer vision and related areas applications. This library can be used to identify photos and videos to recognize items, people, and even human handwriting and is widely used in computer vision, machine learning and deep learning applications.

Posenet-

Pose estimation is a phrase used to describe computer vision algorithms that recognize human figures in photos and videos, such as determining where someone's elbow appears in an image. It's vital to remember that posture estimation just guesses the location of major body joints and does not identify who is in an image or video.

The posture estimation models take a processed camera picture as input and output keypoint information. The discovered keypoints are indexed by a component ID and a confidence score ranging from 0.0 to 1.0. The confidence score represents the likelihood of finding a keypoint at that location.

CNN-

The CNN model is a form of neural network that enables users to extract higher representations for picture input. Unlike traditional image recognition, which requires the user to define the image characteristics, CNN works in a different way, it takes the picture's raw pixel data, and then trains the model, and then extracts the features from picture for improved categorization.

FEASIBILITY ANALYSIS

LINKS:

1) Alphabet and Word Videos: The ALS videos (26 alphabets and 10 words) which are taken by our team members for the project are in the following link.
https://drive.google.com/drive/folders/1fFcfQWyJAJp91nSHNfRw9zW0jsjLTFlU?usp=sharing
2) Demo Links:
80% completion video, i.e., tasks before 8 includes everything except pipeline input/output and ASL word detection and 100% completion video, entire project including accuracy and such.
https://www.youtube.com/playlist?list=PL3Sl9Nug7R3FoSqUkKhtRb4VA5CGpWEAi

TASK COMPLETION

S. No	**Task**	**Assigned To**
1	Recording (26+3) *4 ASL sign videos.	Kumar Swamy, Vijay Kumar, Pavan Kumar, Eshwar Pallem
2	Using posenet and developing a palm cropping algorithm using wrist points of hands.	Kumar Swamy, Pallem Eshwar.
3	Validation of the palm detection algorithm.	Vijay Kumar, Pavan Kumar
4	Configuration of CNN 3D model.	Pallem Eshwar, Pavan Kumar
5	F1 score for the above	Kumar Swamy, Vijay Kumar
6	Recording of 10*4 random words using ASL.	Kumar Swamy, Vijay Kumar, Pavan Kumar, Eshwar Pallem
7	Development of Keypoint Series	Pavan Kumar, Vijay Kumar
8	Implementation of Segmentation Algorithm	Kumar Swamy, Pallem Eshwar
9	Recognizing Alphabets with 3D CNN	Vijay Kumar, Kumar Swamy, Pavan Kumar
10	Developing a word recognition algorithm	Pallem Eshwar, Pavan Kumar
11	Automation pipelining	Kumar Swamy, Pallem Eshwar
12	Calculating the accuracy of word recognition and F1 score.	Pavan Kumar, Kumar Swamy
13	Final Report	Pallem Eshwar, Pavan Kumar, Vijay Kumar, Kumar Swamy.

ACKNOWLEDGEMENT

We'd like to express our gratitude to Dr. Ayan Banerjee for his encouragement and assistance with our questions. We'd also want to thank the writers whose work has aided us in developing and understanding earlier study ideas. We'd like to express our gratitude to every one of our team members for their efforts and contributions to the project.

CONCLUSION

In this project, we have proposed an application that can recognize ASL signs through recorded videos. We now have a comprehensive knowledge of how algorithms may be used to translate ASL into other languages. Going through other people's work on this topic, we got an overview of the current research activities in the subject of language translation, as well as a good grasp of the various machine learning methods and their implementations. To increase the accuracy, a variety of ways were investigated. We got on overview of Posenet, a deep learning model that estimates human pose by detecting human parts.

FUTURE WORKS

As this project stands now, has a significant accuracy for the dataset used. We plan to do the following in the future-

- Using a different dataset (Massey Dataset) from the one we used.
- Using OpenPose instead of Posenet, which is more accurate and works on GPU powered systems.
- Using other machine learning models like support vector machines, HOG-SVM, LBP-SVM, CNN-SVM, HOG-LBPLSVM.

Also, given the work indicated above, a stronger GPU-based system and better dataset is required, with the goal of achieving higher accuracy than now.

REFERENCES

1) Huy B.D Nguyen and Hung Ngoc Do, Deep learning for American Sign Language Fingerspelling recognition system, published in 2019 26th International Conference on Telecommunications (ICT).
2) P. Premaratne, Historical Development of Hand Gesture Recognition, 1st ed. Singapore: Springer, March 2014.
3) Y. Qiao, Z. Feng, X. Zhou, and X. Yang, "Principle component analysis-based hand gesture recognition for Android phone using area features," in the 2nd International Conference on Multimedia and Image Processing (ICMIP), March 2017, pp. 108–112.
4) T. Y. Pan, L. Y. Lo, C. W. Yeh, J. W. Li, H. T. Liu, and M. C. Hu, "Real-time sign language recognition in complex background scene based on a hierarchical clustering classification method," in the Second International Conference on Multimedia Big Data (BigMM), 2017, pp. 64–67.
5) H. Zuo, H. Fan, E. Blasch, and H. Ling, "Combining convolutional and recurrent neural networks for human skin detection," IEEE Signal Processing Letters, vol. 24, no. 3, pp. 289–293, March 2017.