Sign Language Recognition with CNN

By,

Rimzim Thube

Shree Gowri Radhakrishna

Vijayalaxmi Nagurkar

1. Abstract

Hand gesture is one of the methods used in sign language for non-verbal communication. It is essentially used by people with hearing or speech impairments to communicate among themselves and with others. However, to communicate using hand gestures, knowledge of sign language is necessary.

We are seeking to create a system that will automatically identify the hand gestures and convert its meaning into text. This will enable anyone to communicate better even without the explicit knowledge of sign language. This could prove beneficial especially in emergency situations.

The project is trained on AUSLAN dataset which is the Australian version of the sign language. It is composed by using gestures with both the hands. To train the model, we collected the images from a YouTube video of the AUSLAN signs. A CNN model was trained on this dataset.

For the demo, a client application was built with Python Flask framework that captures a live video and posts the video frame to a backend server application that runs the trained CNN model to generate the caption.

2. Introduction

A sign language is a language used to communicate between people by visually transmitting the sign patterns to express the meaning. It is a replacement of speech for hearing and speech impaired people. There are various sign languages used around the world. There are over 300 sign languages used around the world, and 70 million deaf people using them. However, the vast majority of communications technologies are designed to support spoken or written language (which excludes sign languages), and most hearing people do not know a sign language. As a result, many communication barriers exist for deaf sign language users.

To solve this barrier, we decided to build an application which will translate the sign language gestures in real-time using Deep Learning models. This model will capture the live hand gestures streamed through a webcam. Based on the gesture, a caption mentioning the letter will be shown immediately. This application can be deployed easily on mobile devices which makes it easy to carry around and use it to communicate with everyone.

This application can be used by employers for their employees with hearing disabilities, shopping centers to talk to their customers etc. This will improve the quality of life of people with hearing disabilities and help them be included in the society in a better way.

3. Related Work

This area as a whole is very new. The earliest attempts at sign recognition began with fingerspelling. Dr G Grimes at AT & T Bell Labs developed the “Digital Data Entry Glove” which was designed for alphanumeric input.

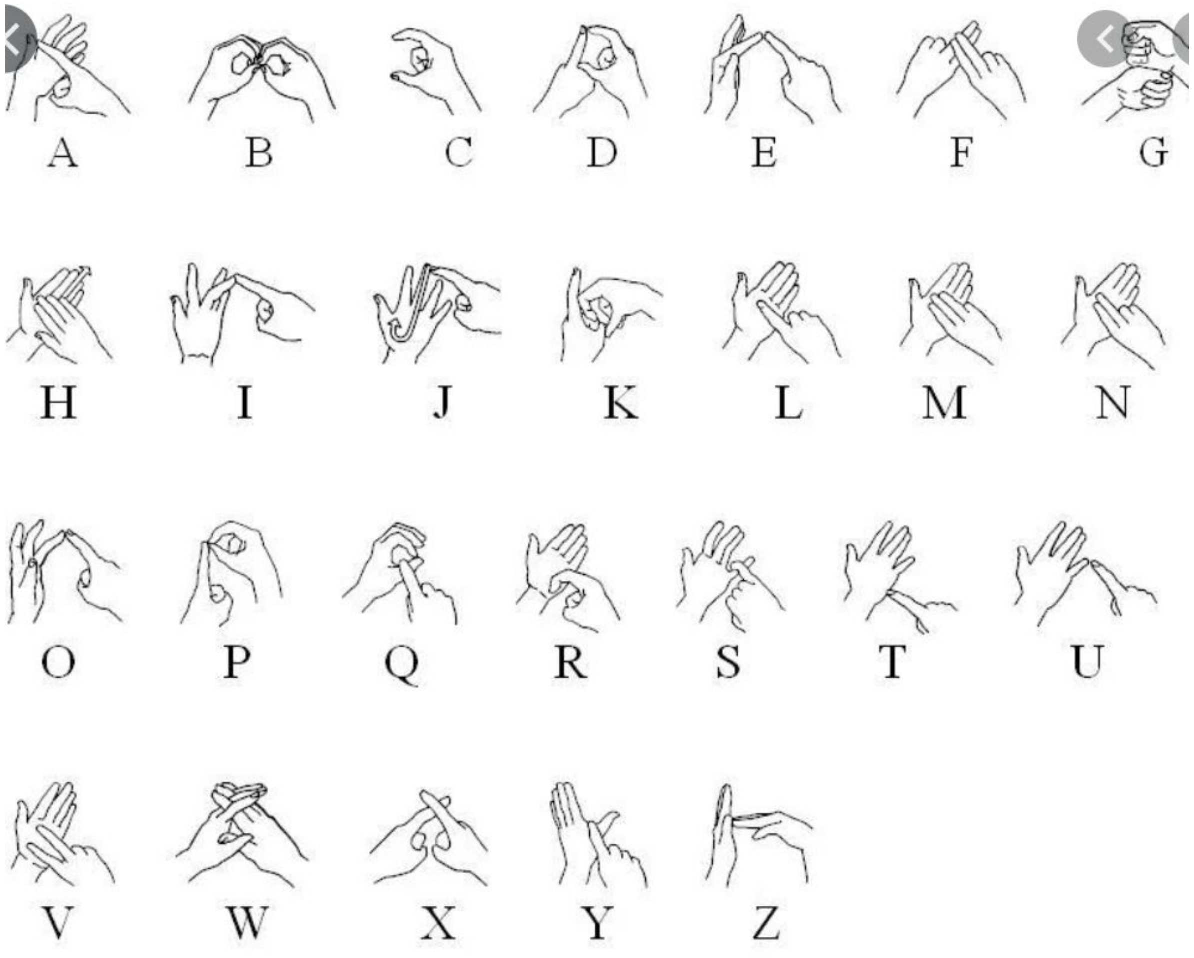
Recognition was accomplished by using present thresholds for the various hand positions.Charayaphan and Marble [CM92] developed a visual system to extract movements of 31 Auslan signs manually, then proceeded to try to work out positions with very simple algorithms. They collected one example of each of the 31 signs and tried to hand-code ways of recognising each sign. They could recognise 27 out of the 31 using relatively simple techniques,based on their approximation on how the signs would vary.

Davis and Shah [DS92] learnt a simple set of seven signs using cameras focused on a hand that had markings on the tip of each finger. However, it required explicit stop and start signals, and could only understand seven simple signs which it was programmed to recognise.

4. Data

Auslan is a sign language used in Australia. There are not many applications available for this language. Hence, we decided to support this language in our application.There is no dataset for the Auslan language available that can be readily used to train the model. Hence, we created our own dataset.

Below are the signs of the Auslan language:



Dataset creation:

- We downloaded a crowdsourced dataset from Amazon DataTurks.
- This dataset is created by downloading the images from YouTube videos demonstrating Auslan sign gestures.
- Each frame of the dataset is separated and stored.
- Each image is manually labeled and a bounding box is drawn capturing the gesture in the image.
- This dataset has the gesture images along with their labels and co-ordinates of the bounding box.
- As a part of dataset preparation, the dataset is downloaded from DataTurk and stored locally.
- The information of labels and the bounding box of every image is stored in separate CSV files.

- While loading the data, it is clubbed in a single data object to have the path to the image, bounding box coordinates and label.

4. Method

The final goal that was to be achieved was to create a web application that uses a computer's webcam to capture a person demonstrating the Auslan alphabet, and translate it in real time. The steps taken to achieve the following were:

1) Data collection
2) Trained a machine learning model (Resnet101) by providing it the collected dataset. It has a set of images to recognise the Auslan alphabets.
3) Post training, we saved the model
4) Build the user interface i.e the client and the application side using the pretrained model.

5. Algorithm

- **YOLO**

YOLO is a clever convolutional neural network (CNN) for doing object detection in real-time. The algorithm applies a single neural network to the full image, and then divides the image into regions and predicts bounding boxes and probabilities for each region. We are using the algorithm to **implement bounding boxes** in our application.

- **ResNet101**

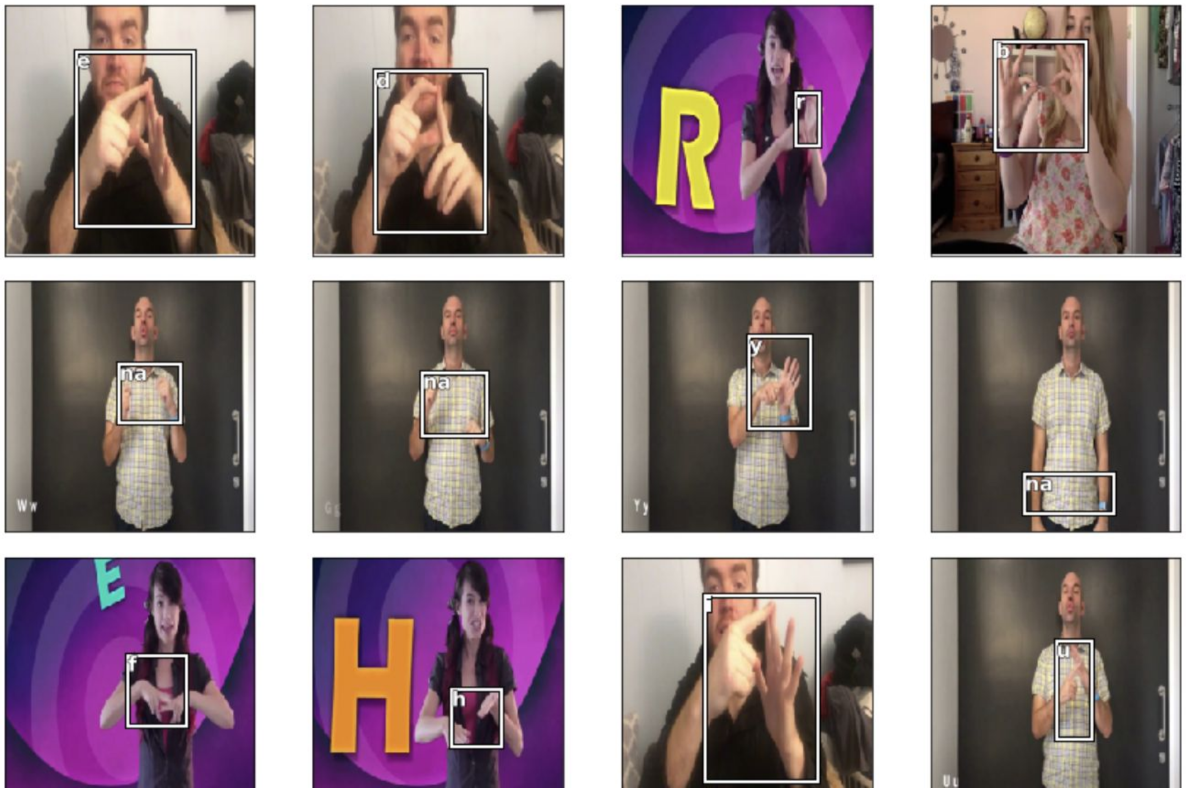
ResNet-101 is a convolutional neural network that is 101 layers deep. A pretrained version of the network trained on more than a million images from the ImageNet database. The pretrained network can classify images into 1000 object categories. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 224-by-224.

We used transfer learning to train the ResNet101 model. Below are the details of the model used -

- Transfer learning using - Resnet101 model
- Size - 224
- Batch size - 32
- Image Augmentation - fastai.ImageClassifierData
- Customised the Resnet model using ConvnetBuilder -

```
model_head = nn.Sequential(
    Flatten(),
    nn.ReLU(),
    nn.Dropout(0.5),
    nn.Linear(100352, 256),
    nn.ReLU(),
    nn.BatchNorm1d(256),
    nn.Dropout(0.5),
    nn.Linear(256, 4 + len(class_model_data.classes))
)
models = ConvnetBuilder(architecture, 0, 0, 0, custom_head=model_head)
```

The trained model was used to test whether the signs were captioned properly. The result is shown below. Every image has a bounding box along with the caption derived from the model.



6. Model Metrics

Input: Live video stream showing the hand gestures from the webcam

Output: Live captions showing the letter of the hand gesture

Model Metrics:

Accuracy	**85%**
L1 loss	**9%**

7. User Interface

The backend service, given an image of a sign, returns the predicted letter, and; the front-end that captures and displays video from the user's webcam and asks the backend for predictions.

For the backend, we are using Flask app (Python) — a POST request with the image as the payload would return us the 4 points of the bounding box and the class (or letter) of the image. On the client side, we are using Javascript to capture the users' webcam with the browser's getUserMedia method and using an invisible canvas we took a frame from the video every 2000ms, request a prediction, and display the results accordingly.

8. Experiments

1. Bounding Box detector: Resnet model trained on the Amazon Turks dataset to automatically detect bounding box around the sign in the image. Experiment with Data Augmentation and fine tuning the number of layers in the model, loss, learning rate and so on.
2. CNN classification: Built a Sequential model with Relu activations, dropouts and batch normalization. Experiments with different values for these hyper-parameters.
3. Tried various values of learning rates and epochs to improve the performance of the model
4. Added more data to the training set to improve accuracy of the model.

9. Conclusion

In the project, we were able to successfully build a CNN model that would recognize the sign language being displayed in the video and emit a caption that corresponds to that sign.The model gave an accuracy of . We used AusLan dataset to train the model and this can be further enhanced to work on other sign languages like the ASL. We also built a Flask application with Python that could capture a live video and call the model to make the prediction.

10. References

- Model based on sign language detection
  https://medium.com/@coviu/how-we-used-ai-to-translate-sign-language-in-real-time-782238ed6bf
- CM92
  C. Charayaphan and A. Marble. Image Processing system for interpreting motion in American Sign Language. Journal of Biomedical Engineering, 14:419--425, September 1992
- DS93
  James Davis and Mubarak Shah. Gesture recognition.Technical Report CS-TR-93-11, University of Central Florida, 1993