Sign Language Recognition

By,

Rimzim Thube

Shree Gowri Radhakrishna

Vijaylaxmi Nagurkar

Objective

- Sign Language used by people with hearing disability to communicate with others
- Difficult to communicate if the language is not understood
- Creates a barrier for hearing disabled to lead a normal life
- To improve their quality of life, develop an application for gesture recognition in real time to communicate
- Used latest Deep Learning models
- Can be used in mobile devices

Dataset

- No ready dataset available for Auslan language
- AUSLAN dataset generated from frames of YouTube videos demonstrating the hand gestures
- Images are annotated with their class labels
- Coordinates of the bounding box covering the gesture
- Images are crowdsourced on Amazon DataTurk for labelling

Data preparation: cleaning, preprocessing

- Frames taken from video showing AusLan sign language
- From Amazon dataturks
- Annotated JSON files containing the image URL, height, width, bounding box vertices
- Save the images in local storage and the metadata in text file to run bounding box regresser

Bounding Box

- Used YOLO algorithm for object detection
- Built a Resnet model with Data Augmentation
- trained, tuned and tested
- Regresser able to detect the bounding box of newly captured video

Methods

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

1) We gathered data
2) Trained a machine learning model by providing it a set of images to recognise the Auslan Alphabet.
3) Post training , we saved the model
4) Build the user interface i.e the client and the application side using the pretrained model.

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 ask 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.

Model Architecture

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

ResNet-101

- convolutional neural network that is 101 layers deep
- trained on more than a million images from the ImageNet database
- pretrained network can classify images into 1000 object categories
- image input size of 224-by-224

Model Architecture

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

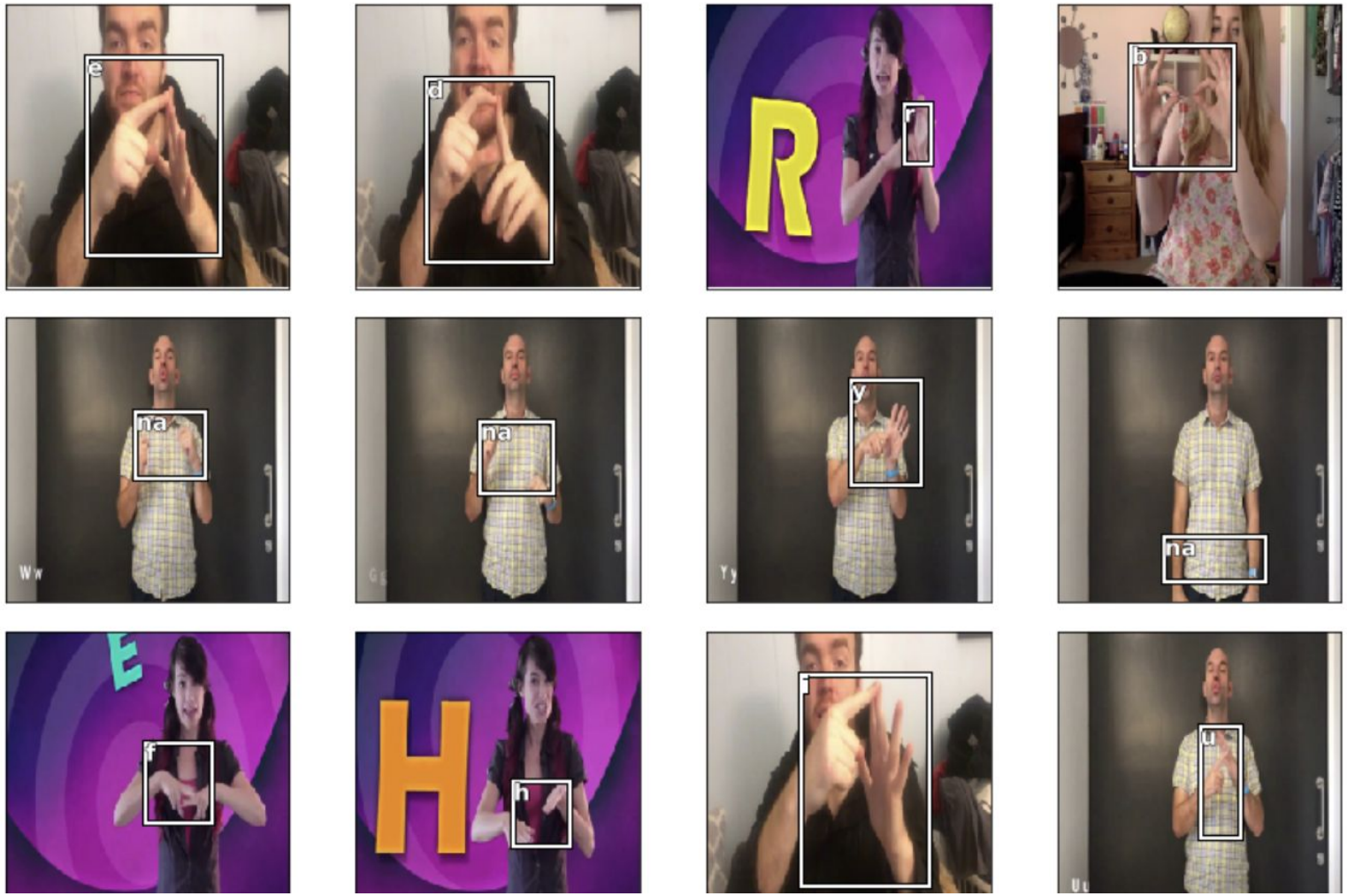
Customised the Resnet model using ConvnetBuilder

```
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)
```

Model Training

- 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.
- CNN classification: Built a Sequential model with Relu activations, dropouts and batch normalization. Experiments with different values for these hyper-parameters.
- Tried various values of learning rates and epochs to improve the performance of the model
- Added more data to the training set to improve accuracy of the model.

Trained model used to test the captions on the images



Metrics of the model

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%

```
In [0]: learn.fit(lrs, 1, cycle_len=2)

        HBox(children=(IntProgress(value=0, description='Epoch', max=2), HTML(value='')))

        epoch      trn_loss   val_loss   detn_acc   detn_l1
            0      40.285558  27.252446  0.851211   9.296417
            1      36.002267  26.065082  0.858131   9.301405

Out[0]: [array([26.06508]), 0.8581314878892734, 9.301405230195465]
```

Existing Model

Model based on sign language detection

https://medium.com/@coviu/how-we-used-ai-to-translate-sign-language-in-real-time-782238ed6bf

Conclusion

- Built a CNN Resnet model to detect sign language
- Trained on AusLan dataset
- Inference tests to validate the model
- Flask end-to-end application to capture live video and predict on the saved model
- Can be extended to other Sign Languages like ASL

Thank you