Real-Time Sign Language Recognition

Deep Learning project

Zubin Bhuyan
(01744486)
zubin_bhuyan@student.uml.edu

Outline

- Introduction
- Motivation
- Related work
- Proposed approach
- Datasets
- Implementation & Results

Introduction

- Sign language:
  - sequence of *structured* handshakes and gestures.
- Hundreds of sign languages
- *American Sign Language* (ASL)
  - Primary language for at least 350, 000 people

Introduction

- American sign language

Introduction

- Sign Language Recognition (SLR)
  - Translating visual signals to text/voice, preferably in real time.

- ASL is a *natural language*.
  - *not universal* and they are *not mutually intelligible.*
  - Hand-shapes, gestures, and even facial expressions

Motivation

Motivation

- SLR can be an alternative method of voice input for deaf people.
  - Improved accessibility- input interface to ICT devices
- Enable easy communication between deaf and hearing individuals.
- Automatic translation of video and audio.
- Facilitate further research of SL linguistics.
  - Automated annotation and meta-data generation.

Related work

Related work: Approaches

- Two broad approaches:
  - Vision-based
  - Sensor-based

Related work: Approaches

- Two broad approaches:
  - Vision-based
    - Single/stereo camera.
    - Structured light projection, eg.- Kinect, LeapMotion.
    - Body markers: Colored gloves, LED, etc.
  - Sensor-based

Related work: Approaches

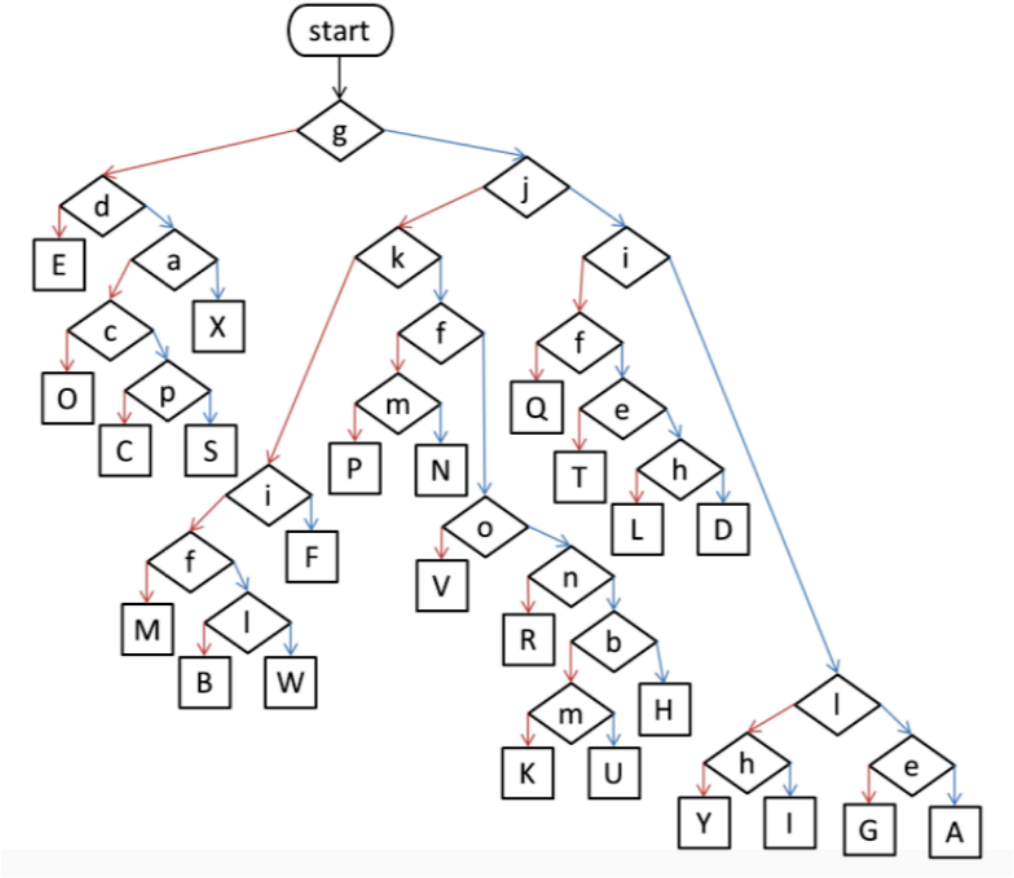
- Two broad approaches:
  - Vision-based
    - Single/stereo camera.
    - Structured light projection, eg.- Kinect, LeapMotion.
    - Body markers: Colored gloves, LED, etc.
  - Sensor-based
    - Intertial measurement units: gyroscopes, accelerometers, etc.
    - EMG
    - WiFi, broad beam RADAR, etc.

Related Work

- "*Sign Language Recognition using Leap Motion Controller*", Funasaka et al, 2015.
  - Uses skeletal image of hand from leap motion controller.
  - 16 kinds of decisions for finger-spelling.
  - Conditions for branching include hand orientation, extension of fingers, etc.
  - Genetic algorithm approach used for ordering decisions.

Related Work

- "*Sign Language Recognition using Leap Motion Controller*", Funasaka et al, 2015.



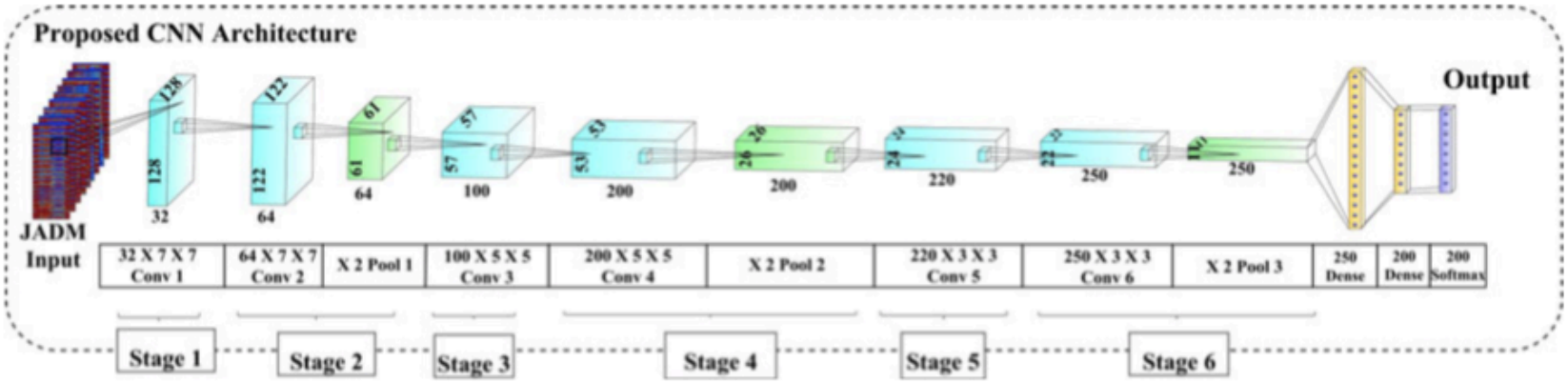
Related Work

- "*Training CNNs for 3-D Sign Language Recognition With Color Texture Coded Joint Angular Displacement Maps*", Kumar et al, IEEE Signal Processing Letters, 2018.

- Uses Joint Angular Displacement Maps (JADM): color-texture image.
  - A video of 3-D skeletal hand image with J joints has J(J-1)/ 2*T joint pairs, T being the number of frames.
  - These joints are projected in 3-D space and joint distance/ angle values encoded in RGB images.

Related Work

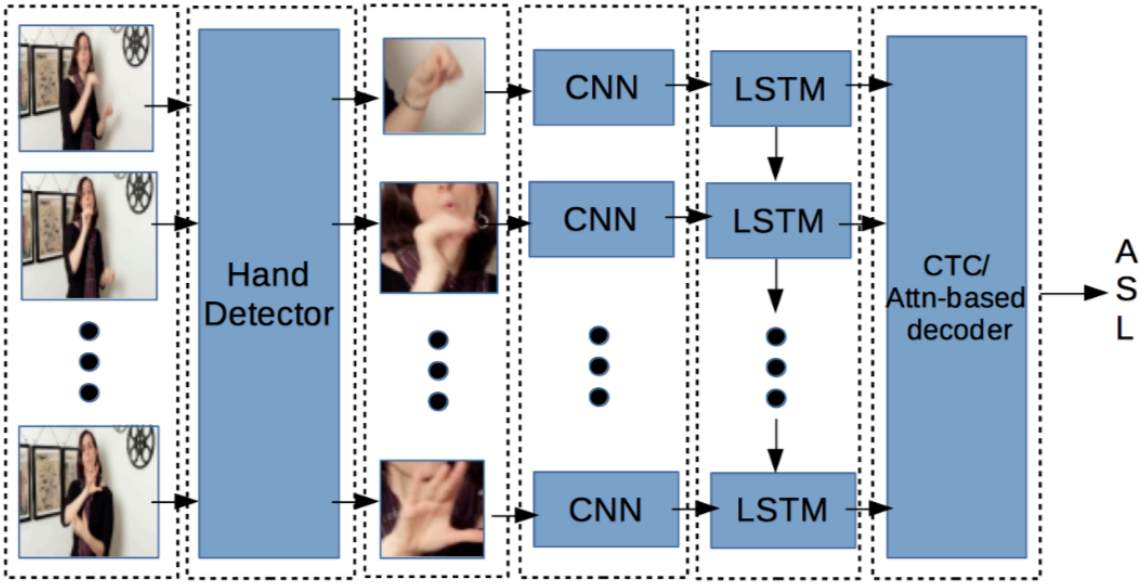
- "*Training CNNs for 3-D Sign Language Recognition With Color Texture Coded Joint Angular Displacement Maps*", Kumar et al, IEEE Signal Processing Letters, 2018.



- Input images were 128x128.
- Tested on 3 datasets.

Related Work

- "*American sign language finger spelling recognition in the wild*", Shi et al, IEEE Spoken Language Technology Workshop, 2018.
- Finger spelling recognition from videos available in websites.
  - Video quality and frame rate variability.



  - Attention-based Encoder-Decoder Recurrent Neural Network.

Proposed Approach

Proposed Approach

- Detect hand sign character in 2 stages:
  - Detect hands (and location) in frame.
  - Check if detected hands conform to a known sign.

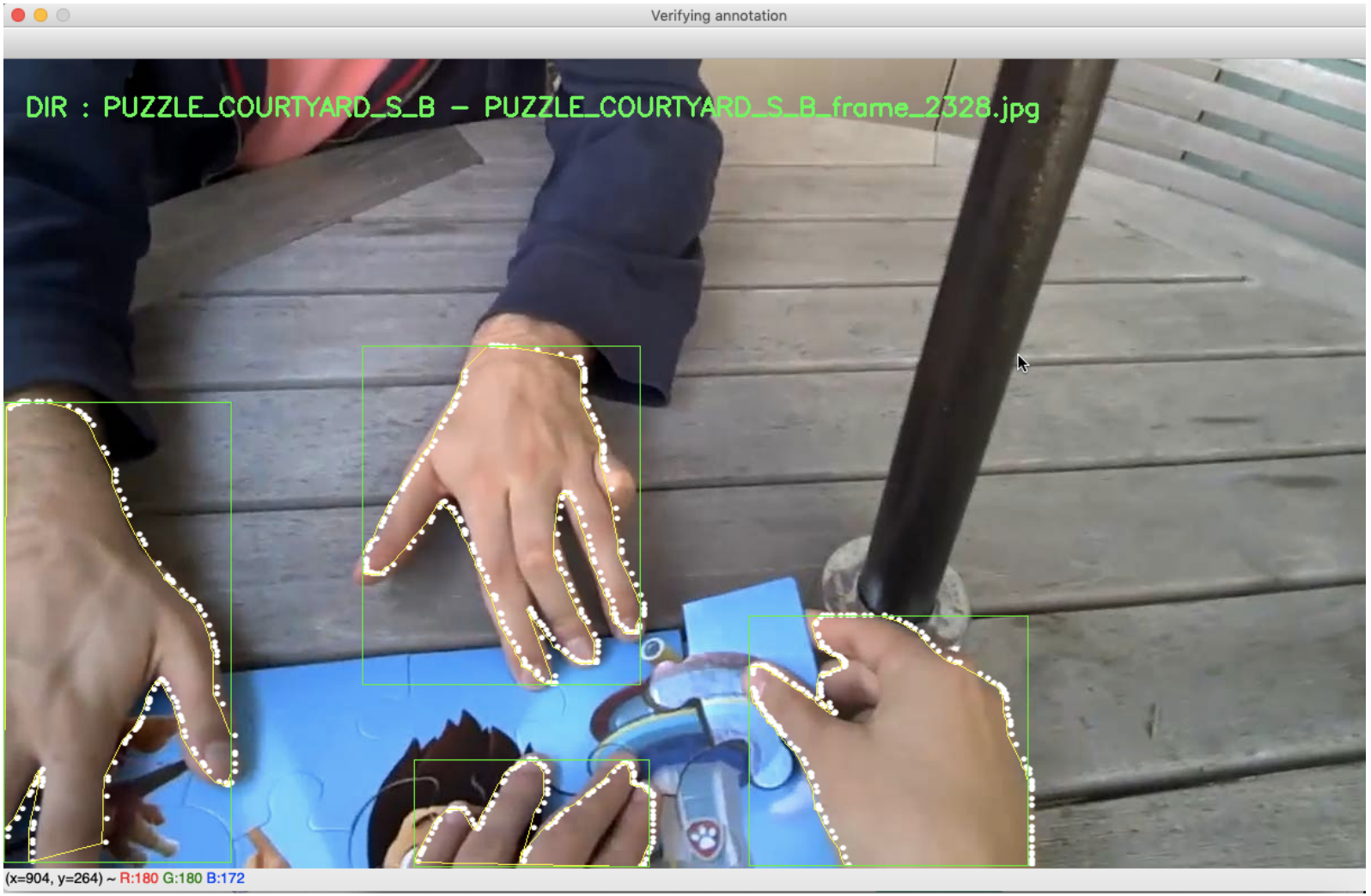
Datasets

Dataset: Egohand†

- 4,800 JPEG images of hand poses (from video of people performing tasks).
- Pixel-level segmentations of hands.
- 15,053 ground-truth labeled hands.

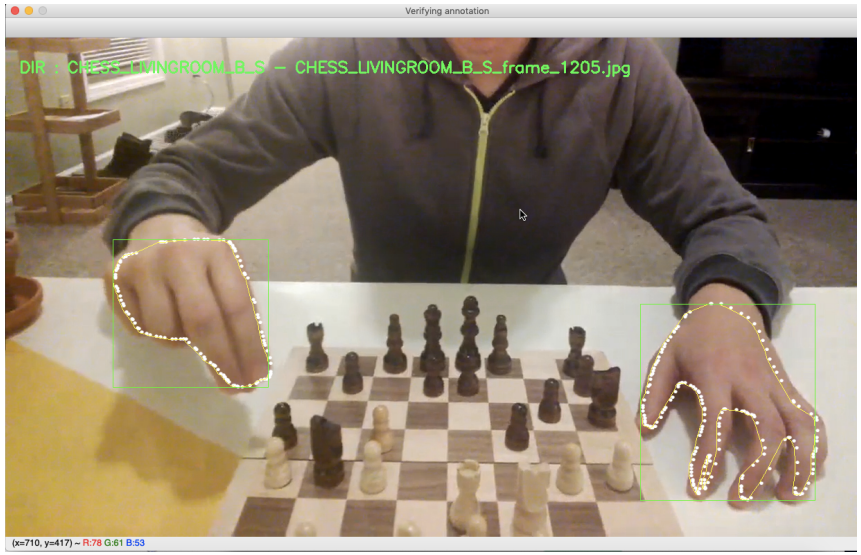
†Bambach, et al, Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions, 2015
http://vision.soic.indiana.edu/projects/egohands/

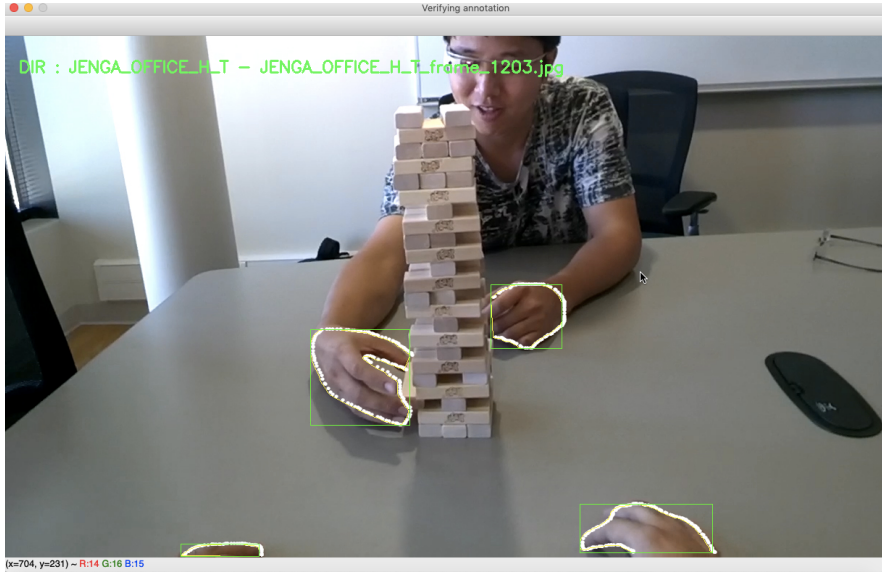
Dataset: Egohand†

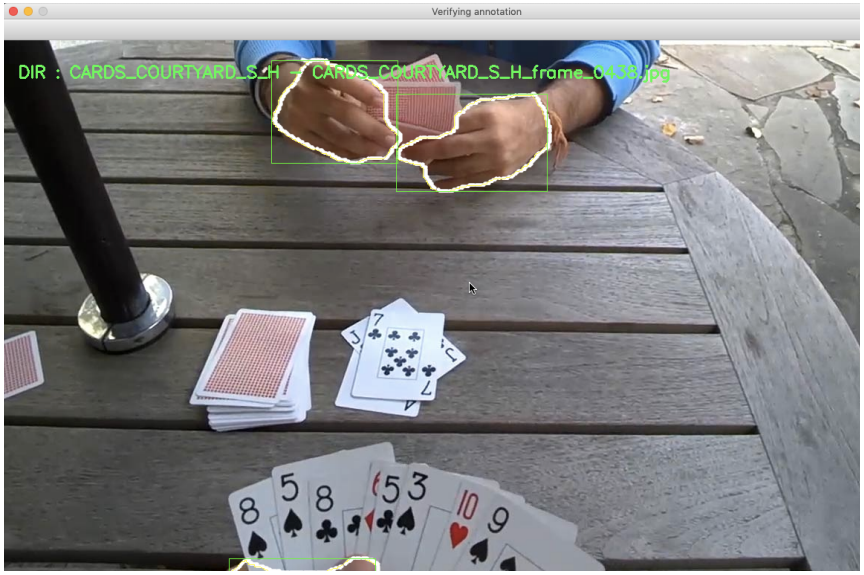


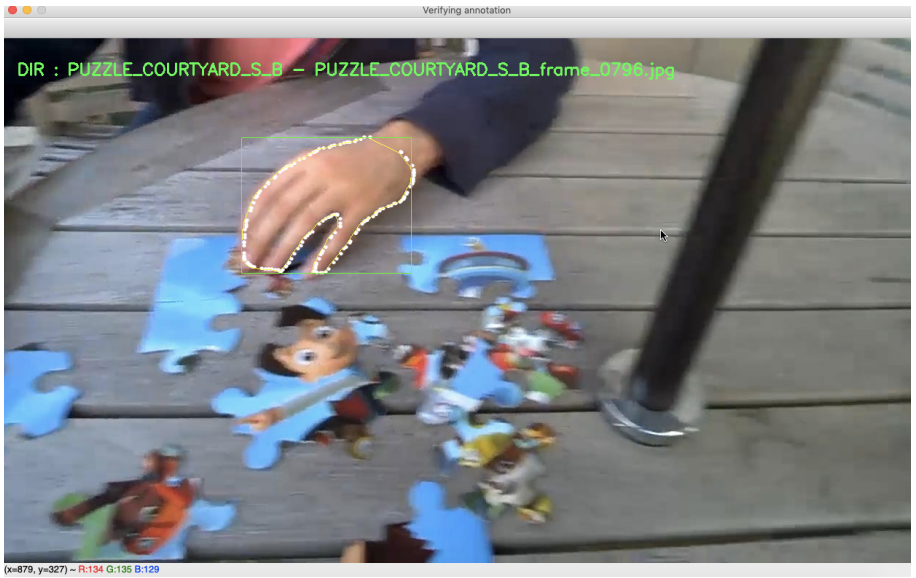
†Bambach, et al, Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions, 2015
http://vision.soic.indiana.edu/projects/egohands/

Dataset: Egohand†



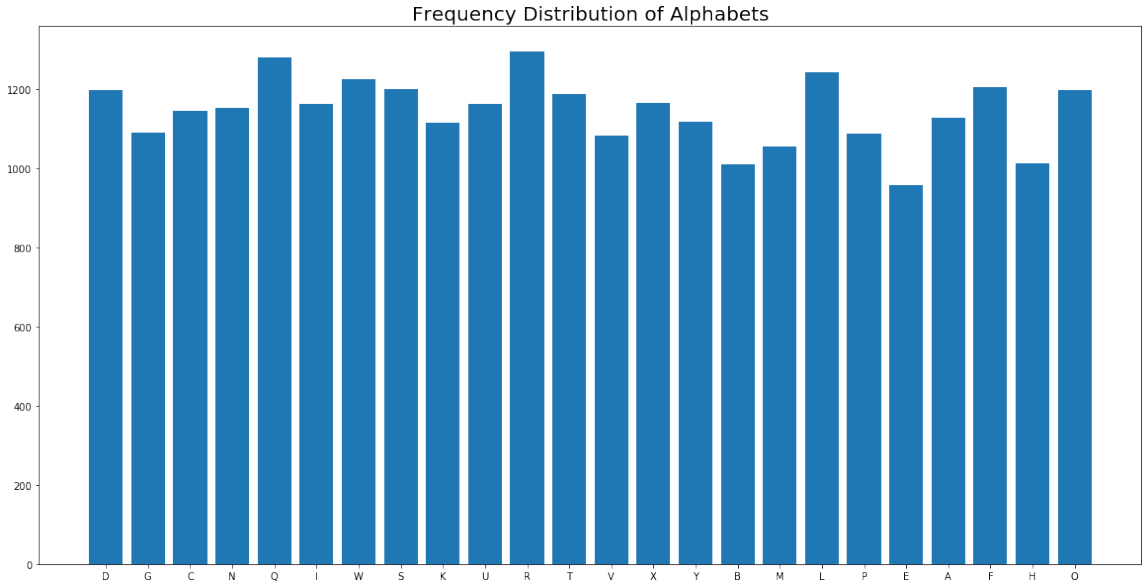






Dataset: MNIST

- Sign Language MNIST†
- 24 classes
- Train data: Labeled pixel data in CSV for 28x28 grayscale (0-255) values, 27,455 labeled cases
- Test data: 7172 labeled cases.



†https://www.kaggle.com/datamunge/sign-language-mnist

Dataset: MNIST

Implementation

Implementation

- Detection steps:
  - Get bounding box of hands in image/frame
  - Detect hand sign (if any)

Implementation: Transfer Learning

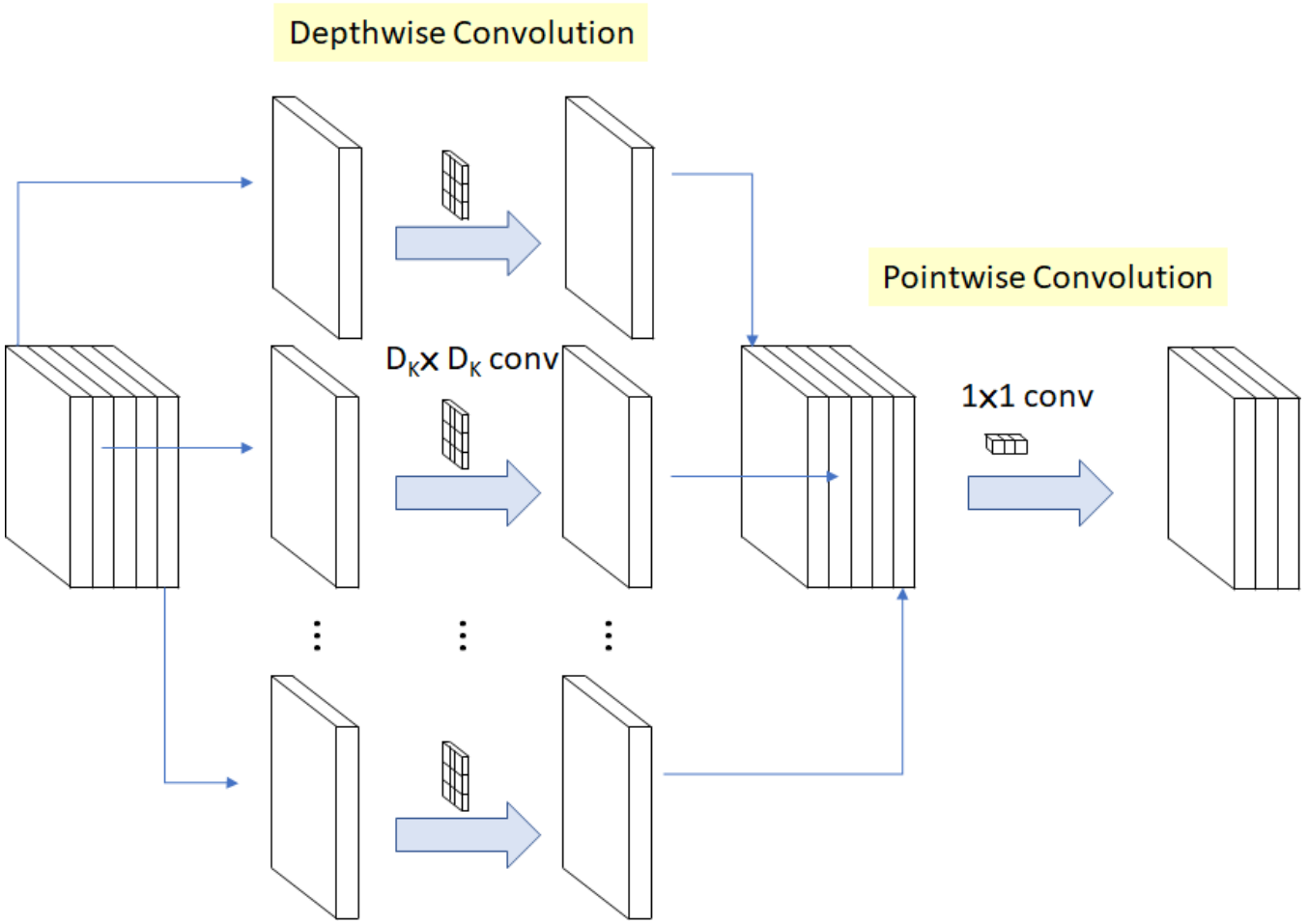
- *Transfer learning: model developed for a task is reused as the starting point for a model on a different task.*

Detecting hands in frame/image:

- Tensorflow detection model zoo [link]:
  - Use SSD mobile net v1
    - Model already pre-tained on COCO.

Implementation: Mobile Net v1

- Depthwise Separable Convolution:

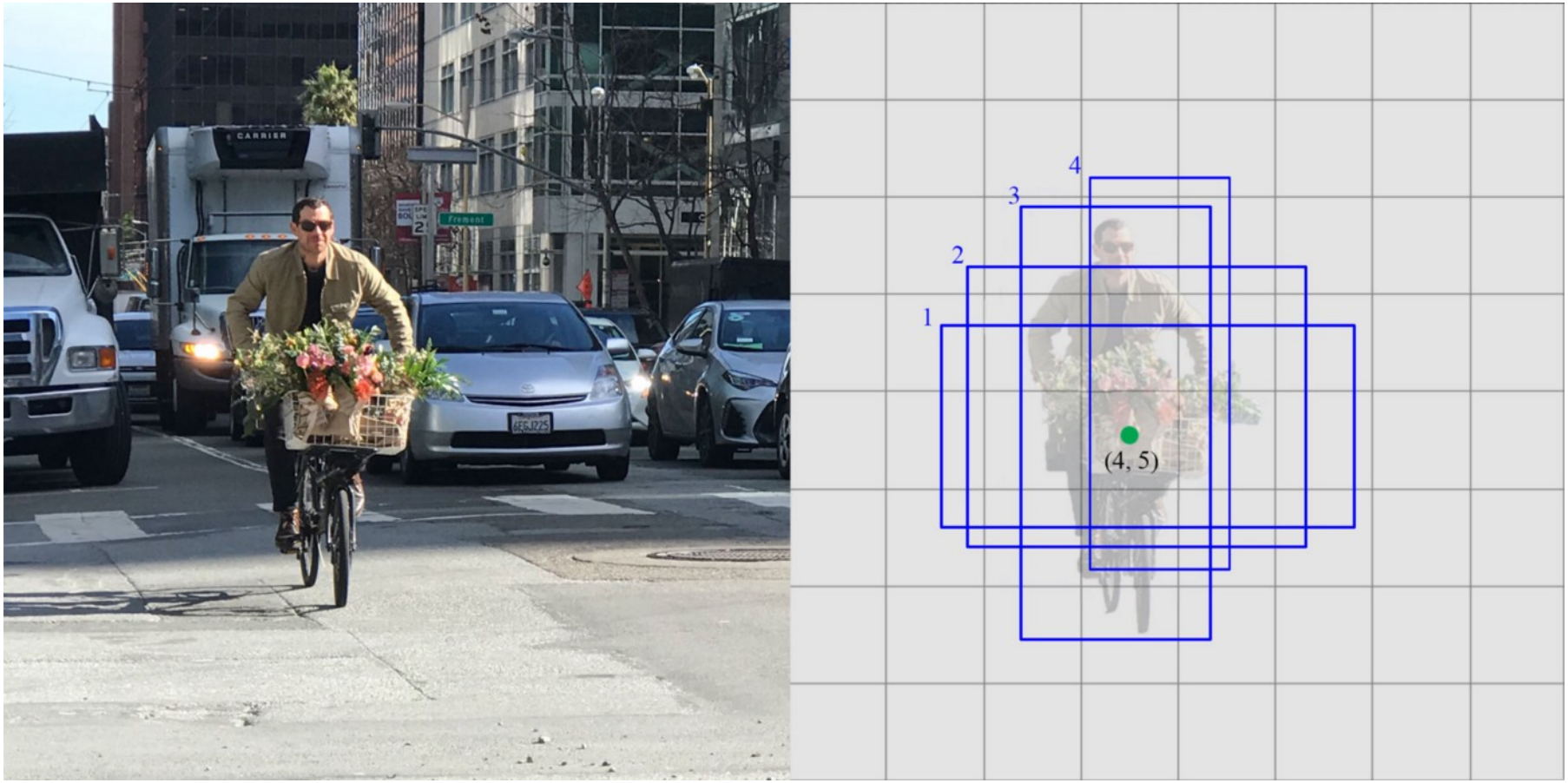


Implementation: Mobile Net v1

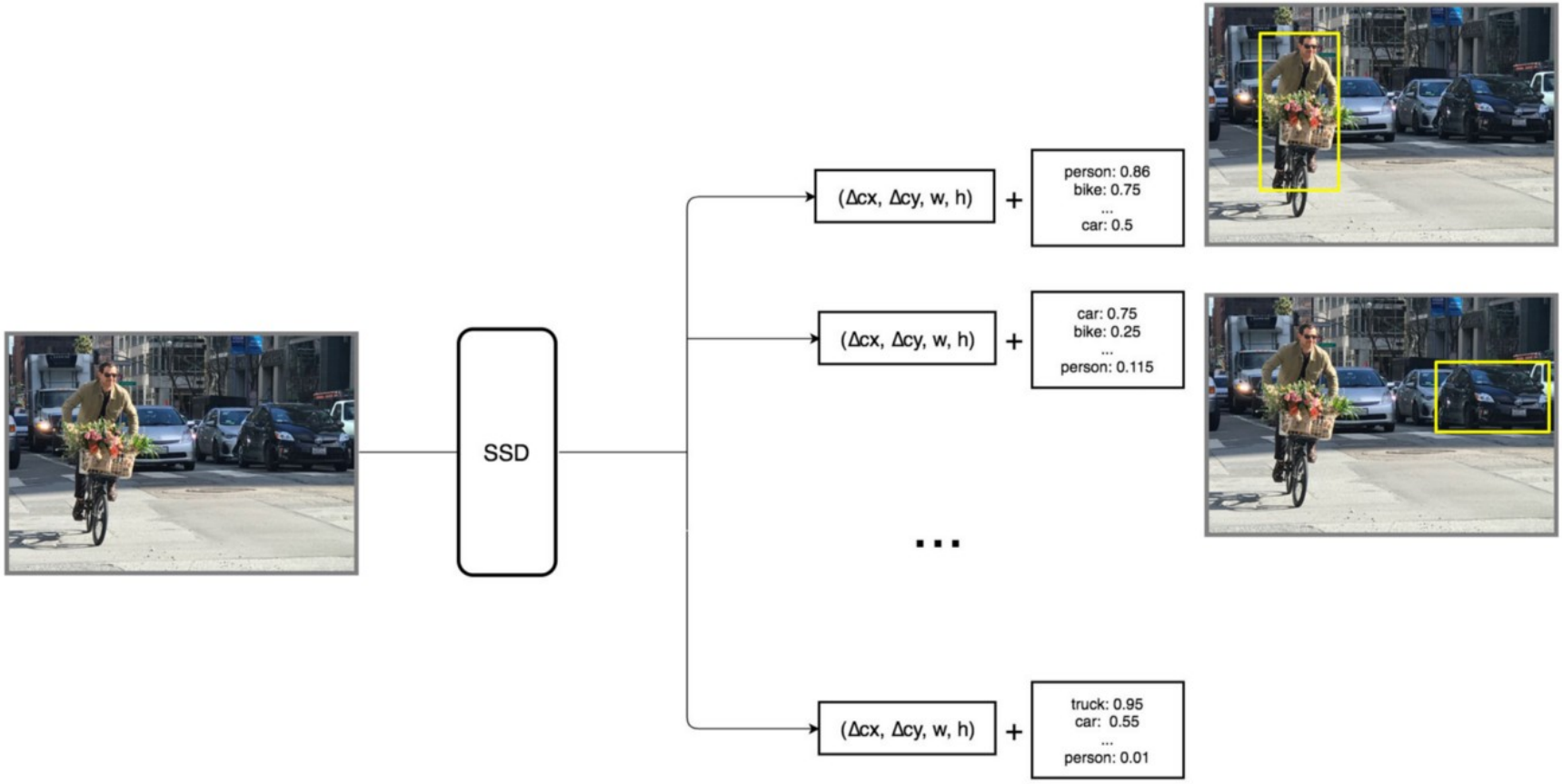
Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Implementation: Mobile Net v1 + SSD



Implementation: Mobile Net v1 + SSD

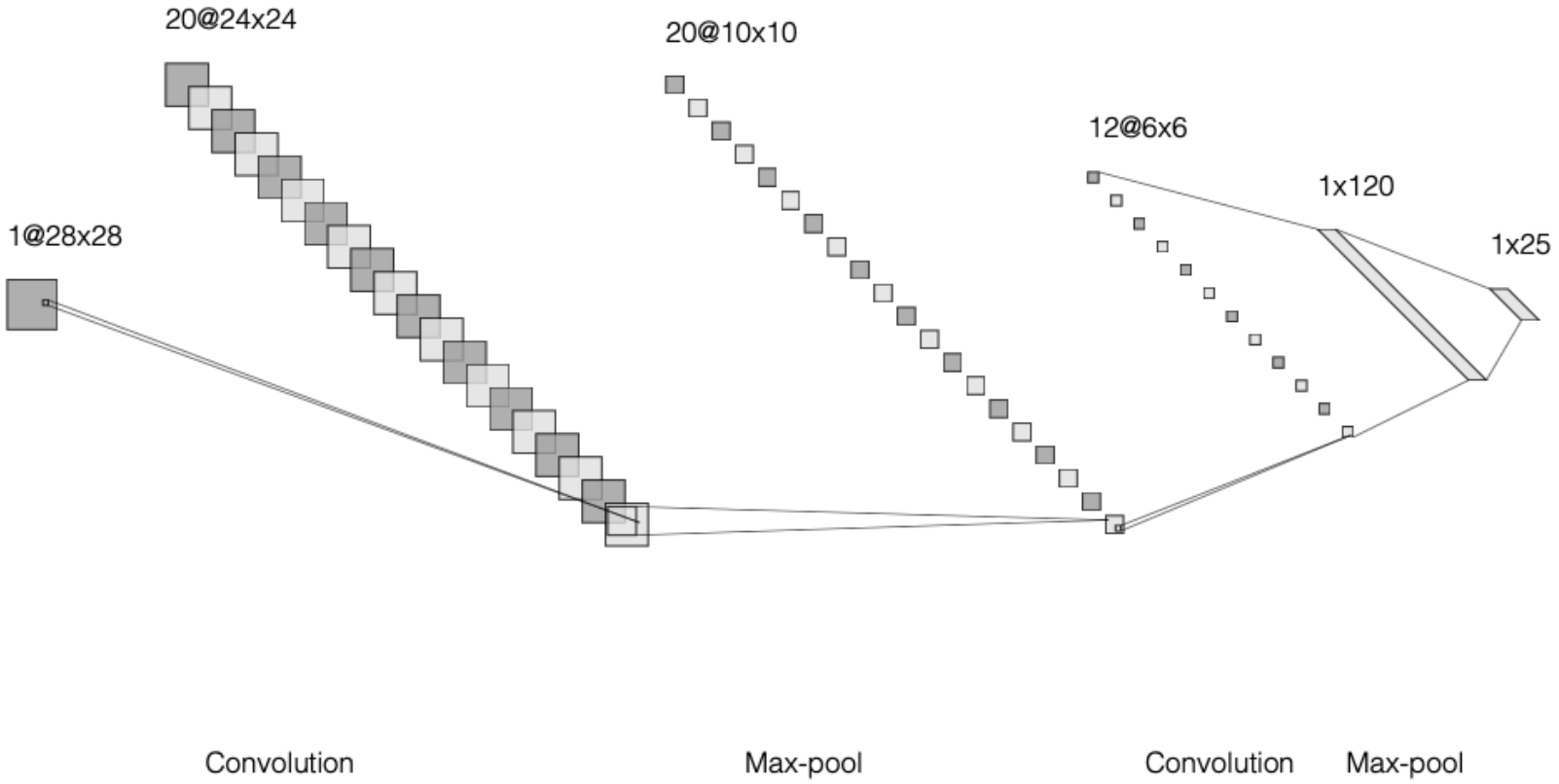


Implementation: CNN

- Task: ***recognize finger spelling hand-shape***.
- CNN to recognize finger spelling hand-shape.
- 2 convolution (stride 1) and 2 max pool layers.
- Batch train = 5
- Learning rate = 0.001
- Momentum = 0.2

Implementation: CNN

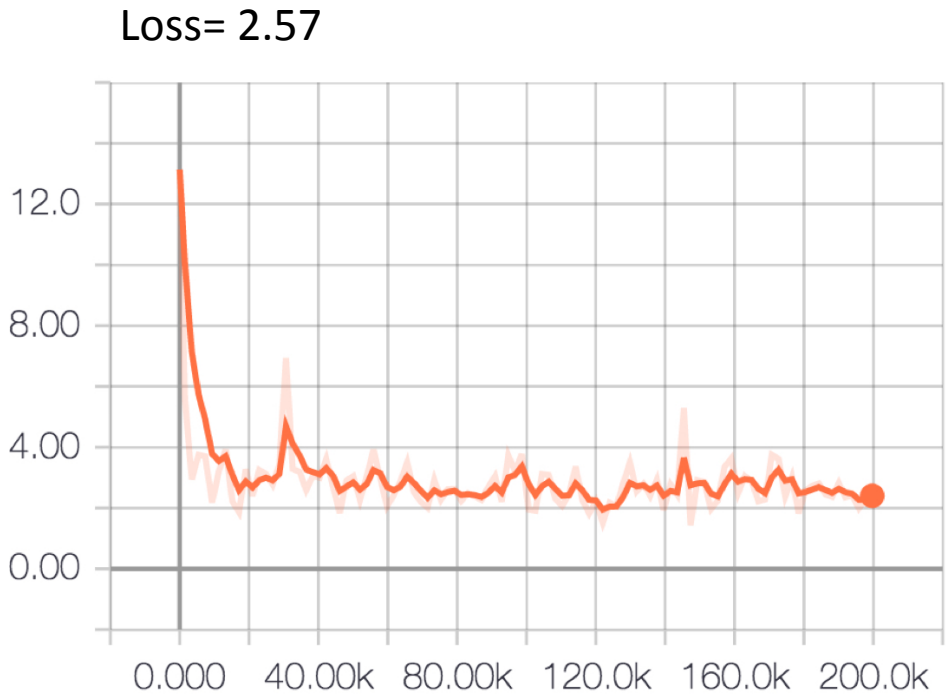
- Task: ***recognize finger spelling hand-shape***.
- Use a CNN to recognize finger spelling hand-shape.
-



Results of training MobileNetv1-SSD (on Egohand dataset)

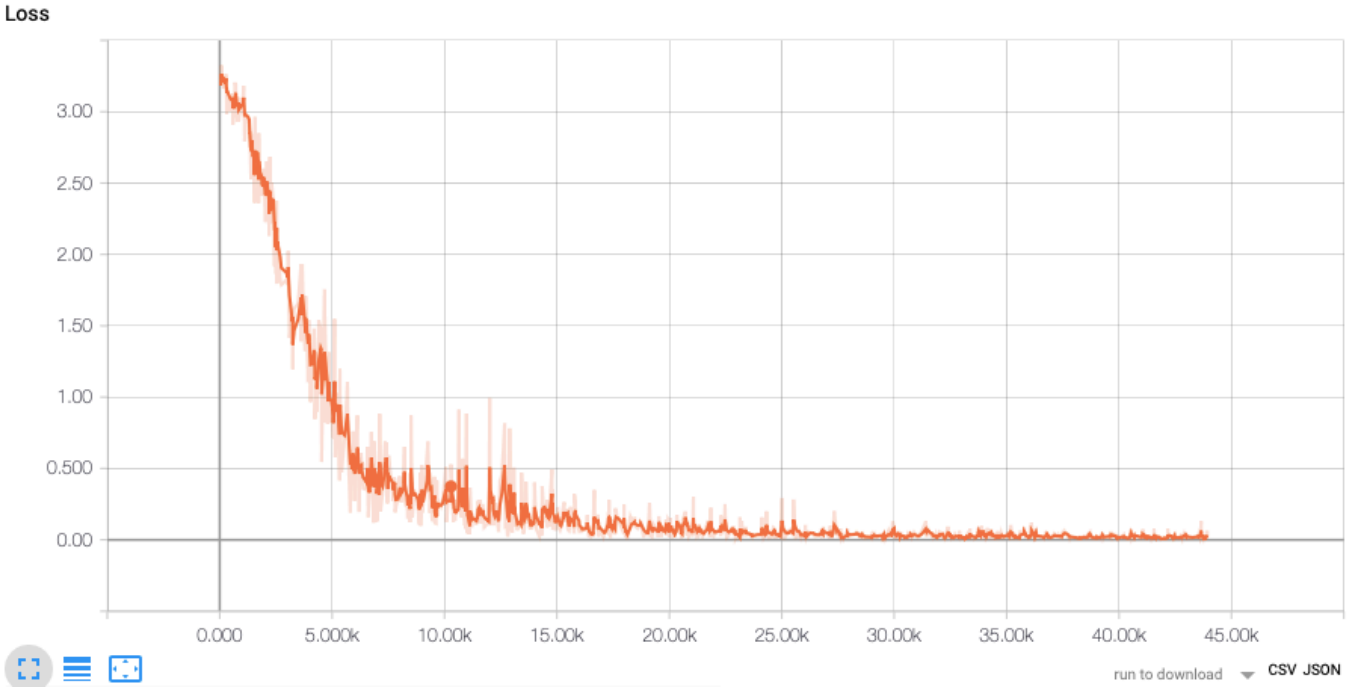
- Precision: what percentage of predictions are correct; how accurate is the precision.
- IoU: Intersection over Union

- AP @ 0.5 IoU = 0.968
- AP @ 0.75 IoU = 0.813
- AP @ 0.5:0.95 IoU = 0.678



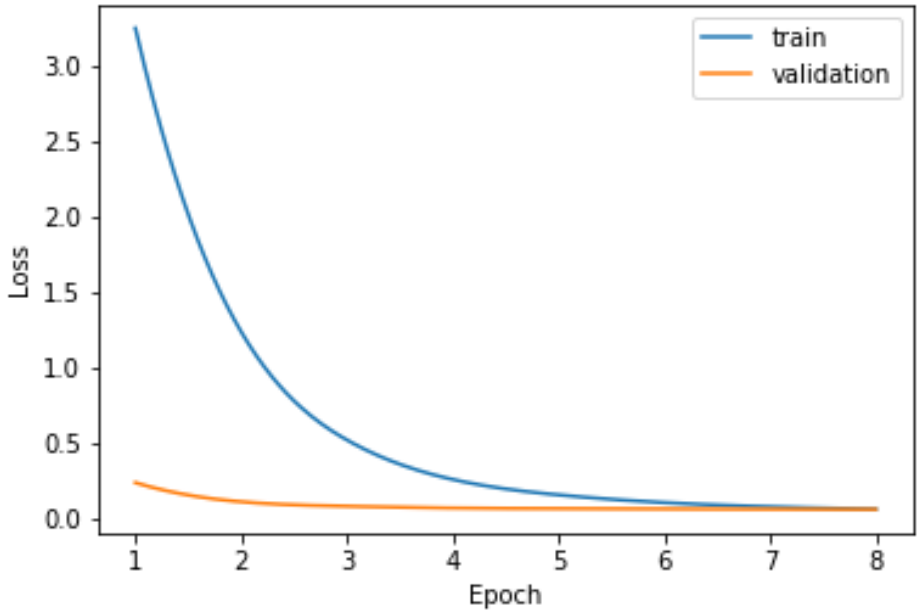
CNN: Results

- CNN to recognize finger spelling hand-shape (MNIST dataset).



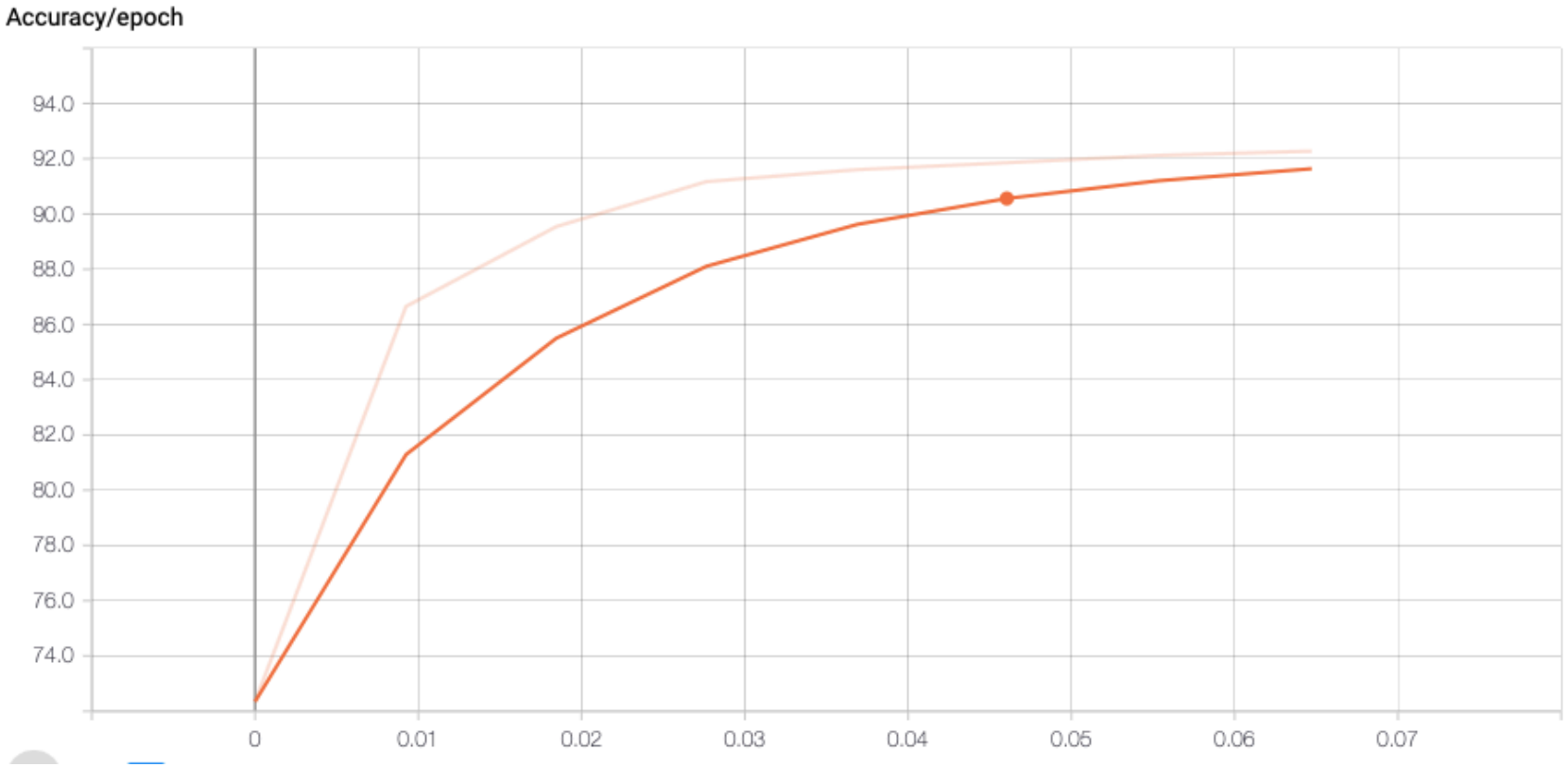
CNN: Results

```
Train Epoch: 1 [0/27455 (0%)]   Loss: 3.255676

Test set: Average loss: 0.2436, Accuracy: 5189/7172 (72%)

Train Epoch: 2 [0/27455 (0%)]   Loss: 1.243511

Test set: Average loss: 0.1168, Accuracy: 6215/7172 (87%)

Train Epoch: 3 [0/27455 (0%)]   Loss: 0.526322

Test set: Average loss: 0.0872, Accuracy: 6422/7172 (90%)

Train Epoch: 4 [0/27455 (0%)]   Loss: 0.264746

Test set: Average loss: 0.0752, Accuracy: 6539/7172 (91%)

Train Epoch: 5 [0/27455 (0%)]   Loss: 0.162337

Test set: Average loss: 0.0715, Accuracy: 6570/7172 (92%)

Train Epoch: 6 [0/27455 (0%)]   Loss: 0.111286

Test set: Average loss: 0.0691, Accuracy: 6588/7172 (92%)

Train Epoch: 7 [0/27455 (0%)]   Loss: 0.083730

Test set: Average loss: 0.0678, Accuracy: 6607/7172 (92%)

Train Epoch: 8 [0/27455 (0%)]   Loss: 0.067452

Test set: Average loss: 0.0669, Accuracy: 6618/7172 (92%)
```



CNN: Results

- Accuracy with MNIST dataset: 92%

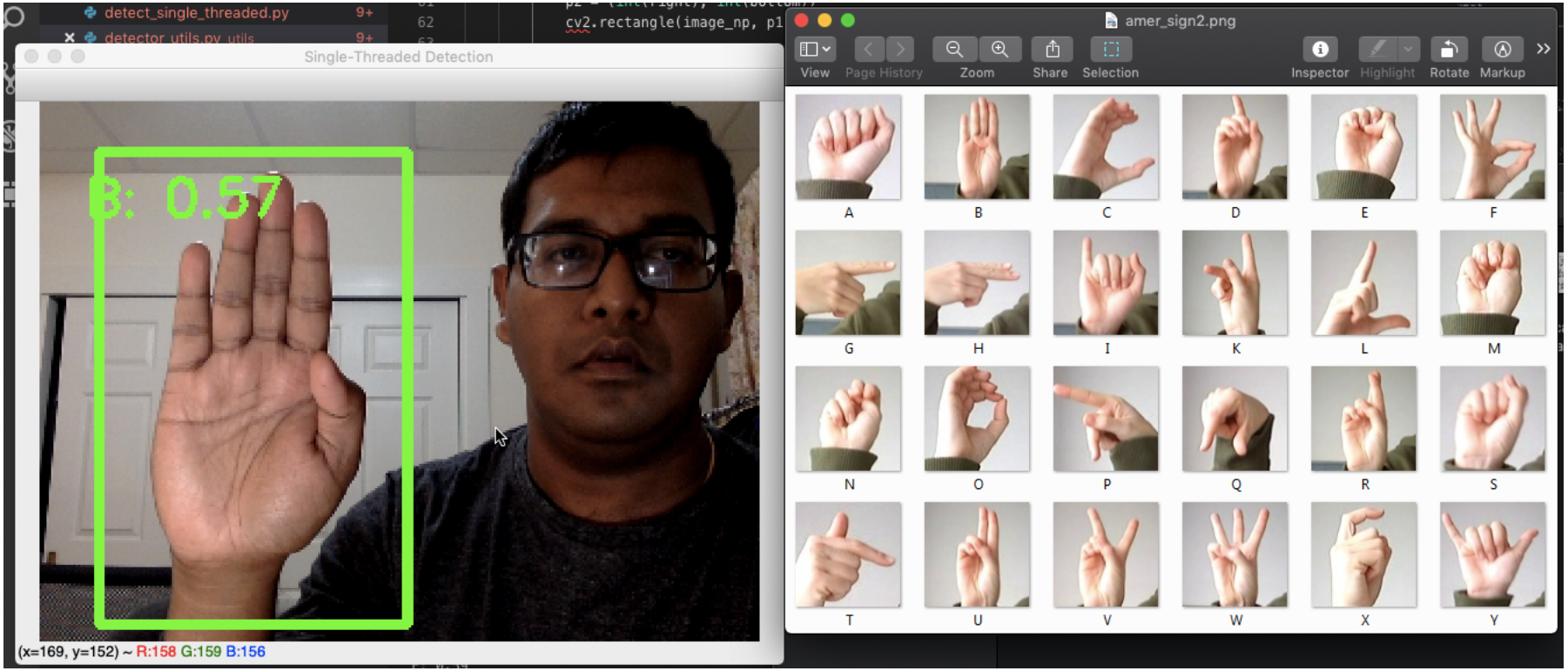


CNN: Accuracy

- Testing with the MNIST test dataset and comparing with other.
- Try out recognition from camera input in real-time for human user.

Realtime implementation

- Open CV for Python.
- Screenshots:

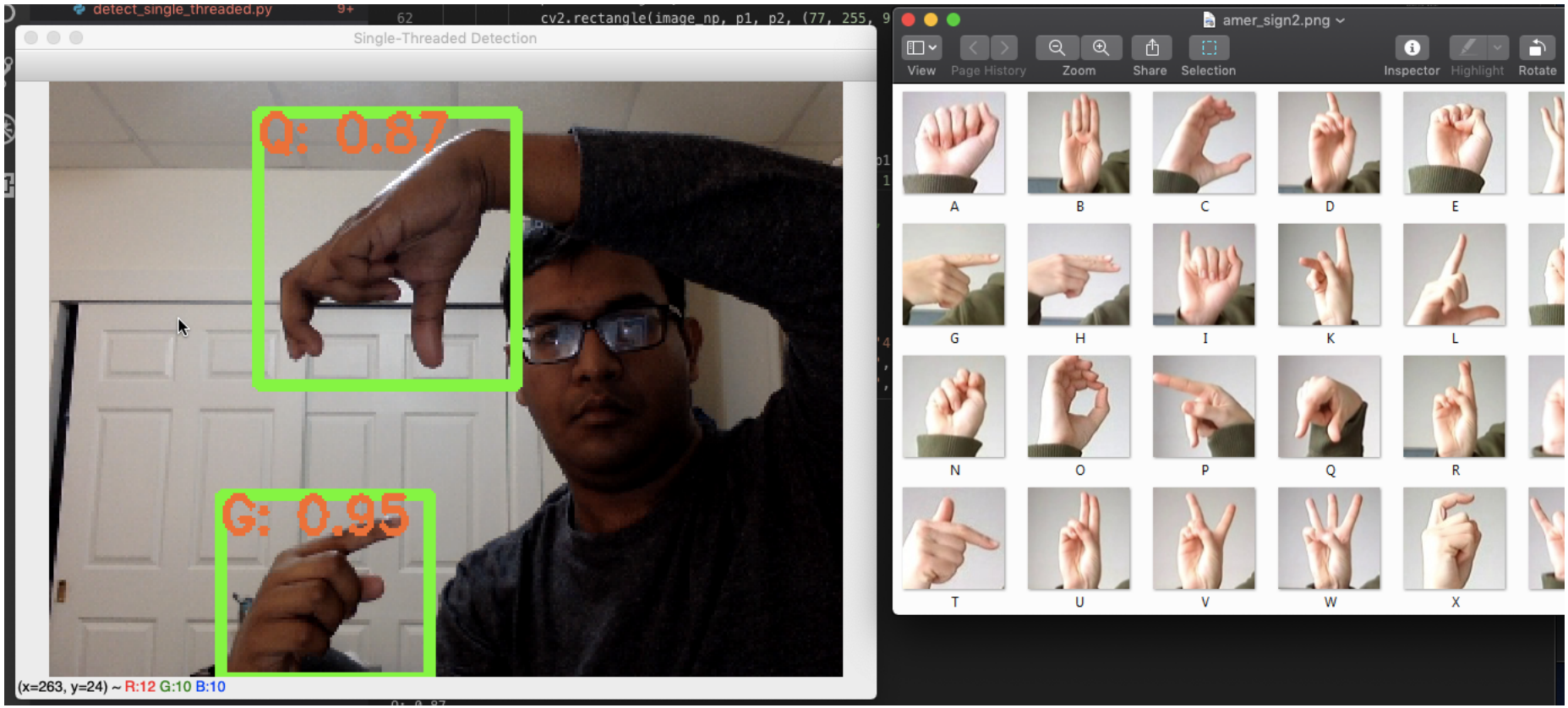


Realtime implementation

- Open CV for Python.
- Screenshots:

Realtime implementation

- Open CV for Python.
- Screenshots (multiple hands):



Thank you.