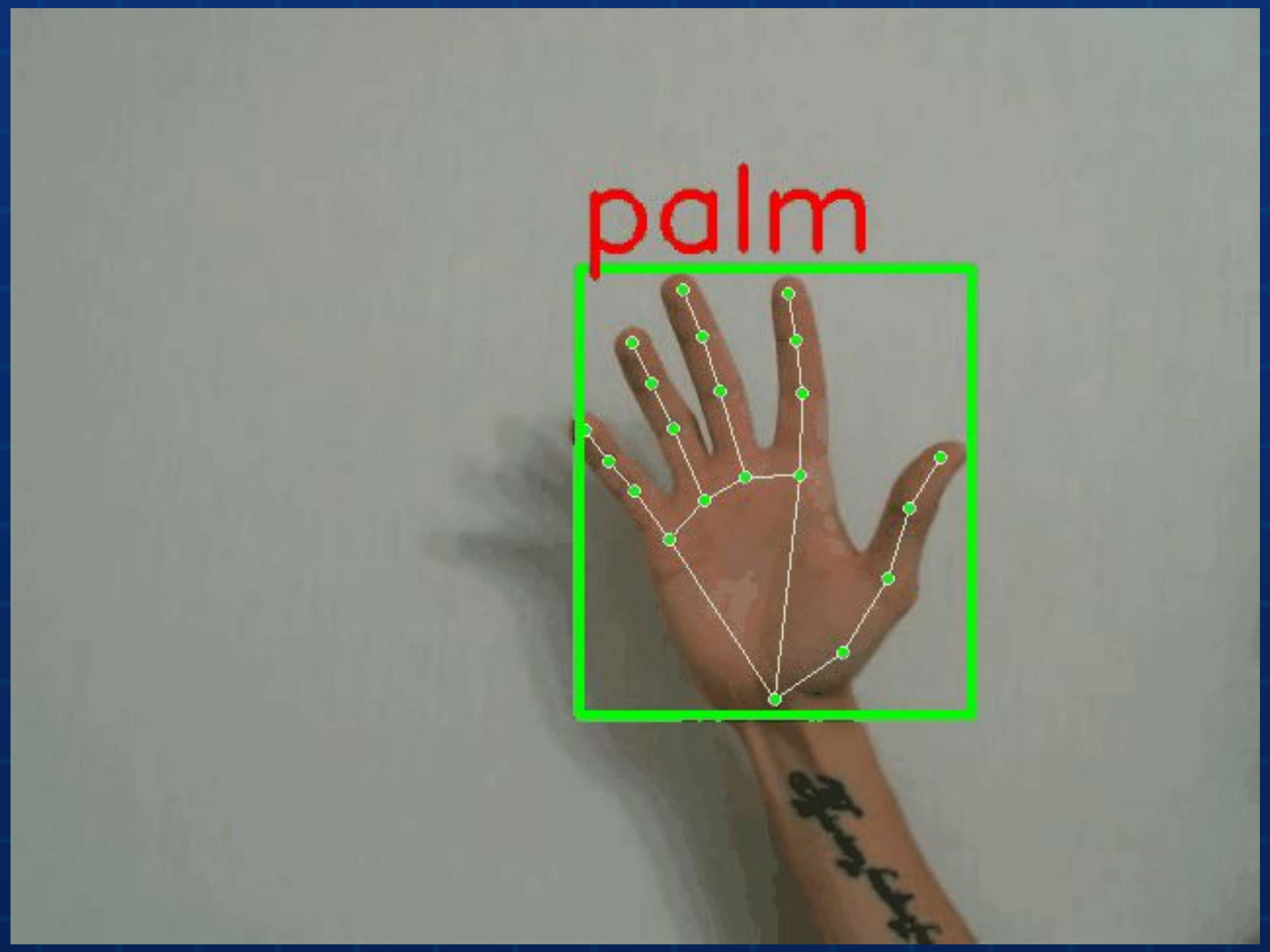
# CS512 - COMPUTER VISION

# Real-time Hand Gesture Recognition



*Source: GitHub-HaGRID dataset*

## Project Presentation

Spring 2023

ILLINOIS INSTITUTE OF TECHNOLOGY

Bastien LEDUC - A20520860
Paul LEGOUT - A20522029

# Problem Statement

# Problem Statement

## Implement AI models for live hand detection and gesture recognition.

Example of applications:

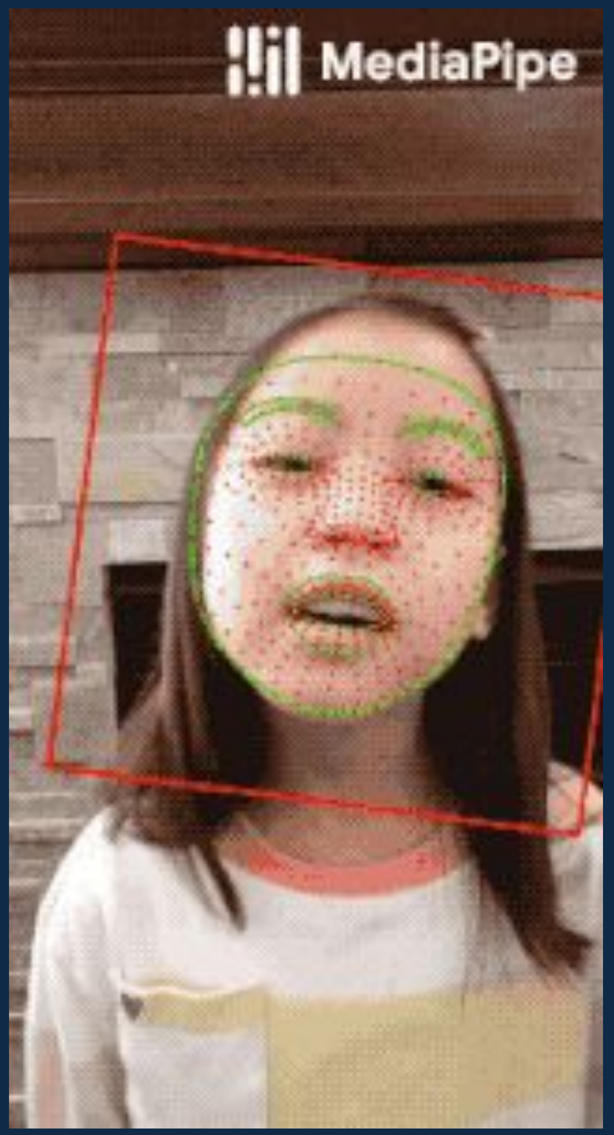
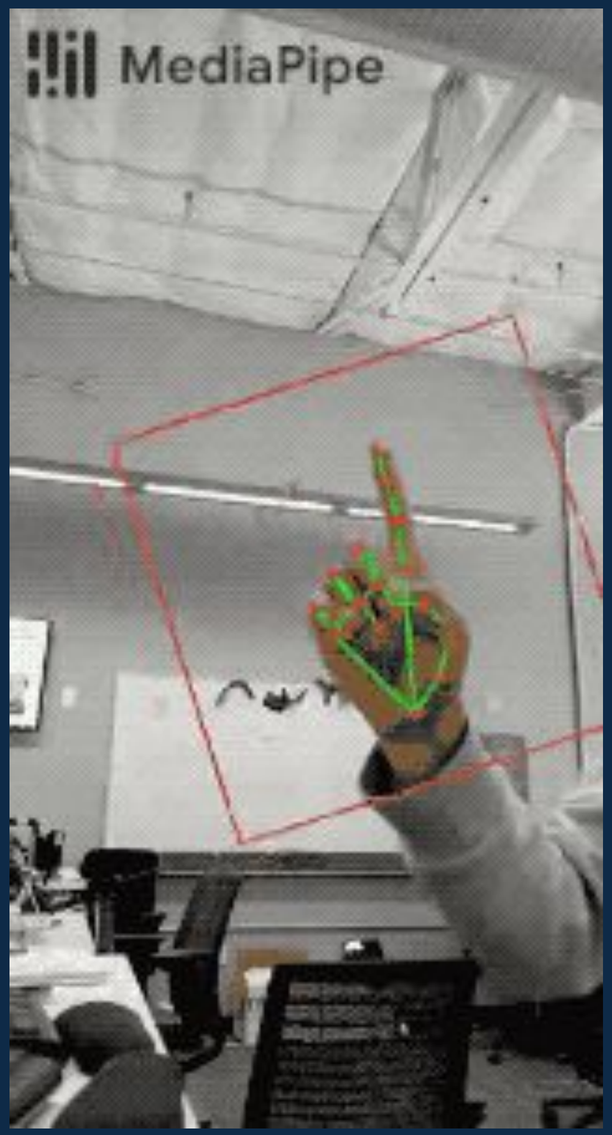
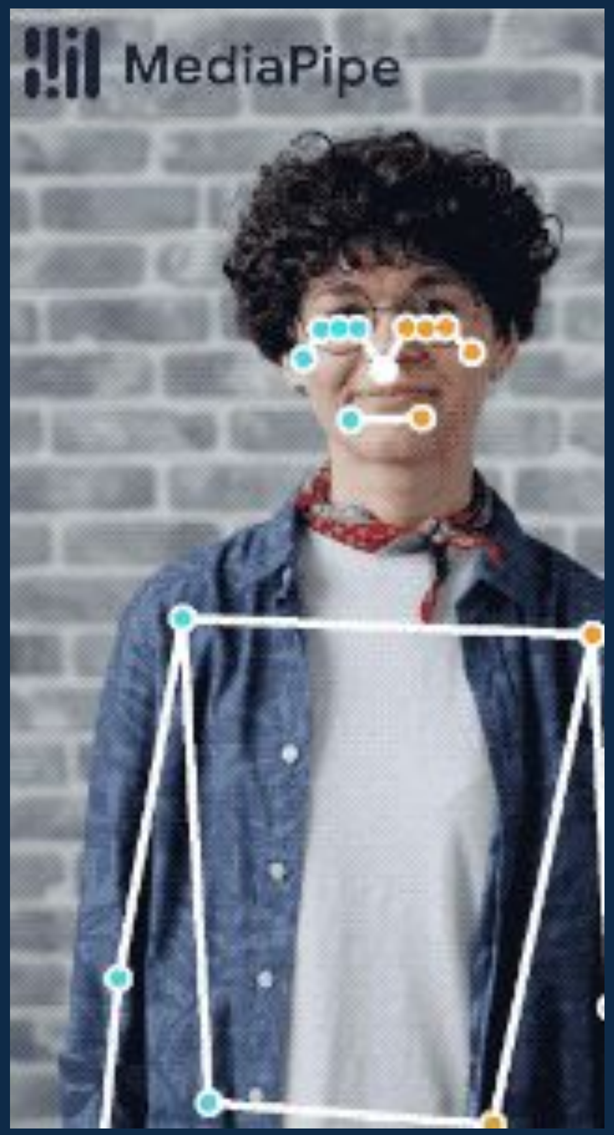
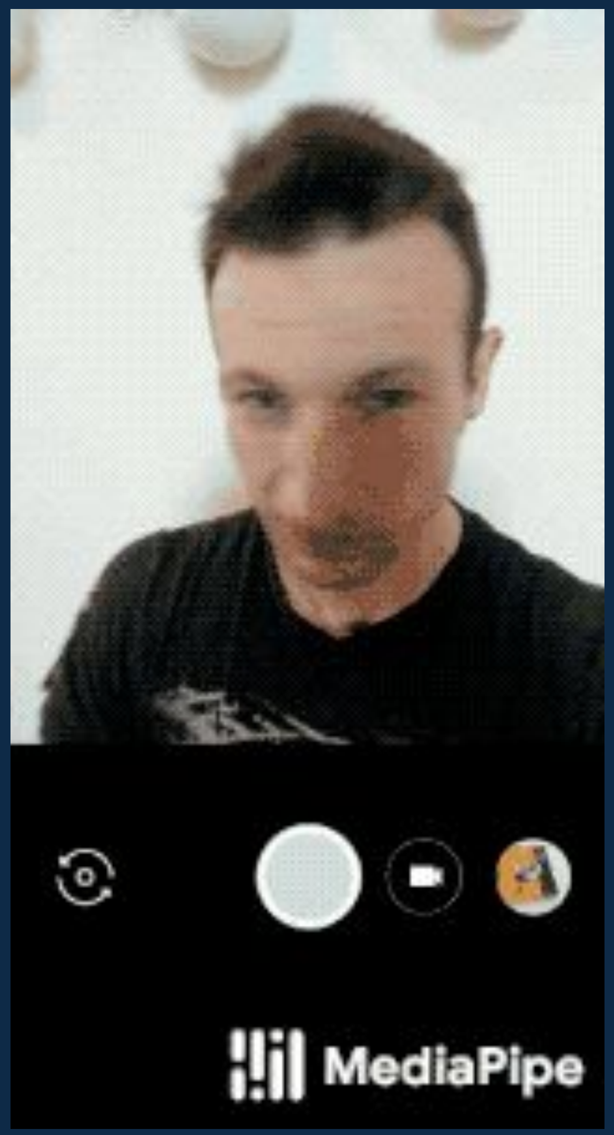
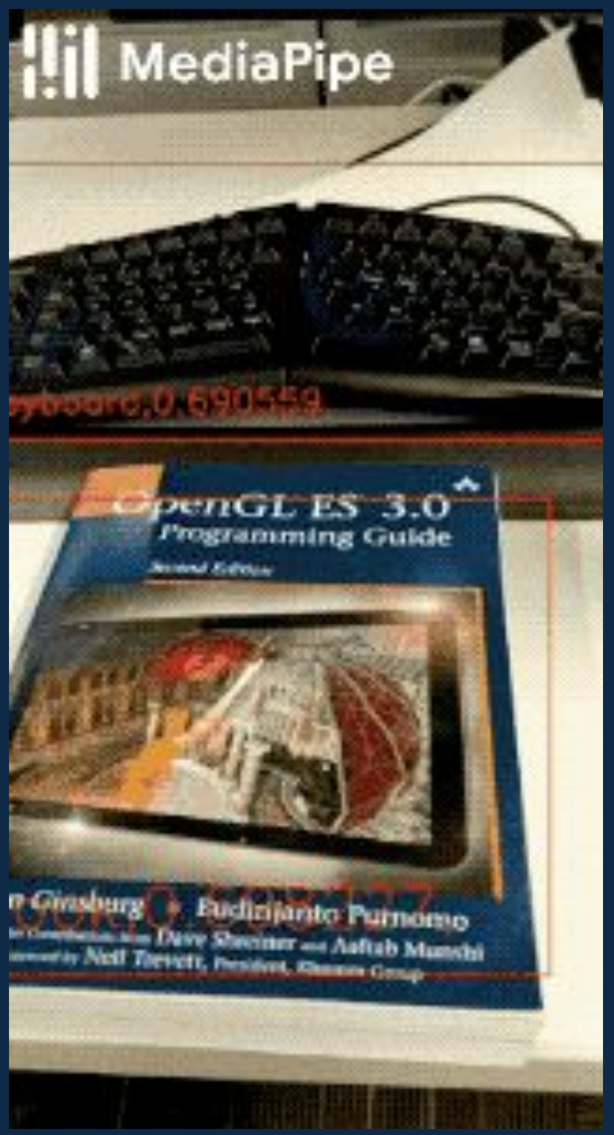
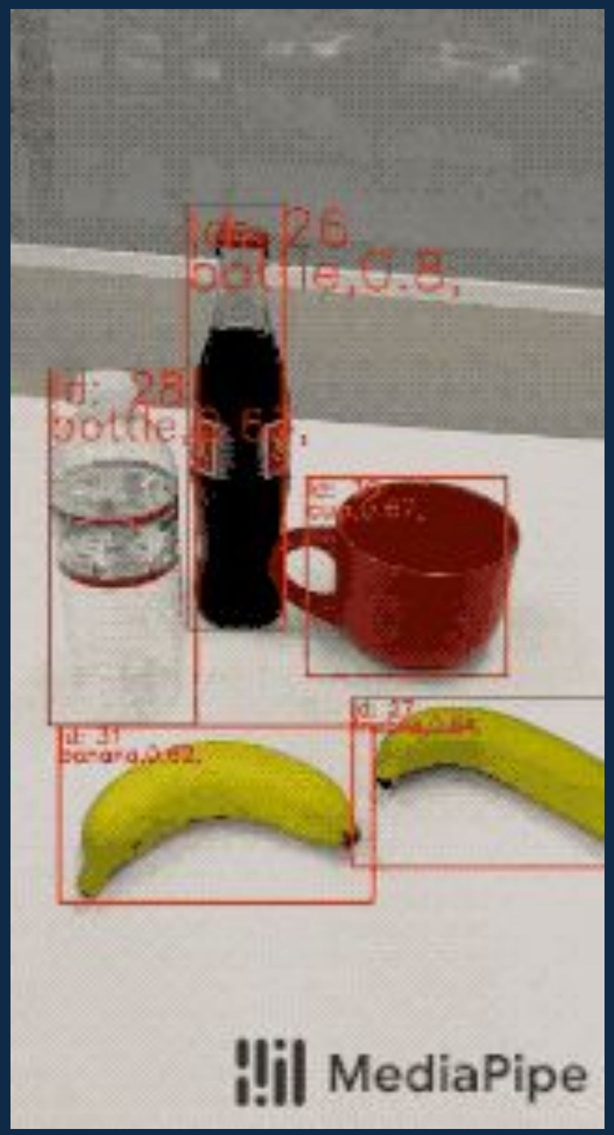
- Video conferencing services
- Home automation systems
- Automotive sector
- Services for people with speech and hearing impairments, etc.
- Human-computer interaction

Paper that we used:

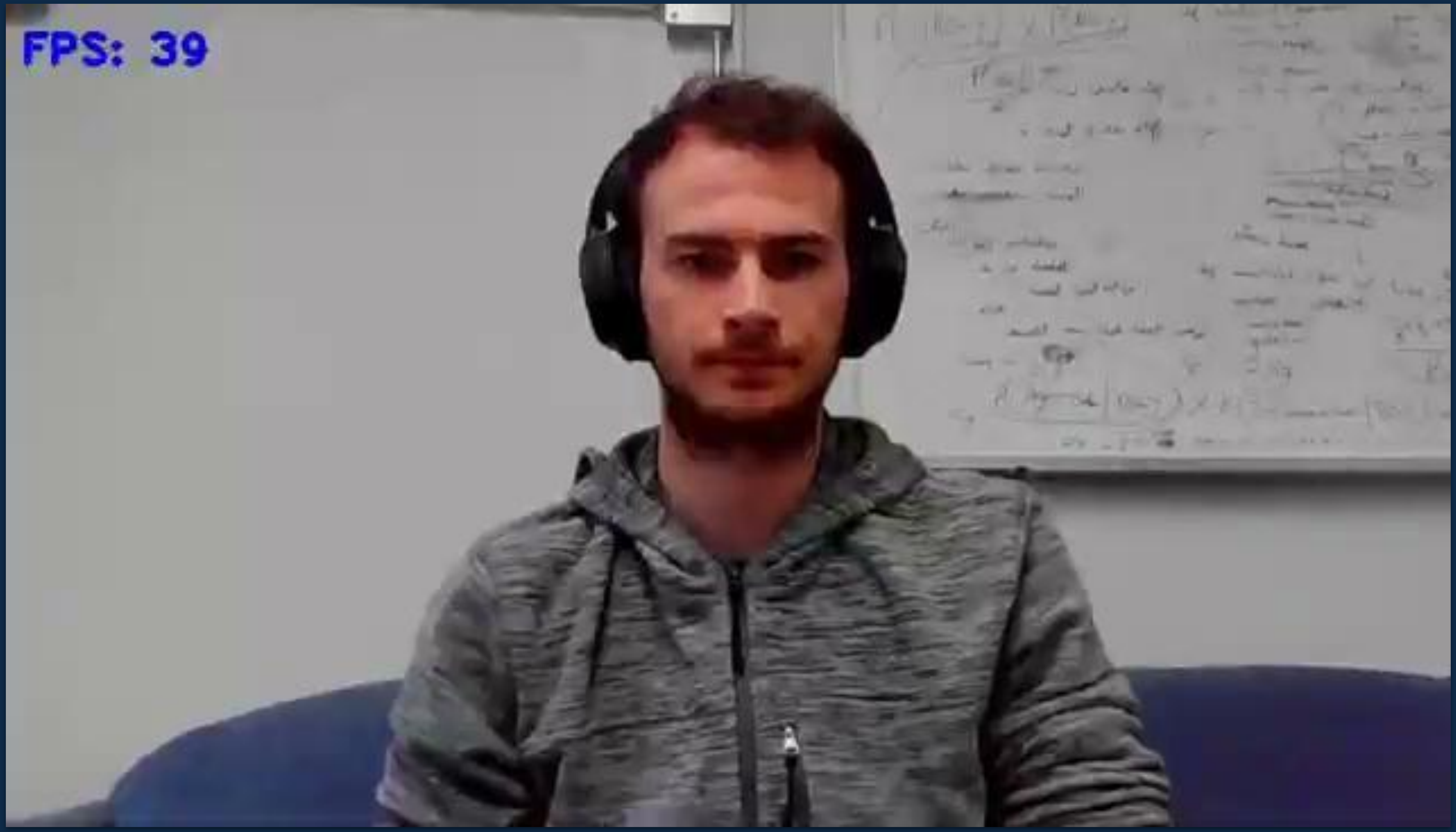
- Dataset: HaGRID — HAnd Gesture Recognition Image Dataset
- MediaPipe: On-device Real-time Hand Gesture Recognition
- Models: Real-time Dynamic Sign Recognition using MediaPipe

# Proposed Solution

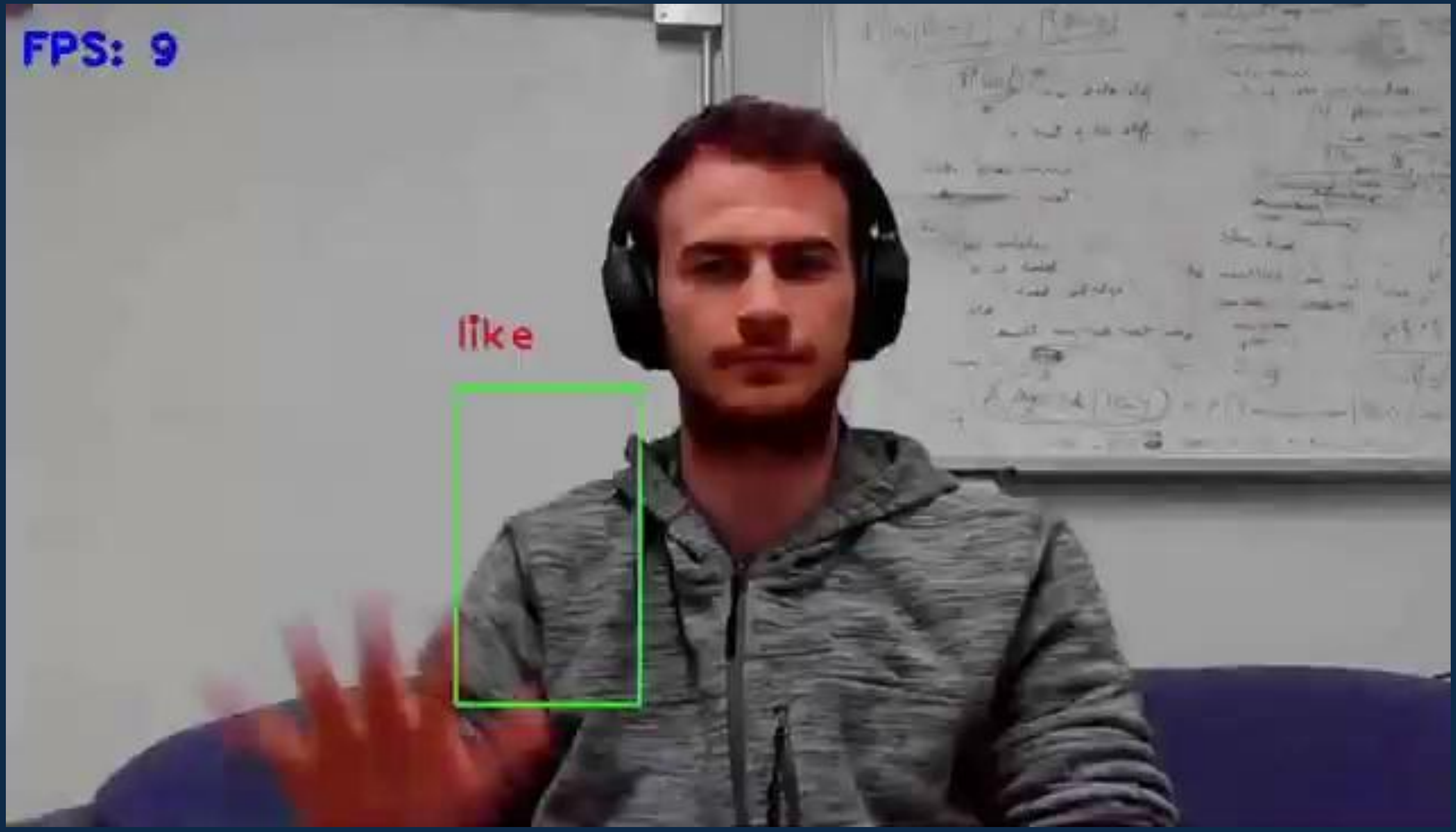
# ML solutions in MediaPipe

| Face Detection | Face Mesh | Iris | Hands | Pose | Holistic |
|---|---|---|---|---|---|
| |  | |  |  | |
| Hair Segmentation | Object Detection | Box Tracking | Instant Motion Tracking | Objectron | KNIFT |
|  |  |  | | | |

# Live Application



MediaPipe Hand Detector (landmarks + boxes) + Conv1D Classifier

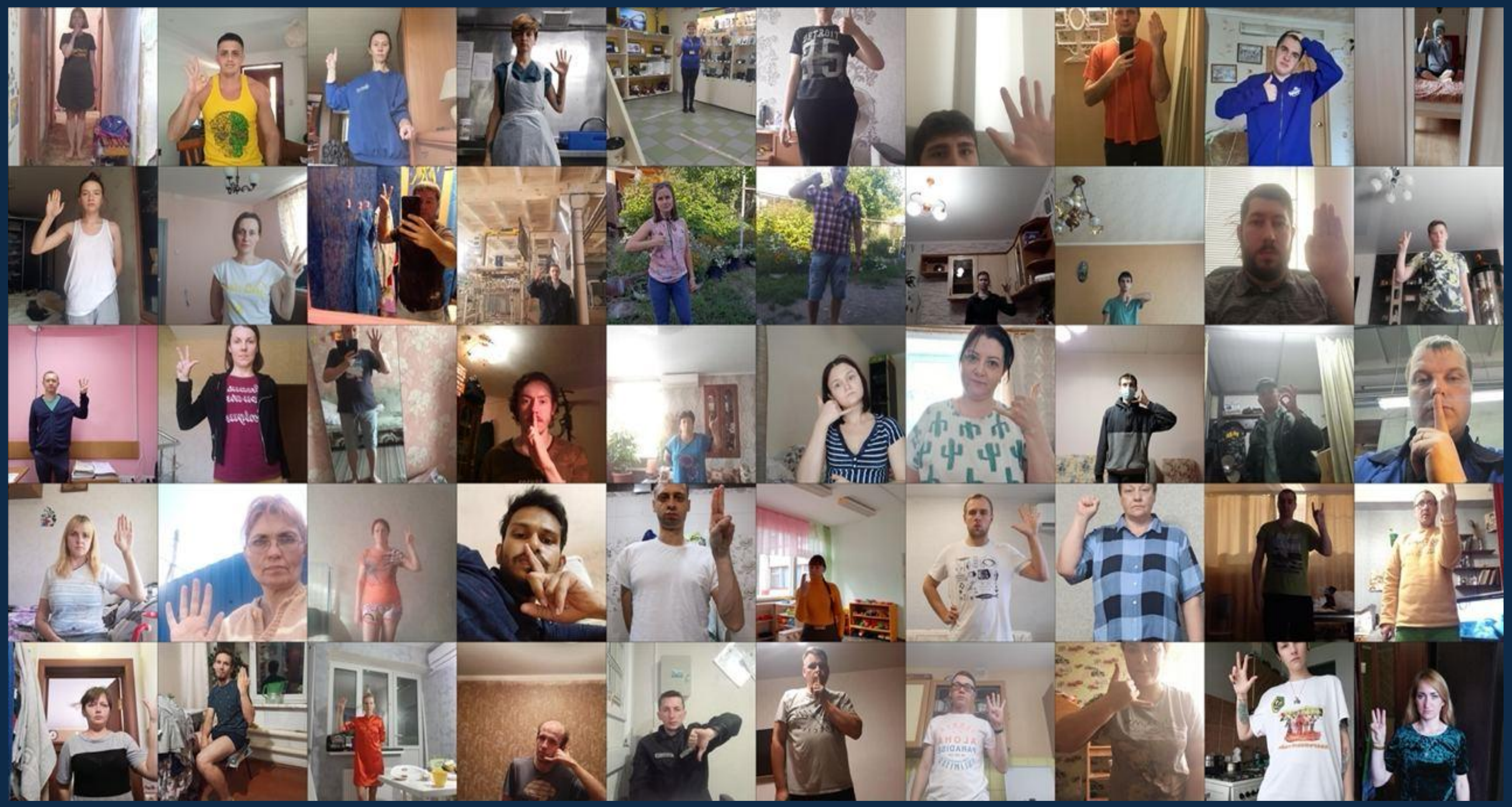


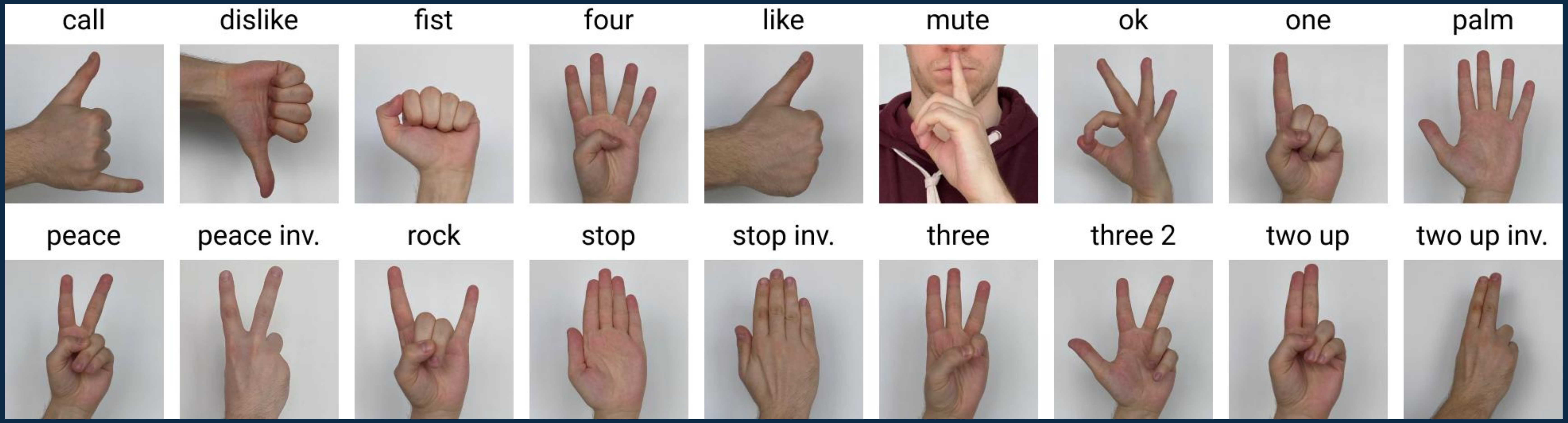
1-Hand MobileNet Classifier (boxes + hand gesture)

# Dataset Presentation

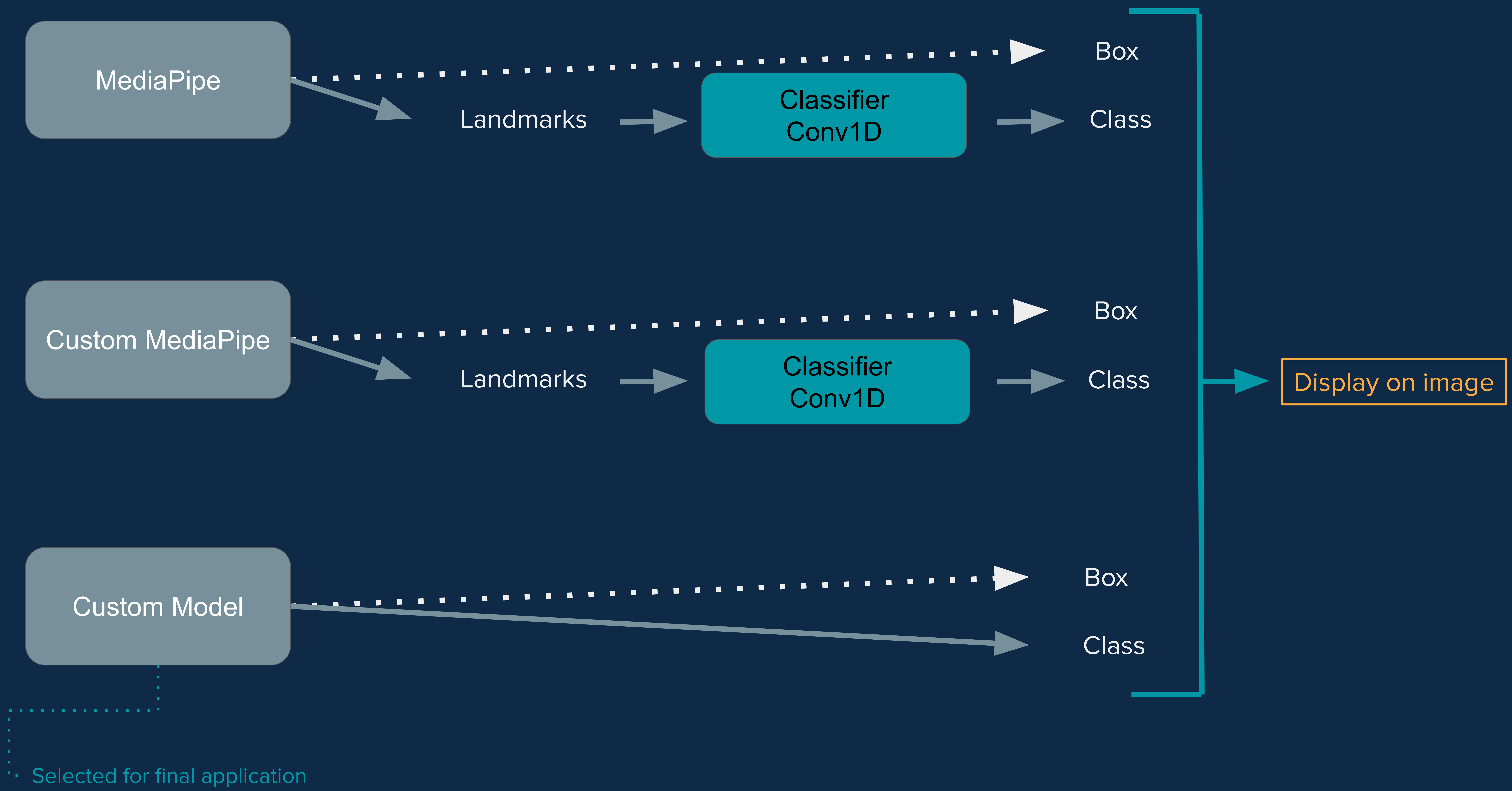
## HaGRID – HAnd Gesture Recognition Image Dataset:

- Created in June 2022.
- Size: 716GB – 552,992 RGB images (with majority FullHD) divided into 18 classes of gestures.
- Extra class “no_gesture” with 123,589 samples.
- Collected mainly indoors in various conditions (lighting variations, blur, distance from camera, …)
- Only kept between 2000 and 3000 samples per class.
- Include whole set of annotations containing image_ID, labels, bboxes, leading hand, label.



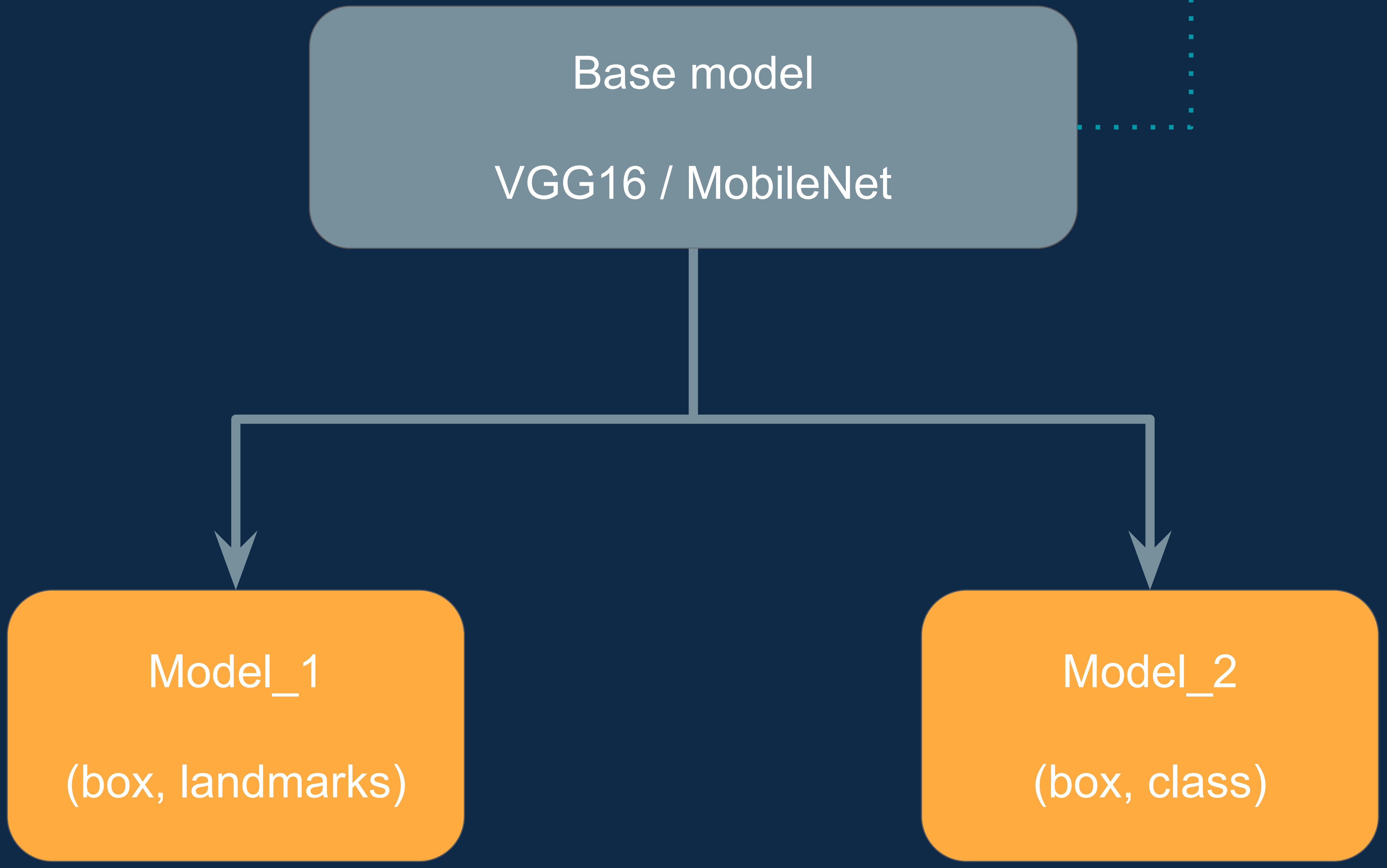


# Pipeline


MediaPipe
Landmarks
Classifier Conv1D
Box
Class
Custom MediaPipe
Landmarks
Classifier Conv1D
Box
Class
Display on image
Custom Model
Box
Class
Selected for final application

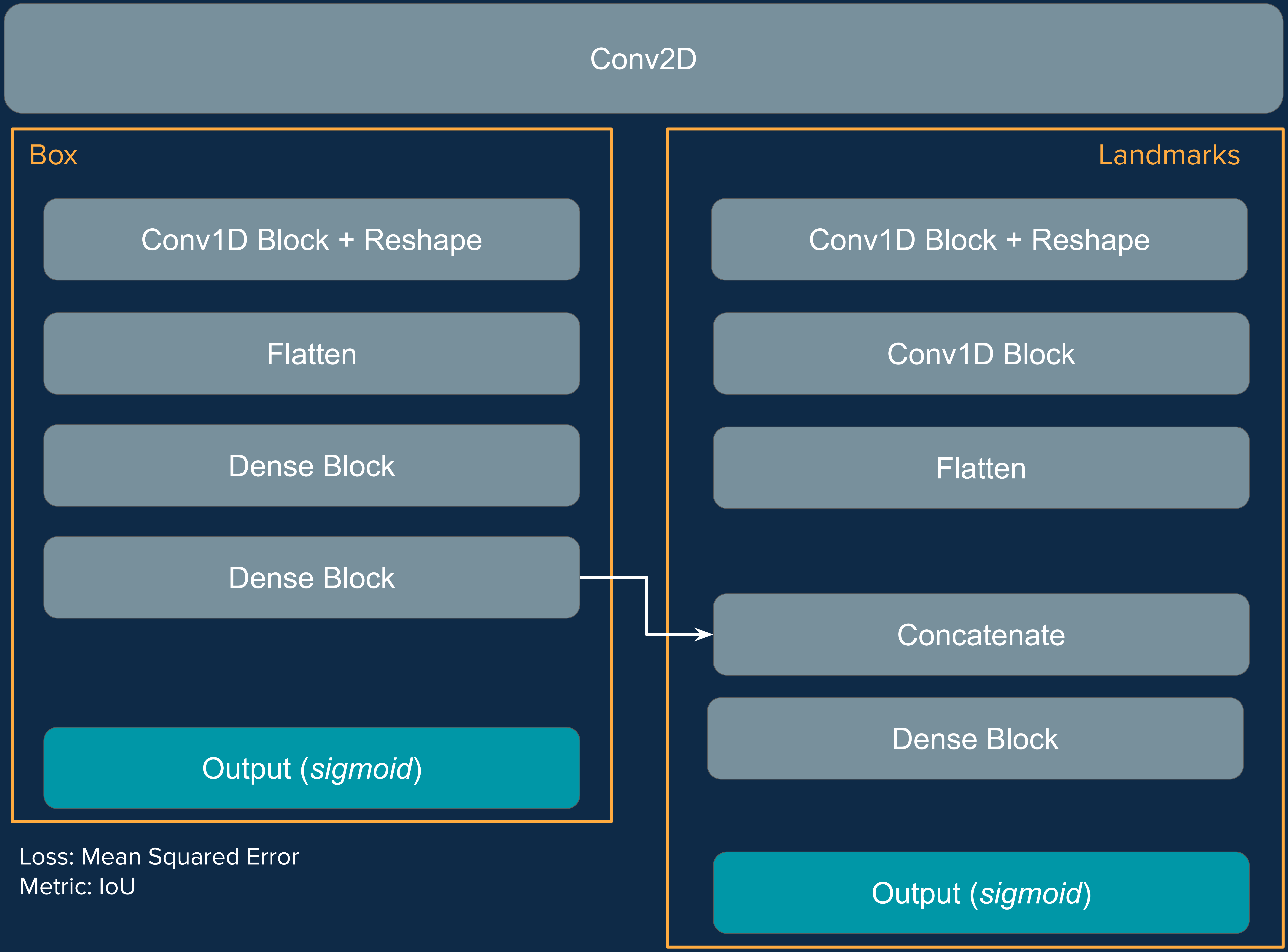
# Implementation Details

# Models



MobileNet selected for final application

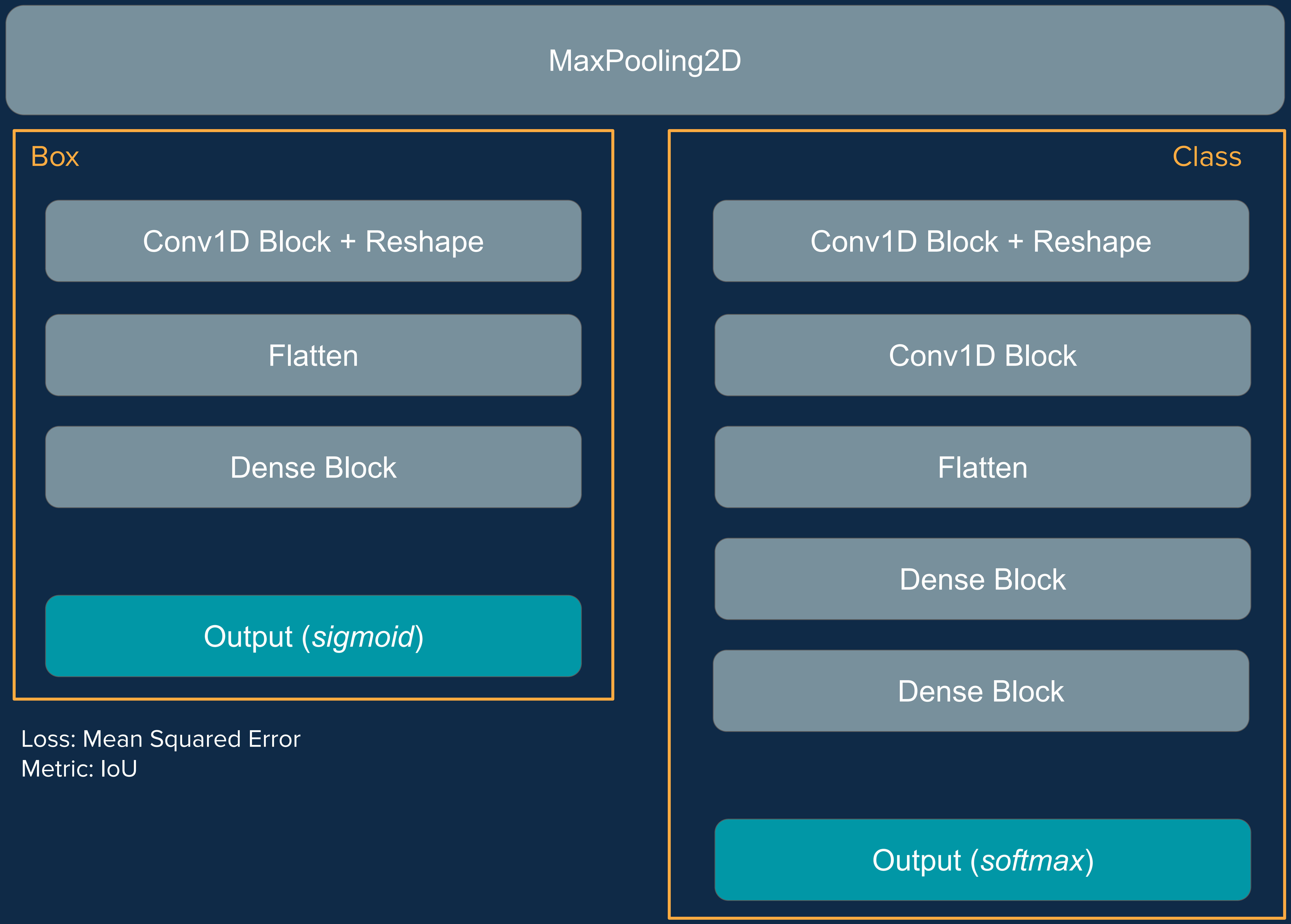
# Model_1 (box, landmarks)



Loss: Mean Squared Error
Metric: IoU

Loss: Huber Loss
Metric: Cosine Similarity

# Model_2 (box, class)



MaxPooling2D

Box

- Conv1D Block + Reshape
- Flatten
- Dense Block
- Output (*sigmoid*)

Class

- Conv1D Block + Reshape
- Conv1D Block
- Flatten
- Dense Block
- Dense Block
- Output (*softmax*)

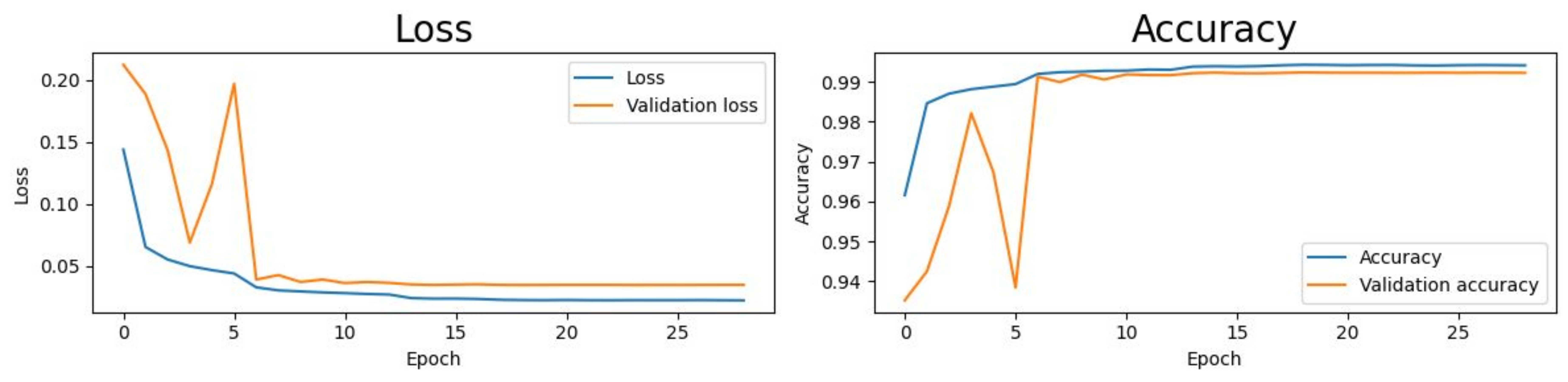
Loss: Mean Squared Error
Metric: IoU

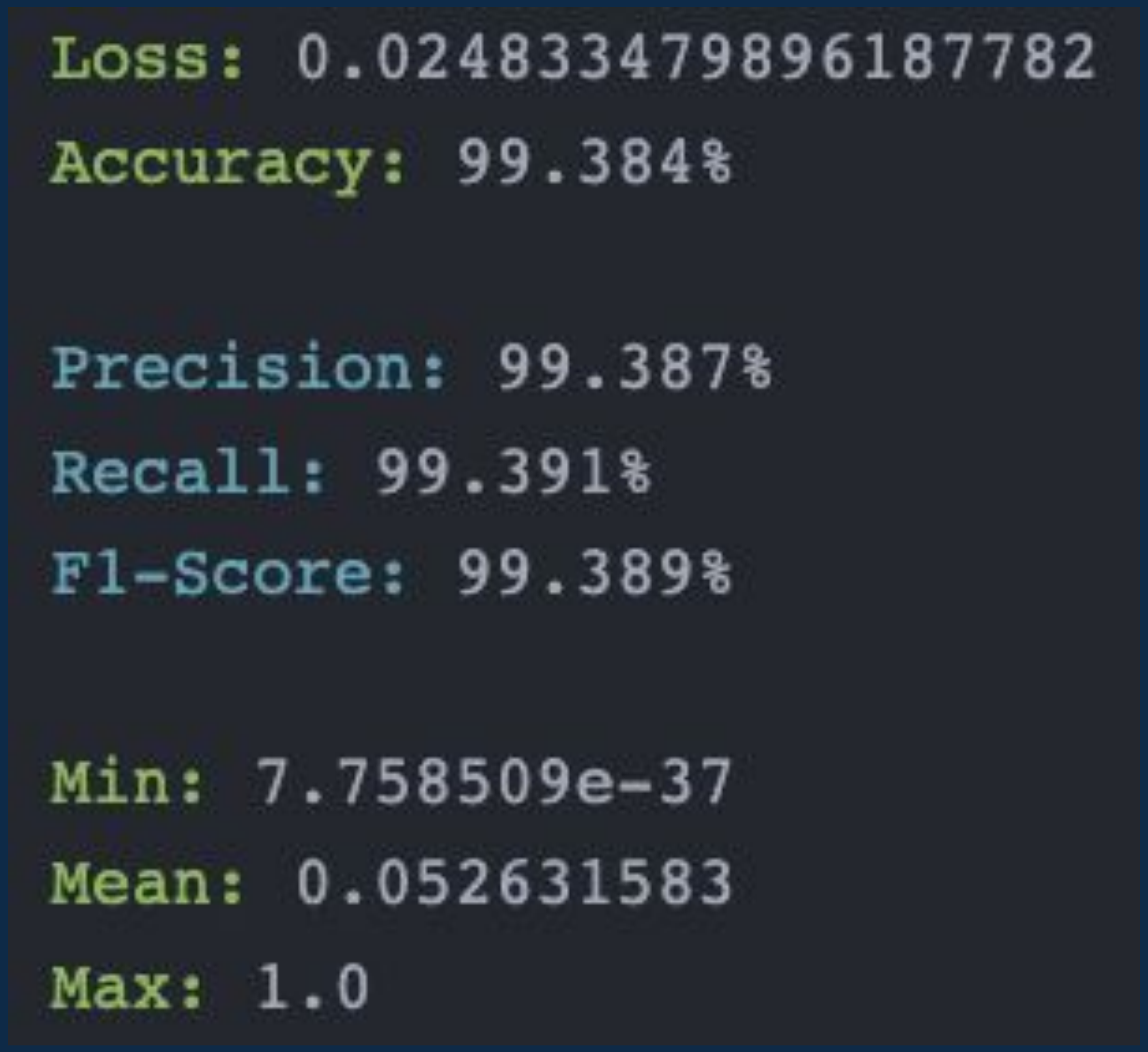
Loss: Categorical Cross Entropy
Metric: Accuracy

# Results & Discussion

# Classifier (Conv1D)

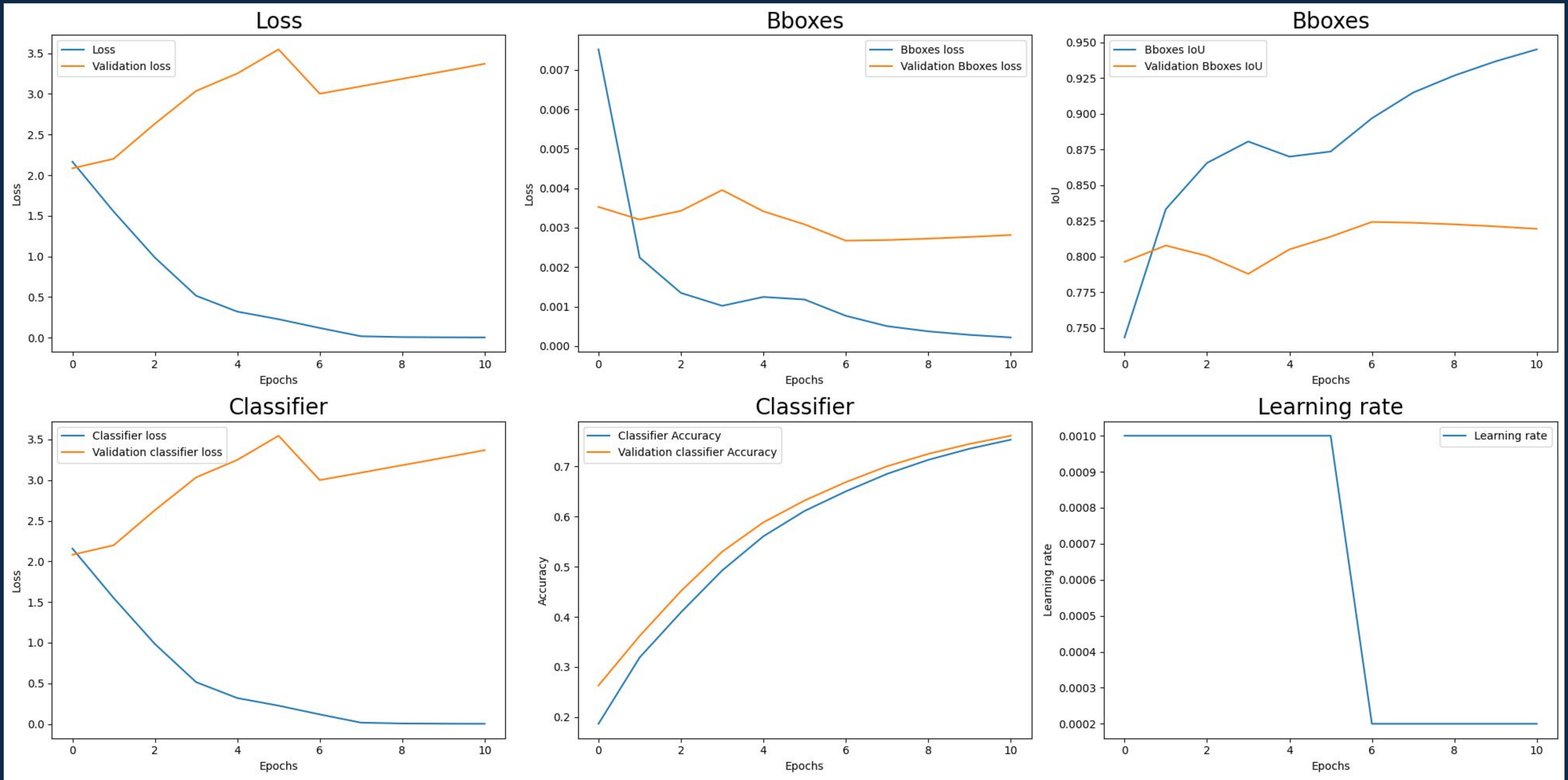
*Trained on more than 490,000 landmarks from dataset annotations*



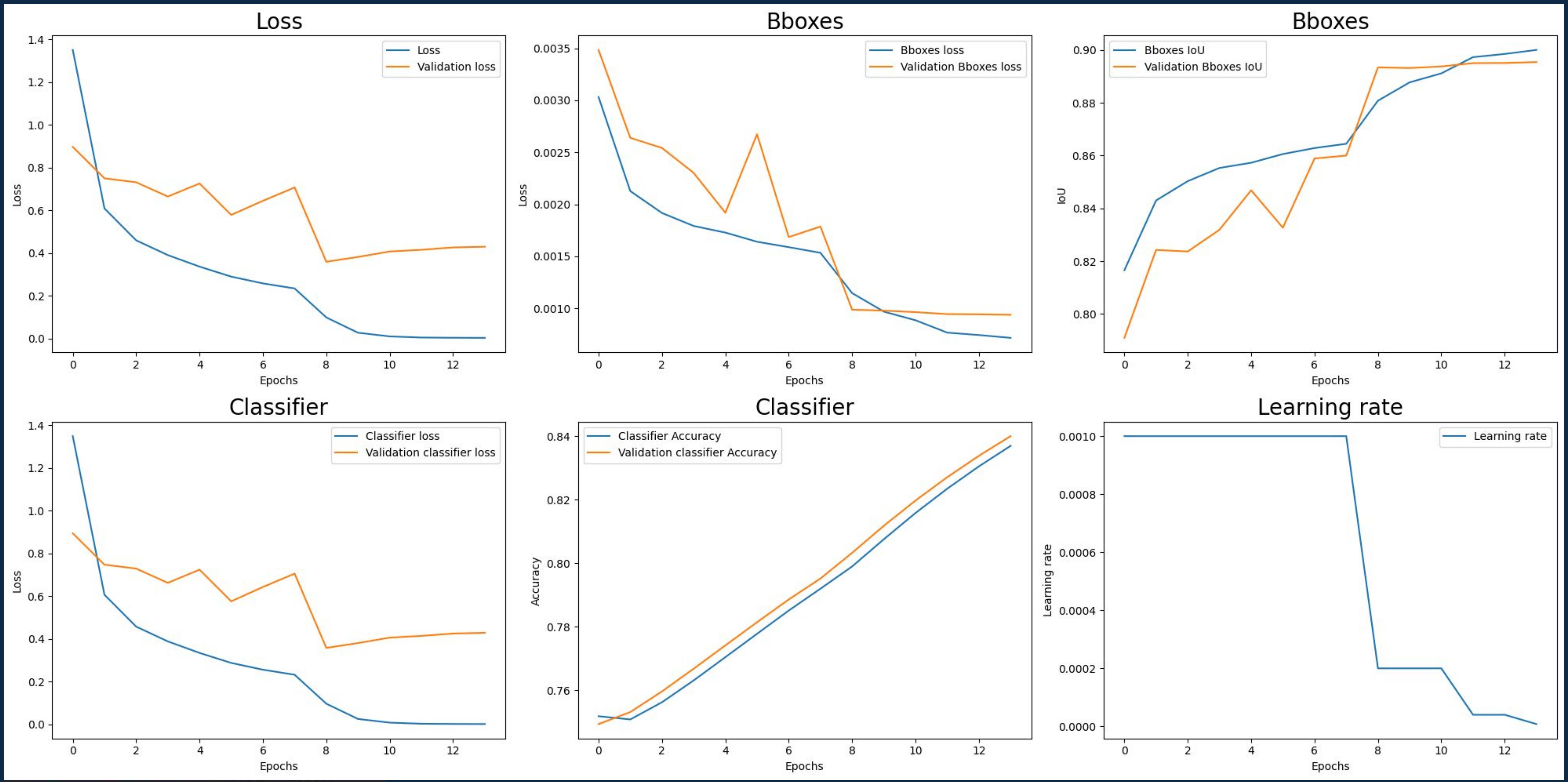

```
Loss: 0.024833479896187782
Accuracy: 99.384%

Precision: 99.387%
Recall: 99.391%
F1-Score: 99.389%

Min: 7.758509e-37
Mean: 0.052631583
Max: 1.0
```

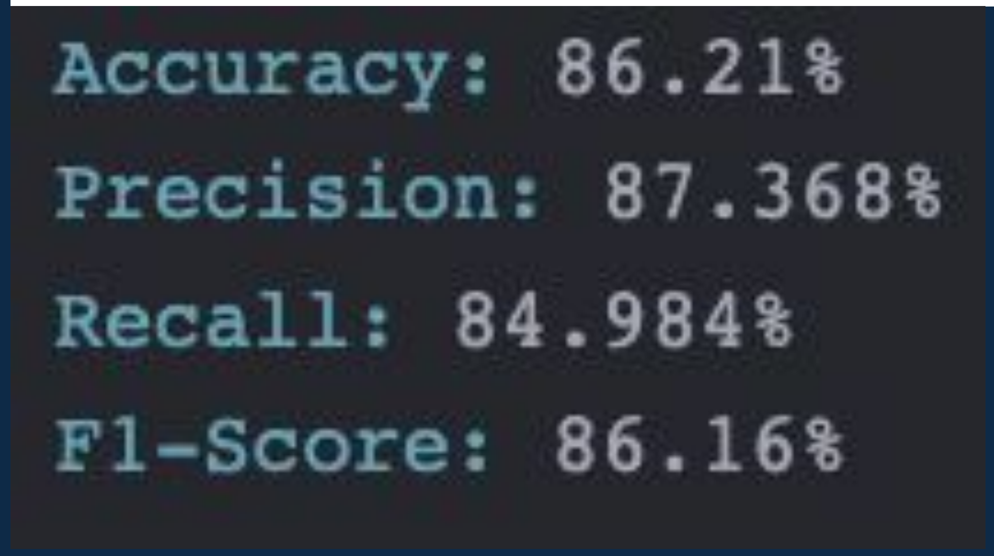
# Model – MobileNet (box, class)

*Trained on more than 32,000 images*



# Model MobileNet (box, class) - Fine tuning





```
Accuracy: 86.21%
Precision: 87.368%
Recall: 84.984%
F1-Score: 86.16%
```

# Problems & Conception choices

# Model architecture and number of parameters

Goal: predictions from live video => need efficiency

=> Discarded VGG16 as base model
=> Selected MobileNet as base model

# From custom MediaPipe to complete custom model

First: Implementation of the classifier from MediaPipe output landmarks

We wanted to implement custom MediaPipe to simply replace it by our model, predict the landmarks and use the same classifier as above

Problem: Predicted landmarks were not accurate enough to get a good accuracy

Solution: Custom model, predicting box and class

# Dataset size

Classifier trained on more than 490,000 landmarks (21x2), from dataset annotations

Problem: Train CNN on 490,000 images very computer intensive

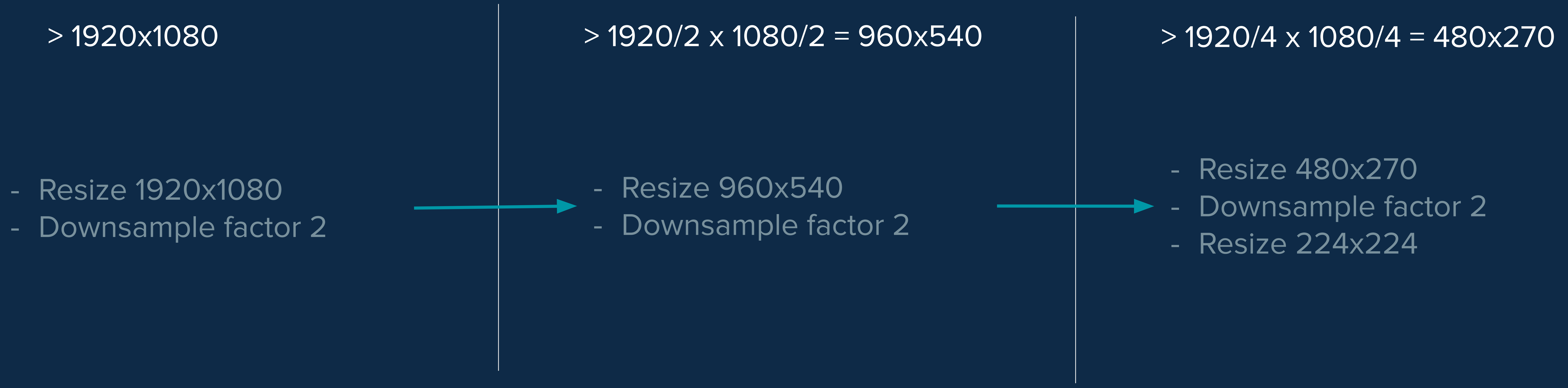
Solution: Sample the data ( ~ 40,000 images)

# MobileNet input size

MobileNet takes an input shape (224, 224, 3)

Problem: Various sizes of the images from the dataset: from more than 1920x1080 to less than 480x270

Solution: Downsample and resize images



> 1920x1080

- Resize 1920x1080
- Downsample factor 2

> 1920/2 x 1080/2 = 960x540

- Resize 960x540
- Downsample factor 2

> 1920/4 x 1080/4 = 480x270

- Resize 480x270
- Downsample factor 2
- Resize 224x224

# Thank you for your attention

Do you have any questions?