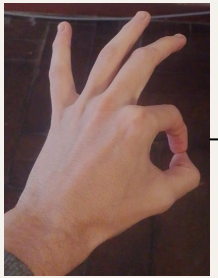
Hand gesture classifier

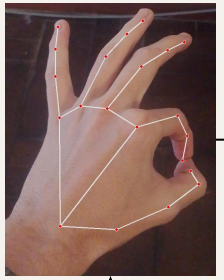
- Modeling



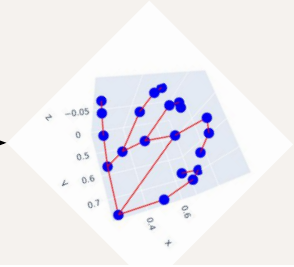
Image

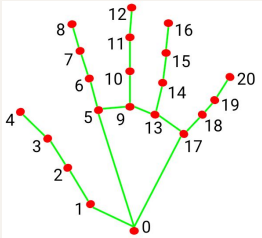
Mimepipe
https://github.com/google-ai-edge/mediapipe

Key points of the hand

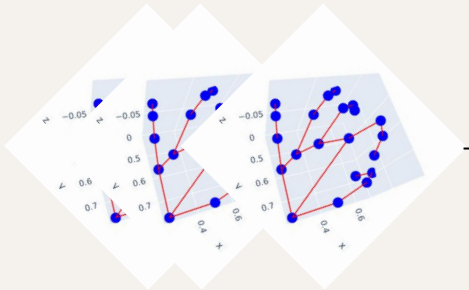


3D Points





0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP



K frames

class_1
class_2
class_C

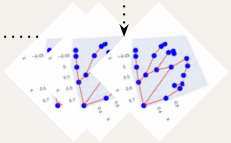
Custom model

Output with C classes

(The last or first is null detection)

● Create dataset

Create a sample dataset (10 videos) → **Manual labeling** → **Convert video frames into a spatiotemporal point cloud**



N x 21 x 3

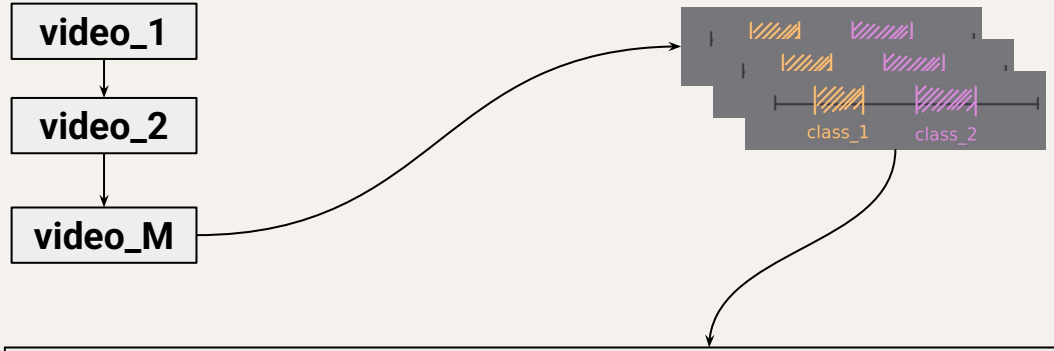
- ➔ **N:** Number of frames in the video.
- ➔ **21:** Number of 3D points.
- ➔ **3:** (x,y,z)

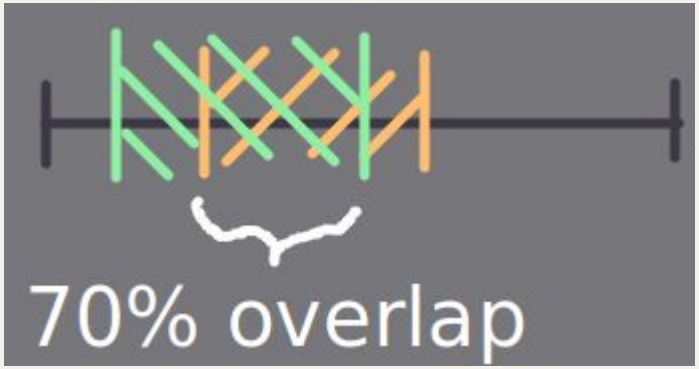
Data augmentation

- **Rotation**
- **Rescaling**
- **Axis translation**
- **Random noise**

→ **Different representations of the same point cloud**

- Handling dataset intervals



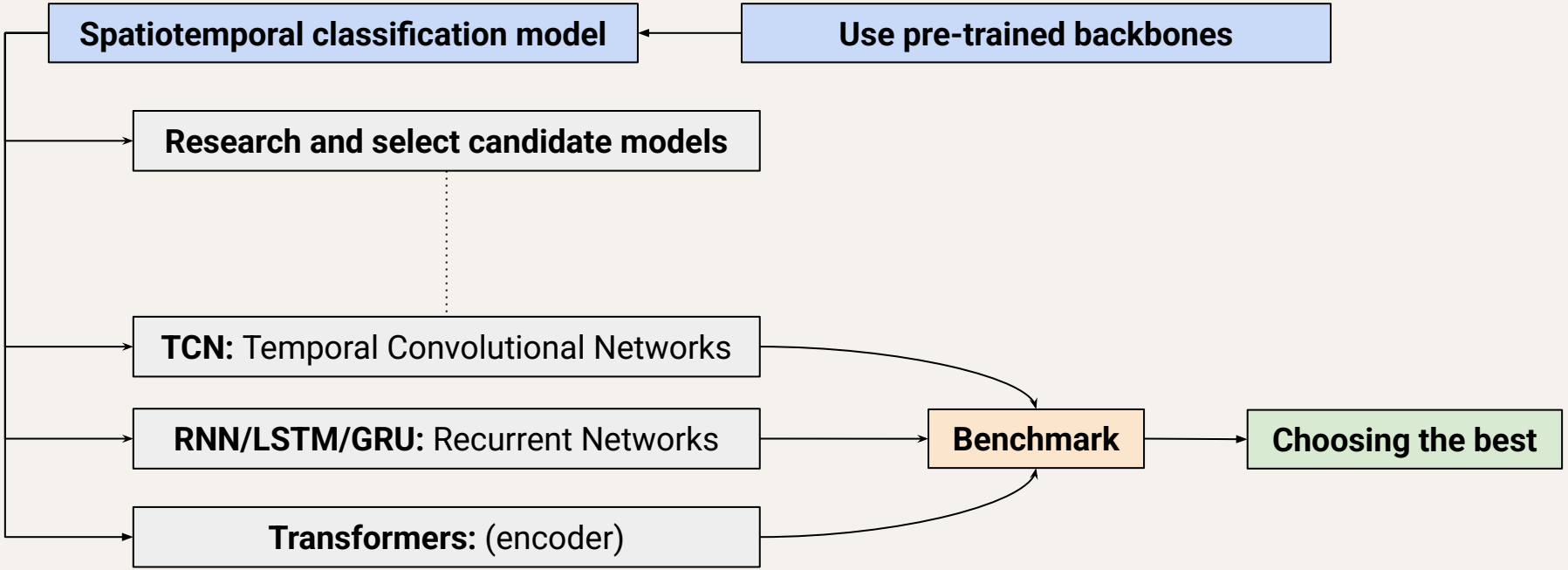


Windows are obtained

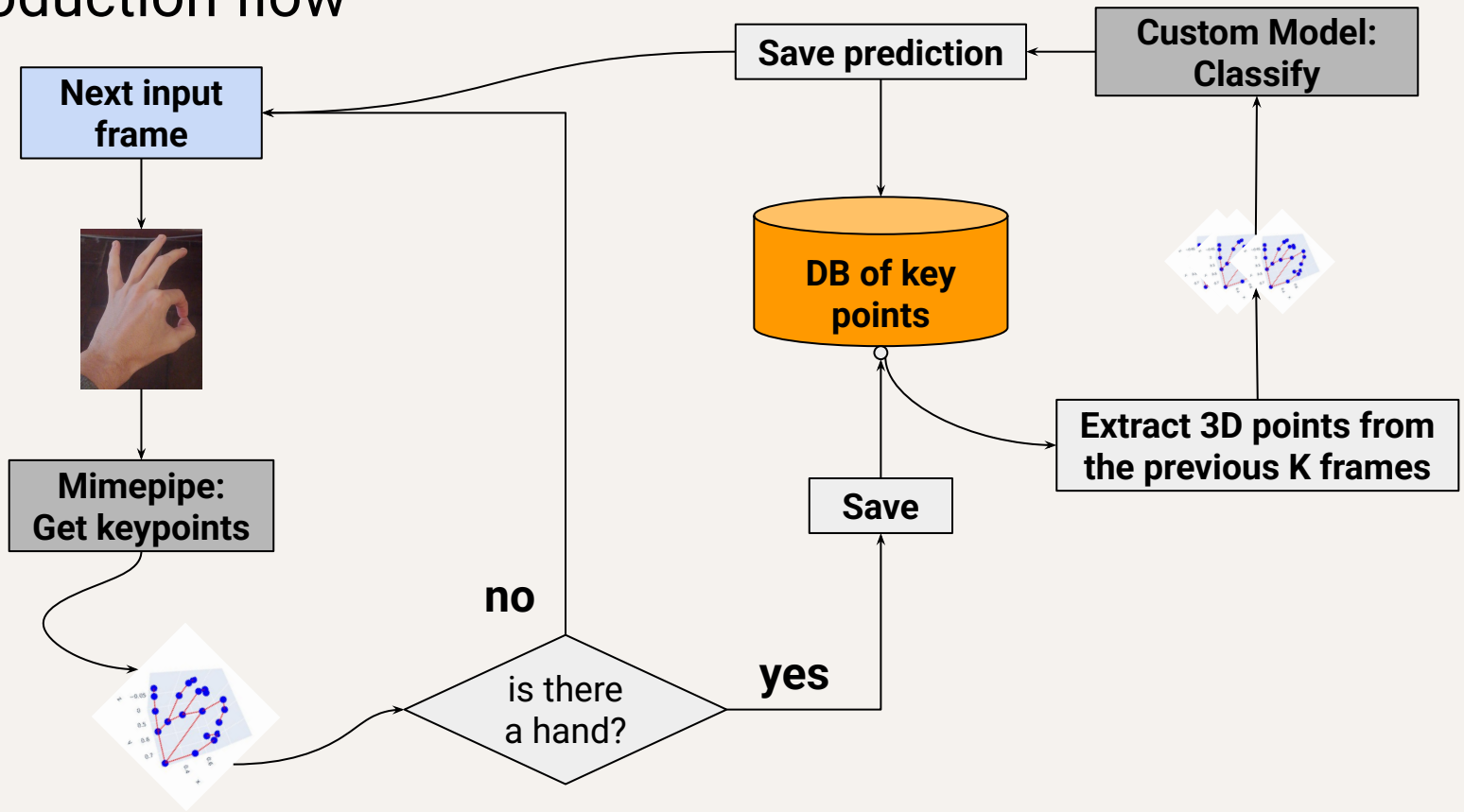
For each window, the percentage of overlap with the class is obtained.

If it exceeds a threshold, it will be that class; otherwise, it will be an empty class.

- Custom model (Try and choose)



- Production flow



- Export .onnx https://docs.pytorch.org/tutorials/beginner/onnx/export_simple_model_to_onnx_tutorial.html

3. Export the model to ONNX format

Now that we have our model defined, we need to instantiate it and create a random 32x32 input. Next, we can export the model to ONNX format.

```
torch_model = ImageClassifierModel()
# Create example inputs for exporting the model. The inputs should be a tuple of tensors.
example_inputs = (torch.randn(1, 1, 32, 32),)
onnx_program = torch.onnx.export(torch_model, example_inputs, dynamo=True)
```

4. Save the ONNX model in a file

Although having the exported model loaded in memory is useful in many applications, we can save it to disk with the following code:

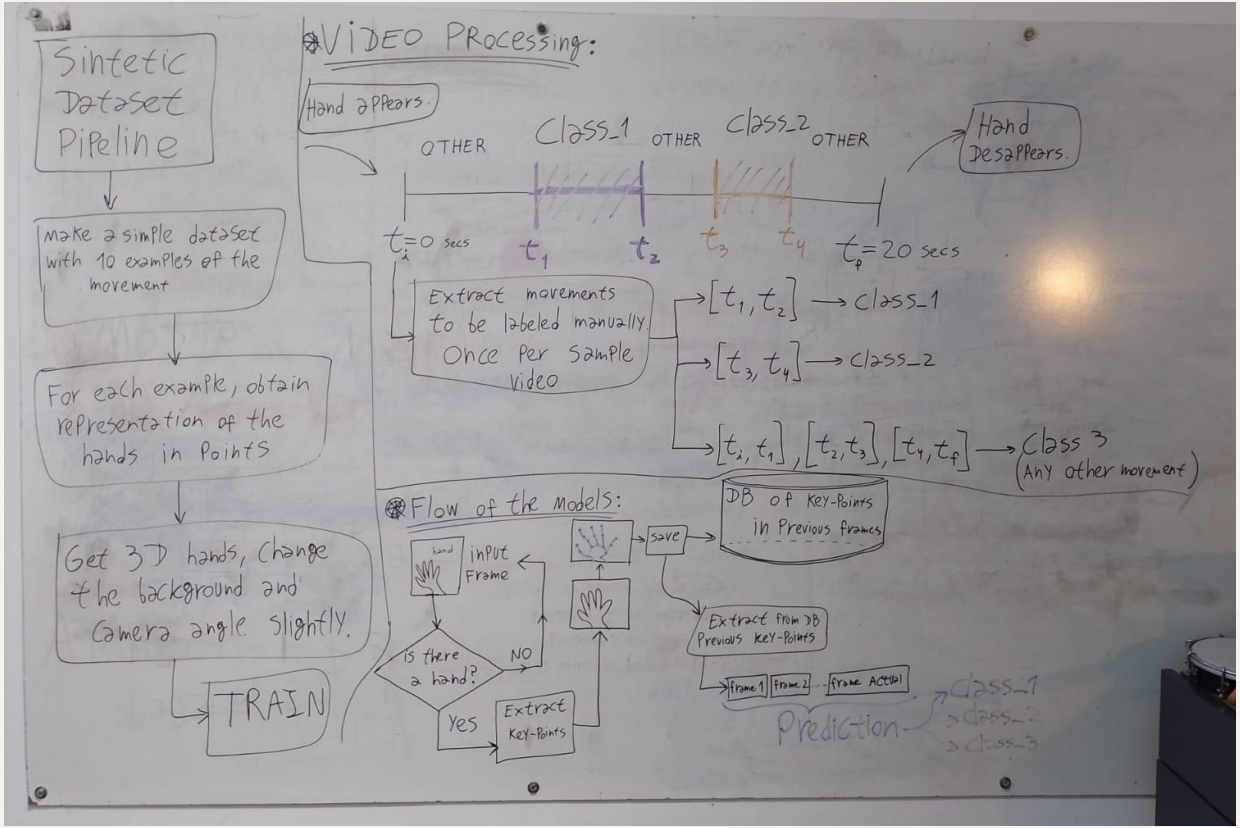
```
onnx_program.save("image_classifier_model.onnx")
```

You can load the ONNX file back into memory and check if it is well formed with the following code:

```
import onnx

onnx_model = onnx.load("image_classifier_model.onnx")
onnx.checker.check_model(onnx_model)
```

- Draft


Sintetic Dataset Pipeline
make a simple dataset with 10 examples of the movement
For each example, obtain representation of the hands in Points
Get 3D hands, change the background and camera angle slightly.
TRAIN
VIDEO Processing:
Hand appears.
OTHER
Class_1
OTHER
Class_2
OTHER
Hand Desappears.
t_i=0 secs
t_1
t_2
t_3
t_4
t_f=20 secs
Extract movements to be labeled manually Once per sample video
$[t_1, t_2]$ → Class_1
$[t_3, t_4]$ → Class_2
$[t_i, t_1]$, $[t_2, t_3]$, $[t_4, t_f]$ → Class 3 (Any other movement)
Flow of the Models:
hand
input Frame
Is there a hand?
NO
Yes
Extract Key-Points
save
DB of Key-Points in Previous frames
Extract from DB Previous Key-Points
Frame 1
Frame 2
Frame Actual
Prediction
Class_1
Class_2
Class_3