MACIEJ DENGUSIAK
PATRYK FLAMA

SZYMON FICA
MARCIN BANAK

# MULTI-STAGE ACTION RECOGNITION CLASIFICATION

# Goal: Improve action recognition using keypoints guidance

- Problem - Recognise specific actions:
  - Jump
  - Pullup
  - Pushup
  - Wave
  - Sit
- We believed that keypoints might be helpfull





# Why did we choose it?

- Interested in computer vision
- Found great database
- Interesting use cases (cinect, CCTV)

# HMDB51 Dataset

```python
class ActionsBaselineModel(nn.Module):
    def __init__(self, num_actions=5):
        super().__init__()

        base_model = models.resnet18(pretrained=True)
        self.cnn_backbone = nn.Sequential(*list(base_model.children()
        self.feature_dim = base_model.fc.in_features
        for param in self.cnn_backbone.parameters():
            param.requires_grad = False

        self.classifier = nn.Sequential(
            nn.Linear(self.feature_dim, 1024),
            nn.BatchNorm1d(1024),
            nn.ReLU(),
            nn.Dropout(0.5),

            nn.Linear(512, 256),
            nn.ReLU(),

            nn.Linear(256, num_actions)
        )
```

# Baseline

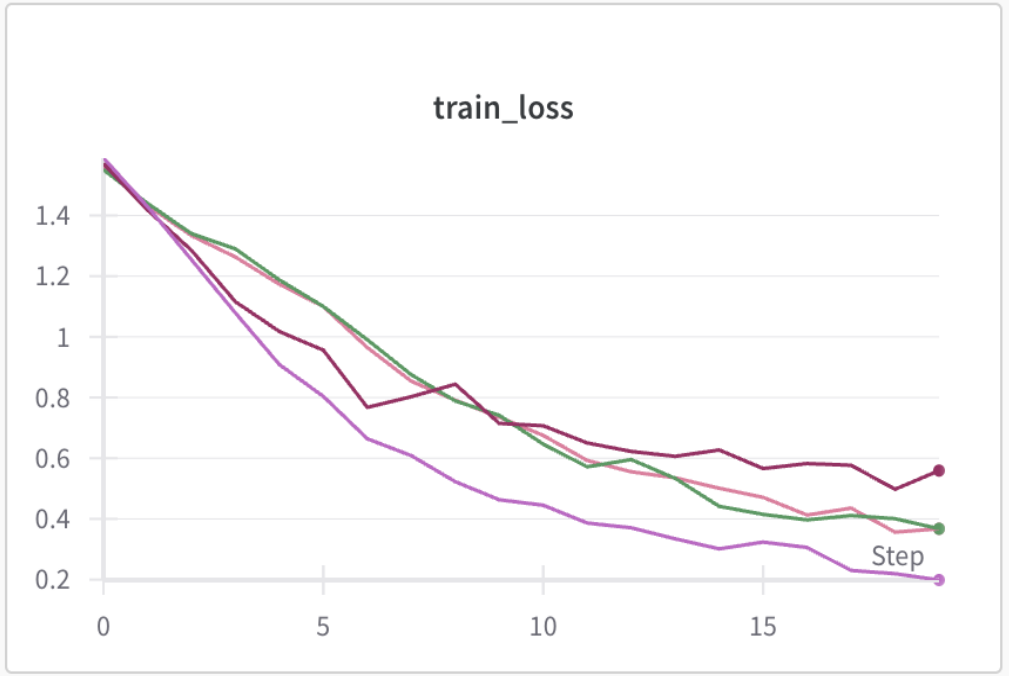
- Pretrained ResNet18 backbone
- Dense Linear layers for classifier
- Cross Entropy Loss
- Adam optimizer
- Input data Resized and Normalized
- Optional augmentation:
  - Blur, Grayscale, Jitter, Sharpness

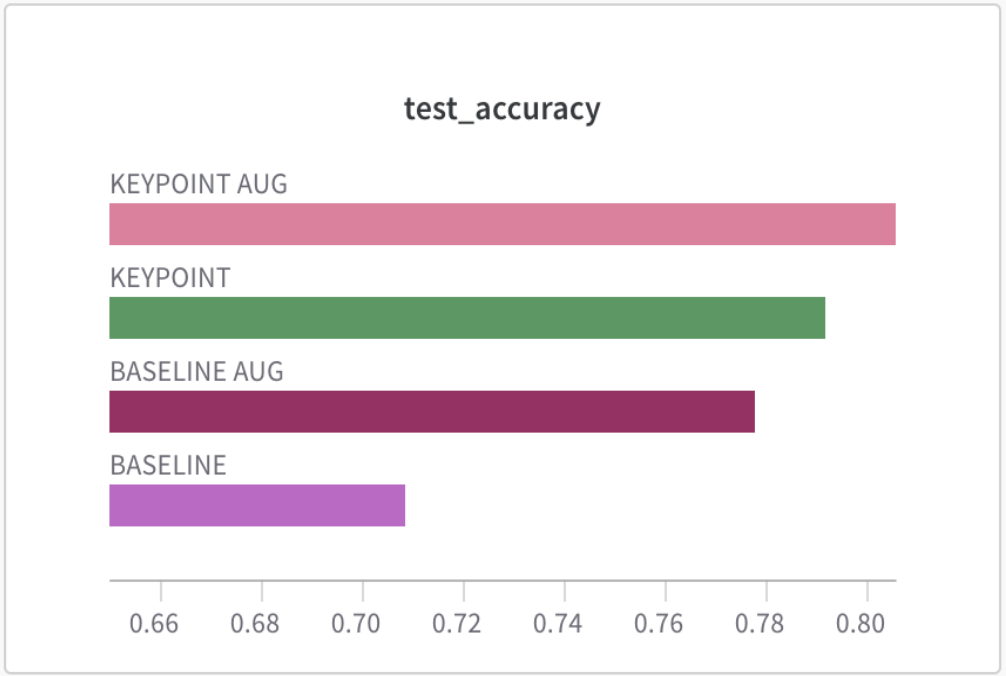
```python
class ActionsFusionModel(nn.Module):
    def __init__(self, num_keypoints=NUM_KEYPOINTS, num_actions=10):
        super().__init__()
        base_model = models.resnet18(pretrained=True)
        self.cnn_backbone = nn.Sequential(*list(base_model.children())[:-1])
        self.feature_dim_img = base_model.fc.in_features
        for param in self.cnn_backbone.parameters():
            param.requires_grad = False

        self.keypoint_dim = num_keypoints * 2
        self.keypoint_mlp = nn.Sequential(
            nn.Linear(self.keypoint_dim, 128),
            nn.ReLU(),
            nn.Dropout(0.3),
            nn.Linear(128, 128),
            nn.ReLU()
        )

        self.classifier = nn.Sequential(
            nn.Linear(self.feature_dim_img + 128, 512),
            nn.BatchNorm1d(512),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(512, 256),
            nn.ReLU(),
            nn.Linear(256, num_actions)
        )
```

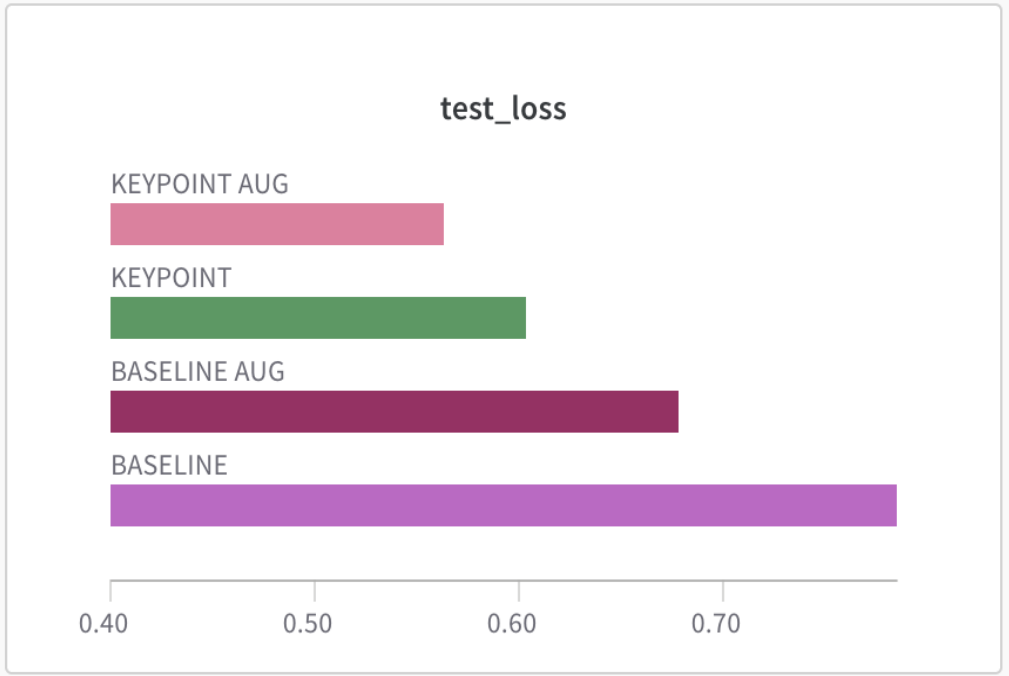
# Keypoint Based

- Pretrained ResNet18 for images
- Dense layers for keypoints
- Both outputs combined with Dense Layers

# Action Detection Comparison

# Additionally… we trained keypoints

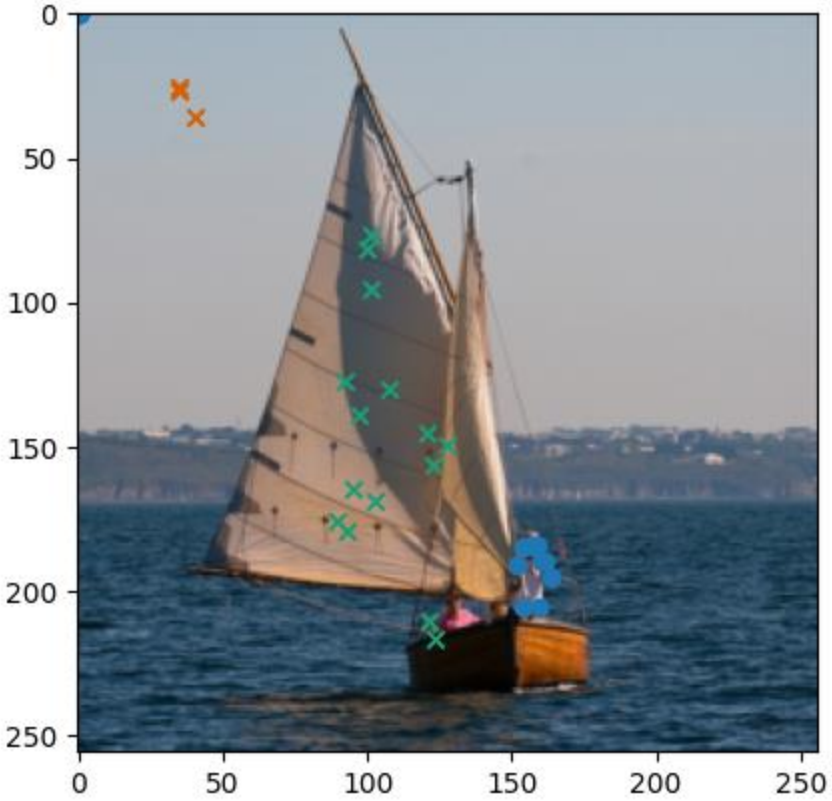
## BASELINE

- Detect the keypoints of each person
- Directly on entire image
- Improve model with yolo style grid

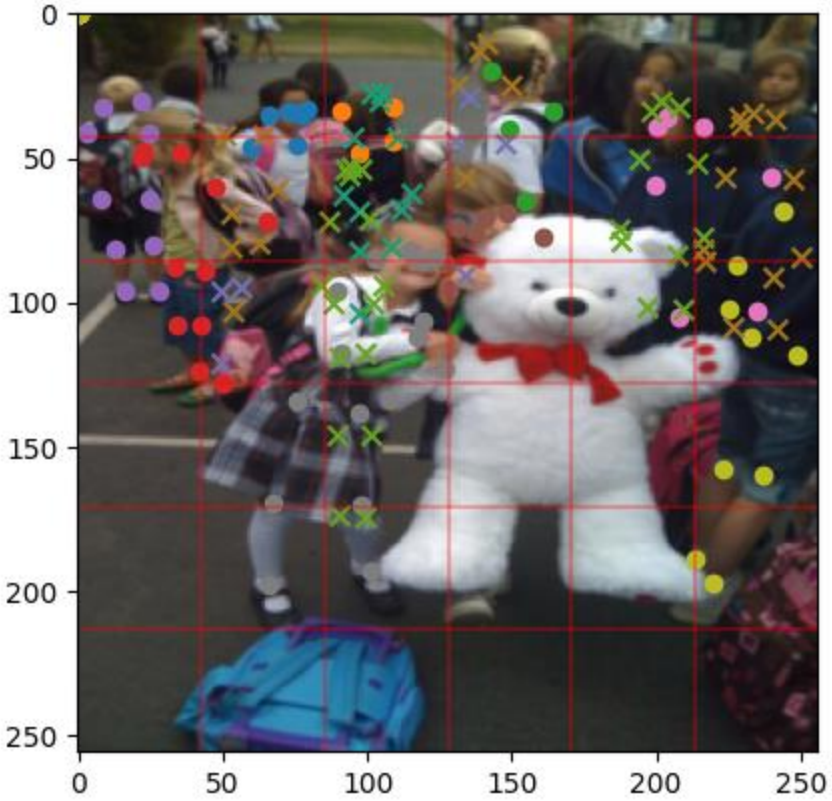
## IMPROVED VERSION

- Detect the bounding box of each person
- Train the model to predict keypoints on the cropped image
- Focused only on the person).

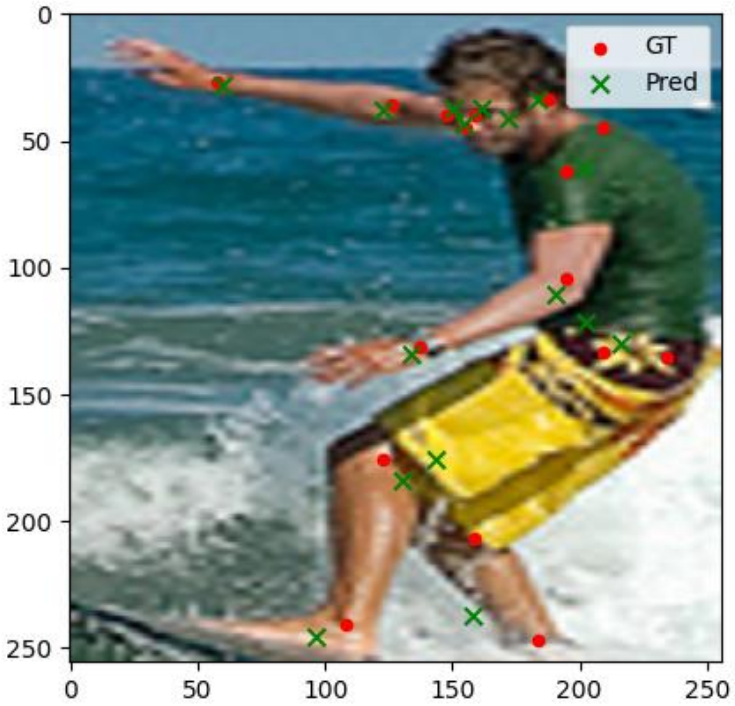
# COCO Dataset




Simple

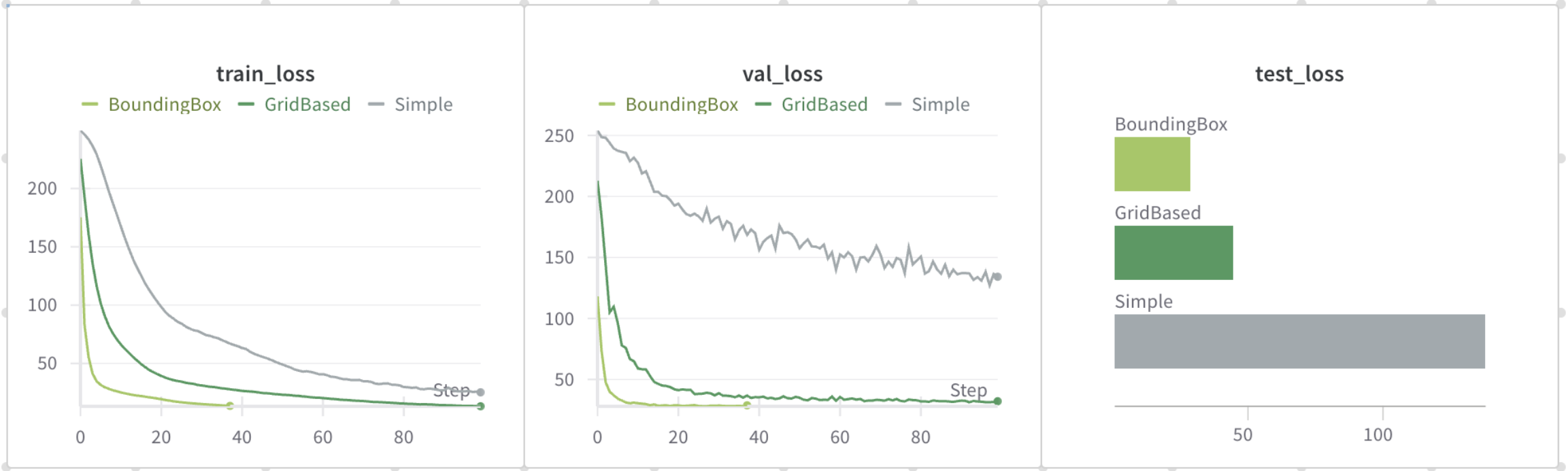



Grid Based




Bounding Box based

# Keypoint Detection Comparison




# Keypoint Detection Comparison

# Conclusions

**What have you learned**

Experience in computer vision problems and transfer learning.

**What was good or bad**

Model performed well and achieved higher accuracy than expected.

**What could have been different**

Utilizing models with more complex architecture and usage of more compute power.

**What could you do next?**

We could use RNN architecture to implement it in real-time action detection.

# Links

- Github: https://github.com/MaciejDengusiak/NN-Project
- Dataset download: https://github.com/MaciejDengusiak/NN-Project/tree/main/data
- Wandb report: https://api.wandb.ai/links/fejowo5522-/aai2ttsd