# CenterNet

에이아이스쿨(AISchool) 대표
양진호 (솔라리스)
http://aischool.ai
http://solarisailab.com

## TensorFlow Object Detection API에서 제공하는 다양한 Object Detection을 위한 최신 모델들

- TensorFlow Object Detection API는 다음과 같은 최신 Object Detection 모델의 다양한 backbone을 이용한 구현을 제공합니다.

① Faster R-CNN

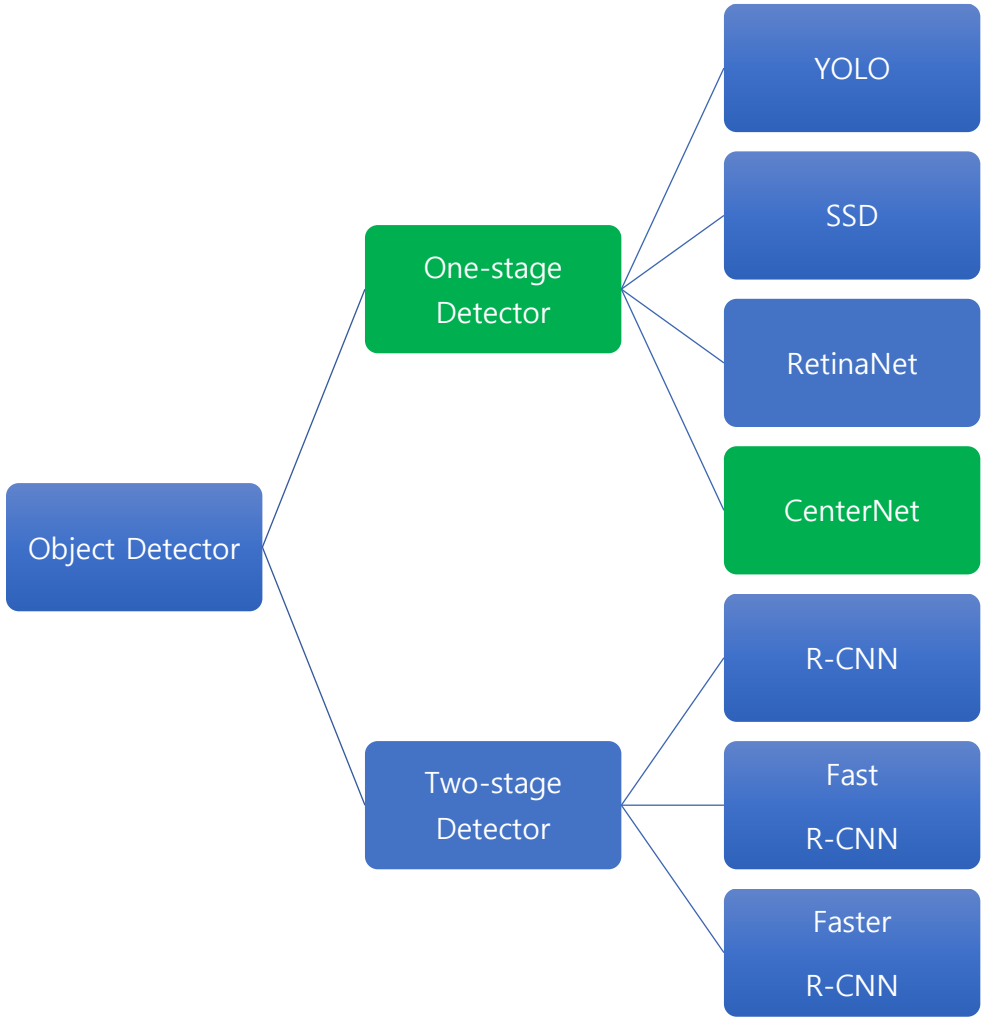
② SSD(Single Shot Multi-box Detector)

③ RetinaNet

④ CenterNet

⑤ EfficientDet

# One-stage Detector vs Two-Stage Detector



# CenterNet

- Zhou, Xingyi, Dequan Wang, and Philipp Krähenbühl. "Objects as points." arXiv preprint arXiv:1904.07850 (2019).
- https://arxiv.org/pdf/1904.07850.pdf

**Objects as Points**


Xingyi Zhou
UT Austin
zhouxy@cs.utexas.edu

Dequan Wang
UC Berkeley
dqwang@cs.berkeley.edu

Philipp Krähenbühl
UT Austin
philkr@cs.utexas.edu





**Abstract**

*Detection identifies objects as axis-aligned boxes in an image. Most successful object detectors enumerate a nearly exhaustive list of potential object locations and classify each. This is wasteful, inefficient, and requires additional post-processing. In this paper, we take a different approach. We model an object as a single point — the center point of its bounding box. Our detector uses keypoint estimation to find center points and regresses to all other object properties, such as size, 3D location, orientation, and even pose. Our center point based approach, CenterNet, is end-to-end differentiable, simpler, faster, and more accurate than corresponding bounding box based detectors. CenterNet achieves the best speed-accuracy trade-off on the MS COCO dataset, with 28.1% AP at 142 FPS, 37.4% AP at 52 FPS, and 45.1% AP with multi-scale testing at 1.4 FPS. We use the same approach to estimate 3D bounding box in the KITTI benchmark and human pose on the COCO keypoint dataset. Our method performs competitively with sophisticated multi-stage methods and runs in real-time.*

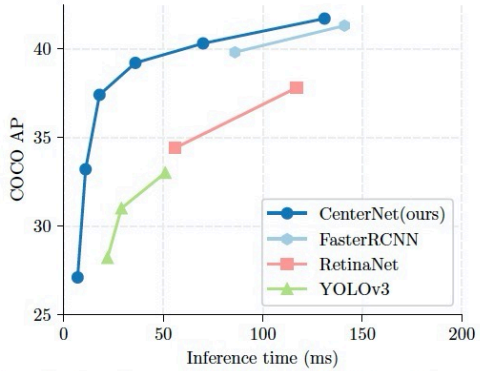




Figure 1: Speed-accuracy trade-off on COCO validation for real-time detectors. The proposed CenterNet outperforms a range of state-of-the-art algorithms.

moves duplicated detections for the same instance by computing bounding box IoU. This post-processing is hard to differentiate and train [23], hence most current detectors are not end-to-end trainable. Nonetheless, over the past

# 기존 Object Detection 모델의 문제점

- 기존 Object Detection 모델의 문제점

① 많은 수의 Anchor Prediction 과정이 필요함 (ReitnaNet의 경우 1장의 이미지에 대한 약 100K 정도의 Anchor Prediction)

② 이로 인해 Post-Processing 과정에서 NMS 처리로 인해 속도가 느려짐

③ 또한 Multi-Anchor Prediction의 결과로 NMS를 적용하더라도 하나의 Object 에 대해 여러개의 Prediction을 만드는 중복 Prediction 문제가 생길 수 있음

# CetnerNet의 핵심 아이디어

- CenterNet의 핵심 아이디어 : Object Detection 문제를 하나의 Object의 중심점을 하나의 Keypoint로 바라보는 Keypoint Estimation 문제로 치환한 CenterNet 구조를 제안하였다.

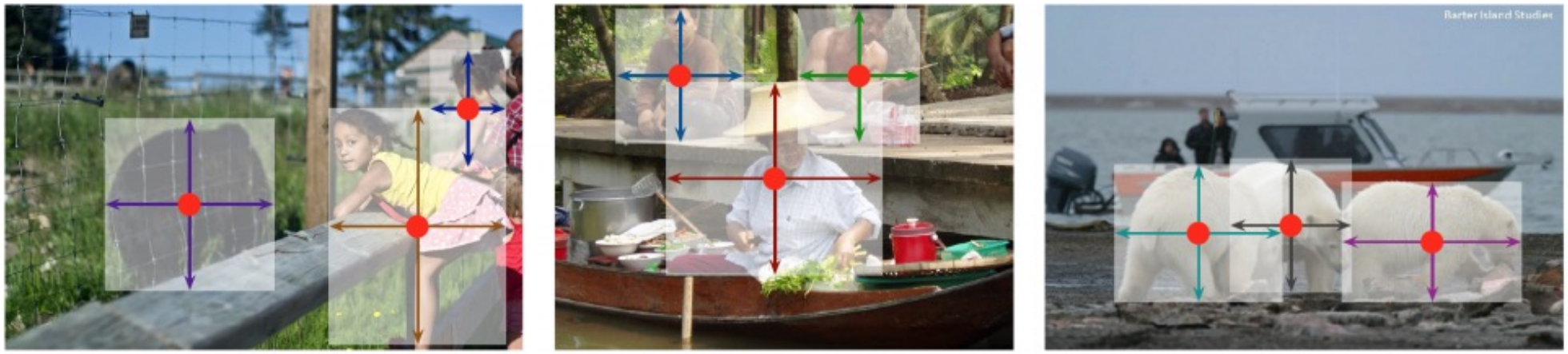


Figure 2: We model an object as the center point of its bounding box. The bounding box size and other object properties are inferred from the keypoint feature at the center. Best viewed in color.

# CenterNet Performance

- 기존 모델 대비 빠른 속도를 가지면서도 높은 성능을 보여 줌

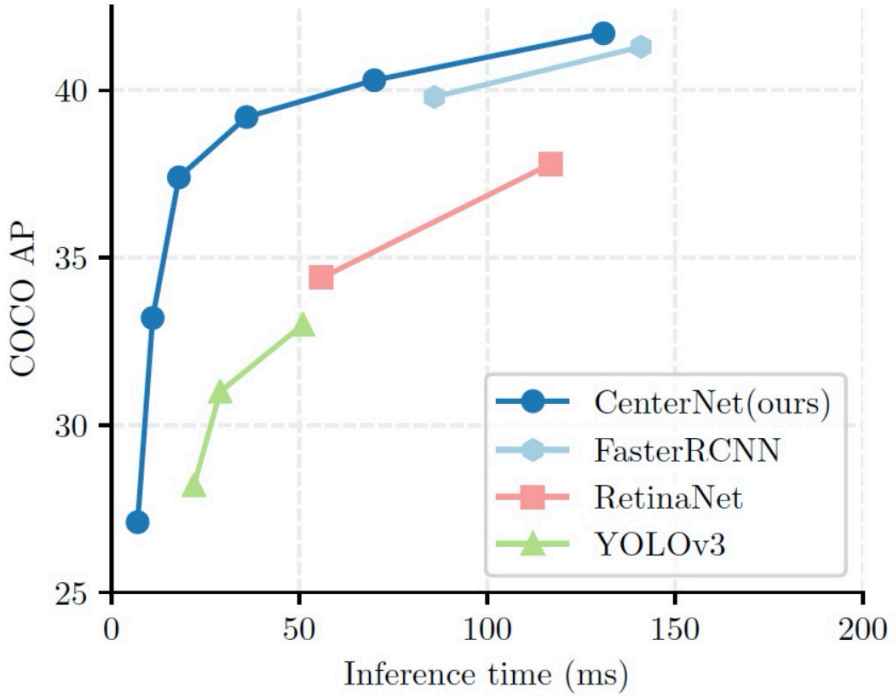



Figure 1: Speed-accuracy trade-off on COCO validation for real-time detectors. The proposed CenterNet outperforms a range of state-of-the-art algorithms.

## CenterNet의 아이디어

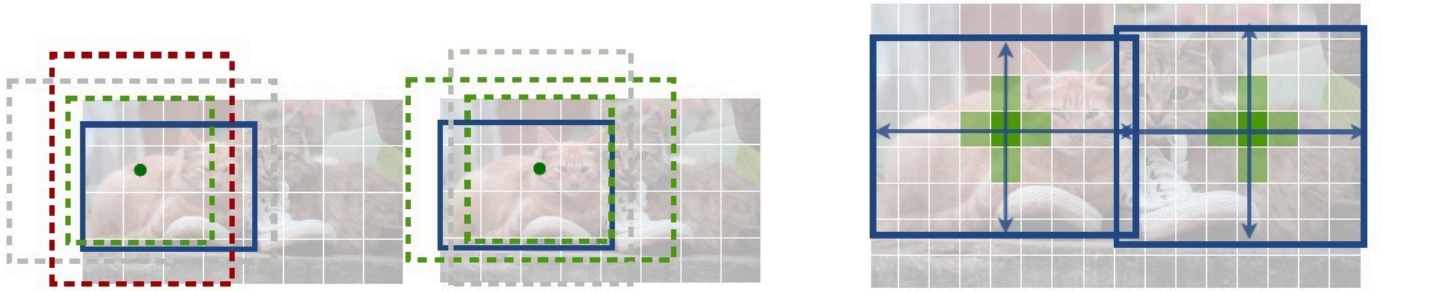
- Our approach is closely related to anchor-based one-stage approaches [33, 36, 43].
- A center point can be seen as a single shape-agnostic anchor (see Figure 3)
- However, there are a few important differences.

① our CenterNet assigns the "anchor" based solely on location, not box overlap [18]. We have no manual thresholds [18] for foreground and background classification.

② we only have one positive "anchor" per object, and hence do not need Non-Maximum Suppression (NMS) [2]. We simply extract local peaks in the keypoint heatmap [4, 39].

③ CenterNet uses a larger output resolution (output stride of 4) compared to traditional object detectors [21, 22] (output stride of 16). This eliminates the need for multiple anchors [47].

# CenterNet의 아이디어



(a) Standard anchor based detection. Anchors count as positive with an overlap $IoU > 0.7$ to any object, negative with an overlap $IoU < 0.3$, or are ignored otherwise.

(b) Center point based detection. The center pixel is assigned to the object. Nearby points have a reduced negative loss. Object size is regressed.

Figure 3: Different between anchor-based detectors (a) and our center point detector (b). Best viewed on screen.

# CenterNet의 확장

- Keypoint Estimation에서 확장할 수 있는 다양한 문제영역에 CenterNet을 사용할 수 있음

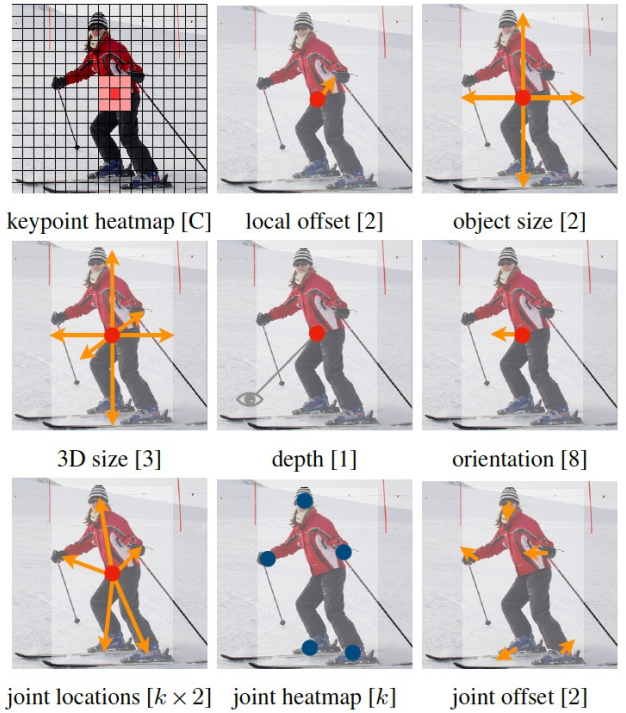



Figure 4: Outputs of our network for different tasks: *top* for object detection, *middle* for 3D object detection, *bottom:* for pose estimation. All modalities are produced from a common backbone, with a different $3 \times 3$ and $1 \times 1$ output convolutions separated by a ReLU. The number in brackets indicates the output channels. See section 4 for details.

# Loss Function

- The **overall training objective** is

$$L_{det} = L_k + \lambda_{size} L_{size} + \lambda_{off} L_{off}. \qquad (4)$$

- We set $\lambda_{size} = 0.1$ and $\lambda_{off} = 1$ in all our experiments unless specified otherwise.
- We use a single network to predict the keypoints $\hat{Y}$, offset $\hat{O}$, and size $\hat{S}$.
- The network predicts a total of **C + 4 outputs** at each location.

keypoint heatmap [C] local offset [2] object size [2]

## Keypoint prediction

- Keypoint heatmap prediction값 $\hat{Y}_{xyc}$는 x,y 좌표가 해당 class c의 keypoint 일 경우 $\hat{\mathbf{Y}}_{\mathbf{xyc}} = \mathbf{1}$, background일 경우 $\hat{\boldsymbol{Y}}_{\boldsymbol{xyc}} = \mathbf{0}$ 을 출력한다.
- 이를 이용해서 Keypoint에 대한 Loss $L_k$를 Focal Loss를 이용해서 계산한다.
- N은 Image I의 Keypoint 개수를 나타낸다.

$$L_k = \frac{-1}{N} \sum_{xyc} \begin{cases} (1 - \hat{Y}_{xyc})^{\alpha} \log(\hat{Y}_{xyc}) & \text{if } Y_{xyc} = 1 \\ (1 - Y_{xyc})^{\beta} (\hat{Y}_{xyc})^{\alpha} & \\ \log(1 - \hat{Y}_{xyc}) & \text{otherwise} \end{cases} \quad (1)$$

## Offset Prediction

- 다음으로 discretization에 의해 생긴 error를 보정하기 위해 Keypoint에 대한 offset x, y 보정값 $\hat{O}$를 예측한다.
- Offset에 대한 Loss $L_{off}$는 아래와 같이 계산한다.

$$L_{off} = \frac{1}{N} \sum_{p} \left| \hat{O}_{\tilde{p}} - \left( \frac{p}{R} - \tilde{p} \right) \right|. \qquad (2)$$

- Keypoint 위치 $\tilde{p}$ 에서만 Loss가 계산되고 나머지 위치는 무시된다.

## Size Prediction

- 마지막으로 Bounding box에 대한 width와 height값 $\hat{S}$를 예측한다.
- Size에 대한 Loss $L_{size}$는 아래와 같이 계산한다.

$$L_{size} = \frac{1}{N} \sum_{k=1}^{N} \left| \hat{S}_{p_k} - s_k \right| . \qquad (3)$$

## Training

- We train on an input resolution of 512 x 512.
- This yields an output resolution of 128x128 for all the models.
- We use **random flip, random scaling** (between 0.6 to 1.3), **cropping**, and color **jittering as data augmentation**, and use Adam [28] to optimize the overall objective.
- we train with a batch-size of 128 (on 8 GPUs) and learning rate 5e-4 for 140 epochs, with learning rate dropped 10 at 90 and 120 epochs, respectively (following [55]).
- Resnet-101 and DLA-34 train in 2.5 days on 8 TITAN-V GPUs, while Hourglass-104 requires 5 days.

# CenterNet Performance

| | Backbone | FPS | $AP$ | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|---|---|
| MaskRCNN [21] | ResNeXt-101 | **11** | 39.8 | 62.3 | 43.4 | 22.1 | 43.2 | 51.2 |
| Deform-v2 [63] | ResNet-101 | - | 46.0 | 67.9 | 50.8 | 27.8 | 49.1 | 59.5 |
| SNIPER [48] | DPN-98 | *2.5* | 46.1 | 67.0 | 51.6 | 29.6 | 48.9 | 58.1 |
| PANet [35] | ResNeXt-101 | - | 47.4 | 67.2 | 51.8 | 30.1 | **51.7** | 60.0 |
| TridentNet [31] | ResNet-101-DCN | 0.7 | **48.4** | **69.7** | **53.5** | **31.8** | 51.3 | **60.3** |
| YOLOv3 [45] | DarkNet-53 | 20 | 33.0 | 57.9 | 34.4 | 18.3 | 25.4 | 41.9 |
| RetinaNet [33] | ResNeXt-101-FPN | 5.4 | 40.8 | 61.1 | 44.1 | 24.1 | 44.2 | 51.2 |
| RefineDet [59] | ResNet-101 | - | 36.4 / 41.8 | 57.5 / 62.9 | 39.5 / 45.7 | 16.6 / 25.6 | 39.9 / 45.1 | 51.4 / 54.1 |
| CornerNet [30] | Hourglass-104 | 4.1 | 40.5 / 42.1 | 56.5 / 57.8 | 43.1 / 45.3 | 19.4 / 20.8 | 42.7 / 44.8 | **53.9** / 56.7 |
| ExtremeNet [61] | Hourglass-104 | 3.1 | 40.2 / 43.7 | 55.5 / 60.5 | 43.2 / 47.0 | 20.4 / 24.1 | 43.2 / 46.9 | 53.1 / 57.6 |
| FSAF [62] | ResNeXt-101 | *2.7* | **42.9** / 44.6 | **63.8** / **65.2** | **46.3** / 48.6 | **26.6** / **29.7** | **46.2** / **47.1** | 52.7 / 54.6 |
| CenterNet-DLA | DLA-34 | **28** | 39.2 / 41.6 | 57.1 / 60.3 | 42.8 / 45.1 | 19.9 / 21.5 | 43.0 / 43.9 | 51.4 / 56.0 |
| CenterNet-HG | Hourglass-104 | 7.8 | 42.1 / **45.1** | 61.1 / 63.9 | 45.9 / **49.3** | 24.1 / 26.6 | 45.5 / **47.1** | 52.8 / **57.7** |

Table 2: State-of-the-art comparison on COCO test-dev. Top: two-stage detectors; bottom: one-stage detectors. We show single-scale / multi-scale testing for most one-stage detectors. Frame-per-second (FPS) were measured on the same machine whenever possible. Italic FPS highlight the cases, where the performance measure was copied from the original publication. A dash indicates methods for which neither code and models, nor public timings were available.

# CenterNet Performance (Backbone 변화에 따른 성능차이)

- Backbone 변화에 따른 성능차이

| | AP | | | $AP_{50}$ | | | $AP_{75}$ | | | Time (ms) | | | FPS | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | N.A. | F | MS | N.A. | F | MS | N.A. | F | MS | N.A. | F | MS | N.A. | F | MS |
| Hourglass-104 | **40.3** | **42.2** | **45.1** | **59.1** | **61.1** | **63.5** | **44.0** | **46.0** | **49.3** | 71 | 129 | 672 | 14 | 7.8 | 1.4 |
| DLA-34 | 37.4 | 39.2 | 41.7 | 55.1 | 57.0 | 60.1 | 40.8 | 42.7 | 44.9 | 19 | 36 | 248 | 52 | 28 | 4 |
| ResNet-101 | 34.6 | 36.2 | 39.3 | 53.0 | 54.8 | 58.5 | 36.9 | 38.7 | 42.0 | 22 | 40 | 259 | 45 | 25 | 4 |
| ResNet-18 | 28.1 | 30.0 | 33.2 | 44.9 | 47.5 | 51.5 | 29.6 | 31.6 | 35.1 | **7** | **14** | **81** | **142** | **71** | **12** |

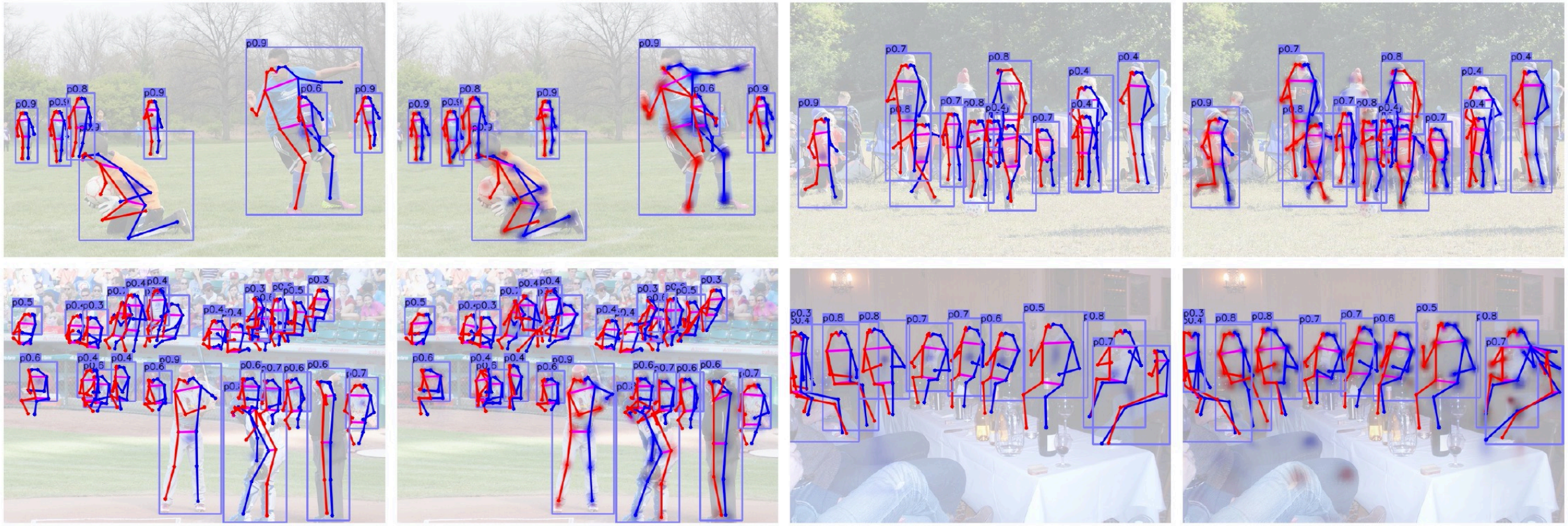
Table 1: Speed / accuracy trade off for different networks on COCO validation set. We show results without test augmentation (N.A.), flip testing (F), and multi-scale augmentation (MS).

# CenterNet Performance – Object Detection

- Object Detection 문제영역에 대한 CenterNet 예측결과

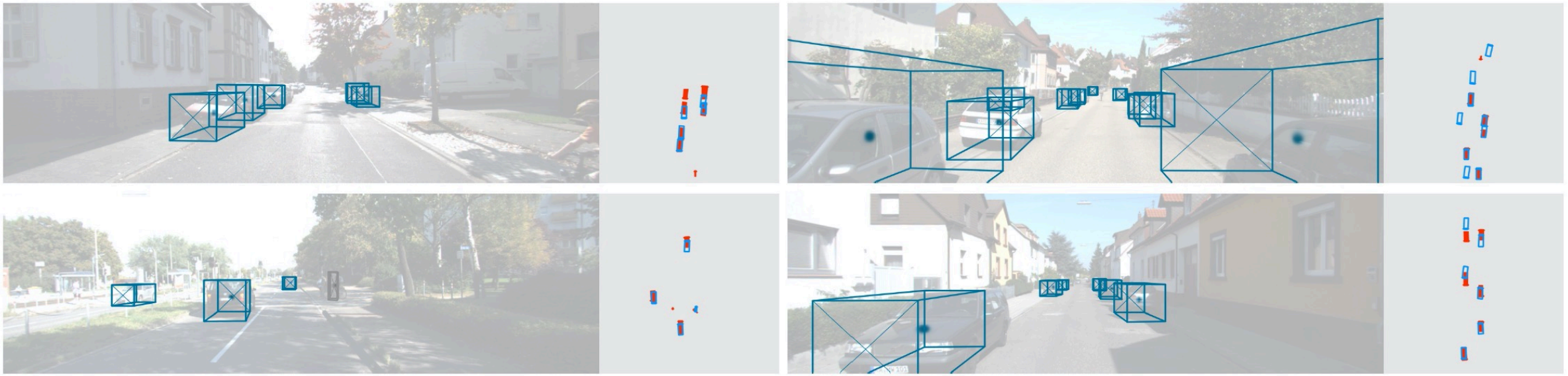
# CenterNet Performance – Human Pose Estimation

- Human Pose Estimation 문제영역에 대한 CenterNet 예측결과



# CenterNet Performance – 3D Bounding Box Estimation

- 3D Bounding Box Estimation 문제영역에 대한 CenterNet 예측결과



## CenterNet의 의의

- **장점** :

① NMS(Non-Maximum Supression) 과정이 필요없는 빠른 속도의 창의적인 Object Detection 모델을 제안함

② 여러개의 Anchor Prediction으로 발생할 수 있는 중복 Prediction 문제를 해결함

# TensorFlow Detection Model ZOO에서 제공하는 CetnerNet 모델들

- https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

| Model name | Speed (ms) | COCO mAP | Outputs |
|---|---|---|---|
| CenterNet HourGlass104 512x512 | 70 | 41.9 | Boxes |
| CenterNet HourGlass104 Keypoints 512x512 | 76 | 40.0/61.4 | Boxes/Keypoints |
| CenterNet HourGlass104 1024x1024 | 197 | 44.5 | Boxes |
| CenterNet HourGlass104 Keypoints 1024x1024 | 211 | 42.8/64.5 | Boxes/Keypoints |
| CenterNet Resnet50 V1 FPN 512x512 | 27 | 31.2 | Boxes |
| CenterNet Resnet50 V1 FPN Keypoints 512x512 | 30 | 29.3/50.7 | Boxes/Keypoints |
| CenterNet Resnet101 V1 FPN 512x512 | 34 | 34.2 | Boxes |
| CenterNet Resnet50 V2 512x512 | 27 | 29.5 | Boxes |
| CenterNet Resnet50 V2 Keypoints 512x512 | 30 | 27.6/48.2 | Boxes/Keypoints |
| CenterNet MobileNetV2 FPN 512x512 | 6 | 23.4 | Boxes |
| CenterNet MobileNetV2 FPN Keypoints 512x512 | 6 | 41.7 | Keypoints |

# Thank you!