Object Instance Detection Challenge

Quick links

- EvalAI page: visit this page to submit results
- challenge codebase in github: use this git repo to prepare results for submission
- dataset in google drive: use this link to download data
- test-dataset in google drive: use this link to download scene images in test phase

Qianqian Shen, Yunhan Zhao, Shu Kong

Should you have questions, please contact Qianqian Shen (shenqq377@gmail.com) or Yunhan Zhao (yzhao117@gmail.com).

Instance Detection (InsDet) is a practically important task in robotics and AR/VR. For example, For example, a robot can be commanded to search for a customer's suitcase at an airport or my-keys in my cluttered bedroom. Different from Object Detection (ObjDet) which detects all objects belonging to some predefined classes, InsDet aims to detect specific object instances defined by some visual examples.

This challenge focuses on the recently introduced InsDet dataset, which is larger in scale and more challenging than existing InsDet datasets. The major strengths of our InsDet dataset includes (1) both high-resolution profile images of object instances and high-resolution testing images from more realistic indoor scenes, simulating real-world indoor robots locating and recognizing object instances from a cluttered indoor scene in a distance (2) a realistic unified InsDet protocol to foster the InsDet research.

Participants in this challenge will be tasked with predicting the bounding boxes for each given instance from testing images. This exciting opportunity allows researchers, students, and data scientists to apply their expertise in computer vision and machine learning to address instance detection problem.

Important Dates

- March 24, 2025, **InsDet** (6.60 GB) will be released. Please download the dataset from Google Drive.
- March 24, 2025, **EvalAI server** will be open.
- May 1, 2025, **Scenes-Test** (2.43 GB) will be released. Please download the dataset from Google Drive.
- May 5, 2025, **Evaluation scripts** will be uploaded to the challenge github repo.
- June 5, 2025, Challenge will be closed.
- June 6, 2025, Results will be released and participants will be selected to present.
- June 11, 2025, Workshop starts.

Dataset

The **InsDet** dataset contains 100 object instances with multi-view profile images, 200 pure background images and 160 scene images. Participants can download the dataset from the InsDet dataset.

1. **Objects:** Our dataset contains 100 different instances. Each profile image has a resolution of 3072×3072 pixels (some instances are 3456×3456). Each instance is captured at 24 rotation positions (every 15° in azimuth) with a 45° elevation view. When capturing profile images for each instance, inspired by prior arts, we paste a QR code on the tabletop, which enables pose estimation, e.g., using COLMAP. We use the GrabCut toolbox to derive foreground masks of instances in profile images. This removes background pixels (such as QR code regions) in the profile images.

```
Instance folder: raw_data
|-- 000_aveda_shampoo
    |-- images
        |-- 001.jpg
        |-- 002.jpg
        ...
        |-- 024.jpg
    |-- masks
        |-- 001.jpg
        |-- 002.jpg
        ...
        |-- 024.jpg
|-- 001_binder_clips_median
...
|-- 099_mug_blue
```

   In practice, we center-crop foreground instances from profile images and downsize the center-crops to 1024×1024, and save images and masks.

2. **Background images:** 200 high-resolution background images of indoor scenes that do not include any given instances from Objects.

```
Background folder
|-- Background 001.png
|-- Background 002.png
...
|-- Background 100.png
```

3. **Scenes:** 160 high-resolution images (6144×8192) in cluttered scenes, where some instances are placed in reasonable locations. We tag these images as easy or hard based on scene clutter and object occlusion levels. We also provide a JSON file of all annotations in the dataset.

```
Scene folder
|-- easy
    |-- leisure_zone_001
        |-- rgb_000.jpg
        |-- rgb_000.xml
        ...
```

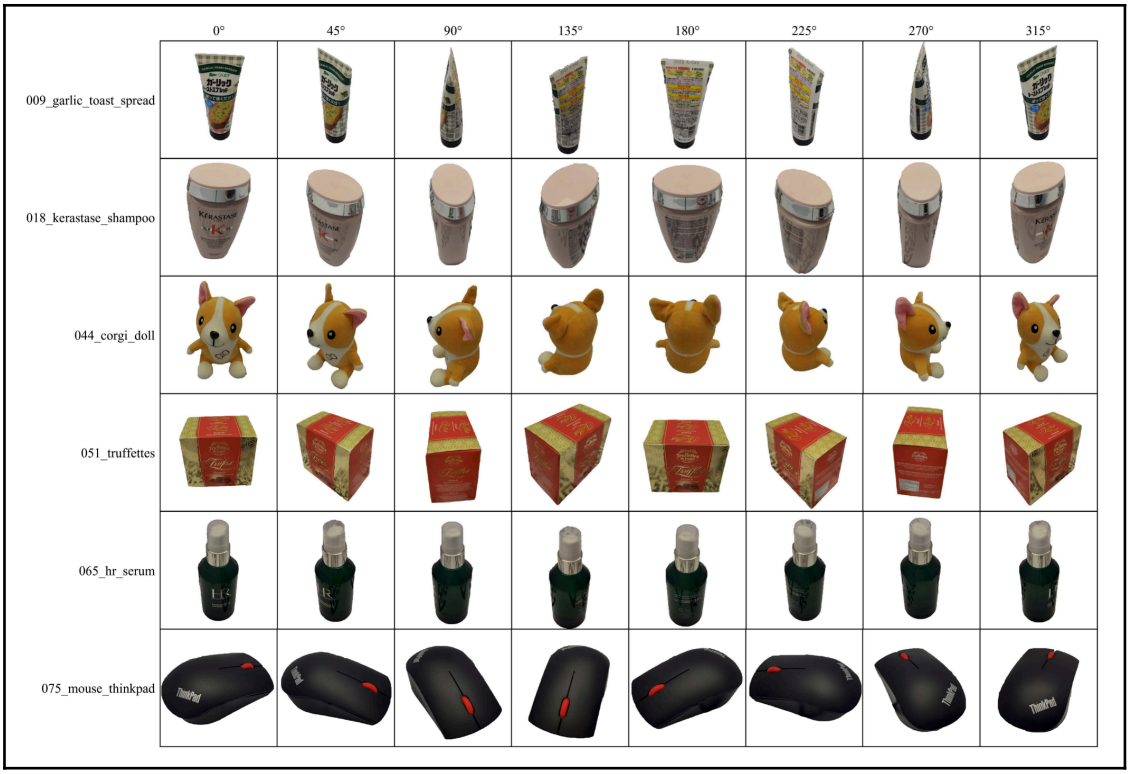
```
    |-- meeting_room_001
    |-- office_002
    |-- pantry_room_002
    |-- sink
|-- hard
    |-- office_001
    |-- pantry_room_001
|-- instance_dev.json
```

The **Scenes-Test** has the same structure of **Scenes**, but most images are captured from different scenarios. This part includes 320 high-resolution images (6144×8192) in cluttered scenes.

Benchmarking Protocols

1. **Goal.** Developing instance detectors using profile images (cf. visuals below) and optionally some random background images (e.g., to apply cut-paste-learn [1]). The detector should detect object instances of interest in real-world testing images.

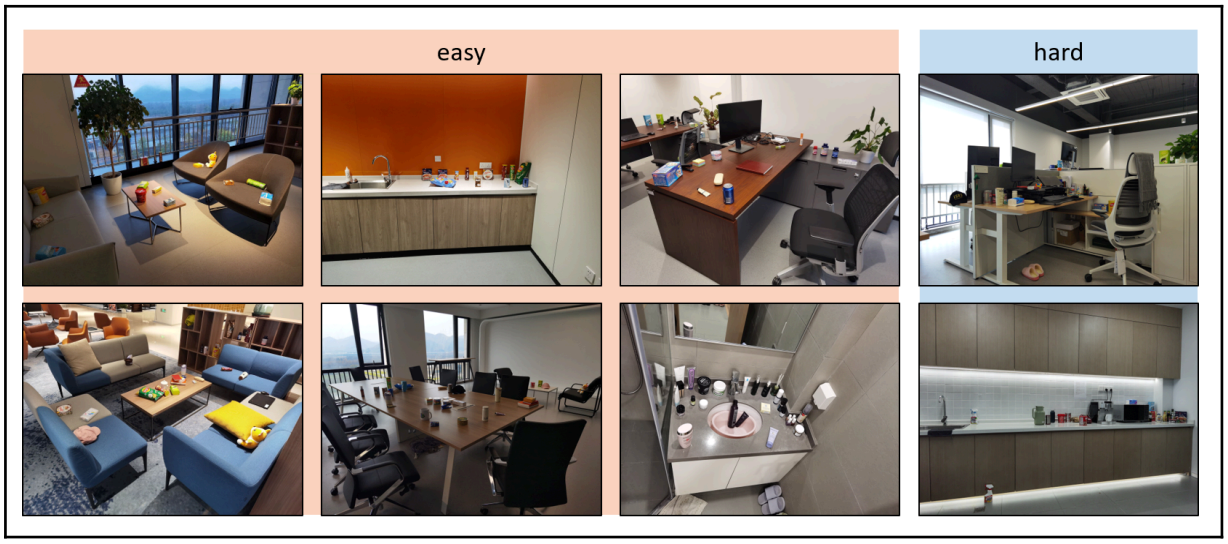
2. **Environment for model development.**
    a. A set of object instances (examples are shown below), each of which has some visual examples captured from multiple views. Participants should develop a model to successfully detect these object instances.



    b. Some random background images (not used in testing). Participants might use them to apply the cut-paste-learn strategy [1]. Participants can also download and use other external background images in training.



3. **Environment for testing.** Real-world indoor scene images, in which participants' algorithms should detect object instances of interest. We provide a few examples of scene images below.



Importantly, participants are not allowed to use the Real-world indoor scene images (i.e., the testing images) to develop models. Furthermore, for participants who are invited for a presentation, we might ask for your code and models for verification. For questions regarding technical details, feel free to submit issues on the challenge Github repo.

Evaluation Metric

Following the COCO dataset [8], we tag testing object instances as small, medium, and large according to their bounding box area. The following 12 metrics are used for characterizing the performance of an instance detector on **InsDet** dataset. Additionally, we will also evaluate AP on *easy* and *hard* scenes separately.

Average Precision (AP):

AP	% AP at IoU=0.50:0.05:0.95
$AP^{IoU=0.50}$	% AP at IoU=0.50
$AP^{IoU=0.75}$	% AP at IoU=0.75

AP Across Scenes:

AP^{easy}	% AP for easy scenes
AP^{hard}	% AP for hard scenes

AP Across Scales:

AP^{small}	% AP for small instances: area < 200^2
AP^{medium}	% AP for medium instances: 200^2 < area < 400^2
AP^{large}	% AP for large instances: area > 400^2

Average Recall (AR):

$AR^{max=1}$	% AR given 1 detection per image
$AR^{max=10}$	% AR given 10 detection per image
$AR^{max=100}$	% AR given 100 detection per image

AR Across Scales:

AR^{small}	% AR for small instances: area < 200^2
AR^{medium}	% AR for medium instances: 200^2 < area < 400^2
AR^{large}	% AR for large instances: area > 400^2

Note:

1. **AP** is average precision over all instances, which is traditionally called mean average precision (mAP).

Submission

1. Output format: One JSON or CSV file of predicted bounding boxes of all test images in a COCO compatible format.


```
[
{"image_id": 0, "category_id": 79, "bbox": [976, 632, 64, 80], "score": 99.32915569311469,
"image_width": 8192, "image_height": 6144, "scale": 1,
"image_name": "easy.leisure_zone.rgb_000.jpg"}, ...
{"image_id": 2, "category_id": 18, "bbox": [323, 0, 1724, 237], "score": 69.3080951903575,
"image_width": 8192, "image_height": 6144, "scale": 1,
"image_name": "easy.leisure_zone.rgb_002.jpg"}, ...
{"image_id": 4, "category_id": 75, "bbox": [1236, 1000, 62, 101], "score":
69.3080951903575, "image_width": 8192, "image_height": 6144, "scale": 1,
"image_name": "easy.leisure_zone.rgb_004.jpg"},
...
{"image_id": 159, "category_id": 9, "bbox": [921, 803, 28, 106], "score":
99.32927090665571, "image_width": 8192, "image_height": 6144, "scale": 1,
"image_name": "hard.pantry_room_001.rgb_019.jpg"}
]
```


Phase

1. **Development Phase**
   Strats on: March 24, 2025 00:00:00 AM GMT
   Ends on: May 30, 2025 11:59:59 PM GMT

   We use testing images in **Scenes** to help participants test how to upload the JSON file for benchmarking. We further provide python scripts in the the challenge Github repo to walk you through how to preprocess raw data in our dataset and how to evaluate the results w.r.t the evaluation metrics.

2. **Test Phase**
   Strats on: March 24, 2025 00:00:00 AM GMT
   Ends on: June 5, 2025 11:59:59 PM GMT

   This is the benchmarking phase. We ask participants to upload a JSON or CSV file that scores for all the testing images in **Scenes-Test**.

Official Baseline

1. **Faster RCNN [2] + Cut-Paste-Learn strategy [1].**
   We train Faster RCNN with data generated using the Cut-Paste-Learn strategy. The detailed hyperparameters used in data generation and model training are mentioned in the paper [3].

2. **SAM [4] + DINOv2 [5] + StableMatching [6,7].**
   We proposed a simple non-learned method using the off-the-shelf class-agnostic segmentation model (Segment Anything Model, SAM), the self-supervised feature representation DINOv2 and classical StableMatching algorithm in the paper [3].

Github Code

Participants can use the challenge Github repo to start. For questions regarding technical details, please submit issues on the challenge repo. Should you have questions, please contact Yunhan Zhao (yzhao117@gmail.com) and Qianqian Shen (shenqq377@gmail.com).

Evaluation Server

We use EvalAI as the evaluation server. Please visit the challenge page here.

Reference

[1] Dwibedi, Debidatta, et al. "Cut, paste and learn: Surprisingly easy synthesis for instance detection." Proceedings of the IEEE international conference on computer vision. 2017.

[2] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

[3] Shen, Qianqian, et al. “A High-Resolution Dataset for Instance Detection with Multi-View Instance Capture.” Thirty-seventh conference on neural information processing systems datasets and benchmarks track. 2023.

[4] Kirillov, Alexander, et al. "Segment anything." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023.

[5] Oquab, Maxime, et al. "Dinov2: Learning robust visual features without supervision." arXiv preprint arXiv:2304.07193 (2023).

[6] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. The American Mathematical Monthly, 69(1):9–15, 1962.

[7] David G McVitie and Leslie B Wilson. The stable marriage problem. Communications of the ACM, 14(7):486–490, 1971.

[8] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In ECCV, 2014.