Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo and Ross Girshick and many other contributors

https://github.com/facebookresearch/detectron2
A research platform and a production library for object detection, mainly built by Facebook AI Research (FAIR)

https://github.com/facebookresearch/detectron2
What is **Object Detection**?
This Page Contains the Demo Video at

https://ai.facebook.com/blog/-detectron2-a-pytorch-based-modular-object-detection-library-
What is **Object Detection**?

recognize, localize, and predict attributes of objects in images
Family of Detection Codebase

- Caffe official fast RCNN 02/2015
- Caffe official faster RCNN 08/2015
- Caffe py-faster-rcnn
- Detectron 01/2018
- Detectron.pytorch 03/2018
- maskrcnn-benchmark 10/2018
- Detectron2 10/2019

- TF object detection API 06/2017
- Matterport/Mask_RCNN 11/2017
- Tensorpack R-CNN
- SimpleDet 01/2019
- mmdetetection 09/2018
- SimpleDet 01/2019
- SimpleDet 10/2019
Detectron ➔ Detectron2

FASTER
PYTORCH
Accurate
Modular
What’s in Detectron2: training speed

Training Throughput of R50-FPN Mask R-CNN

Details at: detectron2.readthedocs.io/notes/benchmarks.html
What’s in Detectron2: accuracy

• Latest SOTA models included
• Same model, more accurate
  • By fixing legacy issues

Mask Average Precision for the Same Model
(R50-FPN Mask R-CNN)

- Mask R-CNN (2017 paper)
- Detectron (2018)
- Detectron2

github.com/facebookresearch/detectron2/tree/master/configs/Detectron1-Comparisons
What’s in Detectron2: model zoo

• Different settings for users to play with
• Standard baselines for researchers
• Efficient models for production (coming soon)

https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md
What’s in Detectron2: Generalized R-CNN Models
(+ a few other types of models: RetinaNet, TensorMask, etc.)
What’s in Detectron2: data / tasks

• Datasets:
  • COCO
  • LVIS
  • CityScapes
  • PascalVOC

• Tasks (data & evaluation):
  • (Rotated) Box Detection
  • {Instance, Semantic, Panoptic} Segmentation
  • Person Keypoint, DensePose

• + Your own data / models / tasks
Extend Detectron2 for Research & Production

• Hack inside the code:  `vim detectron2`
  • Quick & flexible prototyping
  • Not scalable / maintainable

• Extend existing code:  `import detectron2`
  • Some* code duplication
  • Take some time
  • Maintainable

Good research codebase should be
Hackable and Extensible
Extensible / Customizable

- Allow users to plug in custom
  - Models
  - Datasets
  - Data loading routines
  - Augmentations
  - Tasks/Metrics
  - Training logic
  - ...

without having to modify detectron2

```python
@BACKBONE_REGISTRY.register()
class MyBackbone(Backbone):
    ...

@ROI_BOX_HEAD_REGISTRY.register()
class MyBoxHead(nn.Module):
    ...

@ROI_HEADS_REGISTRY.register()
class MyHeads(StandardROIHeads):
    ...

DatasetCatalog.register(...)
```

Docs: https://detectron2.readthedocs.io/tutorials/extend.html
Colab Tutorial: fine-tune on a new dataset in 5min

Train it in Colab!
Real Example: DensePose  (detectron2/projects/DensePose)
Real Example: DensePose  (detectron2/projects/DensePose)

- New heads
- New configs
- New dataset
- New data loader
- New evaluation
- Its own training script
More Research Released by `import detectron2`

DensePose
(CVPR18)

TensorMask
(ICCV 19)

Mesh R-CNN
(ICCV 19)
coming in a month
Extensibility allows us to

• Maintain the projects separately
• Share improvements of core detectron2
• Easily collaborate with each other
• Transfer latest research to products
This Page Contains the Demo Video at
https://ai.facebook.com/blog/smart-camera-portal-advances/
import detectron2 in Production

- Domain-specific production data
- Efficient, low-latency models (open source soon)
- Large scale training on GPU clusters
- Conversion & optimization for deployment on server/mobile/edge (open source soon)
Upcoming Production Features

• R-CNN models based on efficiency-oriented architectures (FBNet, ShuffleNet, MobileNet, etc)
• Conversion, optimization & quantization with ONNX/Caffe2
• Some run in <100ms on mobile/server CPUs
Side Note: Beware of Speed/Accuracy Comparisons

Comparisons across publications/codebases are often *uncontrolled*

- Accuracy varies with hyper-parameters (‘recipe’)
- Speed varies with software (perf tuning) and hardware
- Speed varies with low-level optimization & model-specific optimization
- Speed varies with inference details (e.g., batching, quantization)
- Therefore, speed/acc. results should be taken with a large grain of salt

- We’ll release what works the best for Facebook

slides credit: Ross Girshick
Detectron2

https://github.com/facebookresearch/detectron2

Yuxin Wu, Alexander Kirillov, Francisco Massa,
Wan-Yen Lo and Ross Girshick
and many other contributors