Detectron2

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

https://github.com/facebookresearch/detectron2

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?

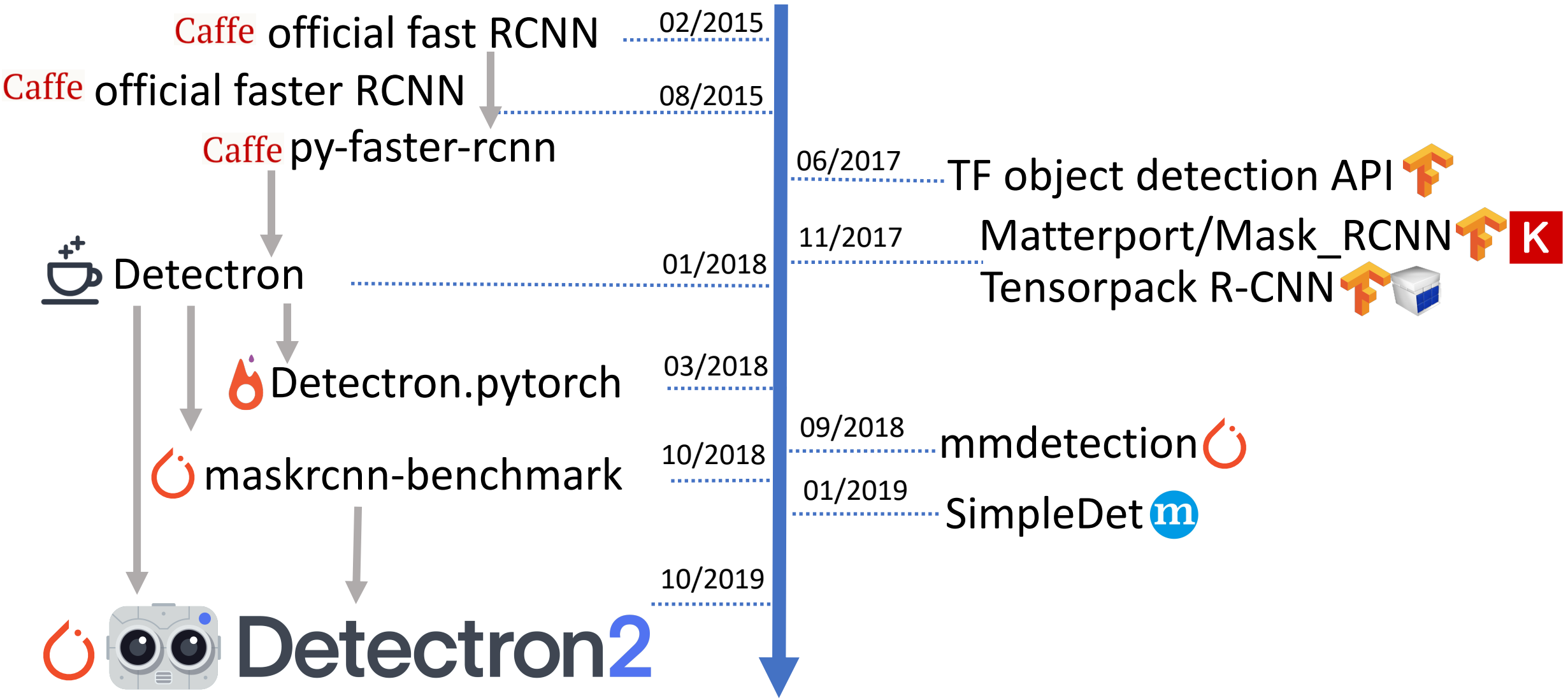
This Page Contains the Demo Video at

https://ai.facebook.com/blog/-detectron2-a-pytorch-based-modular-object-detection-library-

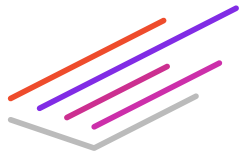
What is Object Detection?

recognize, localize, and predict attributes of objects in images

Family of Detection Codebase

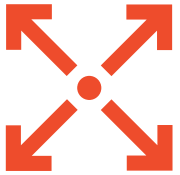


Detectron → Detectron2



FASTER

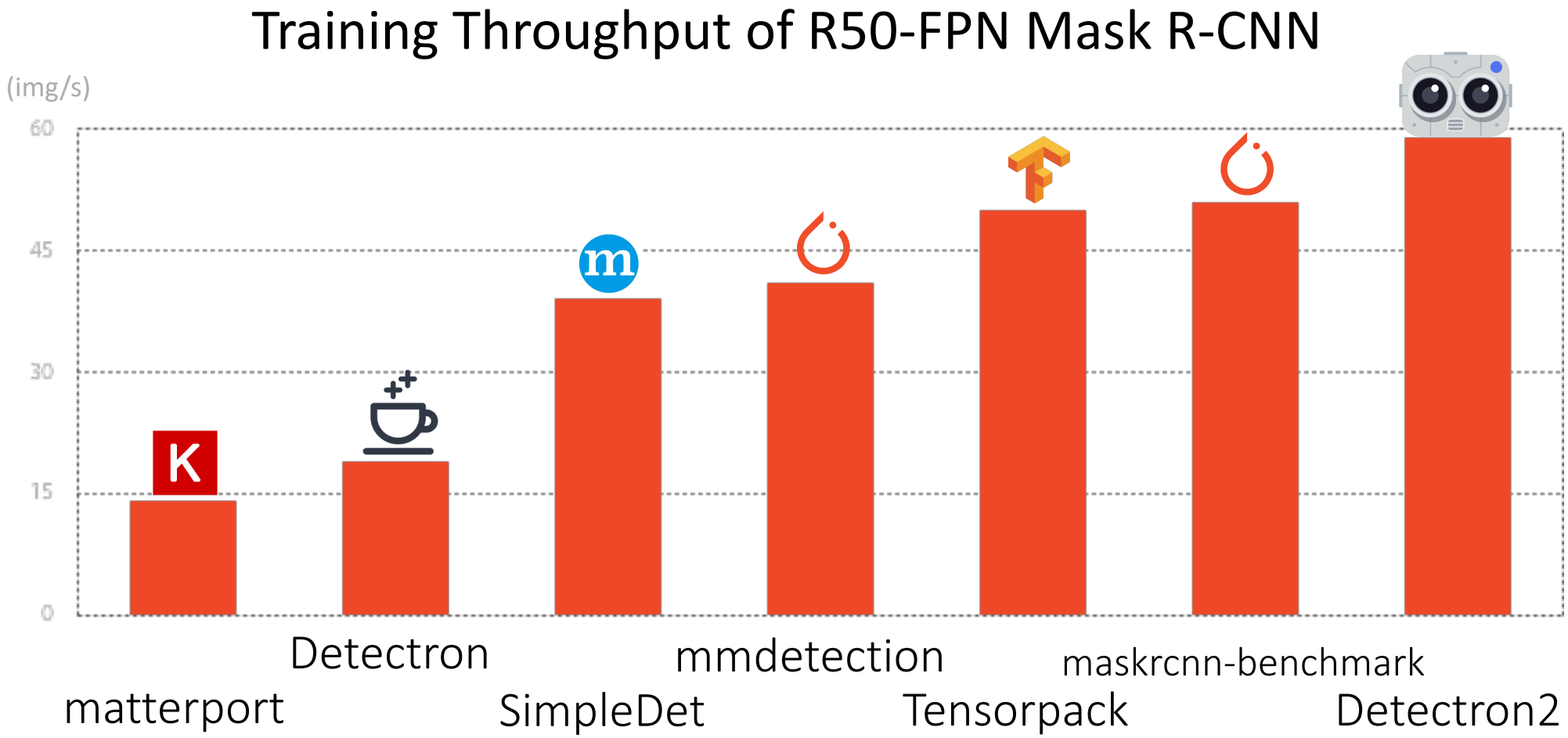
PYTORCH



Accurate

Modular

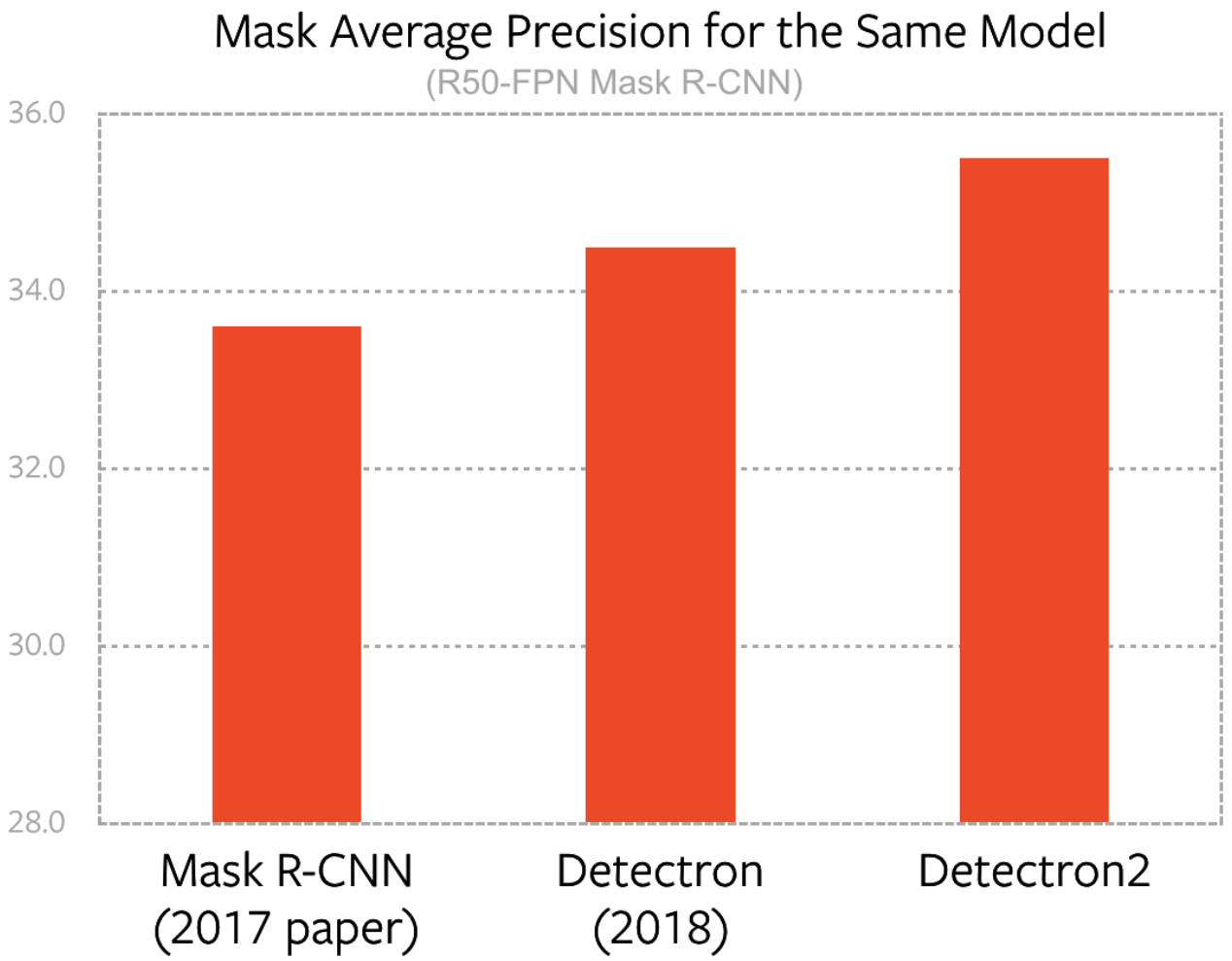
What's in Detectron2: training speed



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



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

Faster R-CNN:

Name	lr sched	train time (s/iter)	inference time (s/im)	train mem (GB)	box AP	model id	download
R50-C4	1x	0.593	0.110	4.8	35.7	137257644	model | metrics
R50-DC5	1x	0.380	0.068	5.0	37.3	137847829	model | metrics
R50-FPN	1x	0.210	0.055	3.0	37.9	137257794	model | metrics
R50-C4	3x	0.589	0.110	4.8	38.4	137849393	model | metrics
R50-DC5	3x	0.378	0.073	5.0	39.0	137849425	model | metrics
R50-FPN	3x	0.209	0.047	3.0	40.2	137849458	model | metrics
R101-C4	3x	0.656	0.149	5.9	41.1	138204752	model | metrics
R101-DC5	3x	0.452	0.082	6.1	40.6	138204841	model | metrics
R101-FPN	3x	0.286	0.063	4.1	42.0	137851257	model | metrics
X101-FPN	3x	0.638	0.120	6.7	43.0	139173657	model | metrics

RetinaNet:

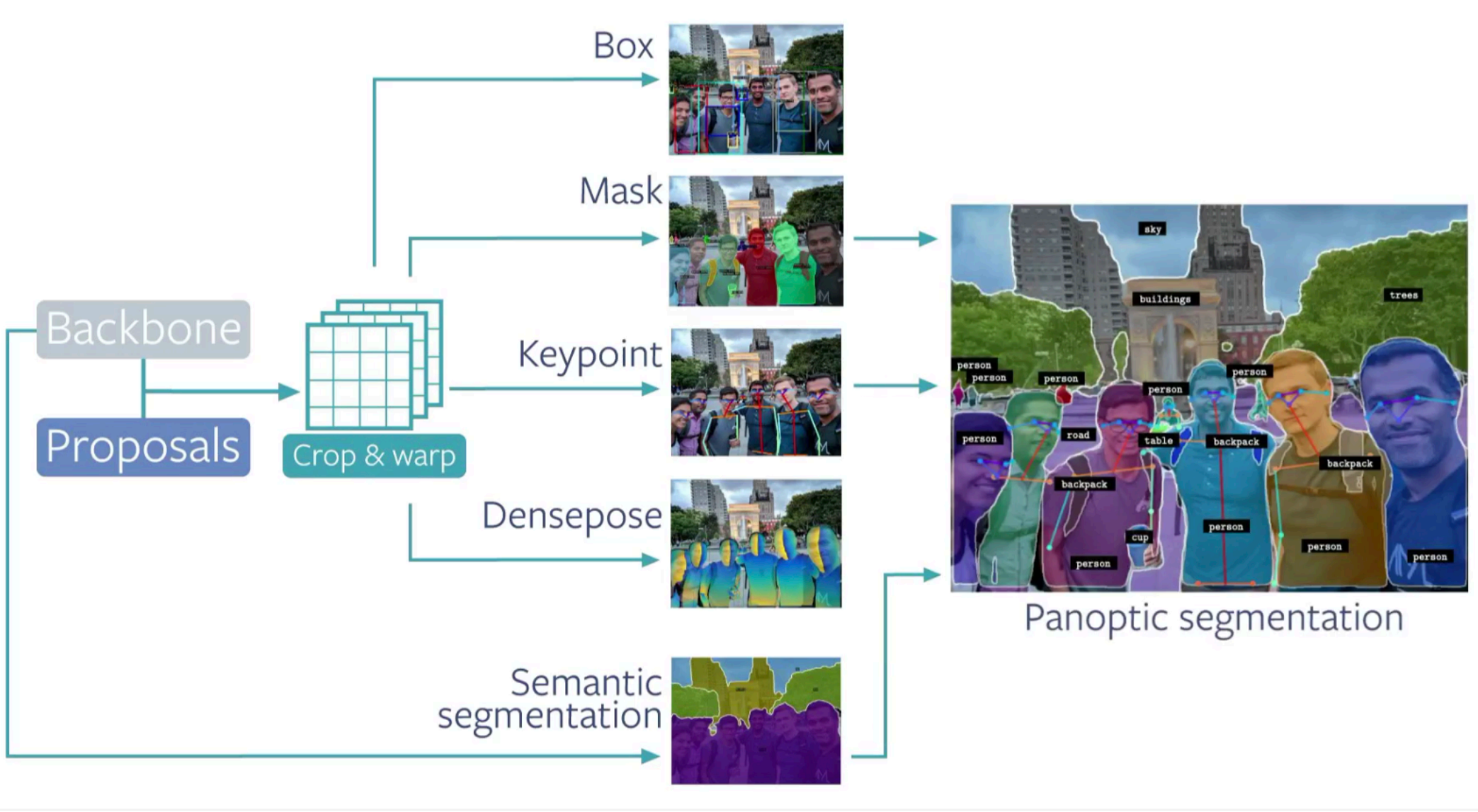
Name	lr sched	train time (s/iter)	inference time (s/im)	train mem (GB)	box AP	model id	download
R50	1x	0.200	0.062	3.9	36.5	137593951	model | metrics
R50	3x	0.201	0.063	3.9	37.9	137849486	model | metrics
R101	3x	0.280	0.080	5.1	39.9	138363263	model | metrics

RPN & Fast R-CNN:

Name	lr sched	train time (s/iter)	inference time (s/im)	train mem (GB)	box AP	prop. AR	model id	download
RPN R50-C4	1x	0.130	0.051	1.5		51.6	137258005	model | metrics
RPN R50-FPN	1x	0.186	0.045	2.7		58.0	137258492	model | metrics

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

new backbone

new box head

new type of head

new dataset

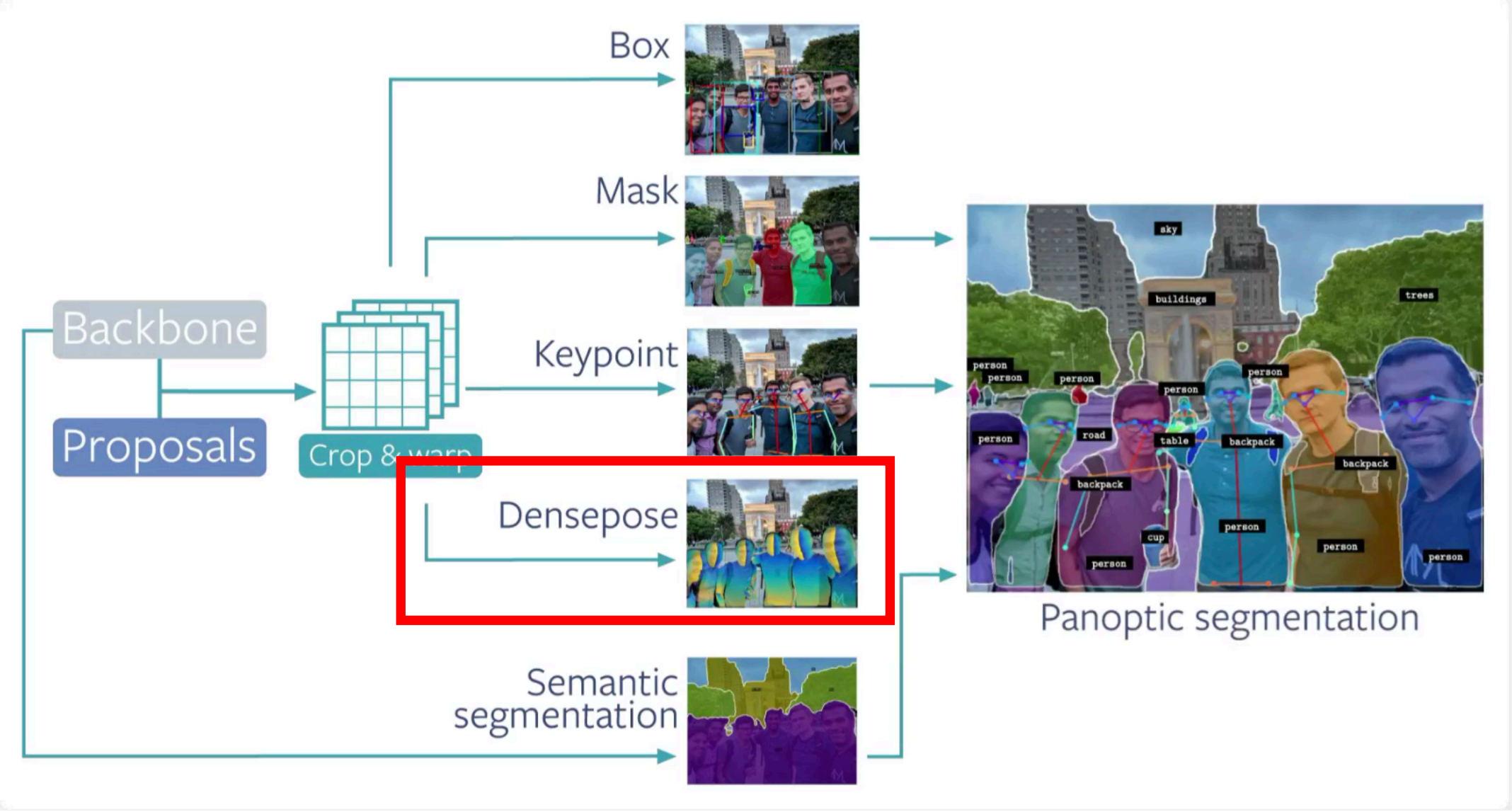
```
@BACKBONE_REGISTRY.register()
class MyBackbone(Backbone):
    ...

@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

```
├── densepose
│   ├── __init__.py
│   ├── config.py
│   ├── dataset.py
│   ├── dataset_mapper.py
│   ├── densepose_coco_evaluation.py
│   ├── densepose_head.py
│   ├── evaluator.py
│   ├── roi_head.py
│   ├── structures.py
└── train_net.py
```

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

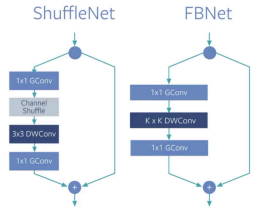
Detectron2 in Portal

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