Alibaba

FastInst: A Simple Query-Based Model for Real-Time Instance Segmentation

Junjie He, Pengyu Li, Yifeng Geng, Xuansong Xie

DAMO Academy, Alibaba Group

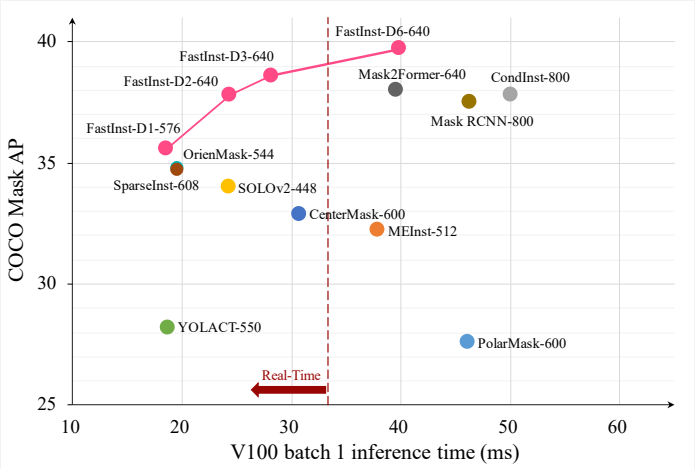
THU-PM-292

JUNE 18-22, 2023
CVPR
VANCOUVER, CANADA

FastInst: A Simple Query-Based Model for Real-Time Instance Segmentation

Junjie He, Pengyu Li, Yifeng Geng, Xuansong Xie
DAMO Academy, Alibaba Group

Introduction of FastInst



The efficient real-time instance segmentation benchmarks are still dominated by convolution-based models.

In this paper, we show the strong potential of query-based models on efficient instance segmentation algorithm designs. We present FastInst, a simple query-based framework that obtains strong performance while staying fast, surpassing most of the existing methods.

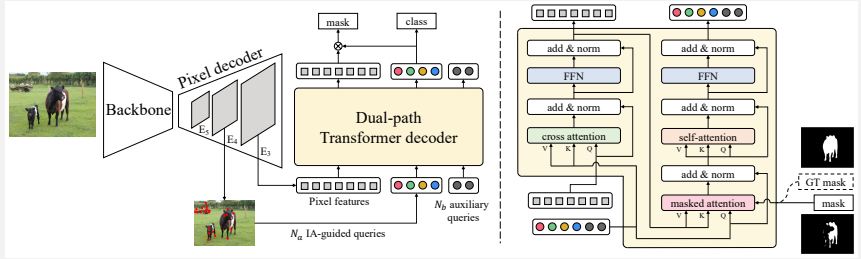
They key components contain:

- Instance activation-guided queries
- Dual-path Transformer decoder
- Ground truth mask-guided learning

Codes and pretrained models are available at

- https://modelscope.cn/models/damo/cv_resnet50_fast-instance-segmentation_coco/summary
- https://github.com/junjiehe96/FastInst

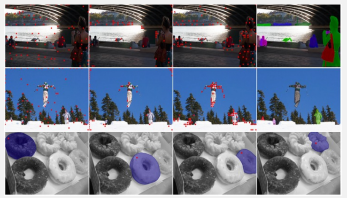
Overview of Model Architecture



Instance activation guided queries

Three simple steps:

- Dynamically pick the pixel embeddings with high semantics from the underlying feature map
- Local-maximum-first selection principle
- Location-sensitive matching during training



Dual-path Transformer decoder

Co-optimize pixel and query feats. The philosophy is inspired from EM algorithm:

- E step: update pixel features according to the centers (queries) they belong to
- M step: update the centers (queries).

Ground truth mask-guided learning

Motivation: masked attention restricts the receptive field of each query and may cause the Transformer decoder to fall into a suboptimal query update process.

So, allow each query to see the whole region of its target predicted object during training.

Specifically, use GT mask instead of predicted masks to forward Transformer decoder again, and use fixed target assignment strategy to supervise outputs.

Experimental results

- Comparison with SOTA (COCO, FPS measured on a V100 GPU)

method	backbone	epochs	size	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MEInst [47]	R50	36	512	26.4	32.2	53.9	33.0	13.9	34.4	48.7
CenterMask [24]	R50	48	600	32.6	32.9	-	-	12.9	34.7	48.7
SOLOv2 [43]	R50	36	448	41.3	34.0	54.0	36.1	10.3	36.3	54.4
OrienMask [15]	D53	100	544	51.0	34.8	56.7	36.4	16.0	38.2	47.8
SparseInst [11]	R50	144	608	51.2	34.7	55.3	36.6	14.3	36.2	50.7
YOLACT [1]	R50	54	550	53.5	28.2	46.6	29.2	9.2	29.3	44.8
FastInst-D1 (ours)	R50	50	576	**53.8**	35.6	56.7	37.2	8.8	53.0	72.8
CondInst [38]	R50	36	800	20.0	37.8	59.1	40.5	**21.0**	40.3	48.7
Mask2Former†	R50	50	640	25.3	38.0	**60.3**	39.8	10.8	54.9	74.3
FastInst-D3 (ours)	R50	50	640	35.5	**38.6**	60.2	**40.6**	10.8	**56.2**	**75.2**
YOLACT [1]	R101	54	700	33.5	31.2	50.6	32.8	12.1	33.3	47.1
FastInst-D1 (ours)	R101	50	640	**35.3**	38.3	60.2	40.5	10.7	55.8	74.8
SOLOv2 [43]	R101	36	800	15.7	39.7	60.7	**42.9**	17.3	42.9	57.4
CondInst [38]	R101	36	800	16.4	39.1	60.9	42.0	**21.5**	41.7	50.9
Mask2Former†	R101	50	640	21.1	39.5	**61.7**	41.6	11.2	56.3	75.8
FastInst-D3 (ours)	R101	50	640	28.0	**39.9**	61.5	42.3	11.4	**57.1**	**76.6**
SOLOv2 [43]	R50-DCN	36	512	32.0	37.1	57.7	39.7	12.9	40.0	57.4
YOLACT++ [2]	R50-DCN	54	550	39.4	34.1	53.3	36.2	11.7	36.1	53.6
SparseInst [11]	R50-d-DCN	144	608	46.5	37.9	59.2	40.2	**15.7**	39.4	56.9
FastInst-D1 (ours)	R50-d-DCN	50	576	**47.8**	38.0	59.7	39.9	10.0	54.9	74.5
FastInst-D3 (ours)	R50-d-DCN	50	640	32.5	**40.5**	**62.6**	**42.9**	11.9	**57.9**	**76.7**

- Unified segmentation (Cityscapes)

	backbone	Cityscapes val AP	PQ	mIoU	#Param. (M)
Mask2Former†	R50	31.4	53.9	74.4	40.9
FastInst-D3	R50	**35.5**	**56.4**	**74.7**	**34.2**

- Visualization of some prediction results

Contents

- Background
- Proposed Method
  - Instance activation guided queries
  - Dual path Transformer decoder
  - Ground truth mask guided learning
- Experiments

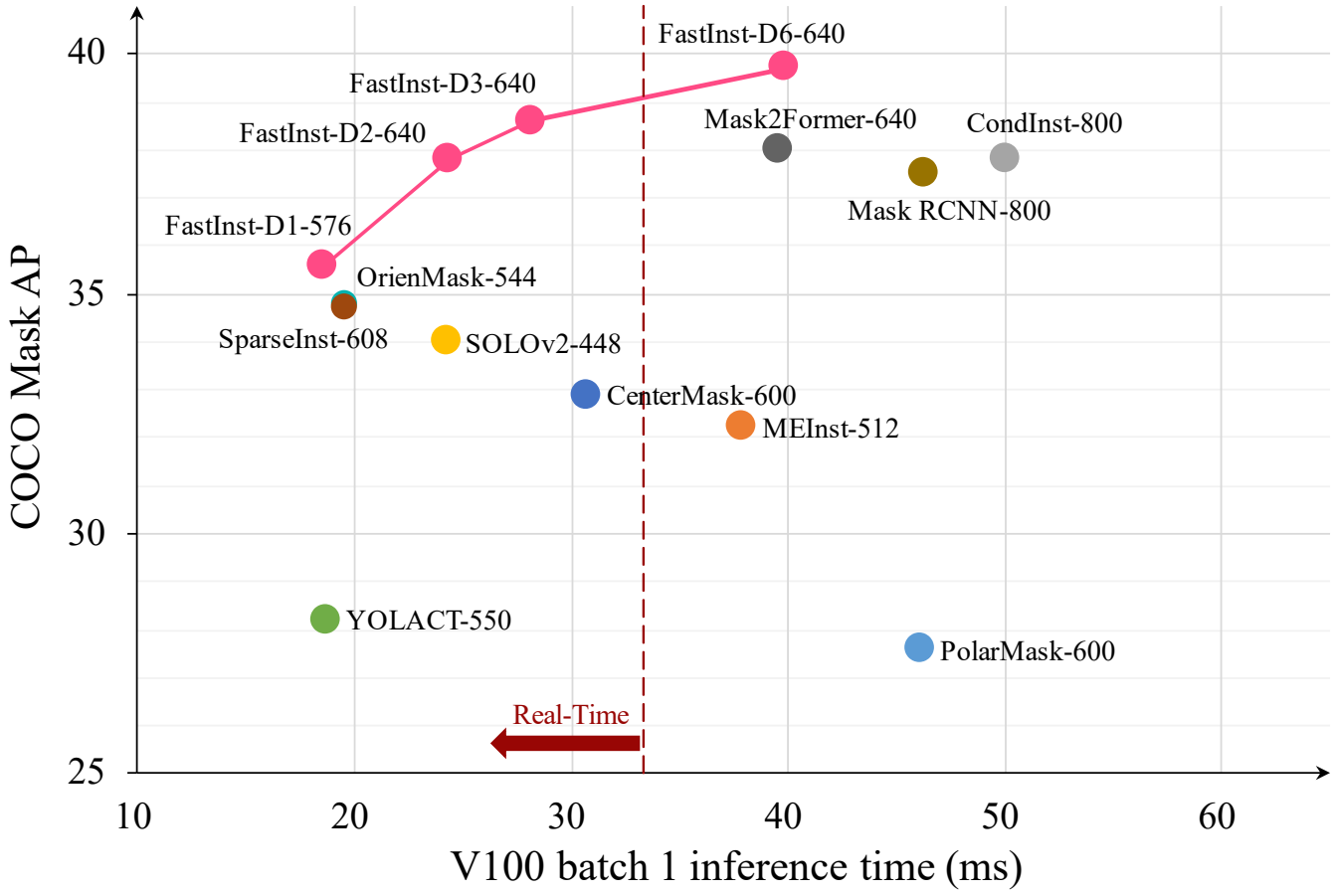
Background

- Mainstream methods like Mask R-CNN follow the design of detection-then-segmentation
  - A lot of duplicate proposals that introduce redundant computations
- Single-stage fully convolutional methods
  - Make dense predictions that still rely on manually-designed post-processing steps like NMS
- Query-based methods
  - Powerful, end-to-end, and NMS-free
  - The efficiency is general unsatisfactory —— the real-time benchmarks are still dominated by classical fully convolutional methods

Proposed Method: FastInst

- Very simple and effective method
- Close the gap between the real-time fully convolutional methods and query-based methods
- Show the great potential of query-based methods in efficient instance segmentation algorithm design

Proposed Method: FastInst



Key techniques

- Instance activation guided queries
- Dual path Transformer decoder
- Ground truth mask guided learning

Instance activation guided queries

Initial queries play a crucial role in query-based architecture

- learnable queries are image-independent and require many Transformer decoder layers to refine

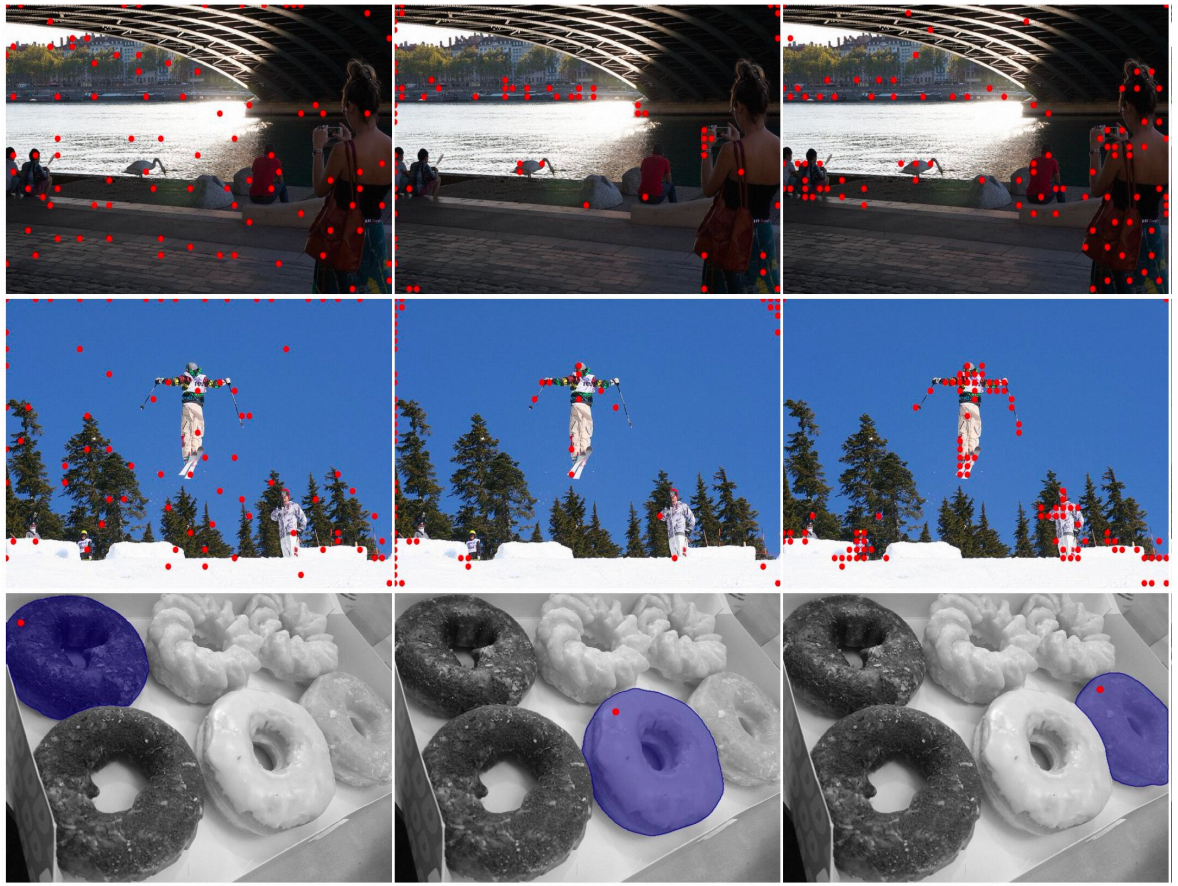
So,

- straightforwardly pick the queries with high semantics from underlying multi-scale feature maps

Instance activation guided queries

Specifically,

- predict the class probabilities for each pixel first
- then dynamically pick the pixel embeddings with high class probabilities
- to avoid redundant query selection, we adopt a local maximum first selection strategy
- during training, we employ Hungarian matching with an instance location cost to assign the learning target



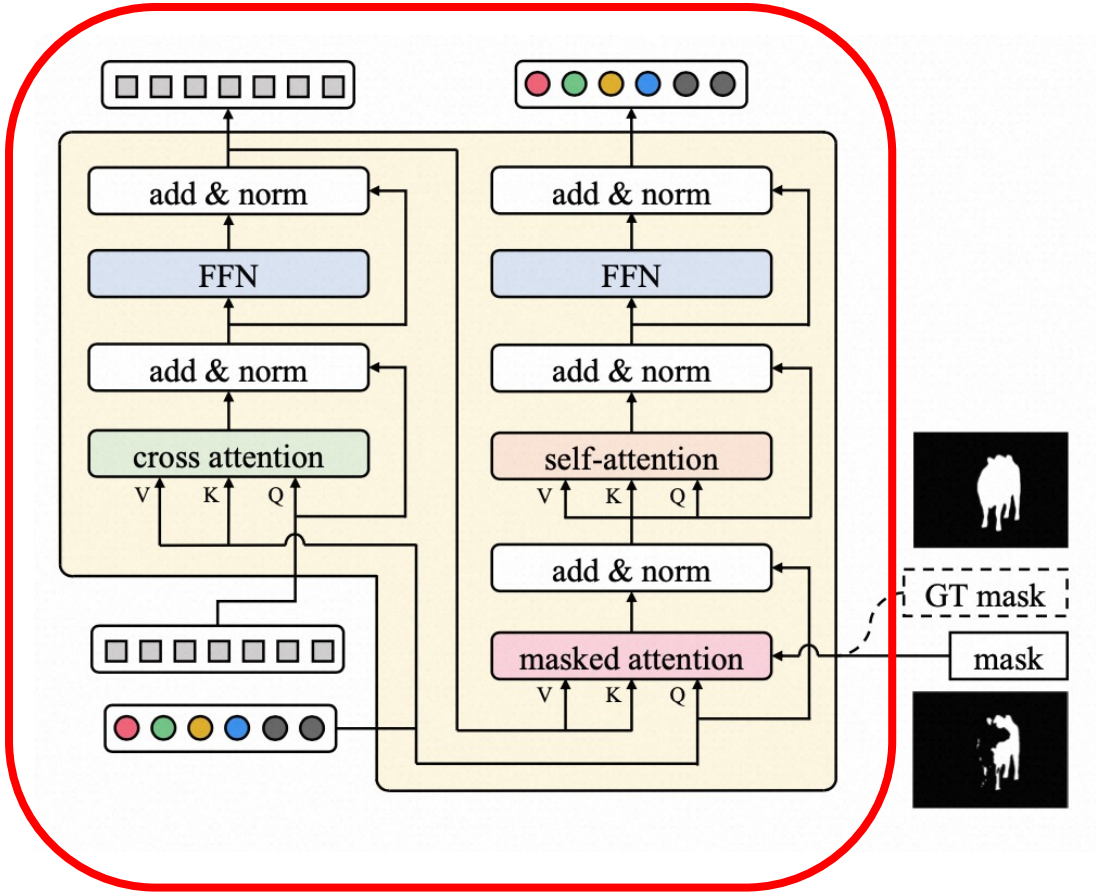
Dual-path Transformer decoder

Co-optimizes the pixel and query features,

- reduce the dependence on heavy pixel decoders
- acquire more fine-grained feature embeddings

The whole process is like an EM algorithm:

- E step: update pixel features according to the centers (i.e., queries)
- M step: update the centers



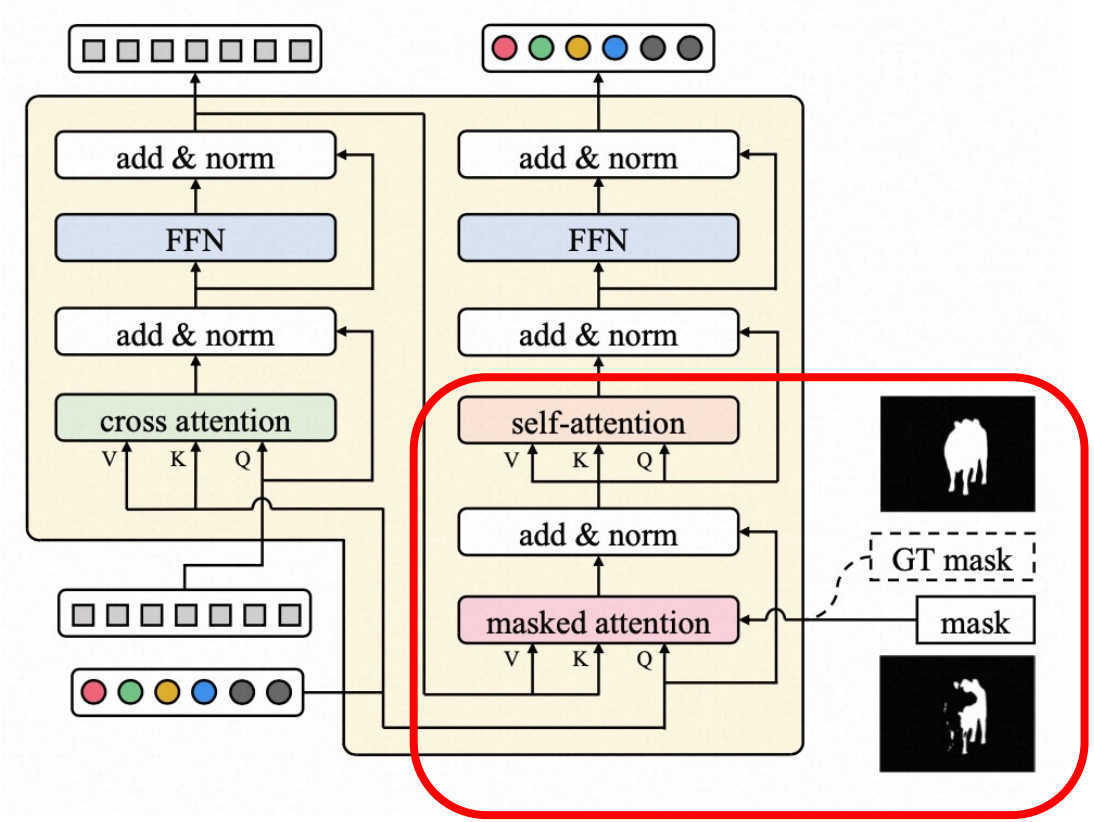
Ground truth mask guided learning

Masked attention

- restricts the receptive field of each query
- may cause the decoder to fall into a suboptimal query update process

So,

- allow each query to see the whole region of its target during training
- that is, use GT mask instead of predicted masks to forward Transformer decoder again, and supervise the new outputs



Experiments

- Ablation study

Instance activate guided queries

	D	AP^{val}	AP_S	AP_M	AP_L	FLOPs	FPS
zero [4]	1	31.5	10.8	33.6	52.6	58.4G	50.0
learnable [9]	1	34.6	13.5	37.5	55.4	58.4G	50.0
resize [33]	1	34.9	13.7	37.9	56.2	58.4G	49.7
IA-guided	1	**35.6**	**14.3**	**38.8**	**56.6**	59.6G	48.8
zero [4]	3	37.2	15.4	40.3	58.6	74.3G	36.1
learnable [9]	3	37.5	15.0	40.6	59.0	74.3G	36.1
resize [33]	3	37.6	15.6	40.4	59.7	74.3G	36.0
IA-guided	3	**37.9**	**16.0**	**40.7**	**60.1**	75.5G	35.5

Table 2. **IA-guided queries.** Our IA-guided queries perform better than other methods, especially when the Transformer decoder layer number (*i.e.*, D) is small.

Dual path Transformer decoder

	D	AP^{val}	AP_S	AP_M	AP_L	FLOPs	FPS
single pixel feat. update	6	32.5	13.9	36.3	50.5	85.4G	35.5
single query update [9]	6	36.9	15.0	39.6	59.8	63.3G	35.0
dual query-then-pixel	3	37.8	**16.0**	40.6	60.0	75.5G	35.5
dual pixel-then-query	3	**37.9**	**16.0**	**40.7**	**60.1**	75.5G	35.5

Table 3. **Dual-path update strategy.** Our dual pixel-then-query update strategy consistently outperforms single-path update strategies. We double the Transformer decoder layers (*i.e.*, D) for the single-path update strategies for a fair comparison.

Ground truth mask guided learning

	backbone	AP^{val}	AP_S	AP_M	AP_L	FLOPs	FPS
w/o GT mask guidance	R50	37.4	15.2	40.6	59.6	75.5G	35.5
w/ GT mask guidance	R50	**37.9**	**16.0**	**40.7**	**60.1**	75.5G	35.5
w/o GT mask guidance	R50-d-DCN	39.7	17.5	43.1	61.9	-	32.5
w/ GT mask guidance	R50-d-DCN	**40.1**	**17.7**	**43.2**	**62.4**	-	32.5

Table 4. **GT mask-guided learning.** Our GT mask-guided learning improves the performance across different backbones.

Experiments

- Comparison with SOTA

method	backbone	epochs	size	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MEInst [47]	R50	36	512	26.4	32.2	53.9	33.0	13.9	34.4	48.7
CenterMask [24]	R50	48	600	32.6	32.9	-	-	12.9	34.7	48.7
SOLOv2 [43]	R50	36	448	41.3	34.0	54.0	36.1	10.3	36.3	54.4
OrienMask [15]	D53	100	544	51.0	34.8	56.7	36.4	16.0	38.2	47.8
SparseInst [11]	R50	144	608	51.2	34.7	55.3	36.6	14.3	36.2	50.7
YOLACT [1]	R50	54	550	53.5	28.2	46.6	29.2	9.2	29.3	44.8
FastInst-D1 (ours)	R50	50	576	**53.8**	35.6	56.7	37.2	8.8	53.0	72.8
CondInst [38]	R50	36	800	20.0	37.8	59.1	40.5	**21.0**	40.3	48.7
Mask2Former†	R50	50	640	25.3	38.0	**60.3**	39.8	10.8	54.9	74.3
FastInst-D3 (ours)	R50	50	640	35.5	**38.6**	60.2	**40.6**	10.8	**56.2**	**75.2**
YOLACT [1]	R101	54	700	33.5	31.2	50.6	32.8	12.1	33.3	47.1
FastInst-D1 (ours)	R101	50	640	**35.3**	38.3	60.2	40.5	10.7	55.8	74.8
SOLOv2 [43]	R101	36	800	15.7	39.7	60.7	**42.9**	17.3	42.9	57.4
CondInst [38]	R101	36	800	16.4	39.1	60.9	42.0	**21.5**	41.7	50.9
Mask2Former†	R101	50	640	21.1	39.5	**61.7**	41.6	11.2	56.3	75.8
FastInst-D3 (ours)	R101	50	640	28.0	**39.9**	61.5	42.3	11.4	**57.1**	**76.6**
SOLOv2 [43]	R50-DCN	36	512	32.0	37.1	57.7	39.7	12.9	40.0	57.4
YOLACT++ [2]	R50-DCN	54	550	39.4	34.1	53.3	36.2	11.7	36.1	53.6
SparseInst [11]	R50-d-DCN	144	608	46.5	37.9	59.2	40.2	**15.7**	39.4	56.9
FastInst-D1 (ours)	R50-d-DCN	50	576	**47.8**	38.0	59.7	39.9	10.0	54.9	74.5
FastInst-D3 (ours)	R50-d-DCN	50	640	32.5	**40.5**	**62.6**	**42.9**	11.9	**57.9**	**76.7**

Experiments

- Unified segmentation

	backbone	Cityscapes `val` AP	PQ	mIoU	#Param. (M)
Mask2Former†	R50	31.4	53.9	74.4	40.9
FastInst-D3	R50	**35.5**	**56.4**	**74.7**	**34.2**

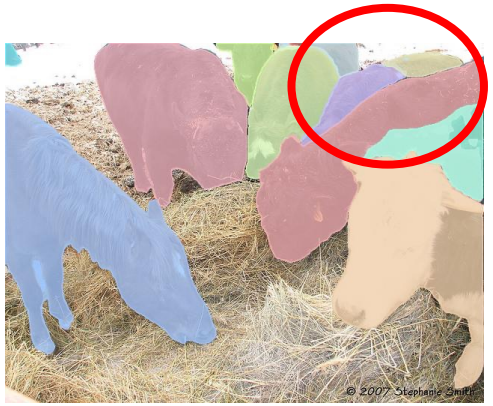
Visualization

- Some predictions on COCO validation set




- Two typical failure cases

Duplicate predictions



Over segmentation

Conclusion

In summary, we present FastInst,

- a simple query-based model for real-time instance segmentation
- achieves excellent performance on instance segmentation task while maintaining a fast inference speed

All codes and pretrained models are available at

- Github: https://github.com/junjiehe96/FastInst
- Modelscope: https://modelscope.cn/models/damo/cv_resnet50_fast-instance-segmentation_coco/summary