Face++ 旷视

MSCOCO & PLACES Challenges 2017

Megvii (Face++) Team

Results

Track	Rank	Ensemble	Single
COCO *(Detection)*	1st	52.8	50.5
PLACES *(InstanceSeg)*	1st	30.7	28.7
COCO *(Keypoint)*	1st	72.6	70.9
COCO *(InstanceSeg)*	2nd	46.4	45.0

Face++ 旷视

I. COCO'17 Detection

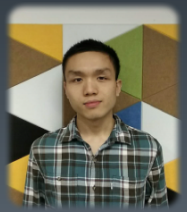
Chao PENG*

Tete XIAO*

Zeming LI*

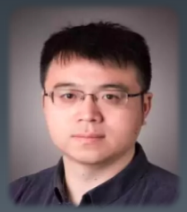
Yuning JIANG

Xiangyu ZHANG



Kai JIA

Gang YU



Jian SUN

Batchsize in Training

Classification (ImageNet)

[A. Krizhevsky, NIPS'12]

[K. Simonyan, ICLR'15]

[K. He, CVPR'16]

[P. Goyal, Arxiv'17]

[Y. You, Arxiv'17]

Detection (MSCOCO)

[S. Ren, NIPS'15]

[S. Ren, TPAMI'16]

[J. Dai, NIPS'16]

[T. Lin, CVPR'17]

[K. He, ICCV'17]

Batchsize in Training

	Classification (ImageNet)	Detection (MSCOCO)
128	[A. Krizhevsky, NIPS'12]	[S. Ren, NIPS'15]
256	[K. Simonyan, ICLR'15]	[S. Ren, TPAMI'16]
	[K. He, CVPR'16]	[J. Dai, NIPS'16]
8K	[P. Goyal, Arxiv'17]	[T. Lin, CVPR'17]
32K	[Y. You, Arxiv'17]	[K. He, ICCV'17]

Batchsize in Training

	Classification (ImageNet)	Detection (MSCOCO)	
128	[A. Krizhevsky, NIPS'12]	[S. Ren, NIPS'15]	2
256	[K. Simonyan, ICLR'15]	[S. Ren, TPAMI'16]	8
	[K. He, CVPR'16]	[J. Dai, NIPS'16]	
8K	[P. Goyal, Arxiv'17]	[T. Lin, CVPR'17]	16
32K	[Y. You, Arxiv'17]	[K. He, ICCV'17]	

Face++ 旷视

Batchsize in Training

	Classification (ImageNet)	Detection (MSCOCO)	
128	[A. Krizhevsky, NIPS'12]	[S. Ren, NIPS'15]	2
256	[K. Simonyan, ICLR'15]	[S. Ren, TPAMI'16]	8
	[K. He, CVPR'16]	[J. Dai, NIPS'16]	
8K	[P. Goyal, Arxiv'17]	[T. Lin, CVPR'17]	16
32K	[Y. You, Arxiv'17]	[K. He, ICCV'17]	

Why batchsize is so small in detection?

Batchsize Constraints

224 *v.s.* 800

Detection eats much more GPU memory!

Small Batchsize Leads to

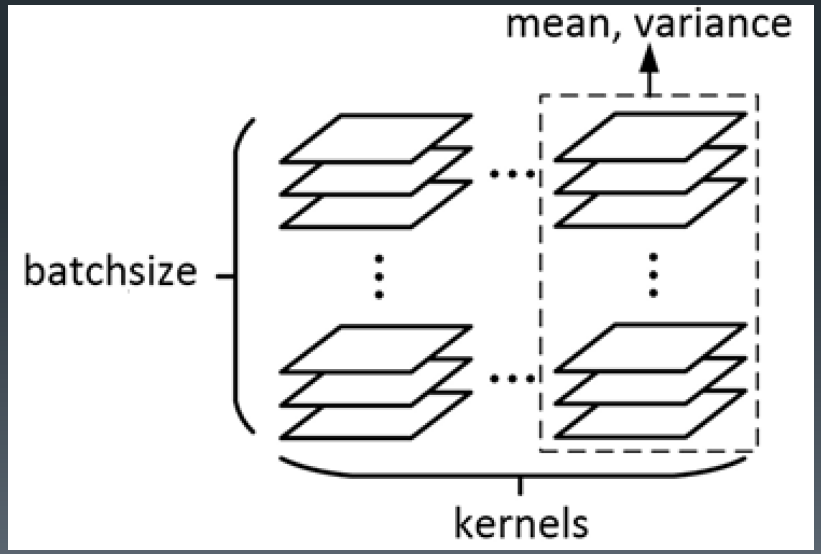
- Unstable gradient

Small Batchsize Leads to

- Unstable gradient

Small Batchsize Leads to

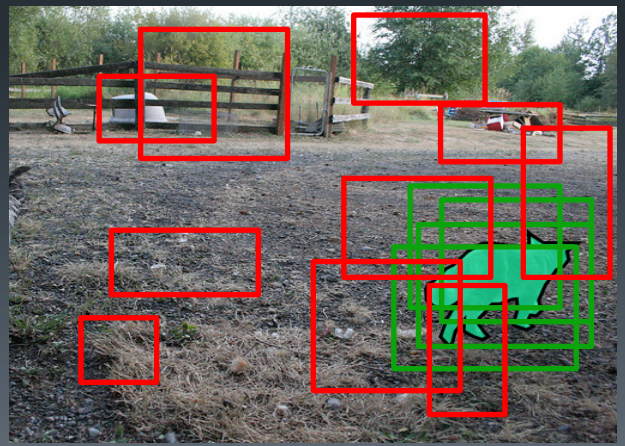
- Unstable gradient
- Inaccurate BN statistics

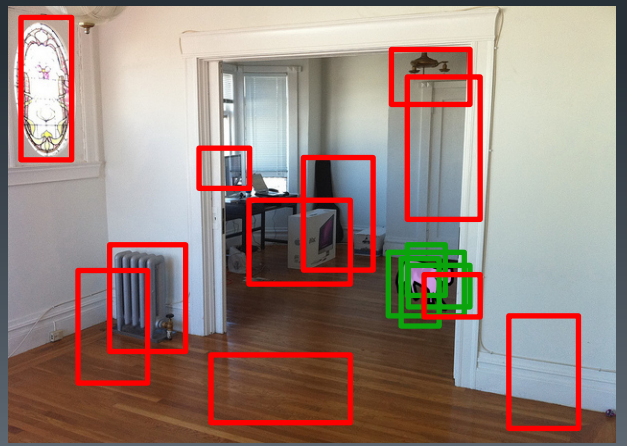


Small Batchsize Leads to

- Unstable gradient
- Inaccurate BN statistics
- Extremely imbalanced data

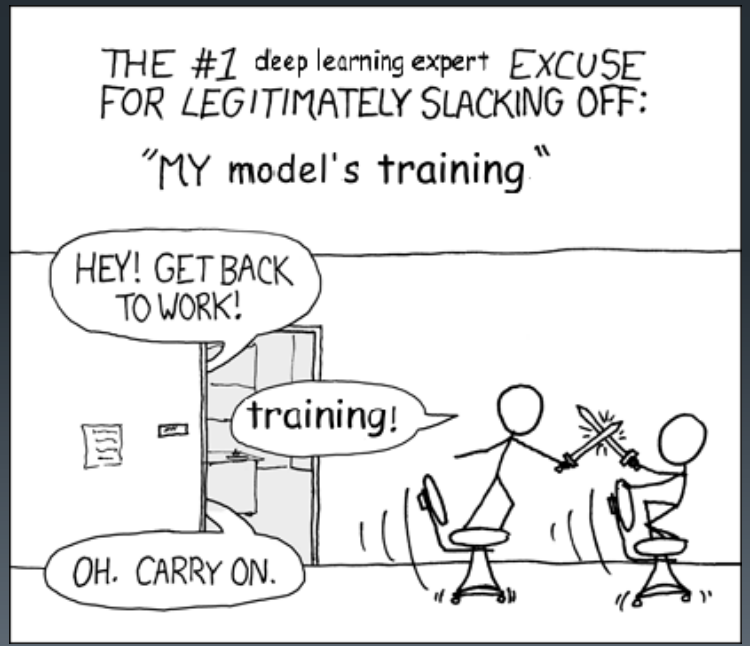
Non-objects >> Objects





Small Batchsize Leads to

- Unstable gradient
- Inaccurate BN statistics
- Extremely imbalanced data
- Very long training period
- ……



MegDet -

The First Large-batch Detector

MegDet -

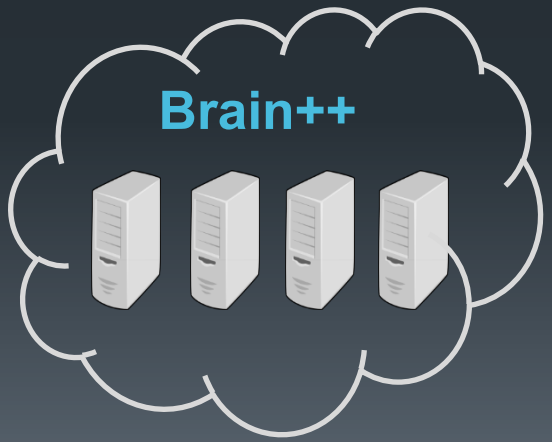
The First Large-batch Detector

- MegBrain & Brain++ empowered

Face++ 旷视

MegBrain & Brain++

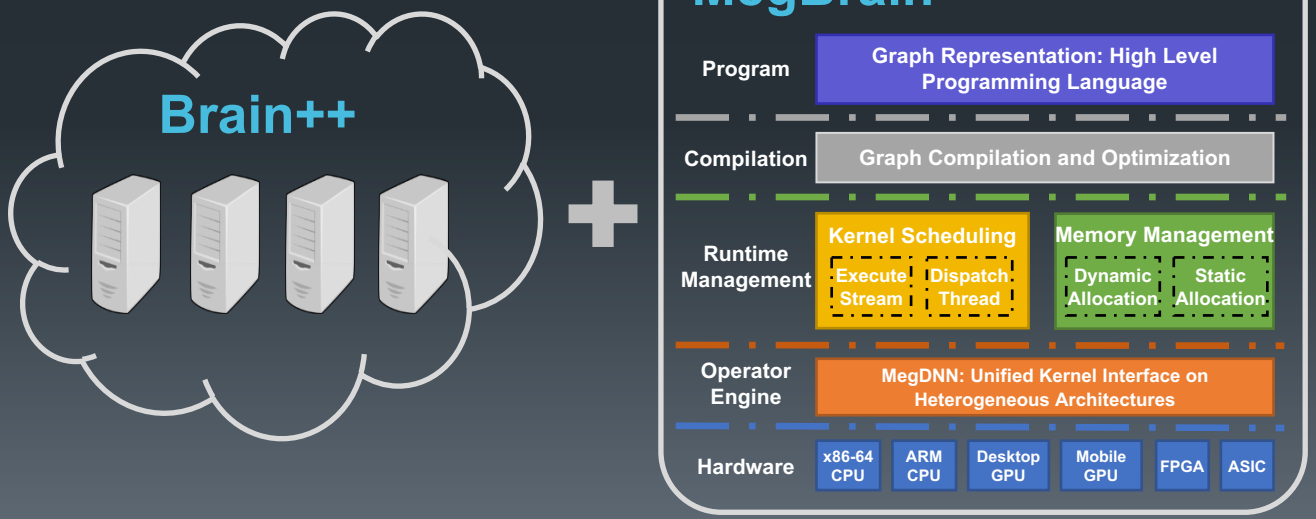
Platform



MegBrain & Brain++

Platform

Framework

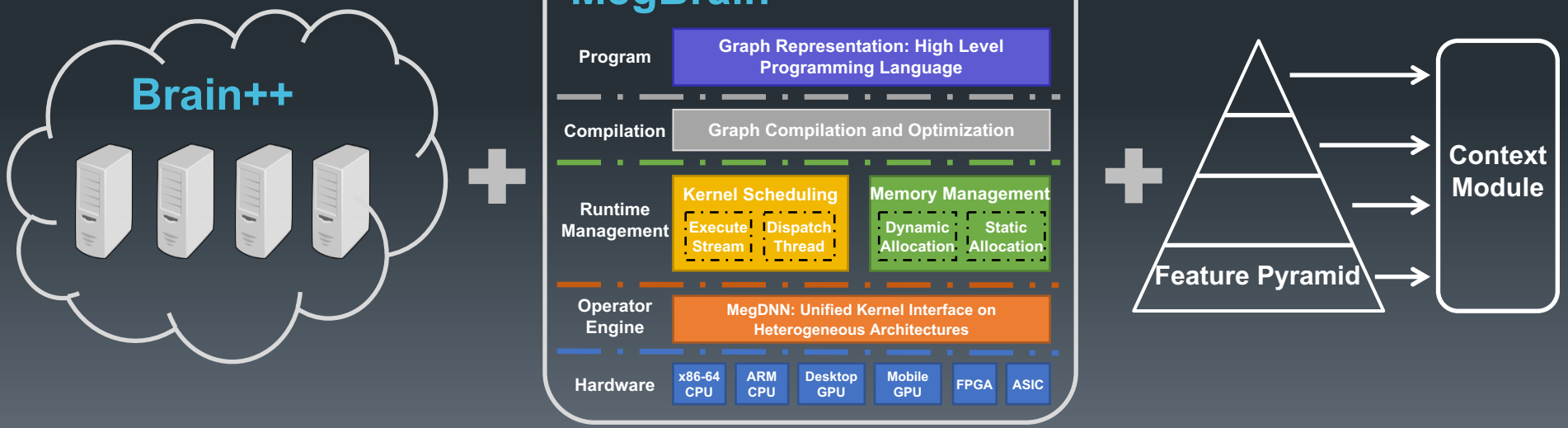


MegBrain & Brain++

Platform

Framework

Algorithm



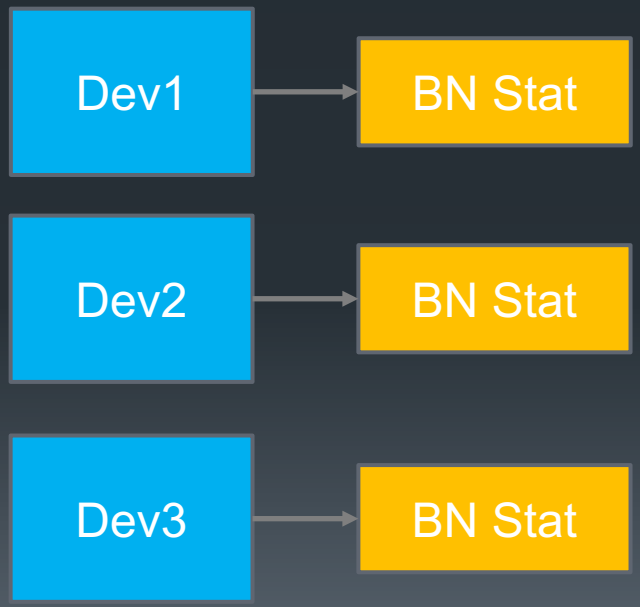
MegDet -

The First Large-batch Detector

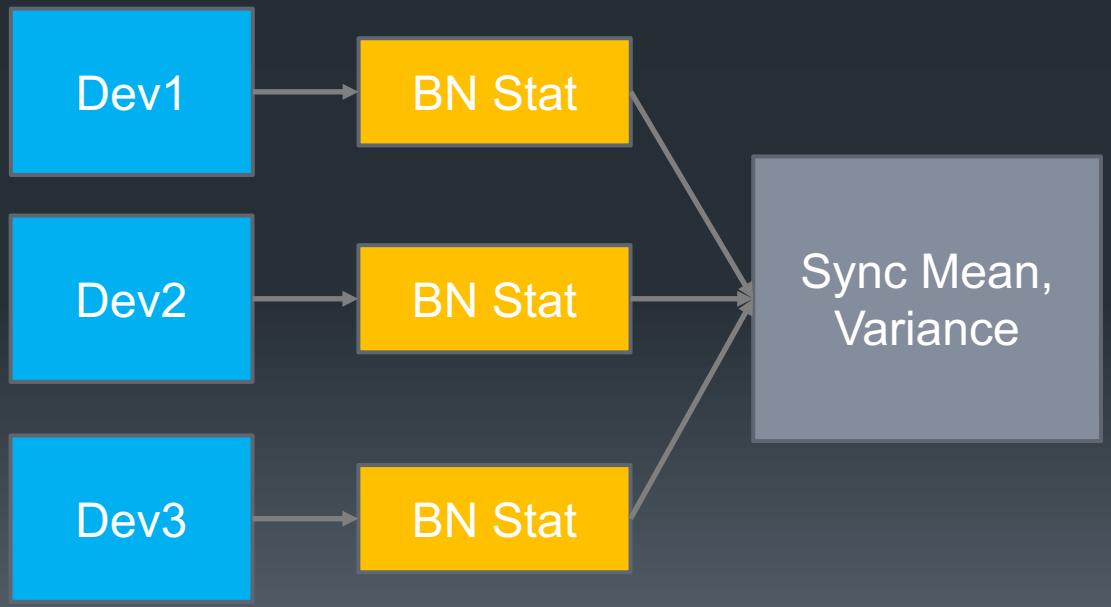
- MegBrain & Brain++ empowered
- Multi-device batch normalization

Multi-device BatchNorm

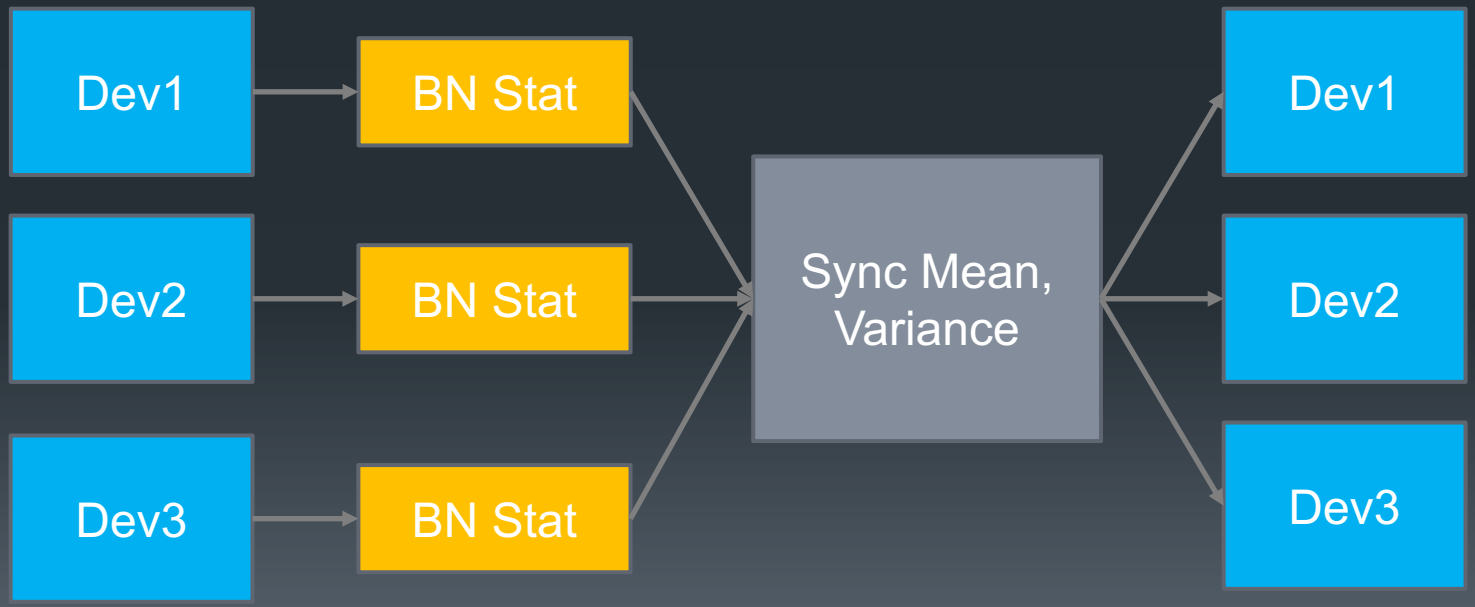
Multi-device BatchNorm



Multi-device BatchNorm



Multi-device BatchNorm

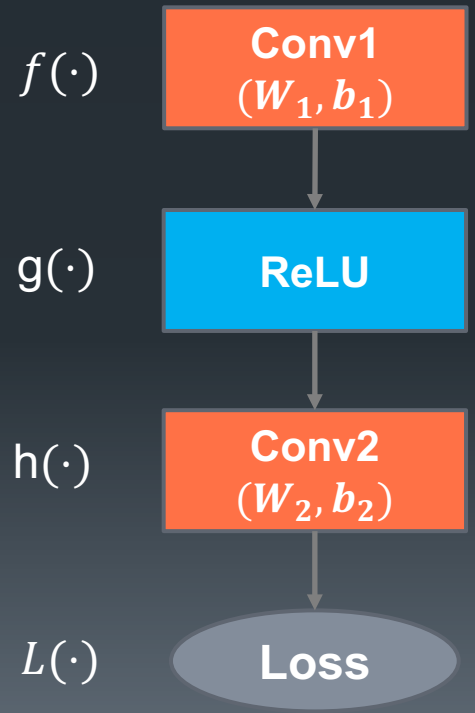


MegDet -

The First Large-batch Detector

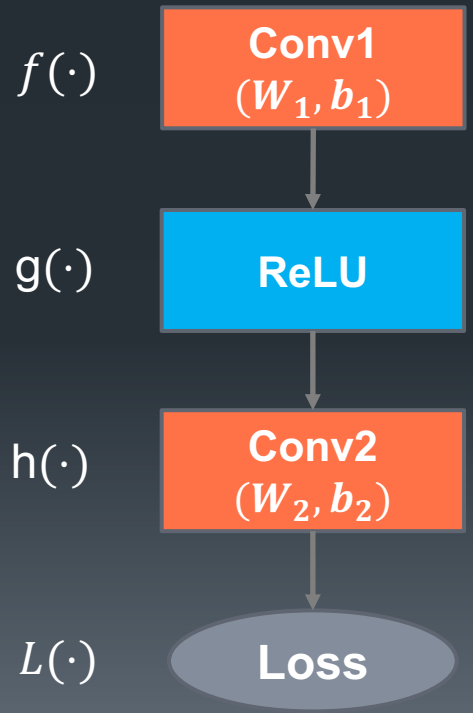
- MegBrain & Brain++ empowered
- Multi-device batch normalization
- Sublinear memory

Sublinear Memory



[T. Chen et al, Arxiv'16]

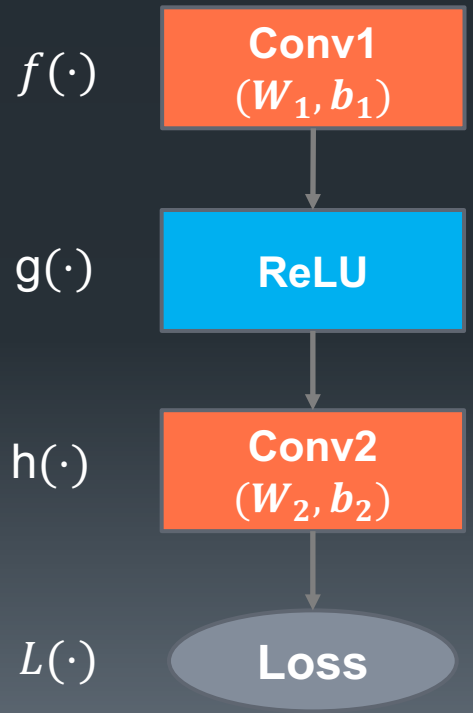
Sublinear Memory



$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h}\frac{\partial h}{\partial W_2}$$

[T. Chen et al, Arxiv'16]

Sublinear Memory

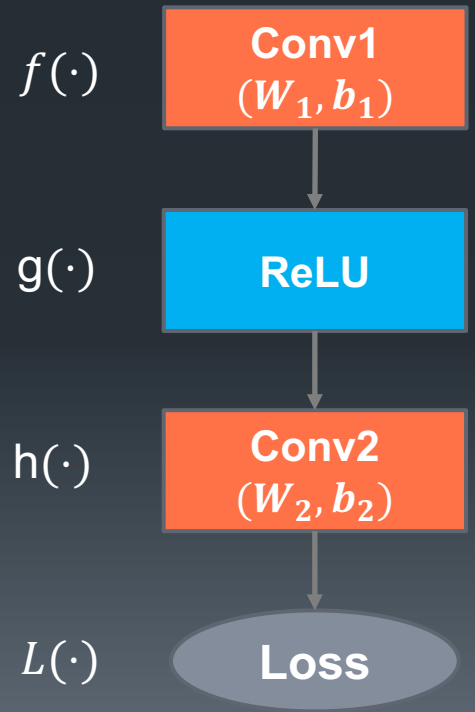


$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h}\frac{\partial h}{\partial W_2}$$

We need store results of Conv2!

[T. Chen et al, Arxiv'16]

Sublinear Memory



$$h(\cdot) = g(f(x)) \qquad \frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h}\frac{\partial h}{\partial W_2}$$

$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial g(f(x))}\frac{\partial g(f(x))}{\partial W_2}$$

We can compute the $h(\cdot)$ in BP and thus don't need to store the Conv2!

[T. Chen et al, Arxiv'16]

MegDet -

The First Large-batch Detector

- MegBrain & Brain++ empowered
- Multi-device batch normalization
- Sublinear memory
- Large learning rate policy

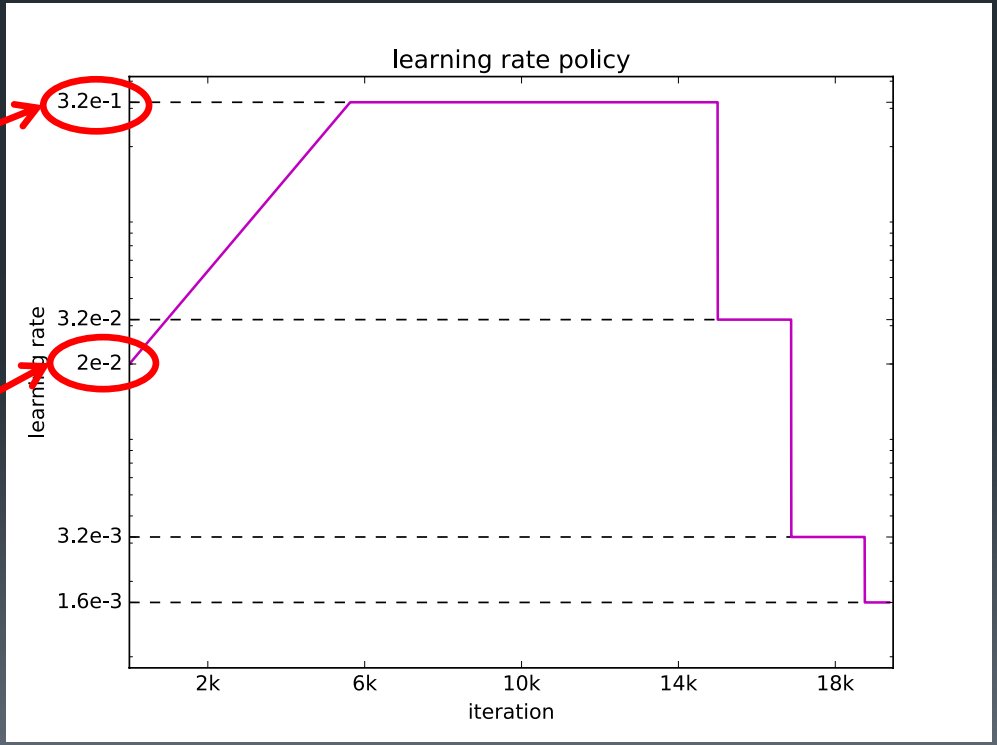
Large Learning Rate

in MegDet: 0.32

II

x 16

in FPN: 0.02



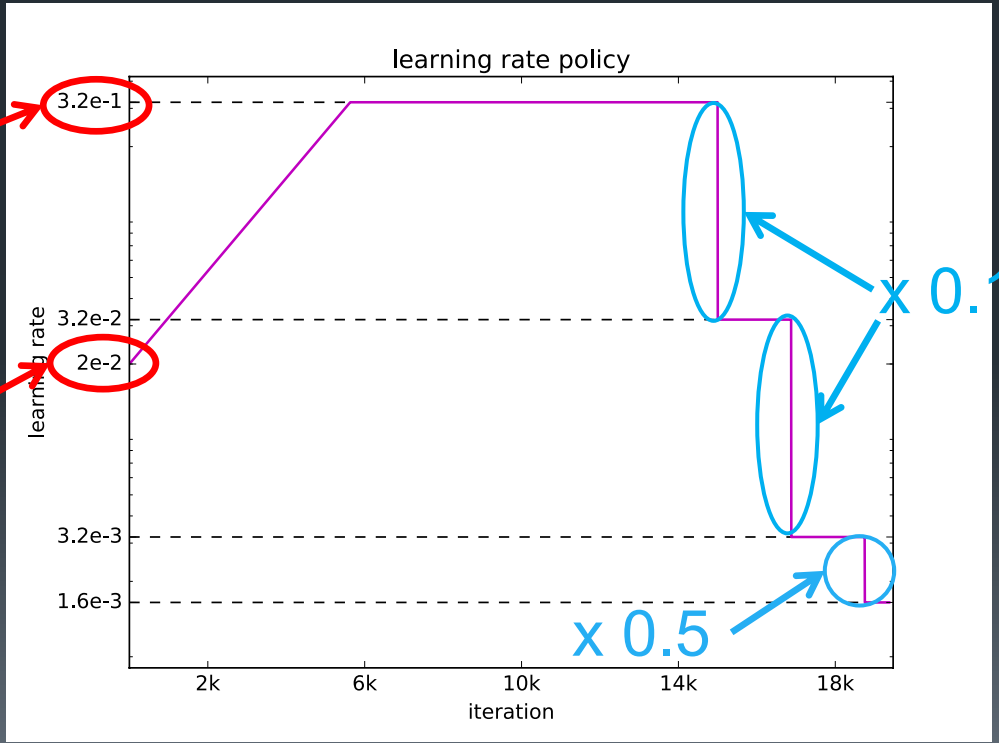
Large Learning Rate

in MegDet: 0.32

II

x 16

in FPN: 0.02



Faster RCNN, 2015
batchsize = 2

[S. Ren et al, Faster R-CNN, NIPS'15]

Faster RCNN, 2015
batchsize = 2

FPN, 2016
batchsize = 16

[T. Lin et al, Feature Pyramid Network, CVPR'17]

Faster RCNN, 2015
batchsize = 2

FPN, 2016
batchsize = 16

MegDet, 2017
batchsize = 256

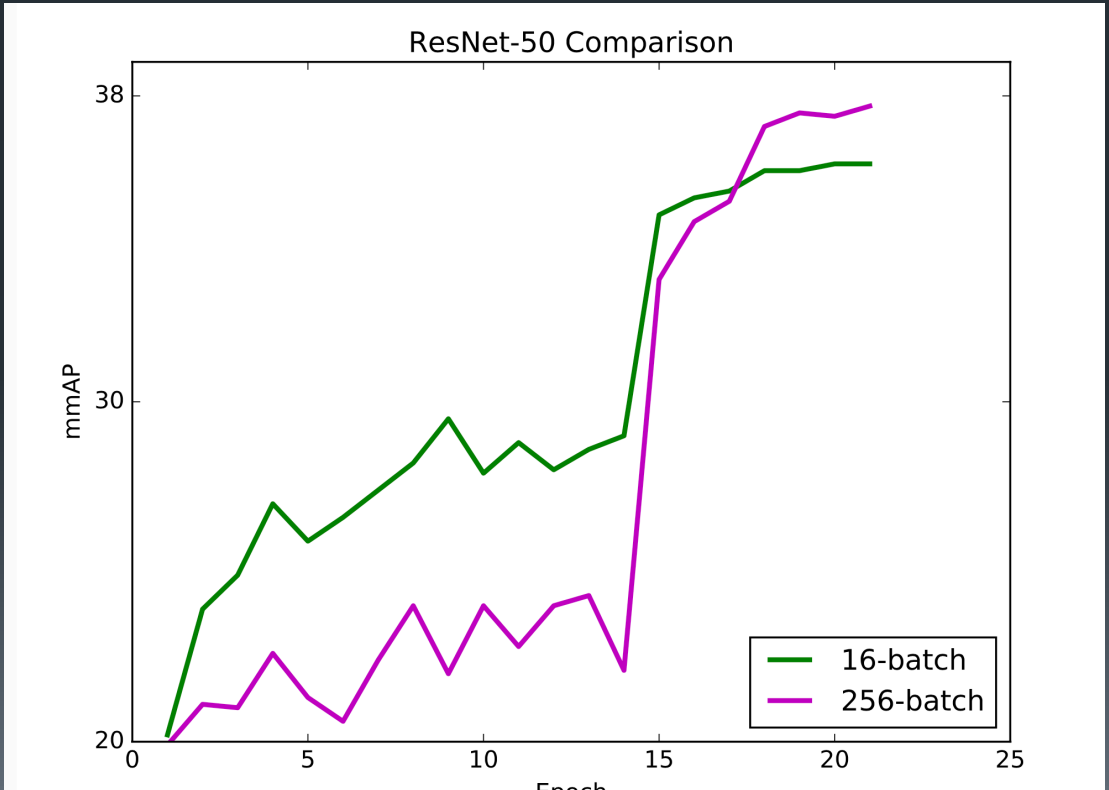
With Large Batch, We Have

- More stable gradient

With Large Batch, We Have

- More stable gradient
- More accurate BN statistics

Large Batch *v.s.* Small Batch



With Large Batch, We Have

- More stable gradient
- More accurate BN statistics
- Faster training (train COCO in hours)

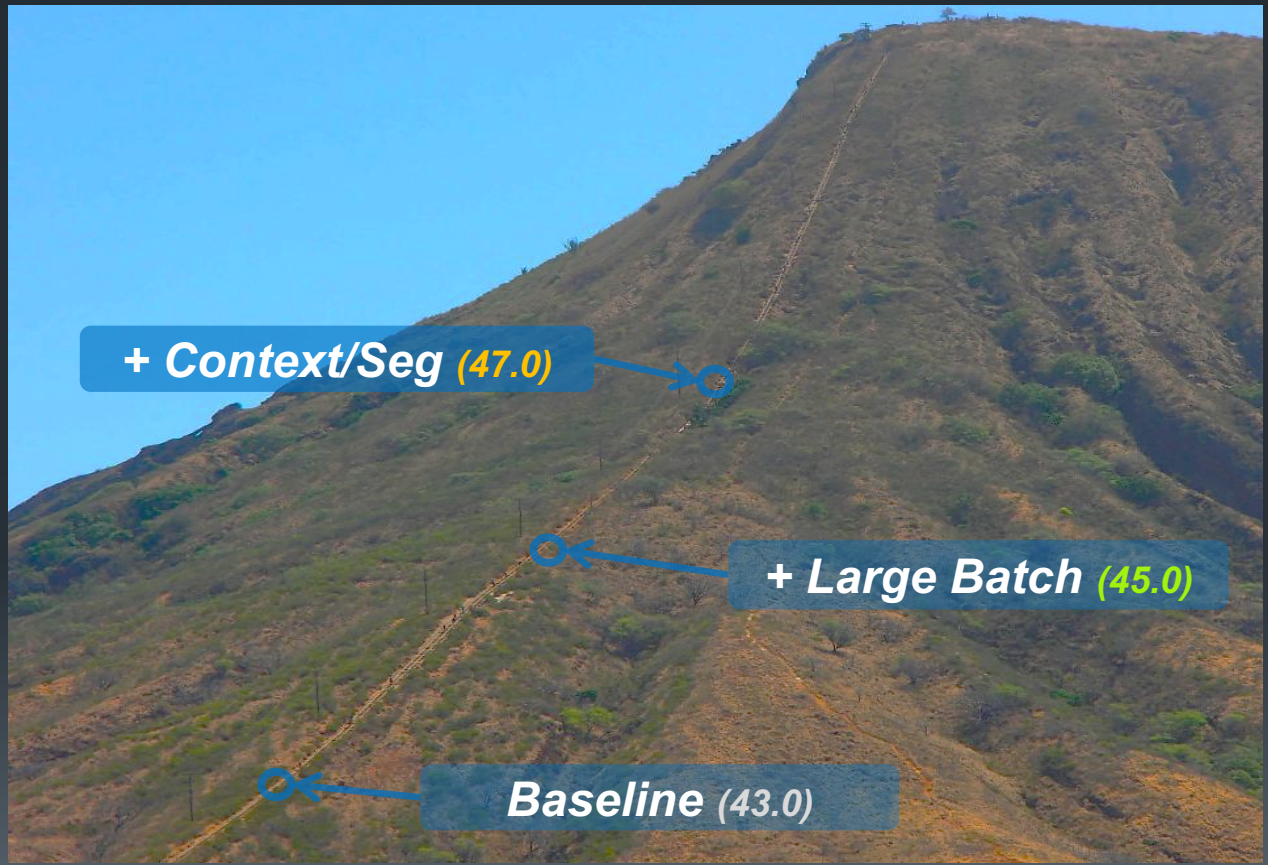
Speed-up Training

Train detector on COCO in Two hour!

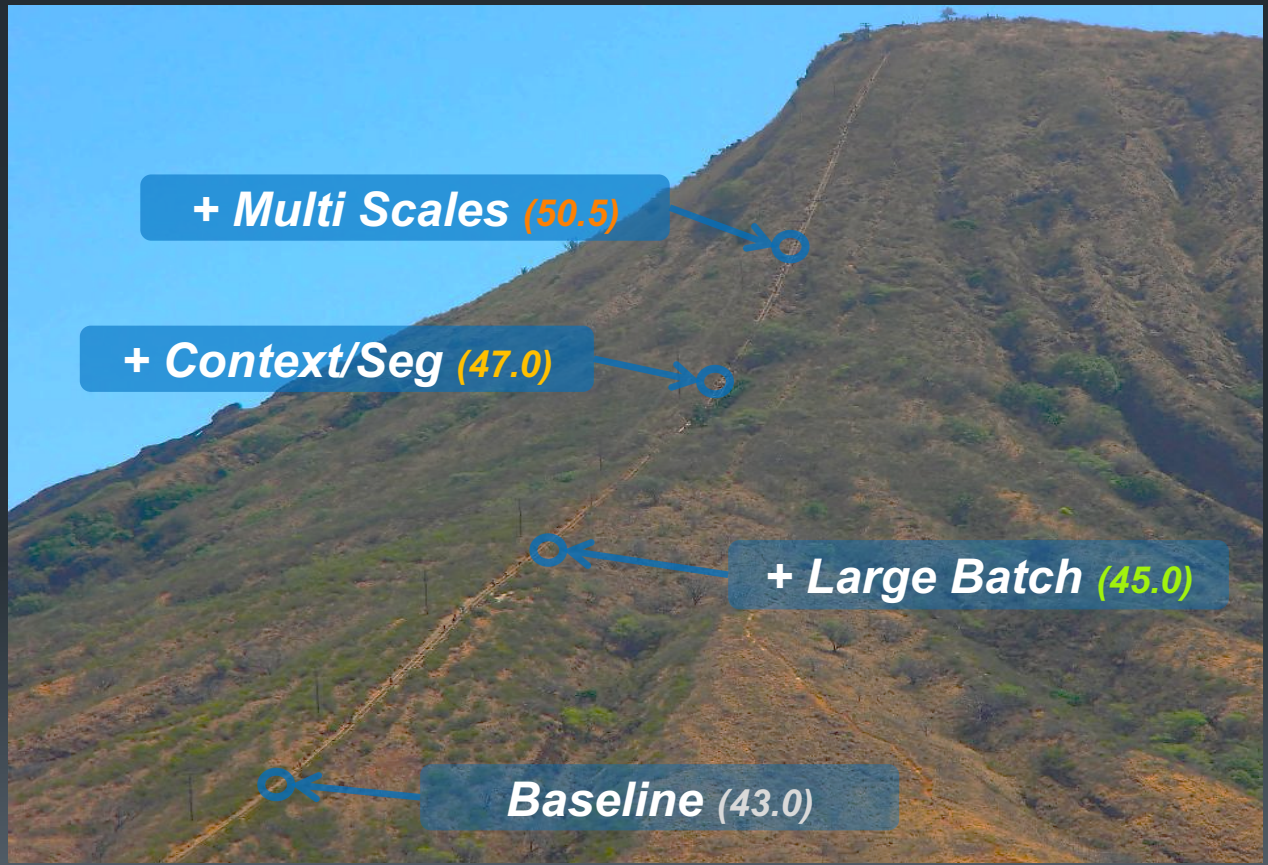
“KOKO Head” in Honolulu, Hawaii

“KOKO Head” in Honolulu, Hawaii

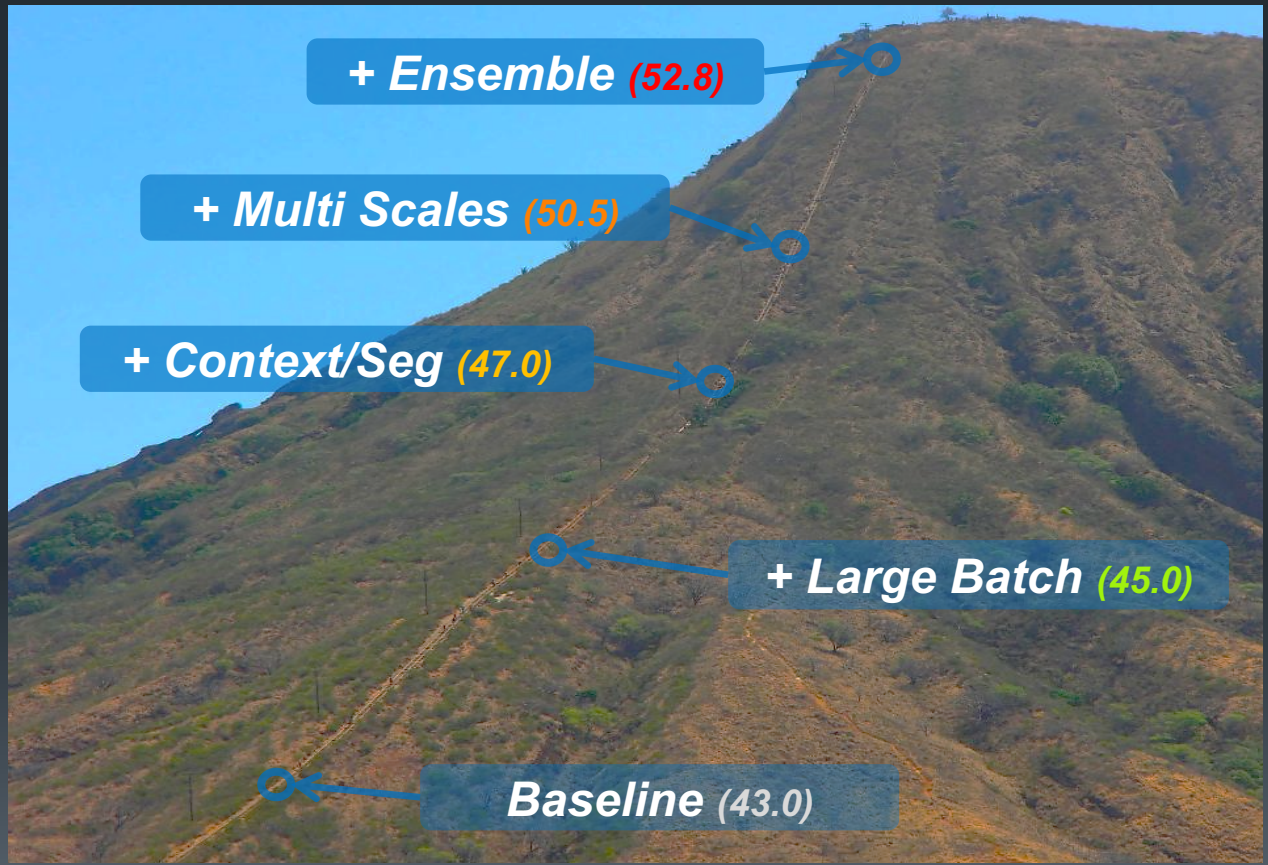
“KOKO Head” in Honolulu, Hawaii



“KOKO Head” in Honolulu, Hawaii



“KOKO Head” in Honolulu, Hawaii

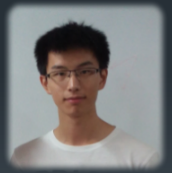


“KOKO Head” in Honolulu, Hawaii

Face++ 旷视

II. COCO & PLACES'17
Instance Segmentation

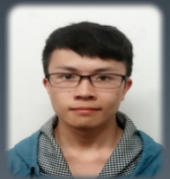

Ruixuan LUO*


Borui JIANG*


Tete XIAO*


Chao PENG*


Yuning JIANG


Zeming LI


Xiangyu ZHANG

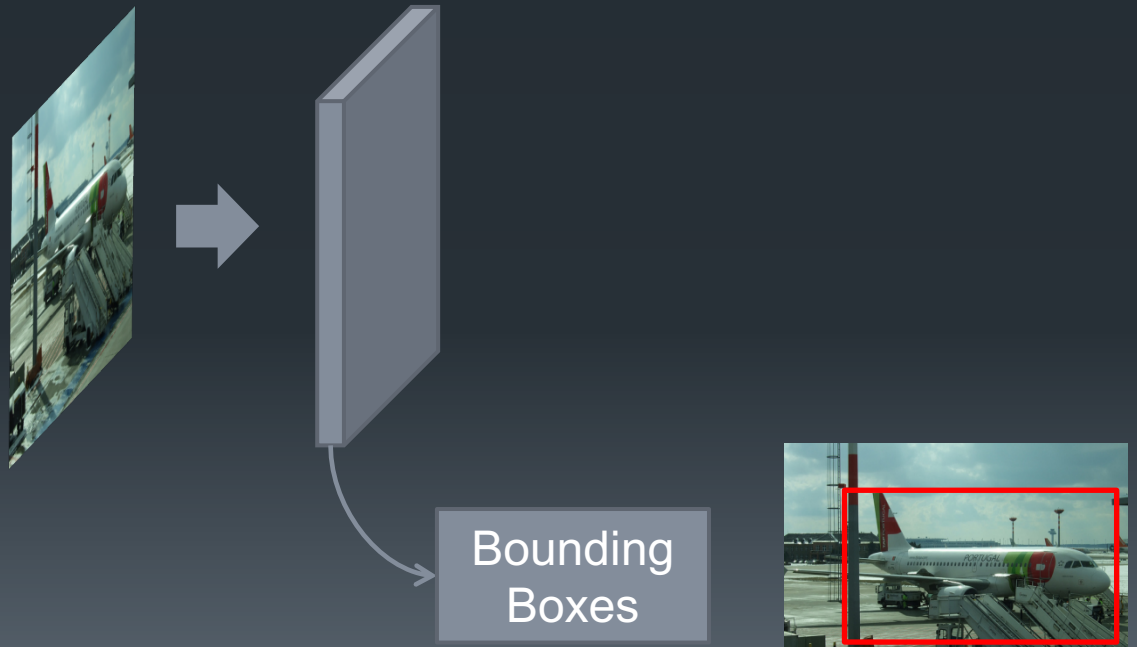

Gang YU


Yadong MU


Jian SUN

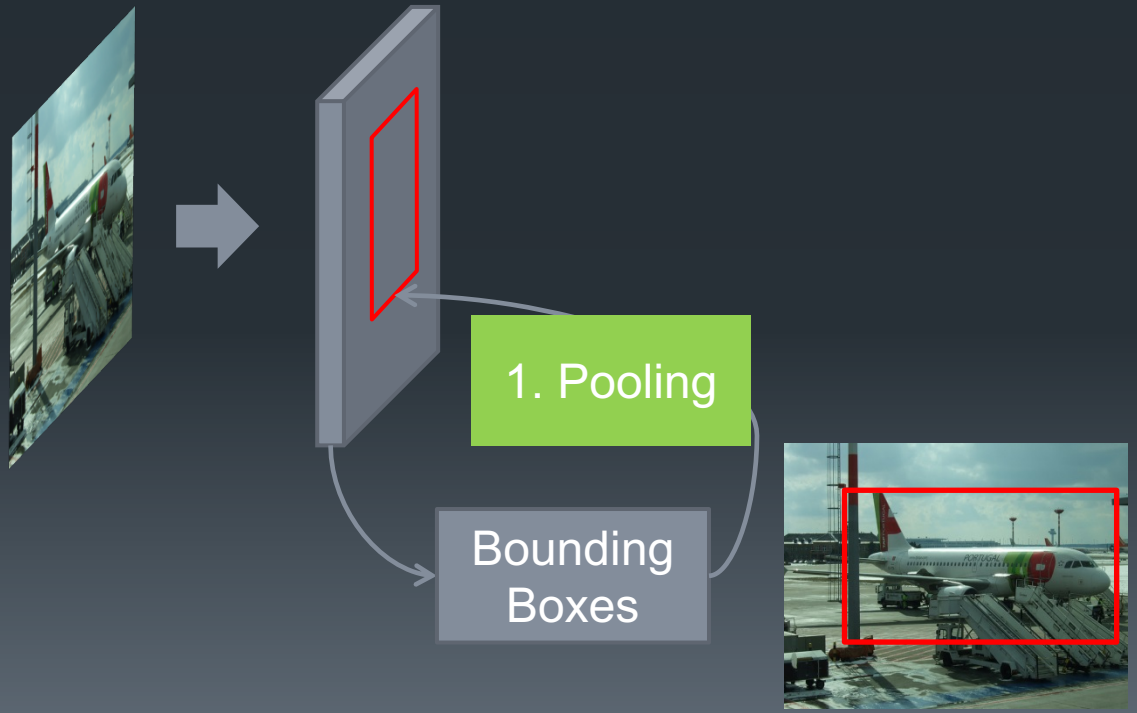
Face++ 旷视

Overview

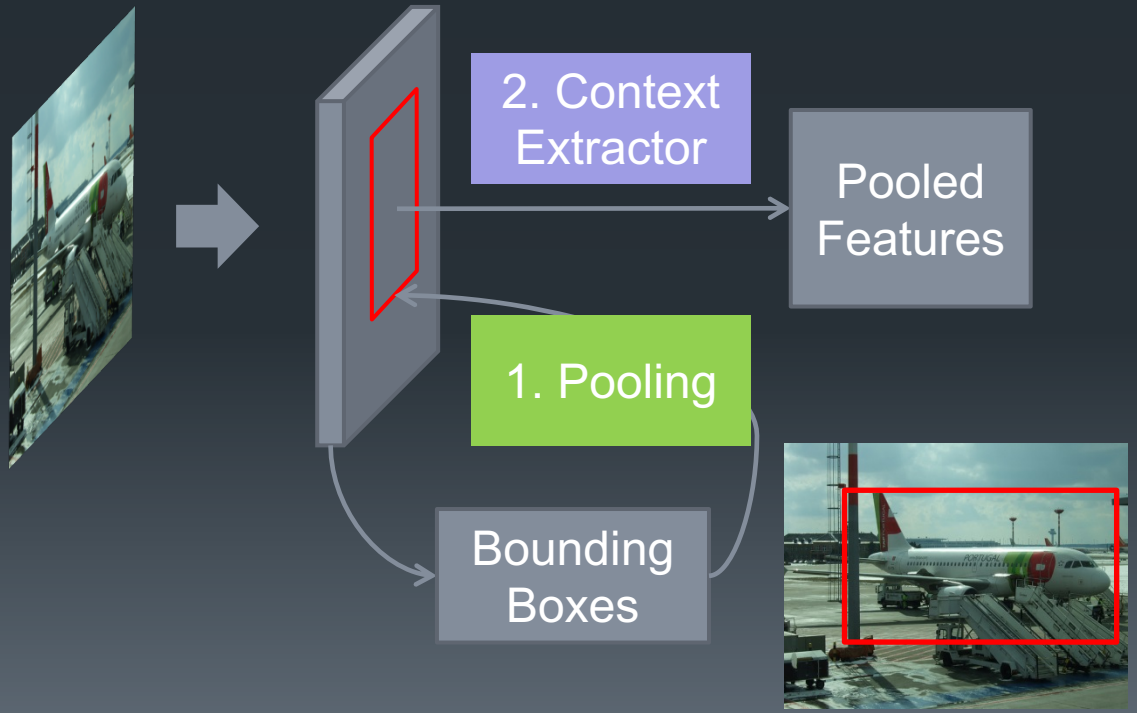


Face++ 旷视

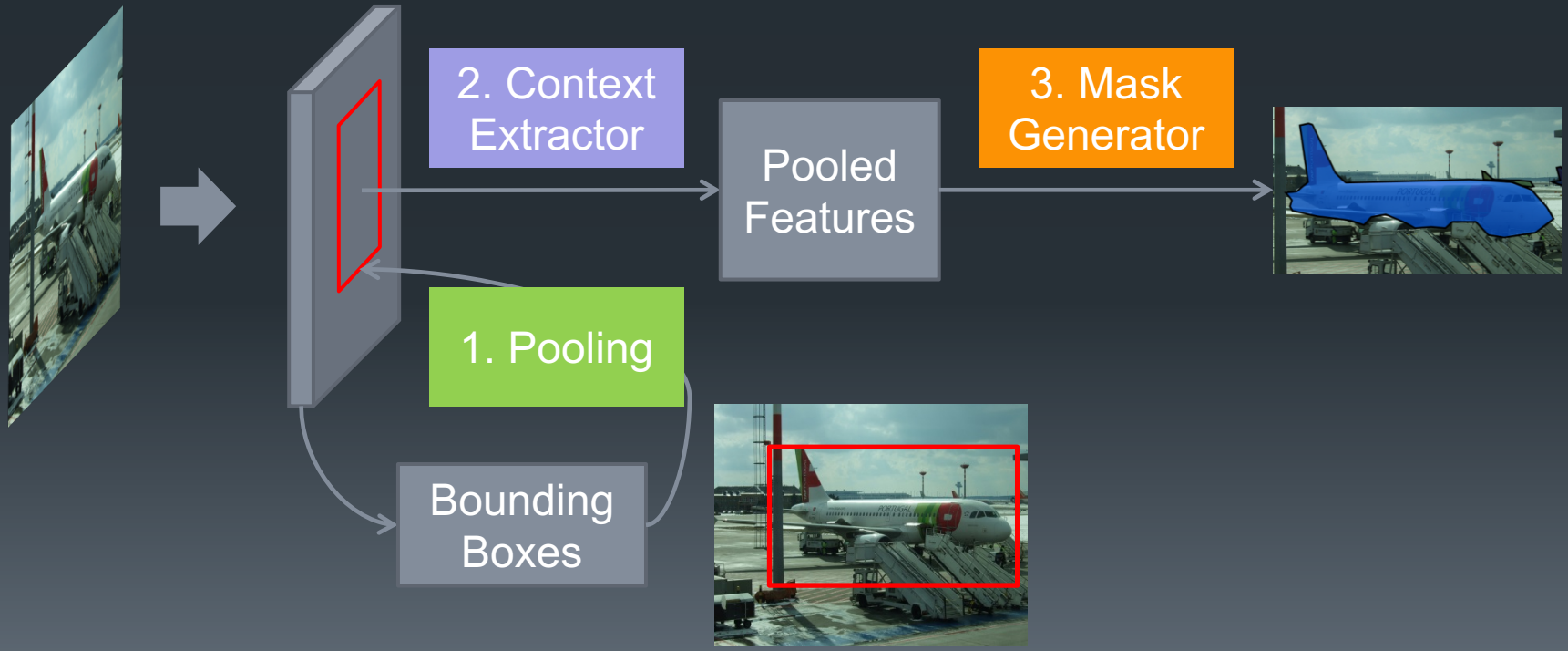
Overview



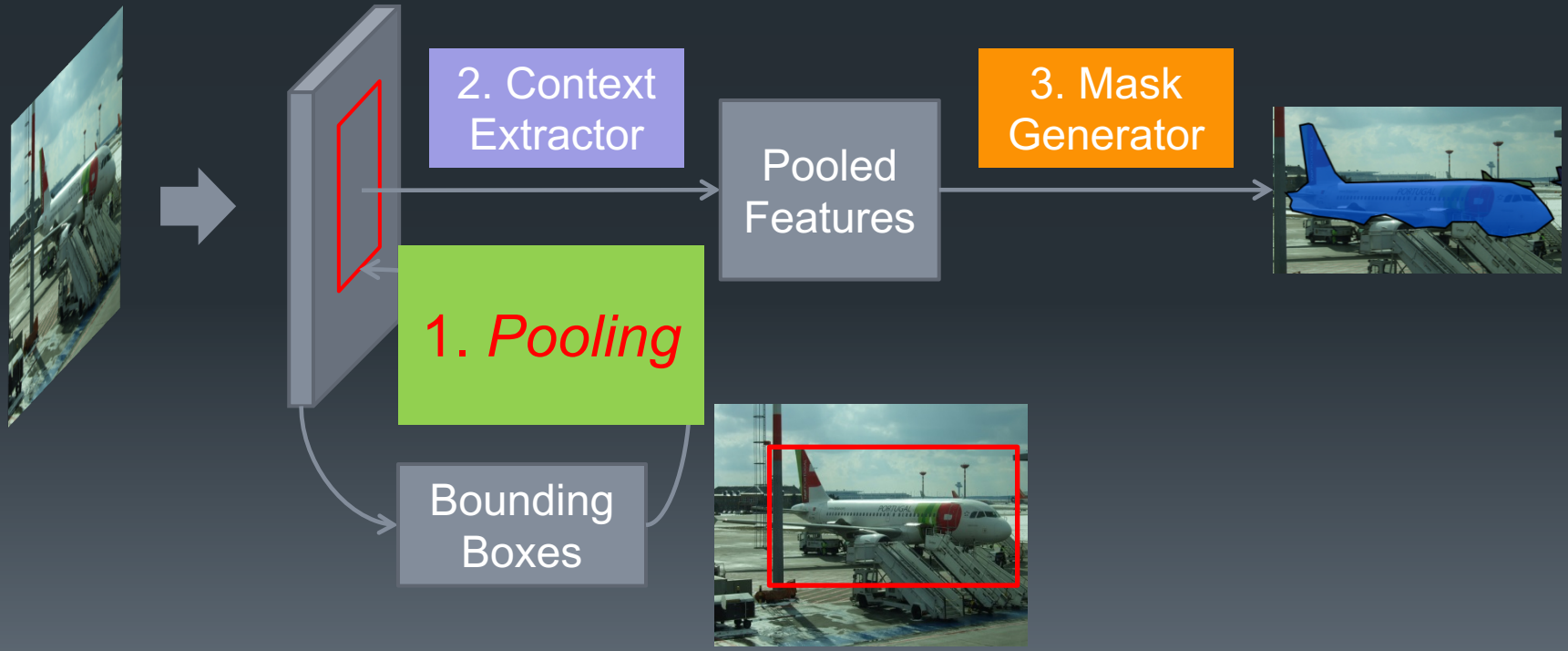
Overview


2. Context Extractor
Pooled Features
1. Pooling
Bounding Boxes

Overview

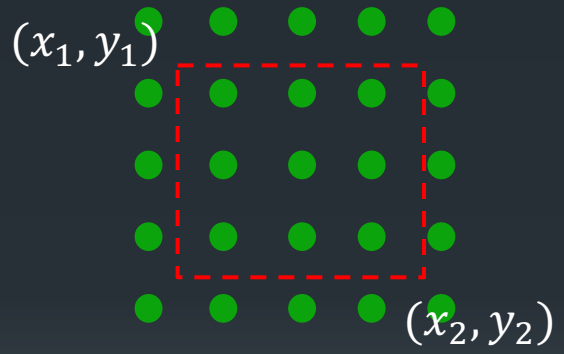

2. Context Extractor
Pooled Features
3. Mask Generator
1. Pooling
Bounding Boxes

Overview


2. Context Extractor
3. Mask Generator
Pooled Features
1. *Pooling*
Bounding Boxes

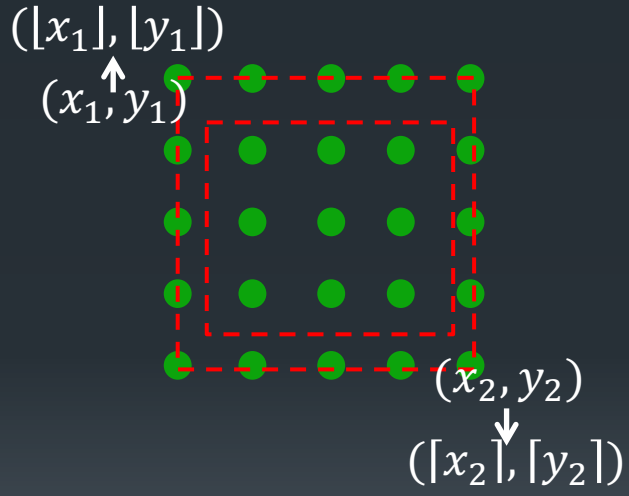
Face++ 旷视

1. Pooling - Vanilla RoI Pooling



1. Pooling - Vanilla RoI Pooling

Discrete



$$\frac{\sum_{i=\lfloor x_1 \rfloor}^{\lceil x_2 \rceil} \sum_{j=\lfloor y_1 \rfloor}^{\lceil y_2 \rceil} w_{i,j}}{(\lceil x_2 \rceil - \lfloor x_1 \rfloor + 1) * (\lceil y_2 \rceil - \lfloor y_1 \rfloor + 1)}$$

1. Pooling - Precise RoI Pooling

$$f(x,y) = \sum_{i,j} g(x,y,i,j) * w_{i,j}$$

where

$$g(x,y,i,j) = \max(0, 1-|x-i|) * \max(0, 1-|y-j|)$$

1. Pooling - Precise RoI Pooling

Discrete

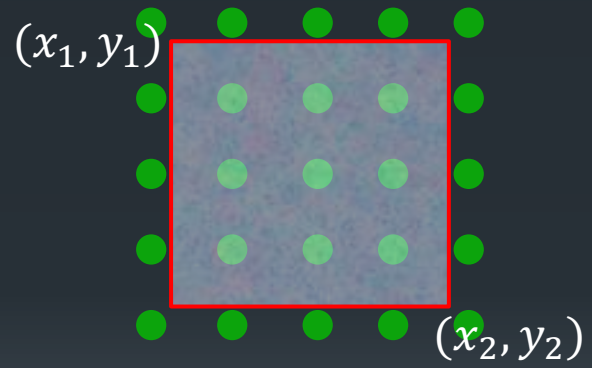
Continuous

$$f(x,y) = \sum_{i,j} g(x,y,i,j) * w_{i,j}$$

where

$$g(x,y,i,j) = \max(0, 1-|x-i|) * \max(0, 1-|y-j|)$$

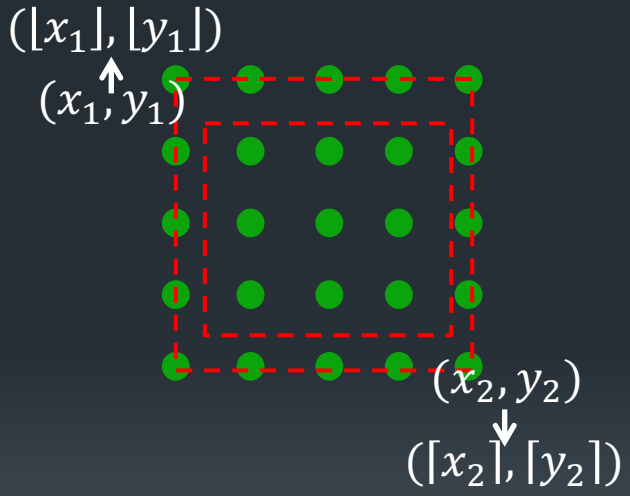
1. Pooling - Precise RoI Pooling



$$pr\_roi\,(x_1, y_1, x_2, y_2) = \iint_{\substack{x_1 \le x \le x_2, \\ y_1 \le y \le y_2}} f(x, y)dxdy \Big/ \left((x_2 - x_1) * (y_2 - y_1)\right)$$

Face++ 旷视

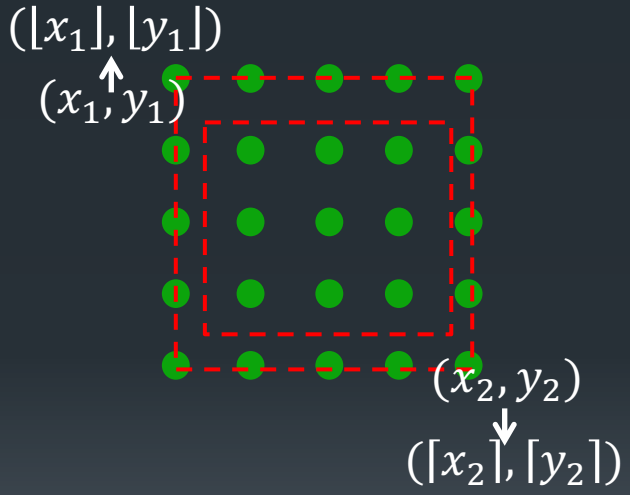
RoI Pooling



$$\frac{\sum_{i=\lfloor x_1 \rfloor}^{\lceil x_2 \rceil} \sum_{j=\lfloor y_1 \rfloor}^{\lceil y_2 \rceil} w_{i,j}}{(\lceil x_2 \rceil - \lfloor x_1 \rfloor + 1) * (\lceil y_2 \rceil - \lfloor y_1 \rfloor + 1)}$$

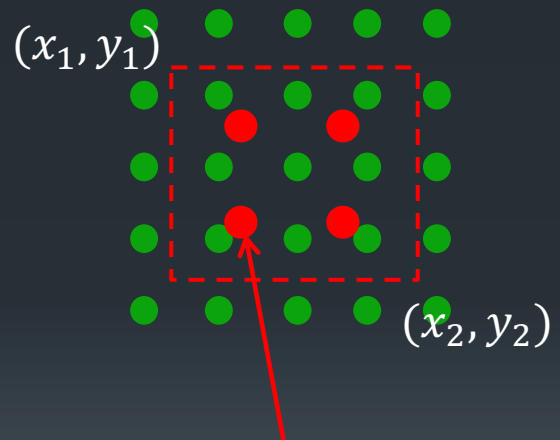
Face++ 旷视

RoI Pooling



$$\frac{\sum_{i=\lfloor x_1 \rfloor}^{\lceil x_2 \rceil} \sum_{j=\lfloor y_1 \rfloor}^{\lceil y_2 \rceil} w_{i,j}}{(\lceil x_2 \rceil - \lfloor x_1 \rfloor + 1) * (\lceil y_2 \rceil - \lfloor y_1 \rfloor + 1)}$$

RoI Align

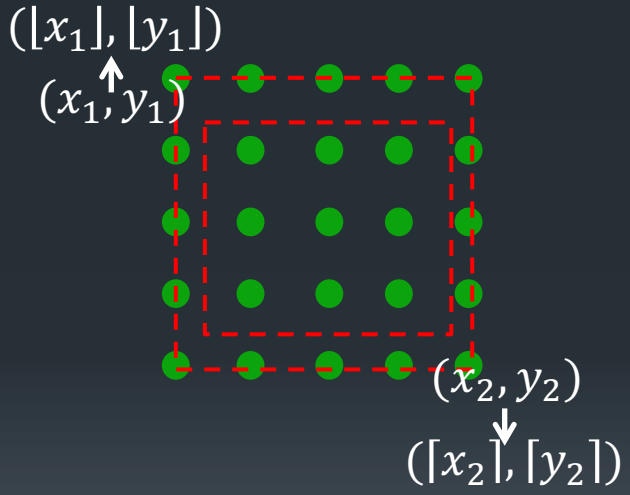


$$\sum_{i=1}^{4} f(a_i, b_i) /4$$

[K. He et al, Mask R-CNN, ICCV'17]

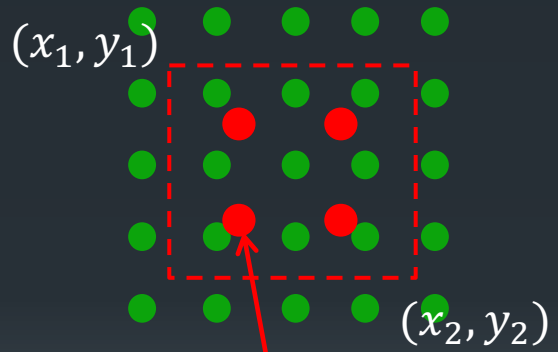
Face++ 旷视

RoI Pooling



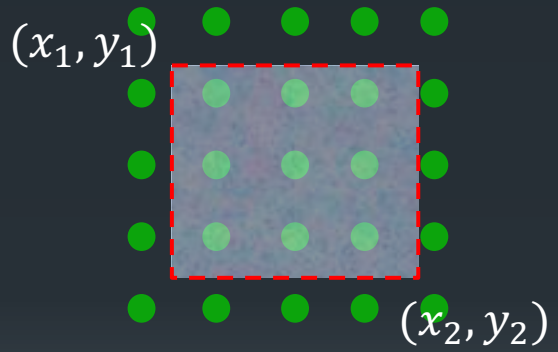
$$\frac{\sum_{i=\lfloor x_1 \rfloor}^{\lceil x_2 \rceil} \sum_{j=\lfloor y_1 \rfloor}^{\lceil y_2 \rceil} w_{i,j}}{(\lceil x_2 \rceil - \lfloor x_1 \rfloor + 1) * (\lceil y_2 \rceil - \lfloor y_1 \rfloor + 1)}$$

RoI Align



$$\sum_{i=1}^{4} f(a_i, b_i) / 4$$

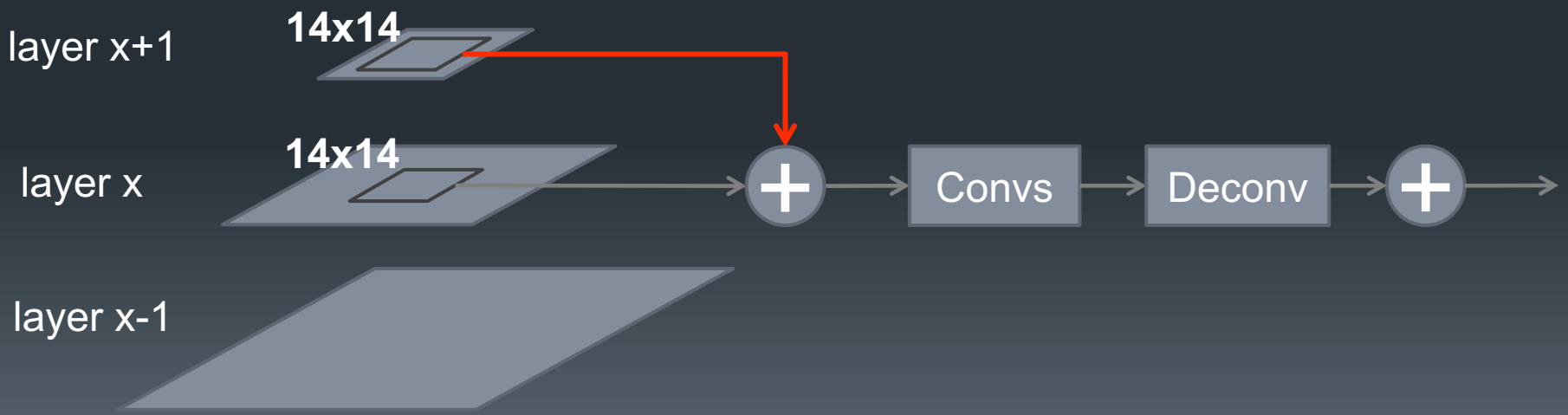
Precise RoI Pooling



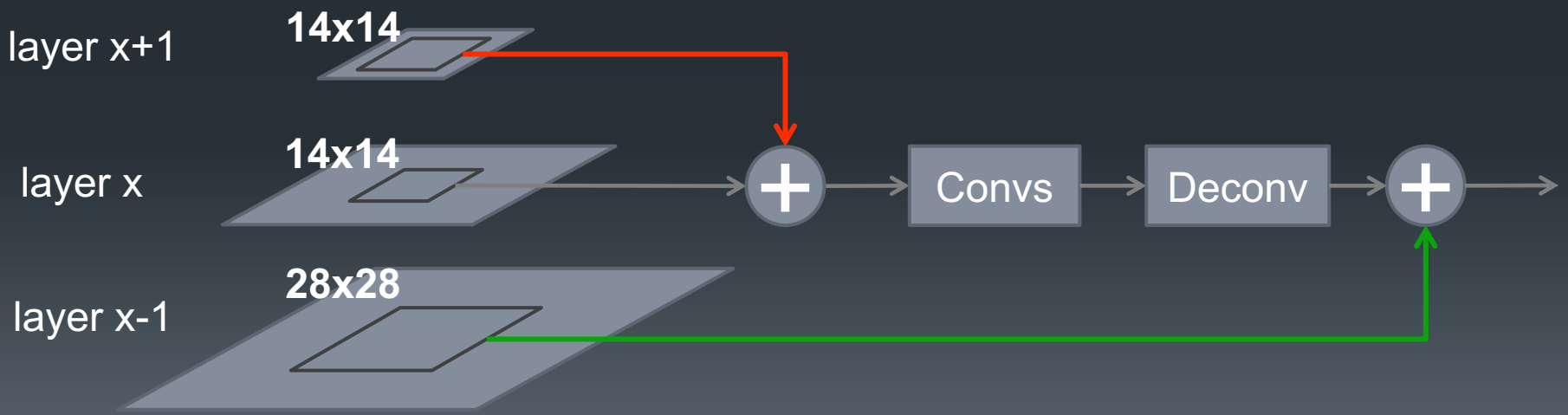
$$\frac{\iint_{\substack{x_1 \le x \le x_2, \\ y_1 \le y \le y_2}} f(x, y) dx dy}{(x_2 - x_1) * (y_2 - y_1)}$$

2. Context Extractor

2. Context Extractor



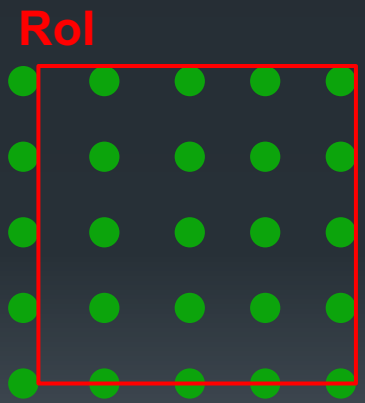
2. Context Extractor



3. Mask Generator

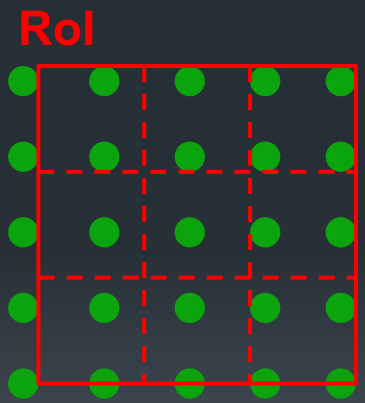
Face++ 旷视

3. Mask Generator

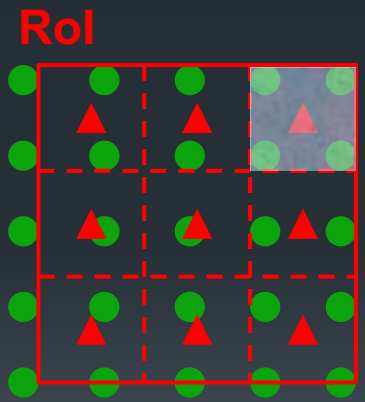


Face++ 旷视

3. Mask Generator

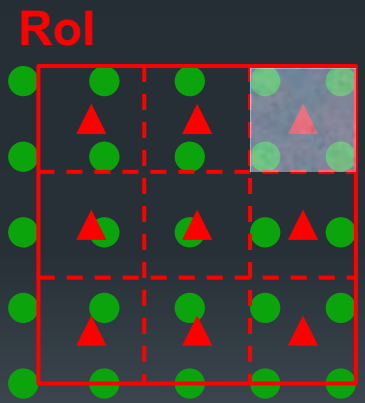


3. Mask Generator



Pooling

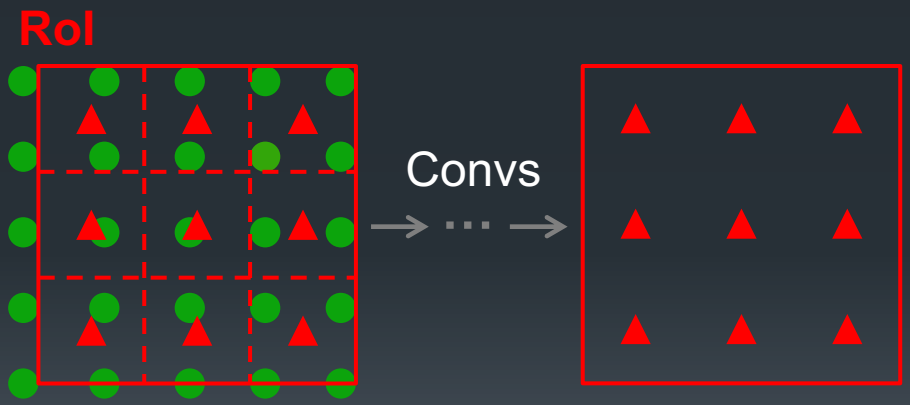
3. Mask Generator



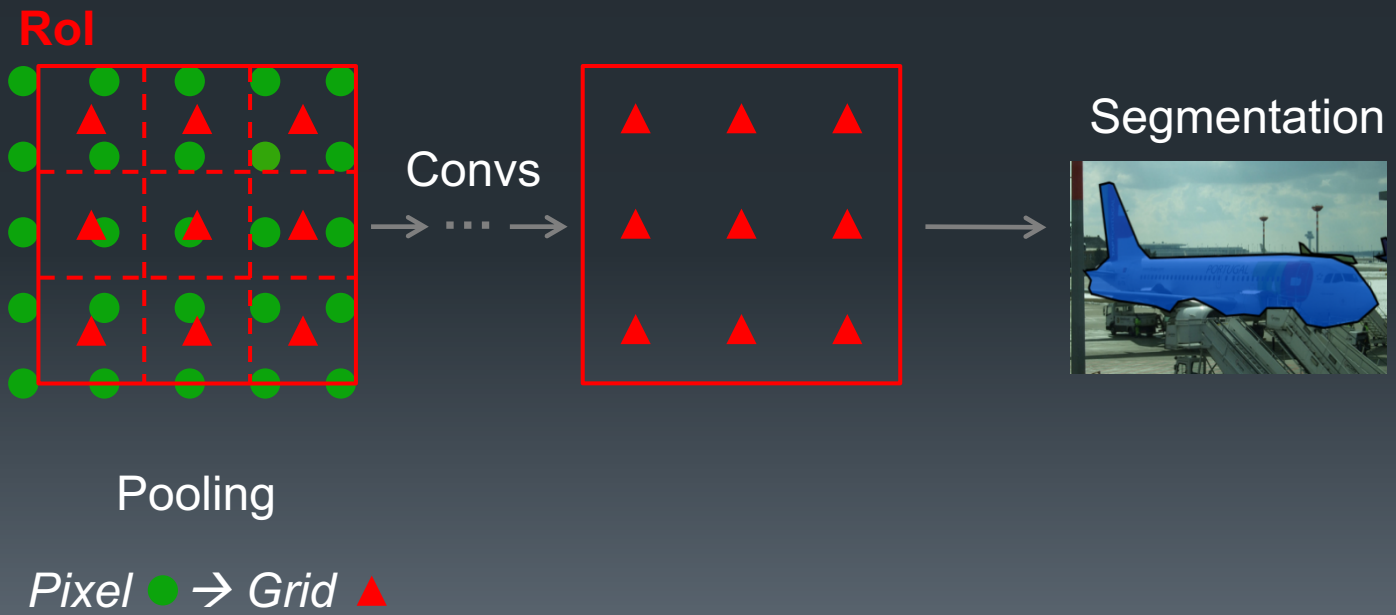
Pooling

Pixel ● → *Grid* ▲

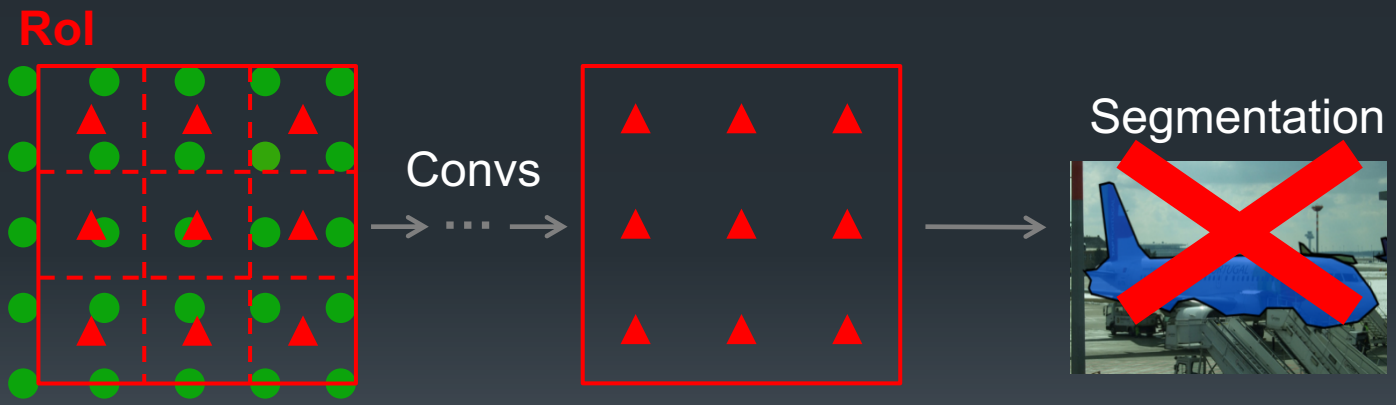
3. Mask Generator



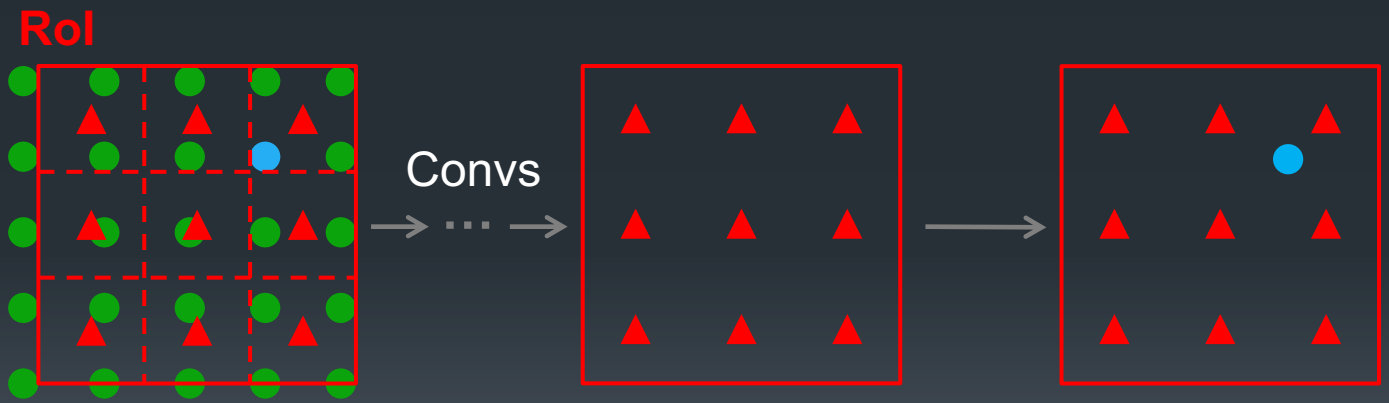
3. Mask Generator



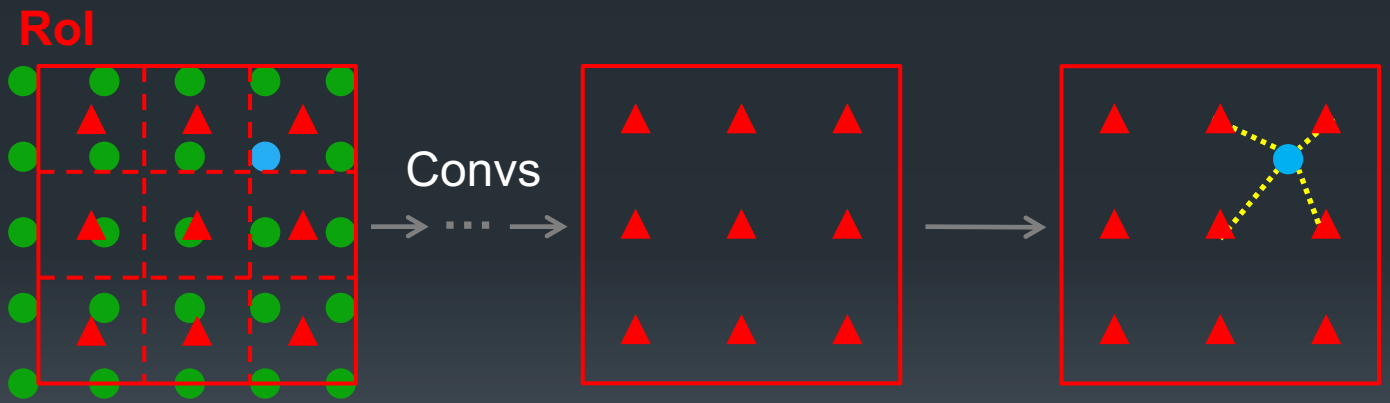
3. Mask Generator



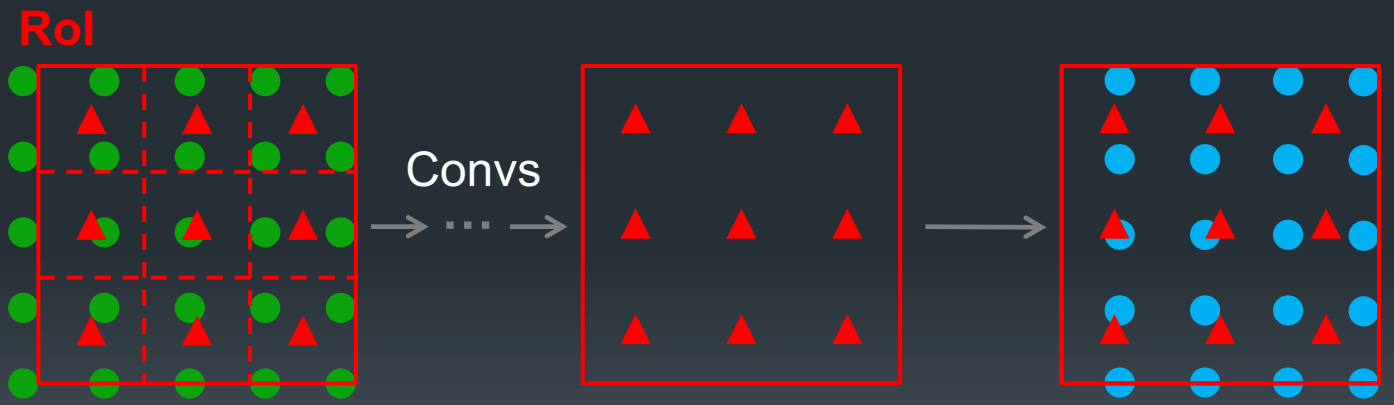
3. Mask Generator



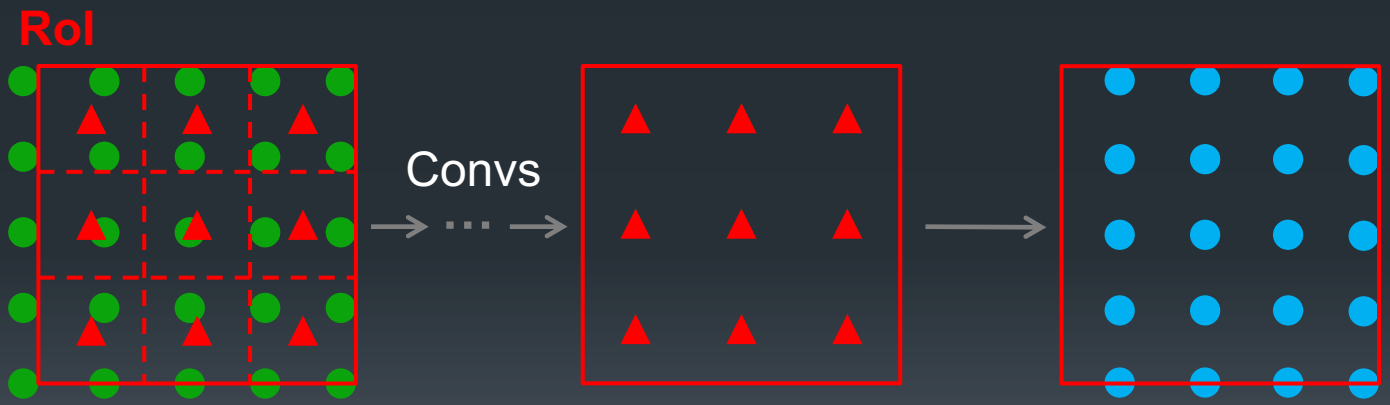
3. Mask Generator



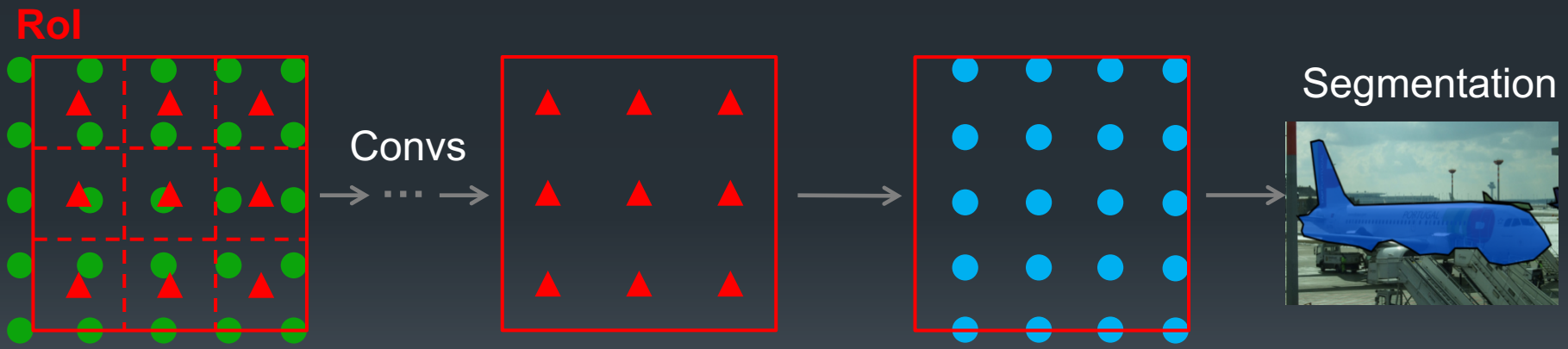
3. Mask Generator



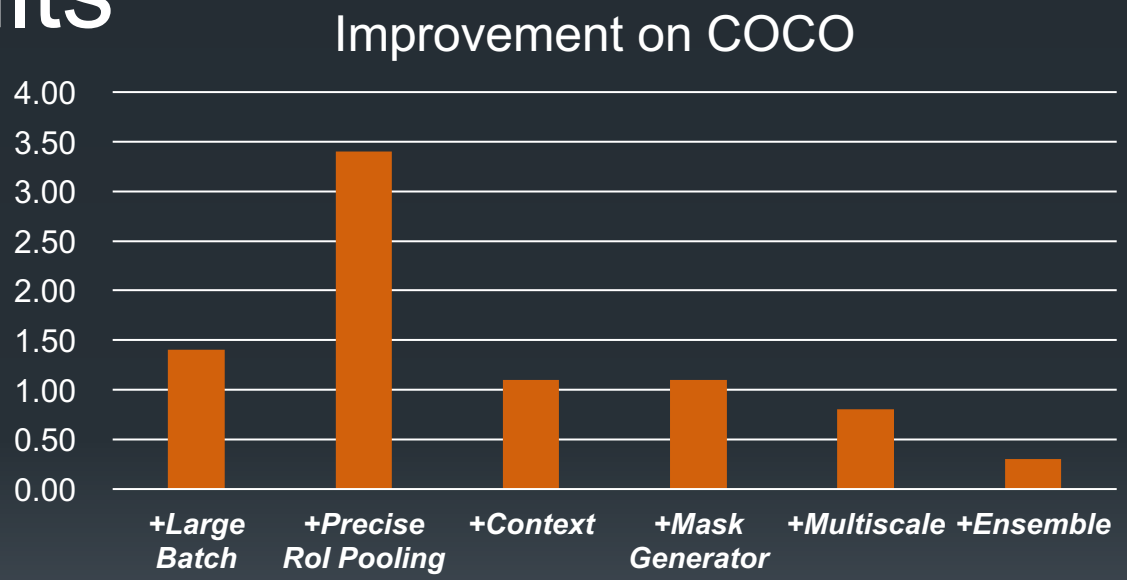
3. Mask Generator



3. Mask Generator



Results



Track	Ensemble	Single	Bbox
COCO	**46.4** (2 models)	46.1	52.8
PLACES	**30.7** (3 models)	29.8	35.1

Conclusion

- MegDet, the first large-batch detector proves successful in detection task.

Conclusion

- MegDet, the first large-batch detector proves successful in detection task.
- For segmentation, the devil lies in pixels.

Conclusion

- MegDet, the first large-batch detector proves successful in detection task.
- For segmentation, the devil lies in pixels.

- ***Training fast is the key for rapid innovation cycle.***

Face++ 旷视

We are hiring!

@Beijing, @Nanjing, @Seattle

career@megvii.com

Thanks & Questions

COCO team on “KOKO Head” :)