School of Computing and Information Systems

Learning from Limited Data for Visual Recognition

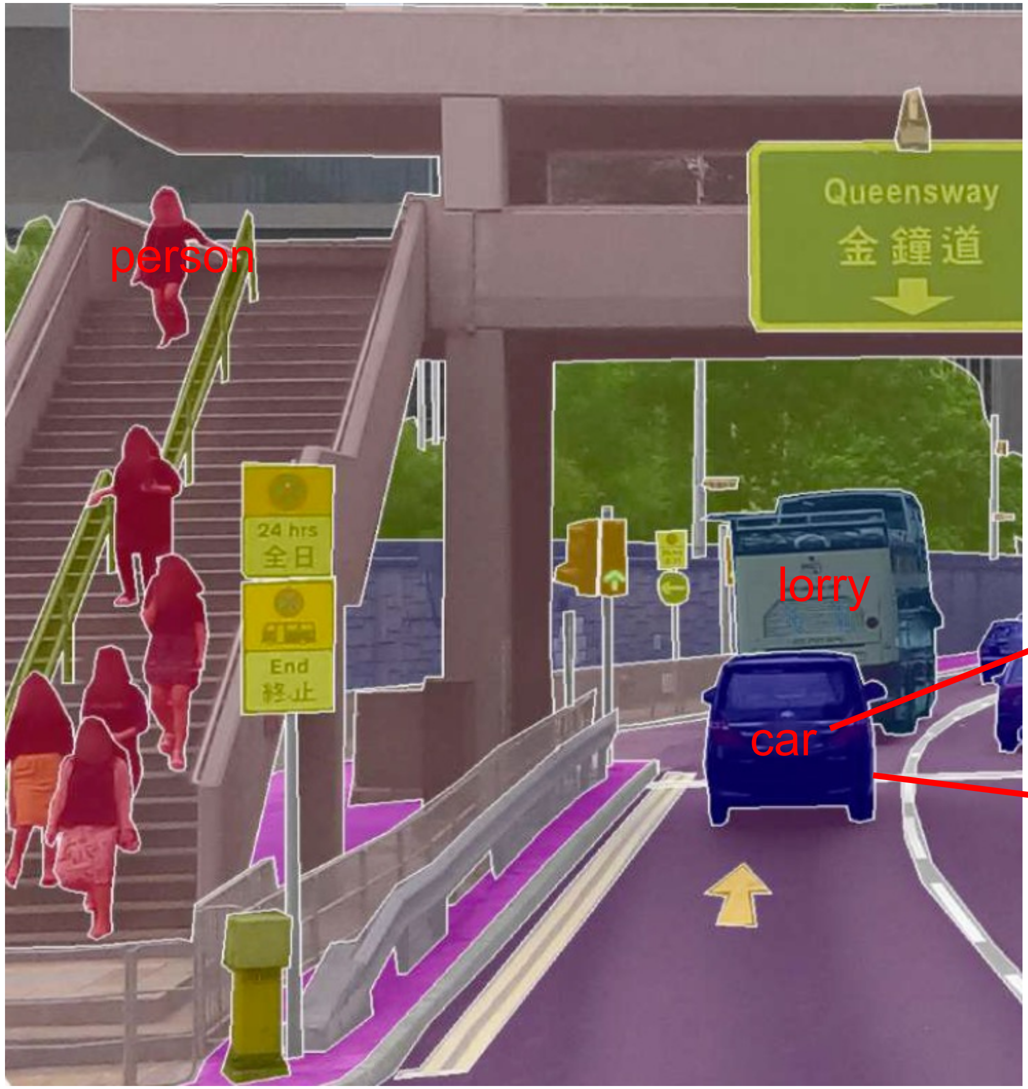
Qianru Sun

https://qianrusun.com/

School of Computing and Information Systems

Singapore Management University

What is the data for visual recognition?



data

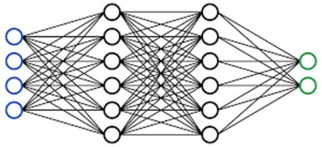
Images + labels → training → Machine model

Object-level labels → **Image classification** Task

Pixel-level labels → **Semantic segmentation** Task

Experimental data vs. Real-world data

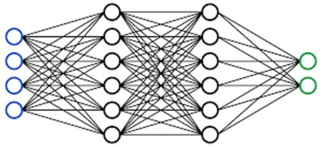
many samples with labels per class



AI Model

Experimental data vs. Real-world data

many samples with labels per class



AI Model

vs.

online data stream, limited labeled data, …

TV shows

Zoo

Before:
seen few animals

When growing up:
learn more new animals

Human

Data-limited image classification

Few-Shot Learning (FSL)

→ **Classifier**

Limited images and labels

Zero-Shot Learning (ZSL)

Seen

+

=

Unseen

No images, but attribute labels

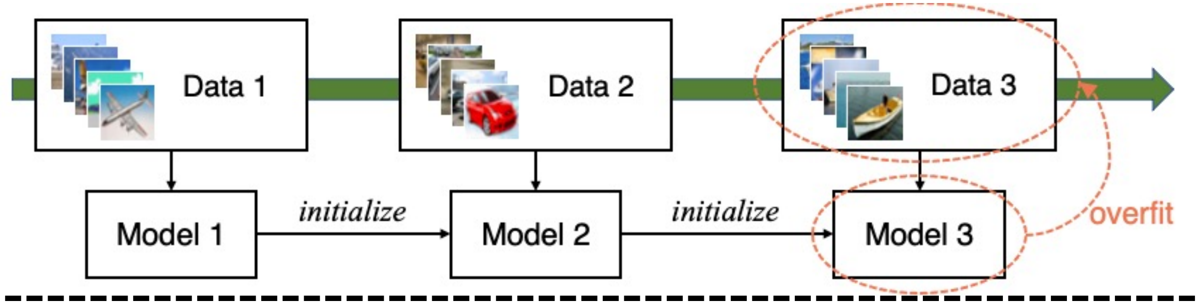
Open-Set Recognition (OSR)

Recognize seen

Reject unseen

No unseen class labels

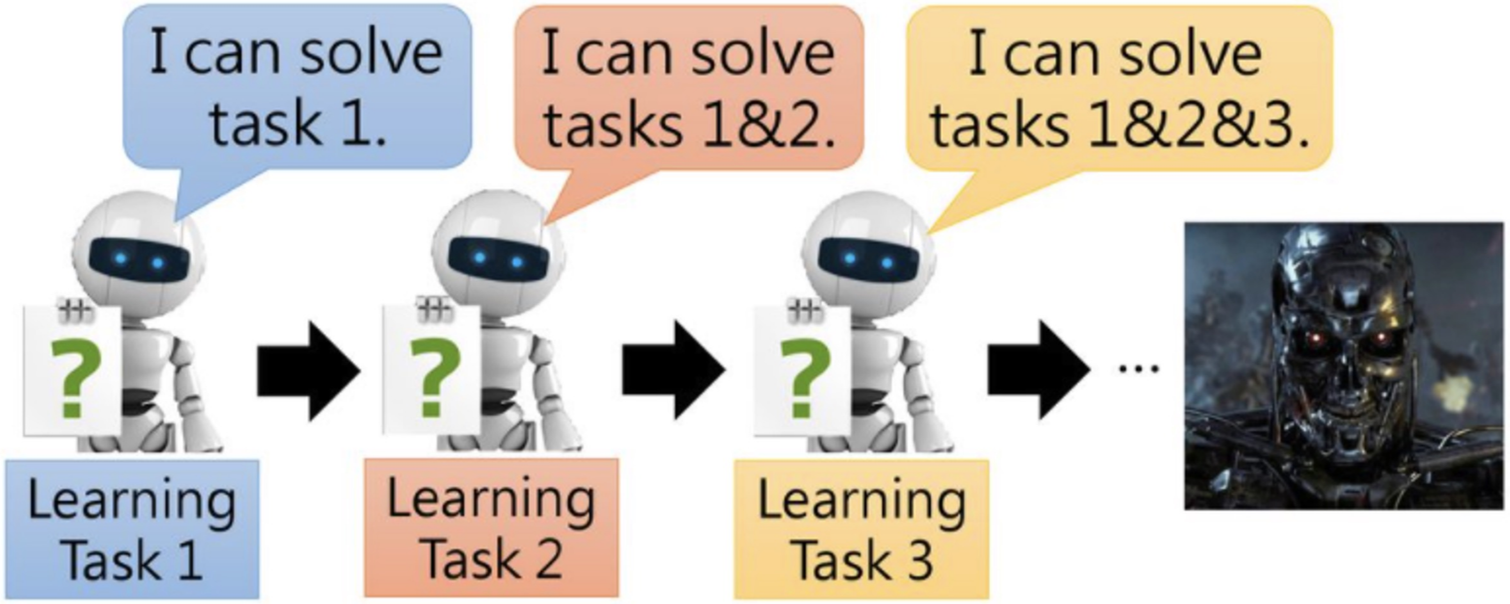
Class-Incremental Learning (CIL)



Challenge: catastrophic forgetting

Little data left for past classes

Class-Incremental Learning (CIL)



Incremental learning

Also known as: continual learning, lifelong learning, ...

Class-Incremental Learning (CIL)

Rebuffi et al.[1] demand the following three properties of an algorithm to qualify as class-incremental:

① Different classes arrive in different phases

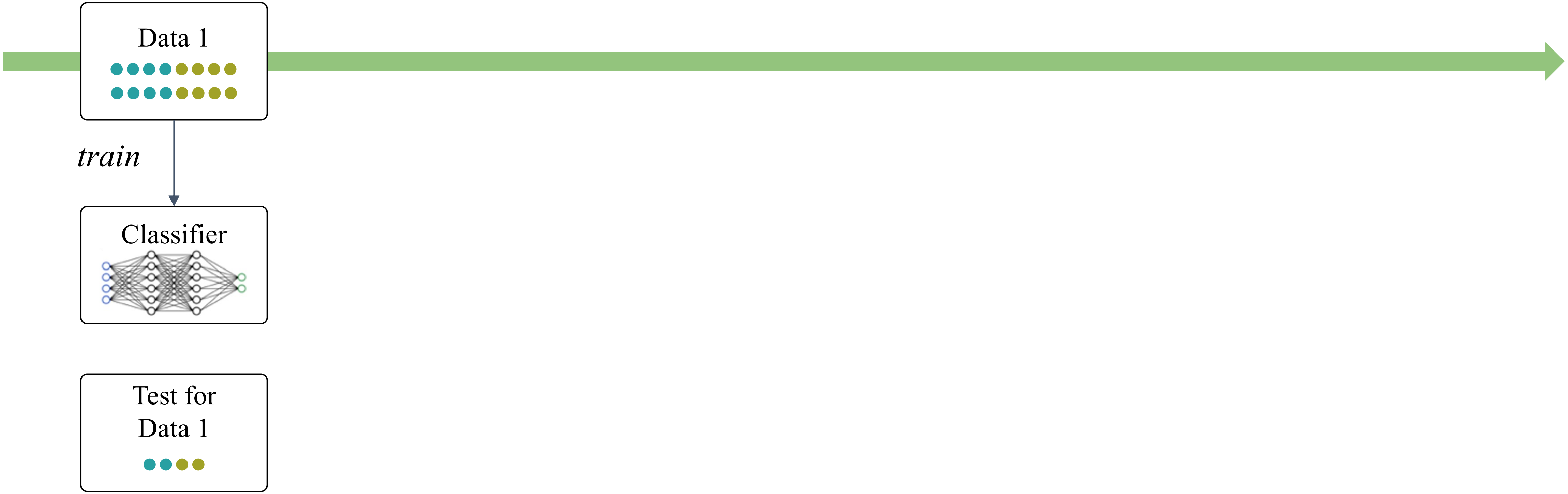
② At any time, provide a classifier for the classes observed so far

③ The memory is limited

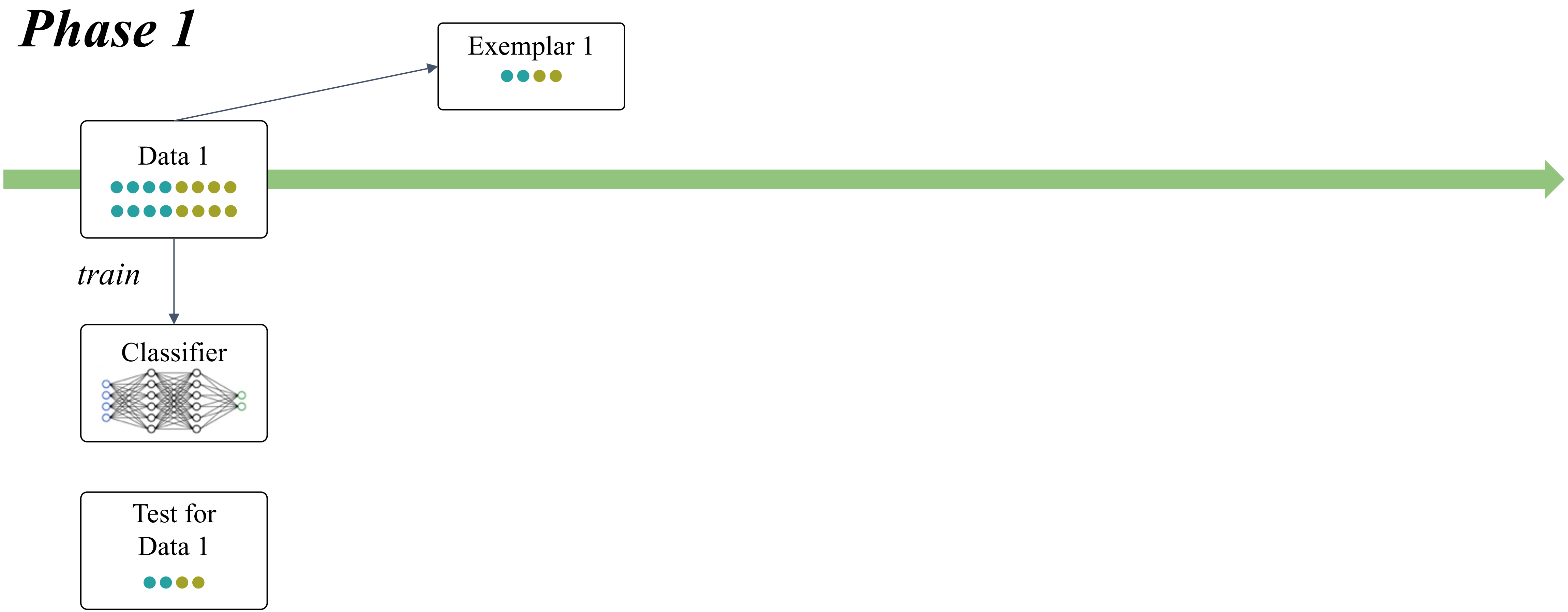
[1] Rebuffi, Sylvestre-Alvise, et al. "icarl: Incremental classifier and representation learning." CVPR 2017.

Class-Incremental Learning (CIL)

Phase 1

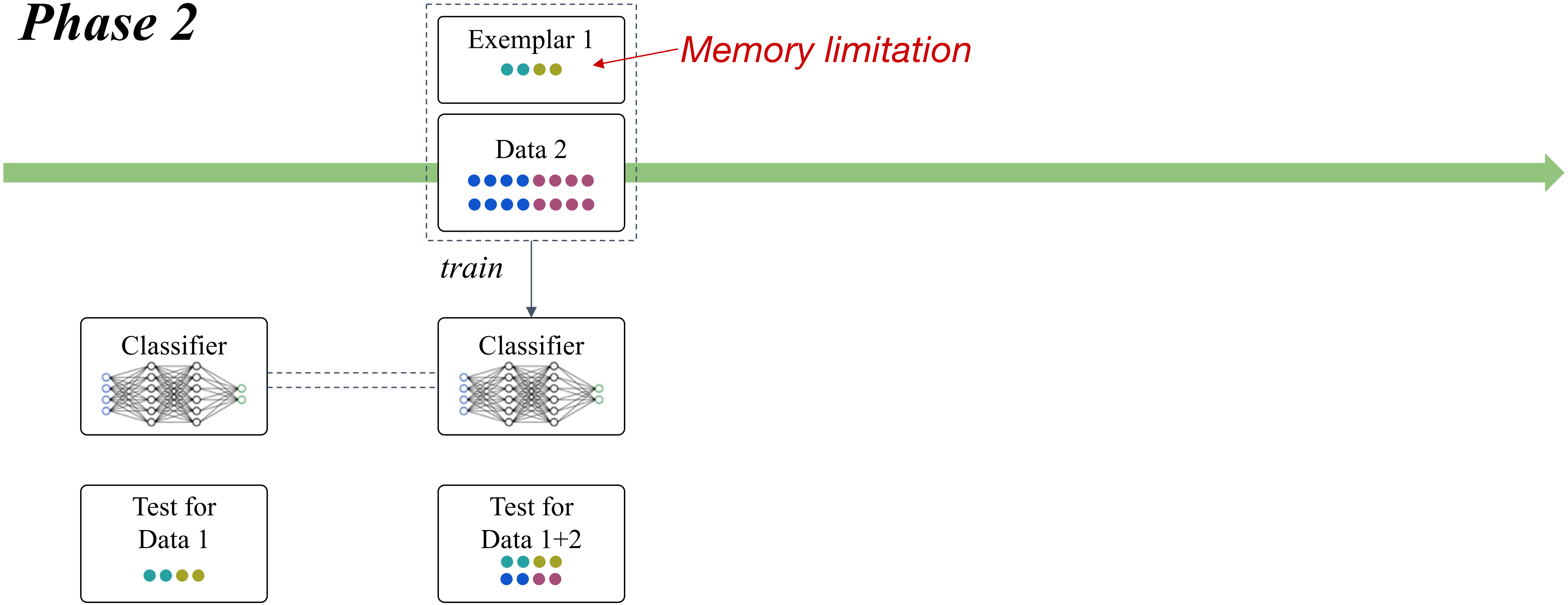


Class-Incremental Learning (CIL)



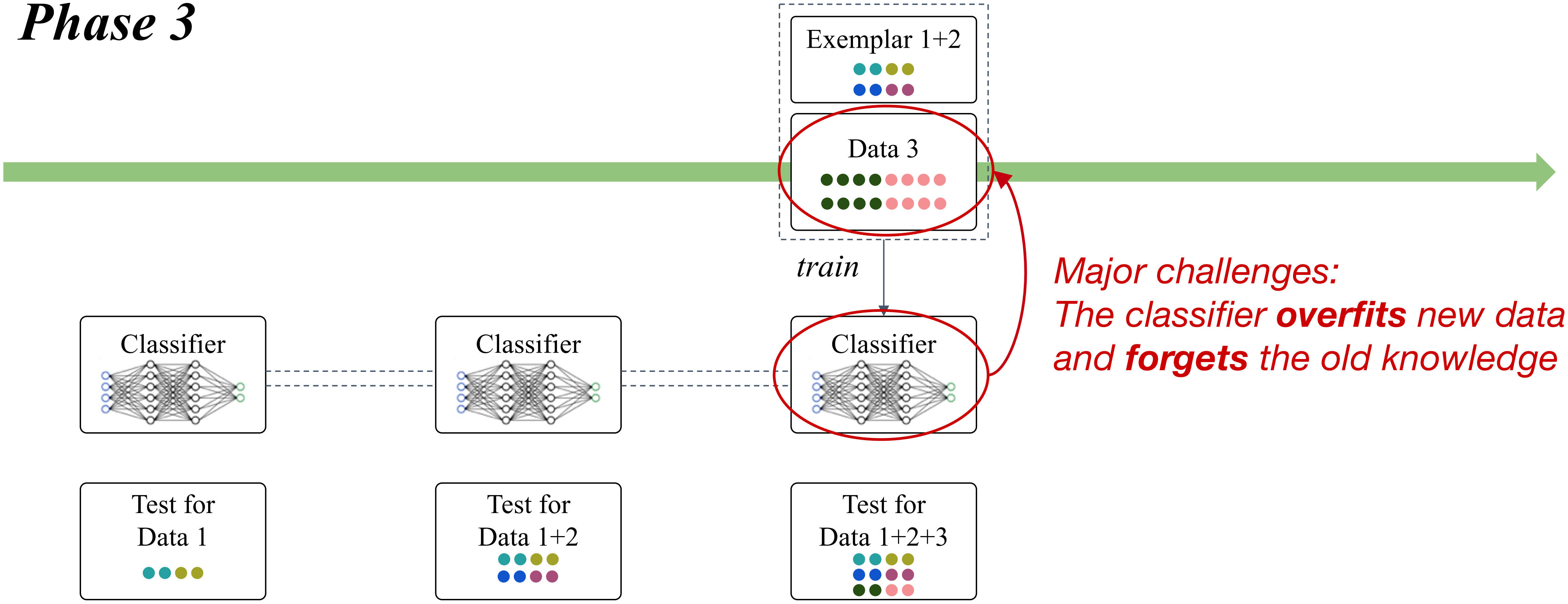
Class-Incremental Learning (CIL)

Phase 2



Class-Incremental Learning (CIL)

Phase 3



Major challenges:
*The classifier **overfits** new data and **forgets** the old knowledge*

Class-Incremental Learning (CIL)

1. Replaying on old class exemplars

Allocating as much memory as possible for the new data[1, 2, 3]

Imbalance between the old and new data

Our proposed solution: use RL to control the memory allocation

2. Using a knowledge distillation loss

Computing the distillation loss on the new data[1, 2, 3]

Hampering the learning of new classes

Our proposed solution: leverage external unlabeled data

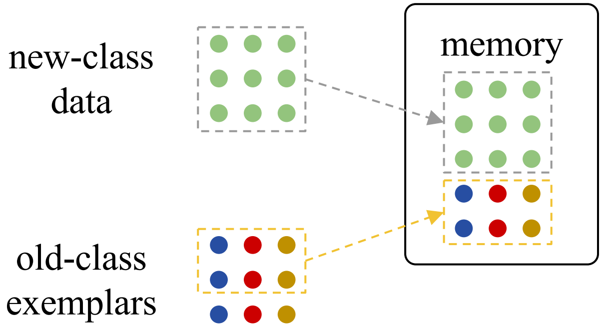
[1] Rebuffi, Sylvestre-Alvise, et al. "icarl: Incremental classifier and representation learning." CVPR 2017;
[2] Hou, Saihui, et al. "Learning a unified classifier incrementally via rebalancing." CVPR 2019;
[3] Wu, Yue, et al. “Large scale incremental learning.” CVPR 2019.

Class-Incremental Learning (CIL)

How to allocate the memory between new-class data and old-class exemplars?

Existing methods [1,2,3]

Allocate as much memory as possible for the new-class data



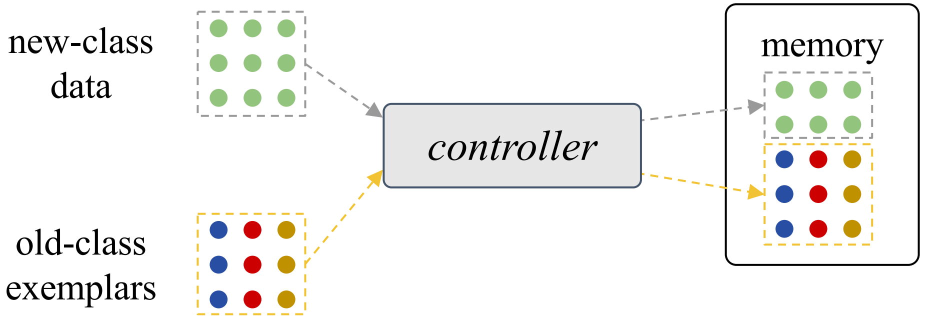
Limitations:

- Data imbalance problem

- Catastrophic forgetting problem

Our idea

Learn a controller to adjust the memory allocation



Benefits:

+ Data is more balanced

+ Overcome the forgetting problem by allocating more memory for exemplars

[1] Rebuffi, Sylvestre-Alvise, et al. "icarl: Incremental classifier and representation learning." CVPR 2017;
[2] Hou, Saihui, et al. "Learning a unified classifier incrementally via rebalancing." CVPR 2019;
[3] Wu, Yue, et al. "Large scale incremental learning." CVPR 2019.

Class-Incremental Learning (CIL)

How to allocate the memory between new-class data and old-class exemplars?

Challenge 1: due to the CIL protocol, we're not allowed to use the ***historical*** and ***future*** data

Challenge 2: the memory allocation is a ***non-differentiable*** operation

Class-Incremental Learning (CIL)

How to allocate the memory between new-class data and old-class exemplars?

Challenge 1: due to the CIL protocol, we're not allowed to use the ***historical*** and ***future*** data

Our solution: generate the ***pseudo CIL tasks***, and train the controller on them

Challenge 2: the memory allocation is a ***non-differentiable*** operation

Our solution: use the ***REINFORCE algorithm***[4] to update the controller

[4] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3-4):229–256, 1992.

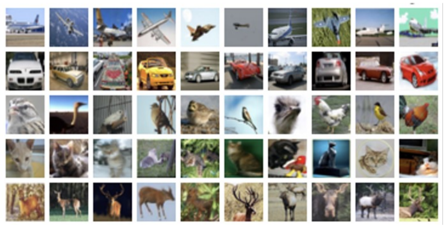
Class-Incremental Learning (CIL)

How to allocate the memory between new-class data and old-class exemplars?

Challenge 1: due to the CIL protocol, we're not allowed to use the ***historical*** and ***future*** data

Our solution: generate the ***pseudo CIL tasks***, and train the controller on them

The data of the initial phase

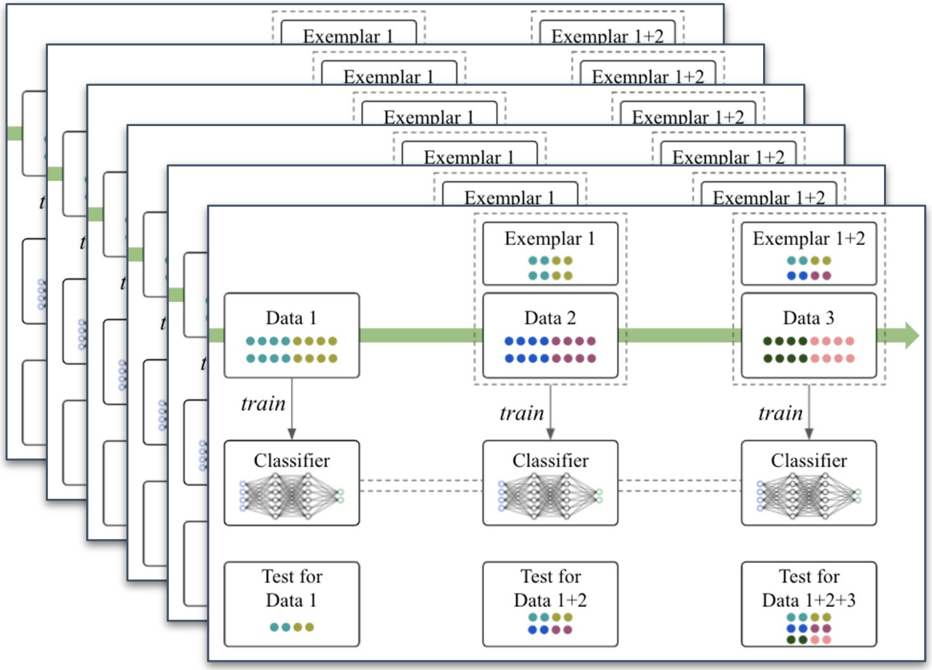


OR

Another dataset

Create

Pseudo CIL tasks



Train

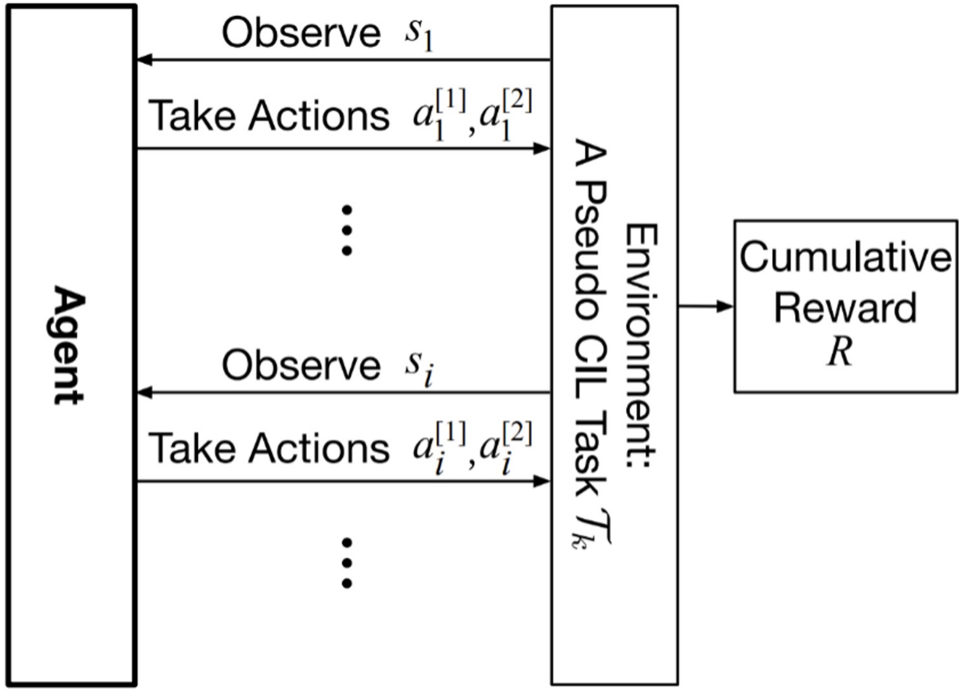
controller

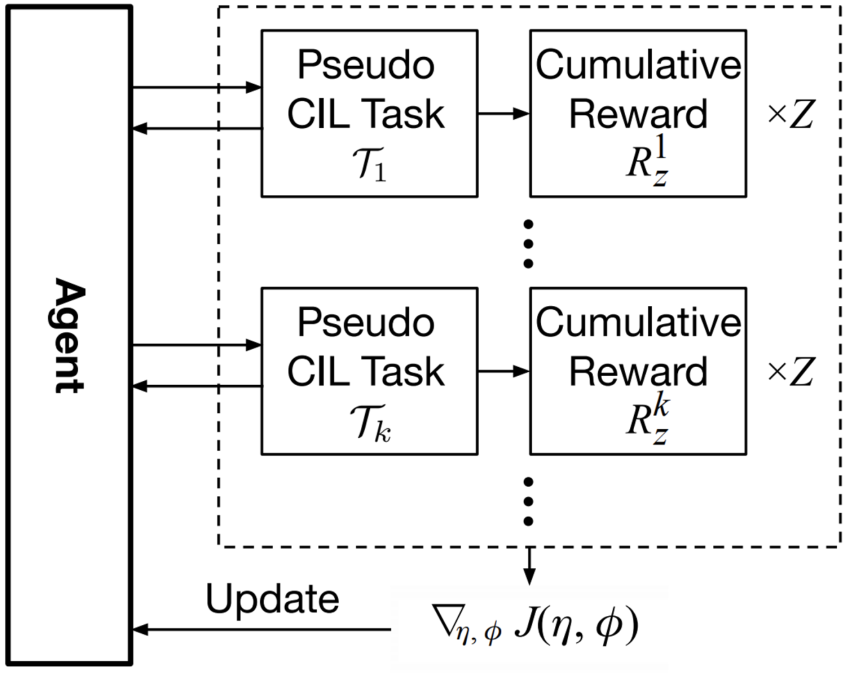
Class-Incremental Learning (CIL)

How to allocate the memory between new-class data and old-class exemplars?

Challenge 2: the memory allocation is a ***non-differentiable*** operation

Our solution: use the ***REINFORCE algorithm***[4] to update the controller





[4] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3-4):229–256, 1992.

Class-Incremental Learning (CIL)

How to allocate the memory between new-class data and old-class exemplars?

Highlighted: our method works especially well in more serious forgetting settings.

Method	CIFAR-100			ImageNet-Subset			ImageNet-Full		
	N=5	10	25	5	10	25	5	10	25
LwF [24]	56.79	53.05	50.44	58.83	53.60	50.16	52.00	47.87	47.49
iCaRL [34]	60.48	56.04	52.07	67.33	62.42	57.04	50.57	48.27	49.44
LUCIR [18]	63.34	62.47	59.69	71.21	68.21	64.15	65.16	62.34	57.37
Mnemonics [26]	64.59	62.59	61.02	72.60	71.66	70.52	65.40	64.02	62.05
PODNet [13]	64.60	63.13	61.96	76.45	74.66	70.15	66.80	64.89	60.28
LUCIR-AANets [25]	66.88	65.53	63.92	72.80	69.71	68.07	65.31	62.99	61.21
w/ RMM (ours)	68.42	67.17	64.56	73.58	72.83	72.30	65.81	64.10	62.23
POD-AANets [25]	66.61	64.61	62.63	77.36	75.83	72.18	67.97	65.03	62.03
w/ RMM (ours)	**68.86**	**67.61**	**66.21**	**79.52**	**78.47**	**76.54**	**69.21**	**67.45**	**63.93**

Class-Incremental Learning (CIL)

How to solve the conflict between distillation and cross-entropy in CIL?

Existing methods and problems

Computing distillation loss on new data[1, 2]

Our idea

Selecting the unlabelled data and
Computing distillation loss on these data

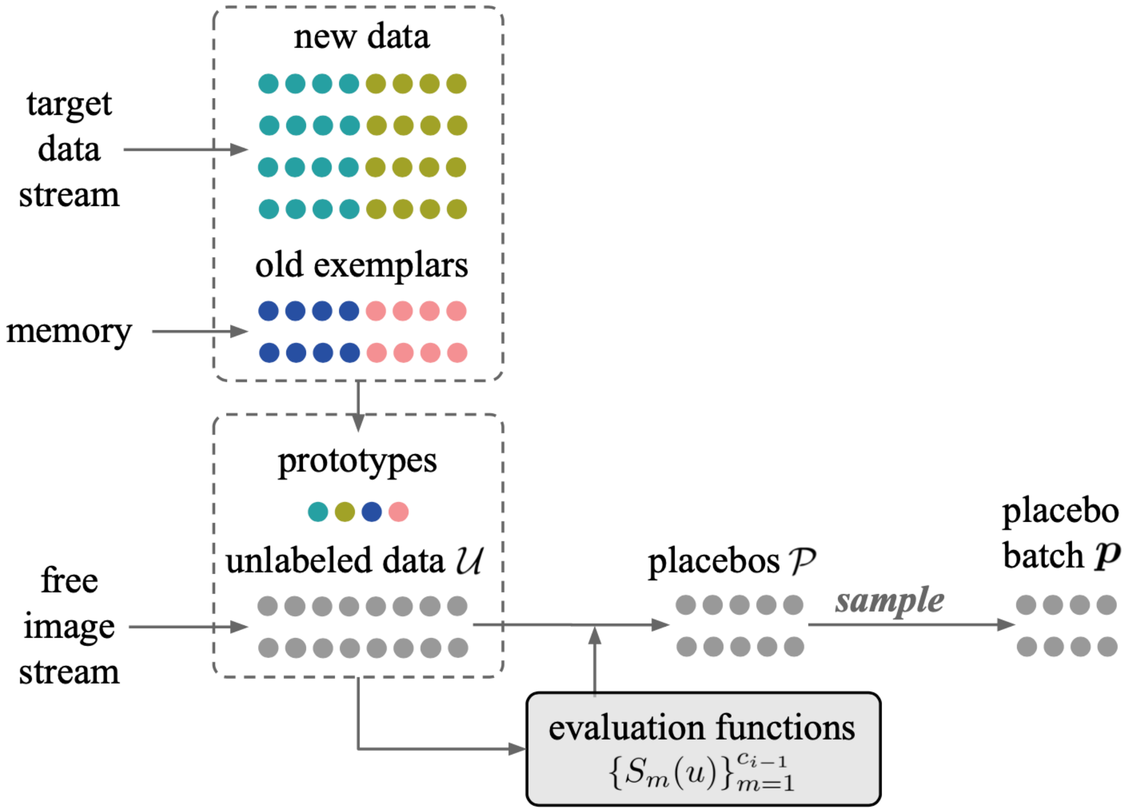
Benefits:

+ No depreciation for new class performance
+ No additional supervision required
+ Easy to train

[1] Rebuffi, Sylvestre-Alvise, et al. "icarl: Incremental classifier and representation learning." CVPR 2017;
[2] Li, Zhizhong, and Derek Hoiem. "Learning without forgetting." TPAMI 2017;

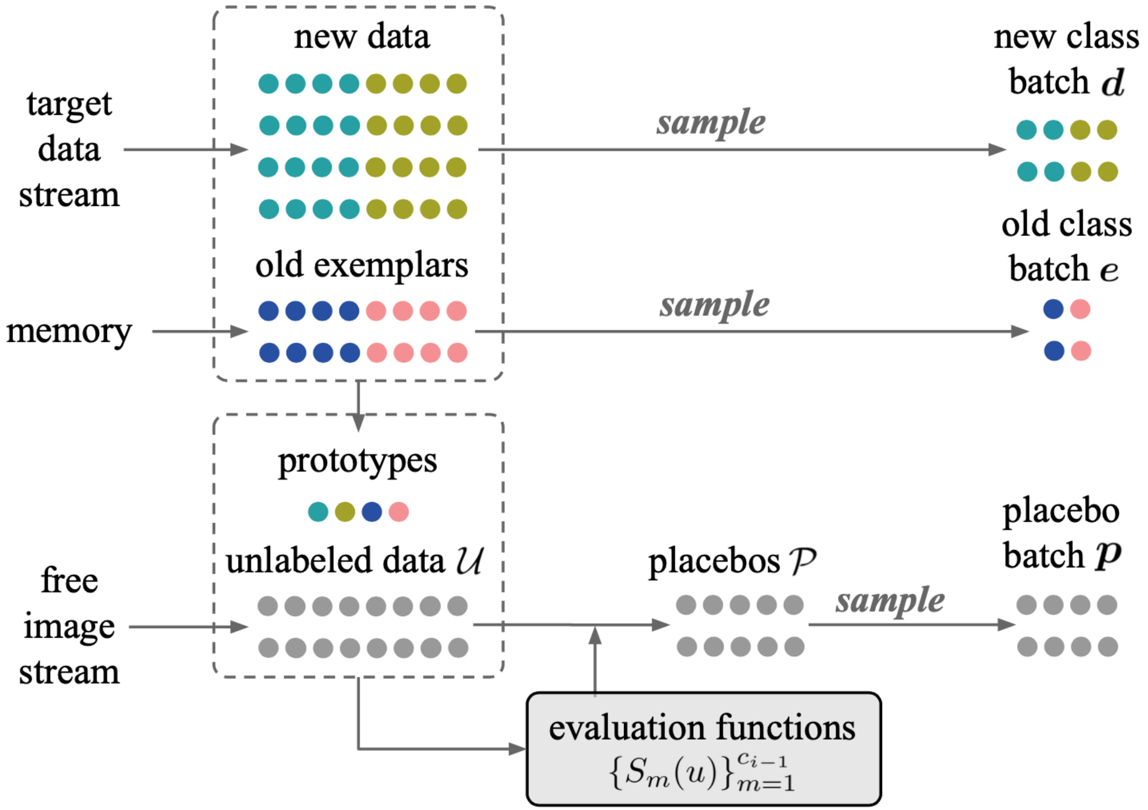
Class-Incremental Learning (CIL)

How to solve the conflict between distillation and cross-entropy in CIL?



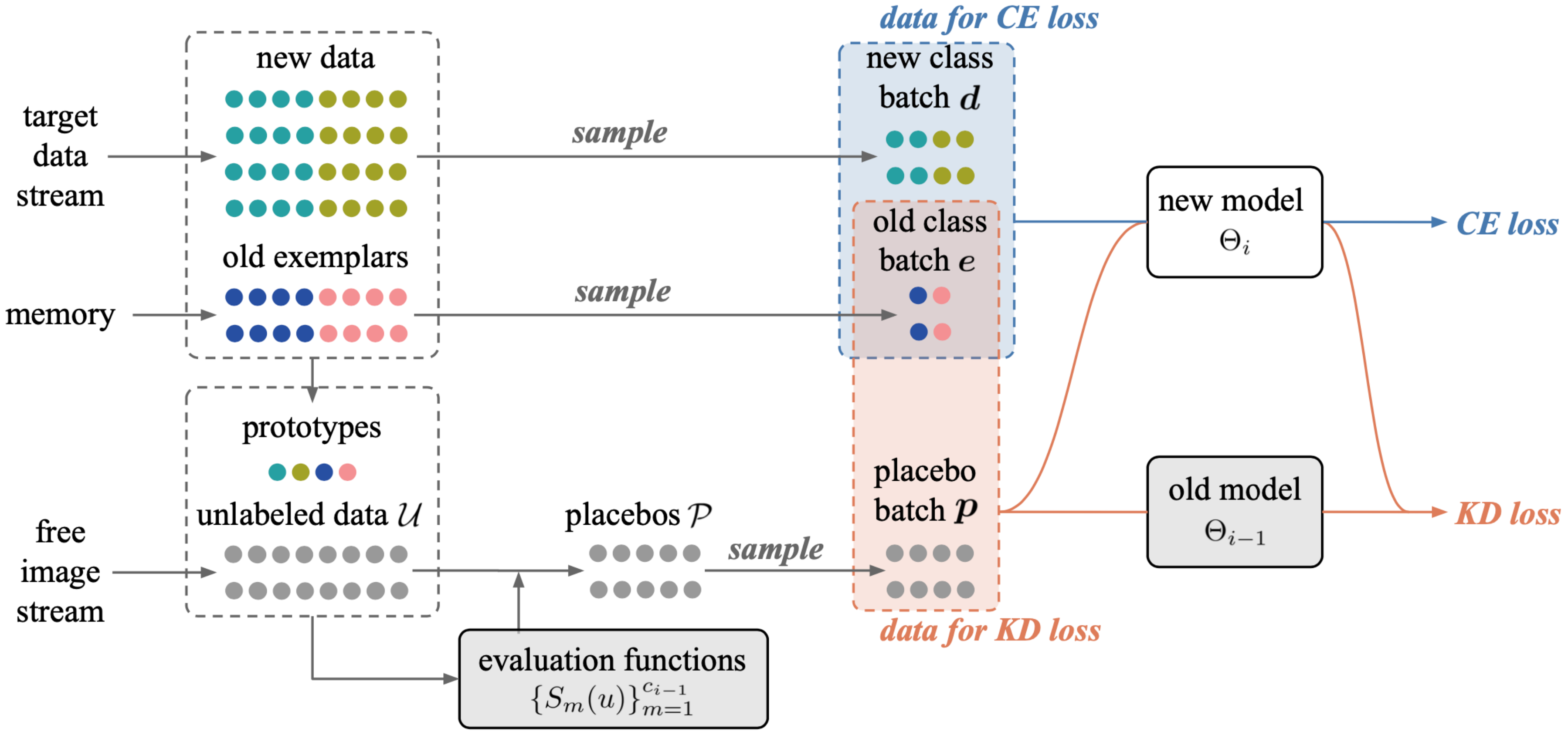
Class-Incremental Learning (CIL)

How to solve the conflict between distillation and cross-entropy in CIL?



Class-Incremental Learning (CIL)

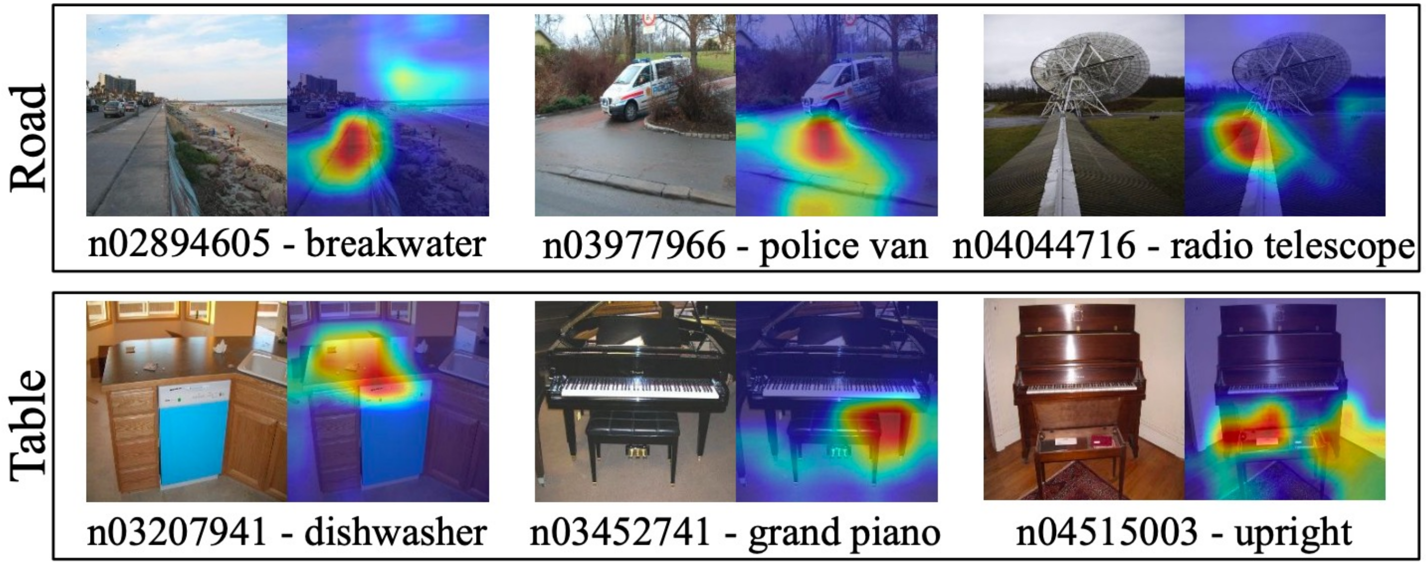
How to solve the conflict between distillation and cross-entropy in CIL?



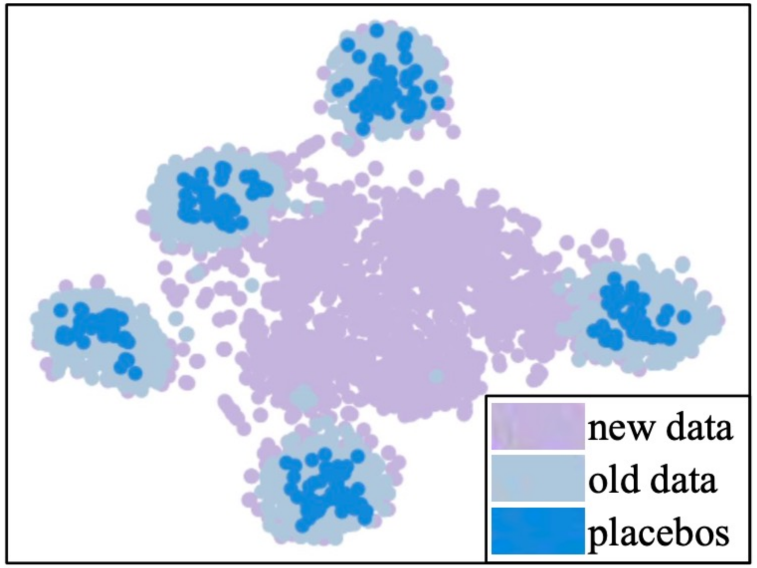
Class-Incremental Learning (CIL)

How to solve the conflict between distillation and cross-entropy in CIL?

Visualization results: related cues are found in the unlabelled images



(a) Selected placebos and GradCAM visualization



(b) t-SNE visualization

Class-Incremental Learning (CIL)

How to solve the conflict between distillation and cross-entropy in CIL?

Quantitative results: our method works especially well in low-shot (in old classes) settings

Method	20 exemplars/class		10 exemplars/class		5 exemplars/class	
	Average	Last	Average	Last	Average	Last
LwF	53.19	43.18	45.96	34.10	35.41	24.91
w/ ours	59.29 +6.10	49.64 +6.46	53.48 +7.52	38.03 +3.93	41.67 +6.26	28.60 +3.69
iCaRL	57.12	47.49	53.43	41.49	43.73	34.33
w/ ours	61.17 +4.05	50.96 +3.47	59.32 +5.89	46.48 +4.99	51.19 +7.46	39.29 +4.96
LUCIR	63.17	53.71	60.50	49.08	51.36	39.37
w/ ours	65.48 +2.31	56.77 +3.06	64.93 +3.89	55.54 +6.46	63.01 +11.65	53.09 +13.72
LUCIR+AANets	66.72	57.77	65.46	55.17	60.28	48.23
w/ ours	67.33 +0.61	59.32 +1.55	65.51 +0.05	55.42 +0.25	64.10 +3.82	53.41 +5.18
POD+AANets	66.12	55.27	61.12	48.83	53.81	42.93
w/ ours	67.47 +1.35	58.91 +3.64	64.56 +3.44	52.60 +3.77	60.35 +6.54	48.53 +5.60

School of Computing and Information Systems

Class-Incremental Learning and related…

T.-S. Chua

B. Schiele

Related works in our team

Z. Luo, Y. Liu, B. Schiele, Q. Sun. Class-Incremental Exemplar Compression for Class-Incremental Learning. CVPR 2023.

Y. Liu, Y. Li, B. Schiele, Q. Sun. Online Hyperparameter Optimization for Class-Incremental Learning. AAAI 2023. Oral.

Q. Sun* Y. Liu*, Z. Chen, T.-S. Chua, B. Schiele. Meta-Transfer Learning through Hard Tasks. T-PAMI 2022.

Y. Liu, B. Schiele, Q. Sun. RMM: Reinforced Memory Management for Class-Incremental Learning. NeurIPS 2021.

Y. Liu, B. Schiele, Q. Sun. Adaptive Aggregation Networks for Class-Incremental Learning. CVPR 2021.

Y. Liu, Y. Su, A.-A. Liu, B. Schiele, Q. Sun. Mnemonics training: Multi-class incremental learning without forgetting. CVPR 2020. Oral.

Y. Liu, B. Schiele, Q. Sun. An Ensemble of Epoch-wise Empirical Bayes for Few-shot Learning. ECCV 2020.

Q. Sun*, Y. Liu*, T.-S. Chua, B. Schiele. Meta-Transfer Learning for Few-Shot Learning. CVPR 2019. 1000+ citations.

X, Li, Q. Sun, Y. Liu, T.-S. Chua, et al. Learning to Self-Train for Semi-Supervised Few-Shot Classification. NeurIPS 2019.

Y. Liu

Label-limited image classification

Few-Shot Learning (FSL)

→ **Classifier**

Limited images and labels

Zero-Shot Learning (ZSL)

Seen

+

=

Unseen

No images, but attribute labels

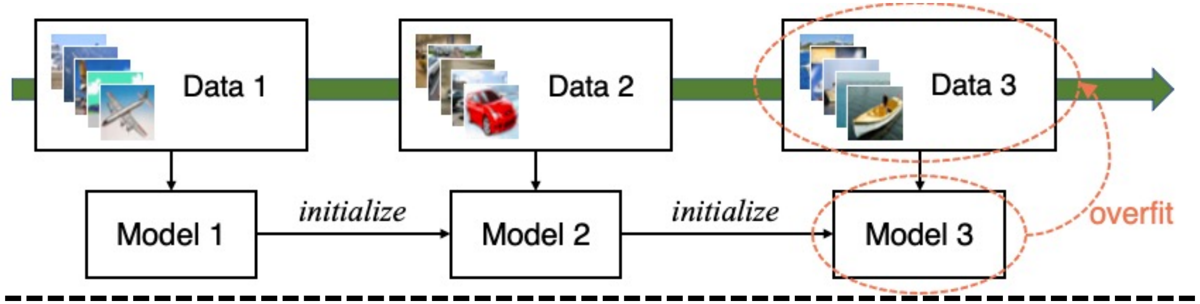
Open-Set Recognition (OSR)

Recognize seen

Reject unseen

No unseen class labels

Class-Incremental Learning (CIL)



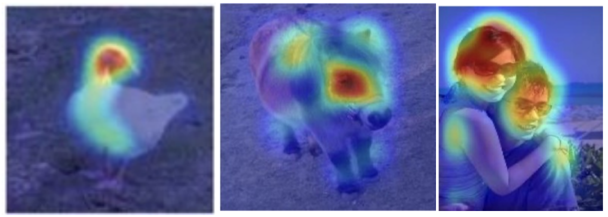
Challenge: catastrophic forgetting

Little data left for past classes

Label-limited semantic segmentation

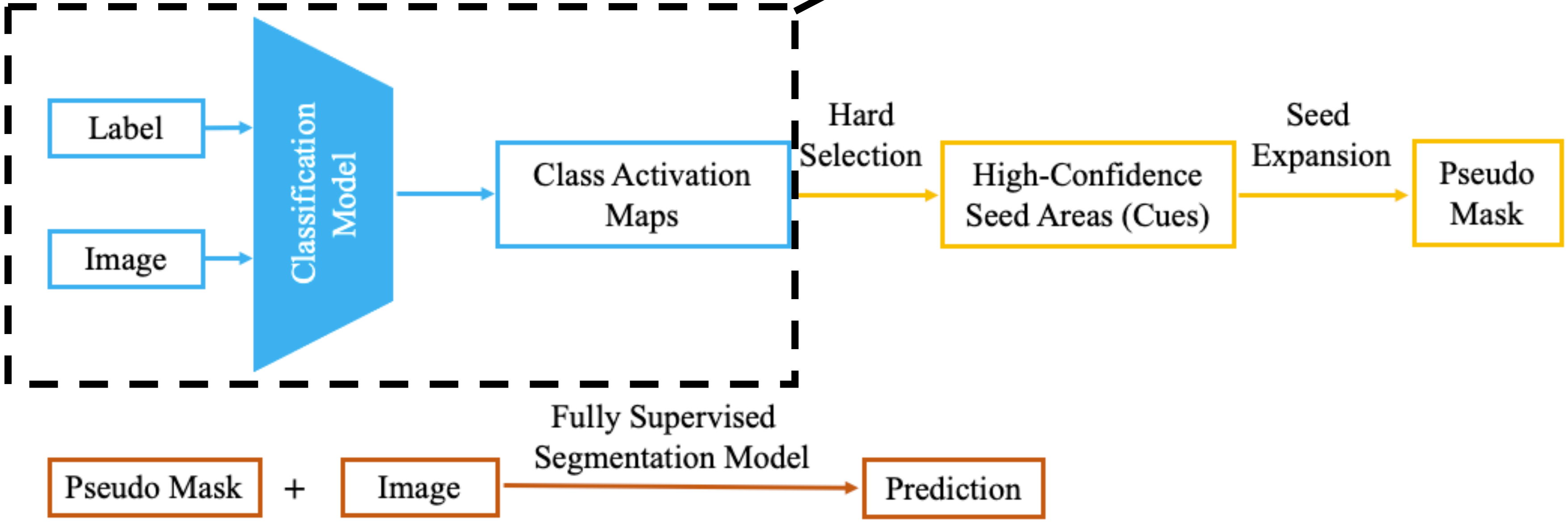
Class Activation Maps (CAM)

CAM



Only image-level labels

Weakly Supervised Semantic Segmentation (WSSS)



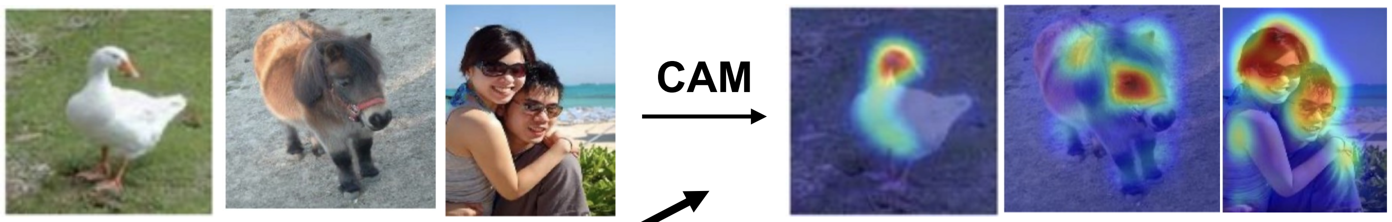
No pixel-level labels; only image level labels

Weakly-Supervised Semantic Segmentation (WSSS)

Why do we need weakly-supervised semantic segmentation techniques?

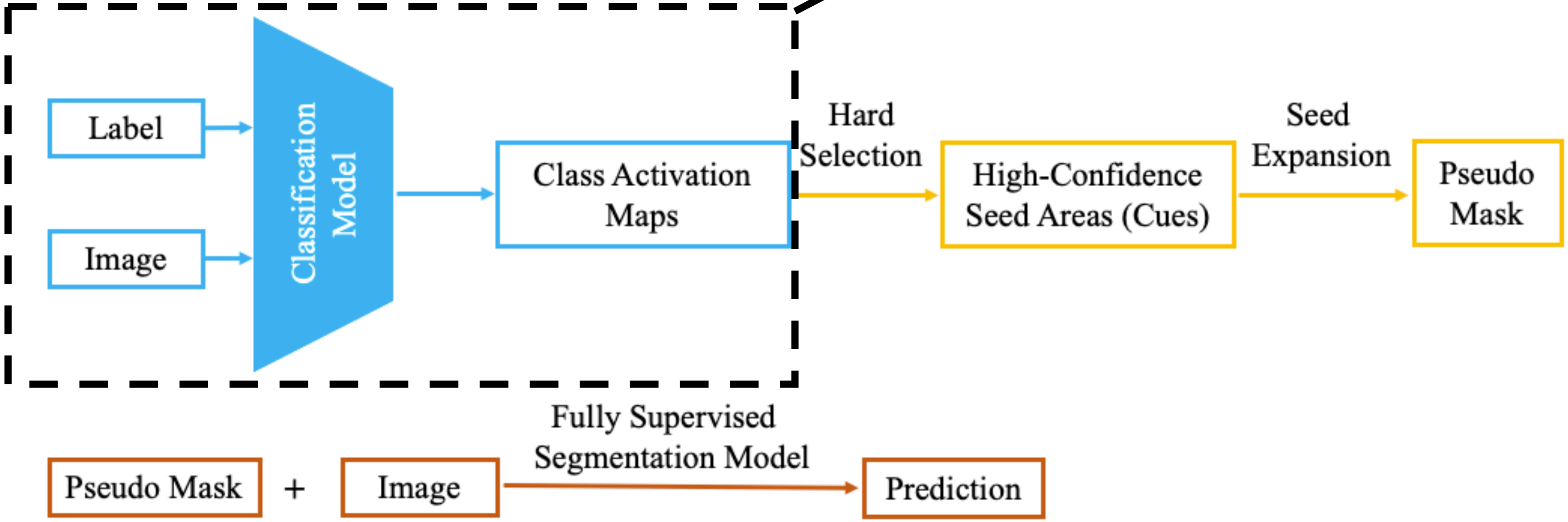
Label-limited semantic segmentation

Class Activation Maps (CAM)



Only image-level labels

Weakly Supervised Semantic Segmentation (WSSS)



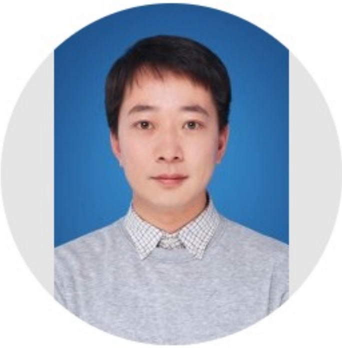
No pixel-level labels; only image level labels

Label-limited semantic segmentation

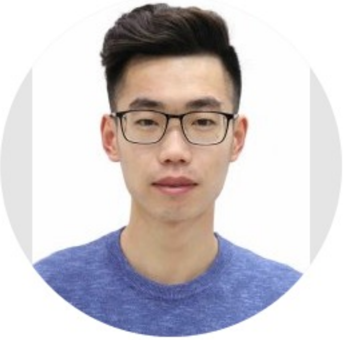
Causal Intervention for Weakly-Supervised Semantic Segmentation
Dong Zhang, Hanwang Zhang, Jinhui Tang, Xian-Sheng Hua, **Qianru Sun**
Neural Information Processing Systems, NeurIPS '20. (Oral Presentation, 1.1%)
[paper] [code]

Class Re-Activation Maps for Weakly-Supervised Semantic Segmentation
Zhaozheng Chen, Tan Wang, Xiongwei Wu, Xian-Sheng Hua, Hanwang Zhang, **Qianru Sun**
2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR '22.
[paper] [code]

Extracting Class Activation Maps from Non-Discriminative Features as well
Zhaozheng Chen, **Qianru Sun**
2023 Conference on Computer Vision and Pattern Recognition, CVPR '23.
[paper] [code]



Dong Zhang



Zhaozheng Chen

Only image-level labels

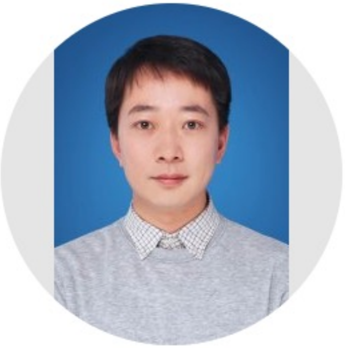
No pixel-level labels; only image level labels

Label-limited semantic segmentation

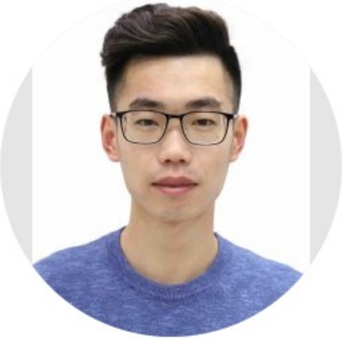
Causal Intervention for Weakly-Supervised Semantic Segmentation
Dong Zhang, Hanwang Zhang, Jinhui Tang, Xian-Sheng Hua, **Qianru Sun**
Neural Information Processing Systems, NeurIPS '20. (Oral Presentation, 1.1%)
[paper] [code]

Class Re-Activation Maps for Weakly-Supervised Semantic Segmentation
Zhaozheng Chen, Tan Wang, Xiongwei Wu, Xian-Sheng Hua, Hanwang Zhang, **Qianru Sun**
2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR '22.
[paper] [code]

Extracting Class Activation Maps from Non-Discriminative Features as well
Zhaozheng Chen, **Qianru Sun**
2023 Conference on Computer Vision and Pattern Recognition, CVPR '23.
[paper] [code]



Dong Zhang



Zhaozheng Chen

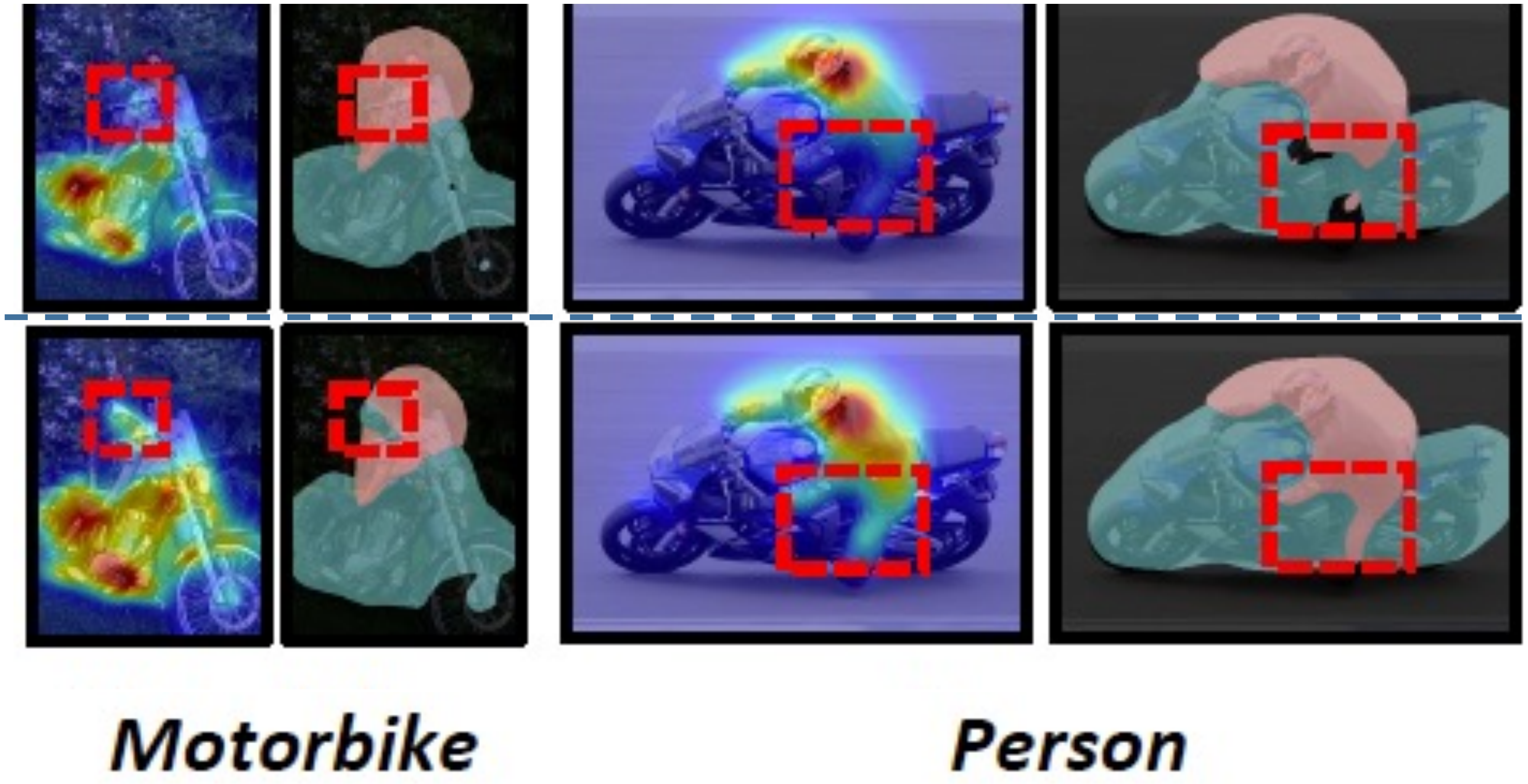
Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

We found many confusing regions are between co-occurring objects

BCE results, i.e., vanilla CAM

Here are better ones:



Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

Why?

We inspect the Sigmoid function in BCE: exp(x)/(1+exp(x))
where x denotes the prediction logit of any individual class e.g., person.

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

What about Softmax CE (SCE)?

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

What about Softmax CE (SCE)?

- 80-class models: BCE and SCE yield equal-quality classifiers but clearly different CAMs

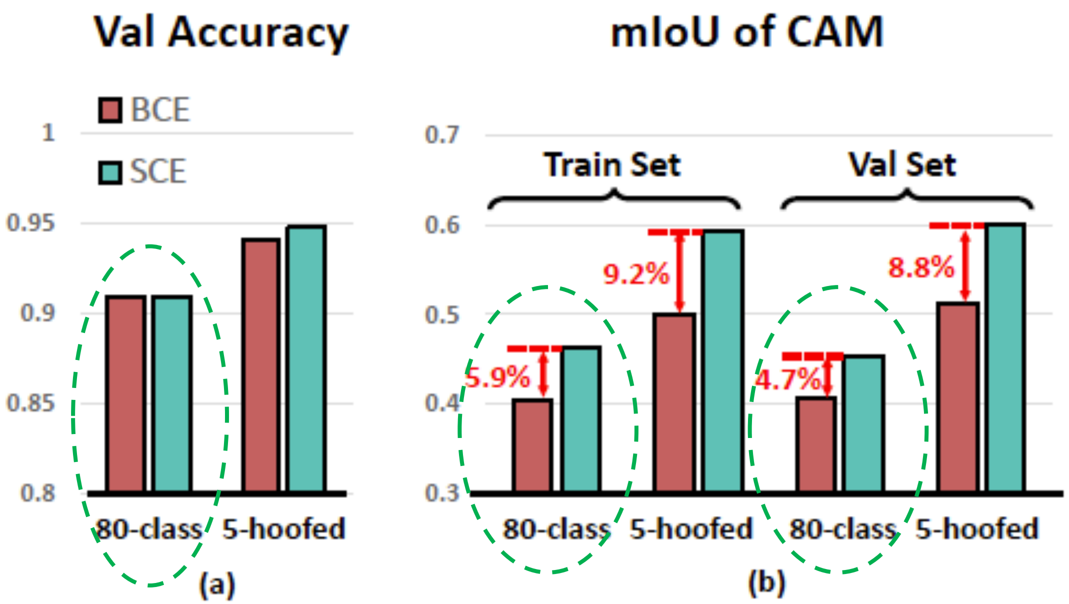


Figure 1. We train two models respectively using binary cross entropy (BCE) and softmax cross entropy (SCE) losses. Our `train` and `val` sets contain only single-label images of MS COCO [30]. "80-class" model uses the complete label set. "5-hoofed" model is trained on only the samples of 5 hoofed animals each causing false positive flaws to another, e.g., between `cow` and `horse`.

School of Computing and Information Systems

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

What about Softmax CE (SCE)?

- The CAMs of SCE models are of higher mIoU.
- This superiority is maintained in validation images.

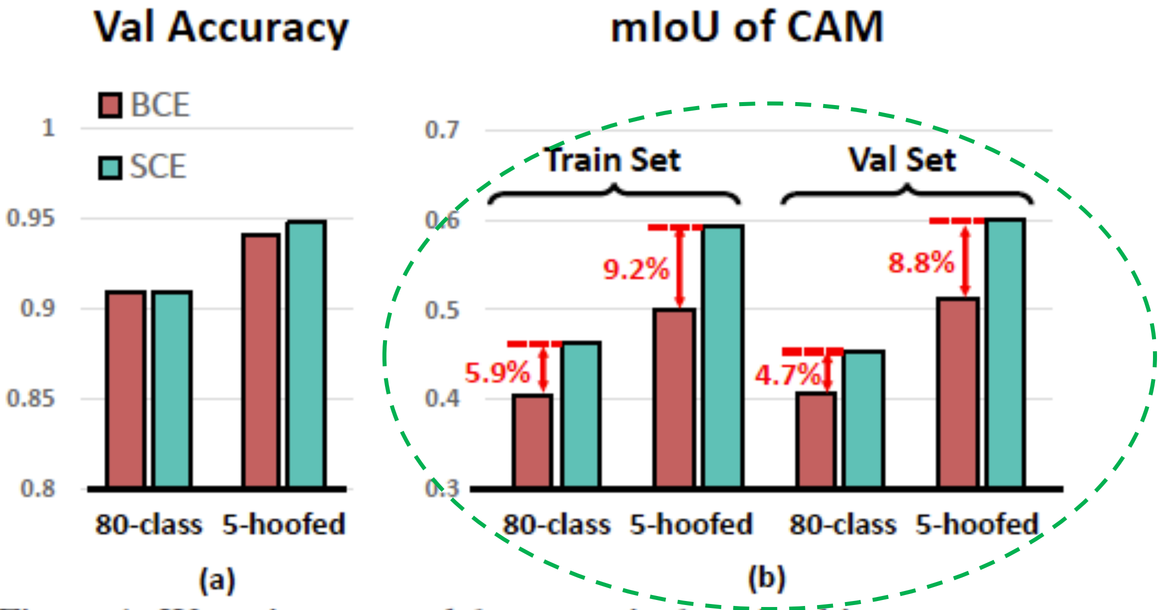


Figure 1. We train two models respectively using binary cross entropy (BCE) and softmax cross entropy (SCE) losses. Our train and val sets contain only single-label images of MS COCO [30]. "80-class" model uses the complete label set. "5-hoofed" model is trained on only the samples of 5 hoofed animals each causing false positive flaws to another, e.g., between cow and horse.

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

Justification: BCE vs. SCE

For the ease of analysis, we consider the binary-class (K = 2) situation with the positive class p and negative class q:

$$① \ \nabla_{z_p}\mathcal{L}_{bce} = \frac{-1}{2+2e^{z_p}} \qquad ② \ \nabla_{z_q}\mathcal{L}_{bce} = \frac{1}{2+2e^{-z_q}}$$

$$③ \ \nabla_{z_p}\mathcal{L}_{sce} = \frac{-1}{1+e^{z_p-z_q}} \qquad ④ \ \nabla_{z_q}\mathcal{L}_{sce} = \frac{1}{1+e^{z_p-z_q}}$$

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

Justification: BCE vs. SCE

For the ease of analysis, we consider the binary-class (K = 2) situation with the positive class p and negative class q:

① $$\nabla_{z_p}\mathcal{L}_{bce} = \frac{-1}{2+2e^{z_p}}$$ ② $$\nabla_{z_q}\mathcal{L}_{bce} = \frac{1}{2+2e^{-z_q}}$$

③ $$\nabla_{z_p}\mathcal{L}_{sce} = \frac{-1}{1+e^{z_p-z_q}}$$ ④ $$\nabla_{z_q}\mathcal{L}_{sce} = \frac{1}{1+e^{z_p-z_q}}$$

For confusing prediction logits, i.e., $z_p \approx z_q$, there are two subcases: both are of small or large numbers. In these cases, either $\nabla_{z_p}\mathcal{L}_{bce}$ or $\nabla_{z_q}\mathcal{L}_{bce}$ is zero.

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

Justification: BCE vs. SCE

For the ease of analysis, we consider the binary-class (K = 2) situation with the positive class p and negative class q:

① $$\nabla_{z_p}\mathcal{L}_{bce} = \frac{-1}{2+2e^{z_p}}$$ ② $$\nabla_{z_q}\mathcal{L}_{bce} = \frac{1}{2+2e^{-z_q}}$$

③ $$\nabla_{z_p}\mathcal{L}_{sce} = \frac{-1}{1+e^{z_p-z_q}}$$ ④ $$\nabla_{z_q}\mathcal{L}_{sce} = \frac{1}{1+e^{z_p-z_q}}$$

For confusing prediction logits, i.e., $z_p \approx z_q$, there are two subcases: both are of small or large numbers. In these cases, both $\nabla_{z_p}\mathcal{L}_{sce}$ and $\nabla_{z_q}\mathcal{L}_{sce}$ are non-zeros.

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

Justification: BCE vs. SCE

For the ease of analysis, we consider the binary-class (K = 2) situation with the positive class p and negative class q:

$$① \nabla_{z_p}\mathcal{L}_{bce} = \frac{-1}{2+2e^{z_p}} \qquad ② \nabla_{z_q}\mathcal{L}_{bce} = \frac{1}{2+2e^{-z_q}}$$

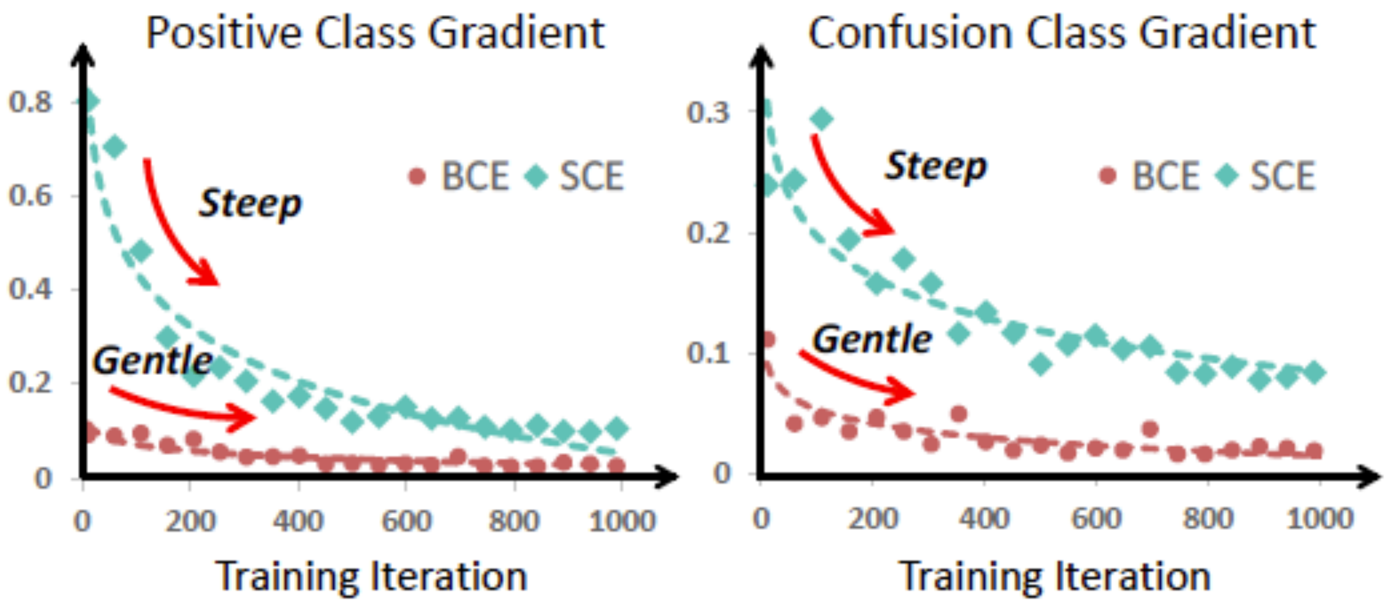
$$③ \nabla_{z_p}\mathcal{L}_{sce} = \frac{-1}{1+e^{z_p-z_q}} \qquad ④ \nabla_{z_q}\mathcal{L}_{sce} = \frac{1}{1+e^{z_p-z_q}}$$

Therefore, **SCE is more active than BCE** to yield gradients for optimization.

Weakly-Supervised Semantic Segmentation (WSSS)

The problem we found in Step 1: binary cross-entropy (BCE) loss is **inefficient**

Justification: BCE vs. SCE

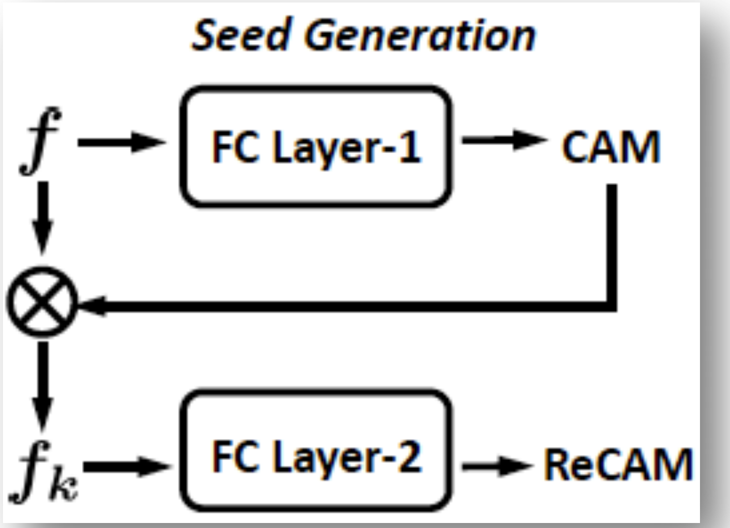


- The gradients of SCE loss change more rapidly for both positive and negative classes
- The SCE model learns more actively

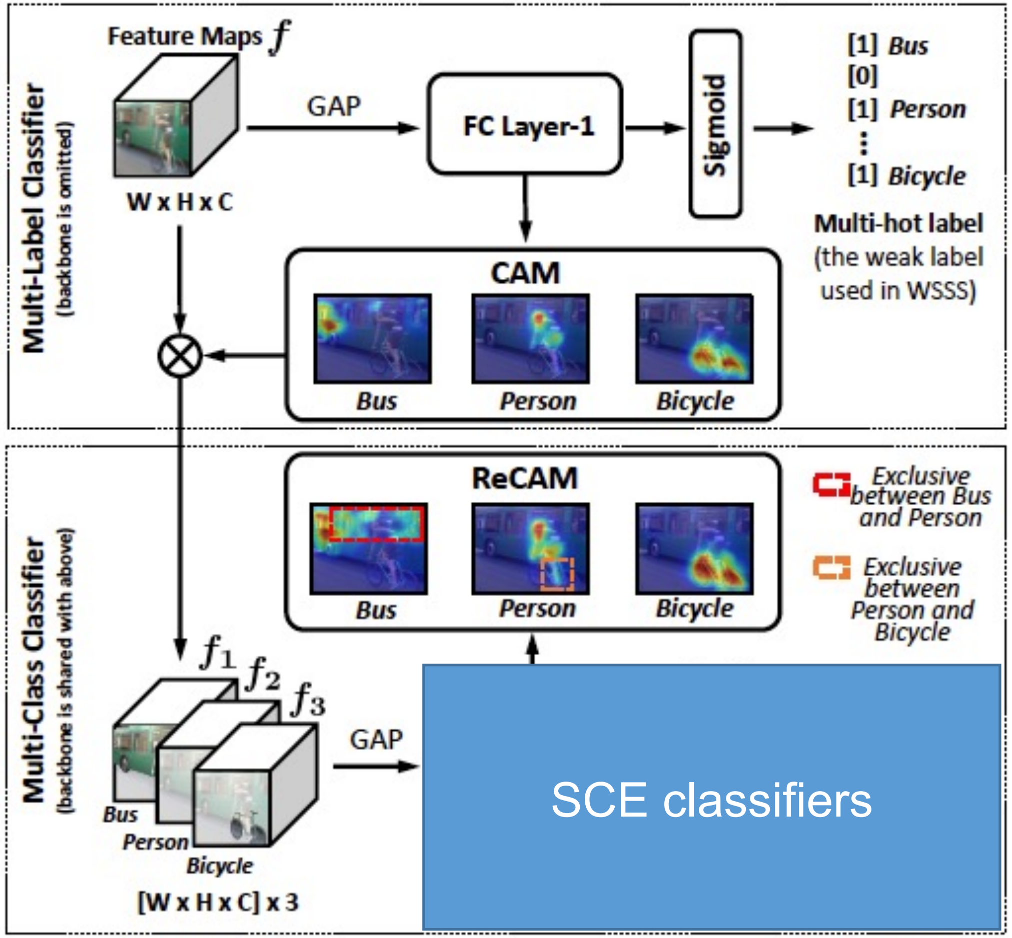
Visualization of Gradient Changes in Training with BCE and SCE

Weakly-Supervised Semantic Segmentation (WSSS)

The solution is introducing SCE in the process of CAM extraction!



implement

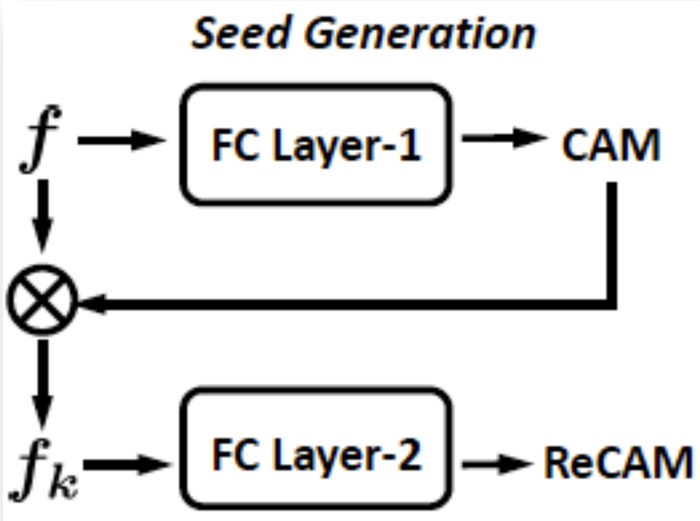


Our method is called ReCAM

https://github.com/zhaozhengChen/ReCAM

Weakly-Supervised Semantic Segmentation (WSSS)

The solution is introducing SCE in the process of CAM extraction!



results

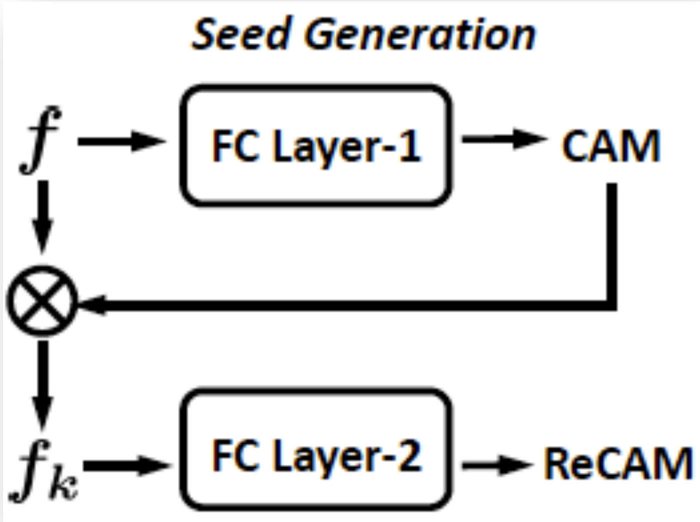
	Methods	CAM mIoU (%)	CAM Time (ut)	ReCAM (ours) mIoU (%)	ReCAM (ours) Time (ut)
VOC	ResNet-50 [51]	48.8	1.0	54.8	1.9
	IRN [1]	66.3	8.2	70.9	9.1
	AdvCAM [23]	55.6	316.3	56.6	317.2
	AdvCAM + IRN	69.9	323.3	70.5	324.2

Our method is called ReCAM

https://github.com/zhaozhengChen/ReCAM

Weakly-Supervised Semantic Segmentation (WSSS)

The solution is introducing SCE in the process of CAM extraction!



results

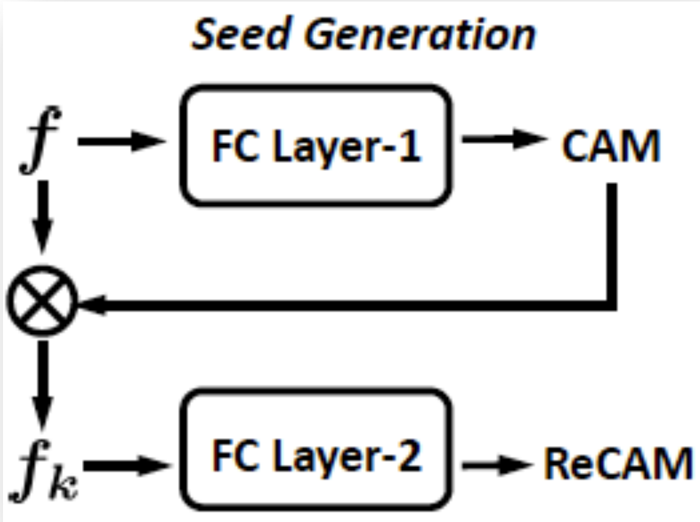
	Methods	CAM mIoU (%)	CAM Time (ut)	ReCAM (ours) mIoU (%)	ReCAM (ours) Time (ut)
VOC	ResNet-50 [51]	48.8	1.0	54.8	1.9
	IRN [1]	66.3	8.2	70.9	9.1
	AdvCAM [23]	55.6	316.3	56.6	317.2
	AdvCAM + IRN	69.9	323.3	70.5	324.2

Our method is called ReCAM

https://github.com/zhaozhengChen/ReCAM

Weakly-Supervised Semantic Segmentation (WSSS)

The solution is introducing SCE in the process of CAM extraction!



results

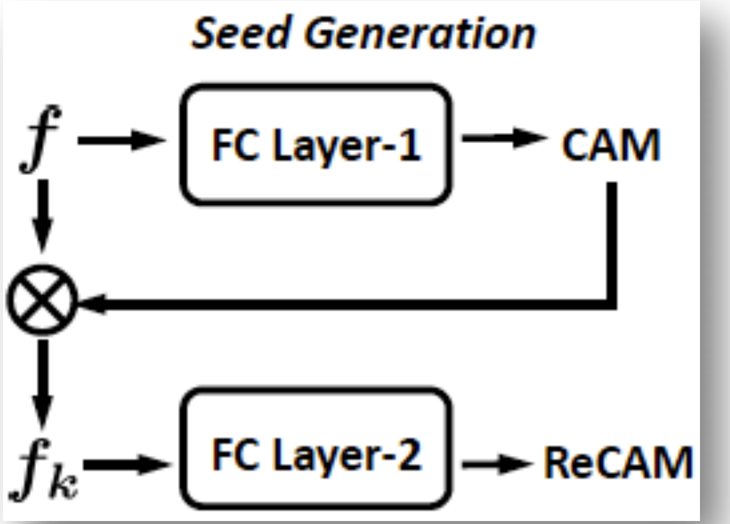
	Methods	CAM		ReCAM (ours)	
		mIoU (%)	Time (ut)	mIoU (%)	Time (ut)
VOC	ResNet-50 [51]	48.8	1.0	54.8	1.9
	IRN [1]	66.3	8.2	70.9	9.1
	AdvCAM [23]	55.6	316.3	56.6	317.2
	AdvCAM + IRN	69.9	323.3	70.5	324.2

Our method is called ReCAM

https://github.com/zhaozhengChen/ReCAM

Weakly-Supervised Semantic Segmentation (WSSS)

The solution is introducing SCE in the process of CAM extraction!



results

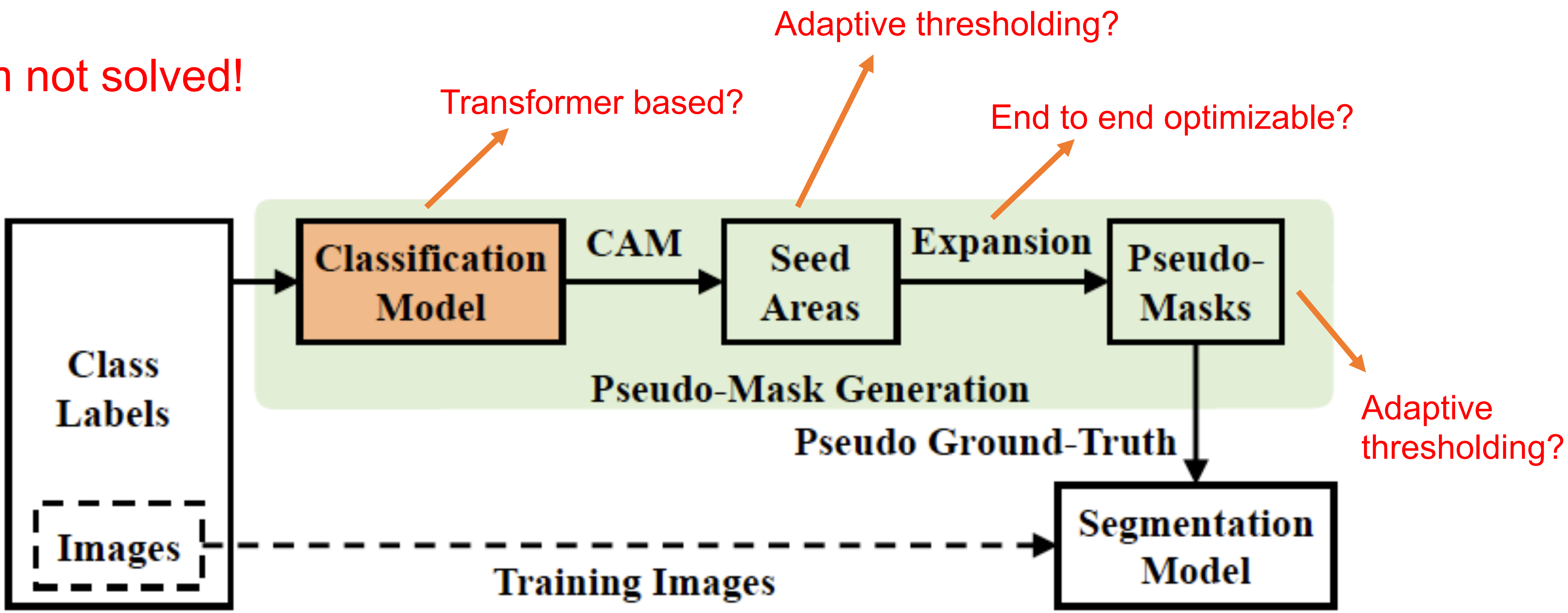
	Methods	CAM		ReCAM (ours)	
		mIoU (%)	Time (ut)	mIoU (%)	Time (ut)
VOC	ResNet-50 [51]	48.8	1.0	54.8	1.9
	IRN [1]	66.3	8.2	70.9	9.1
	AdvCAM [23]	55.6	316.3	56.6	317.2
	AdvCAM + IRN	69.9	323.3	70.5	324.2

Our method is called ReCAM

https://github.com/zhaozhengChen/ReCAM

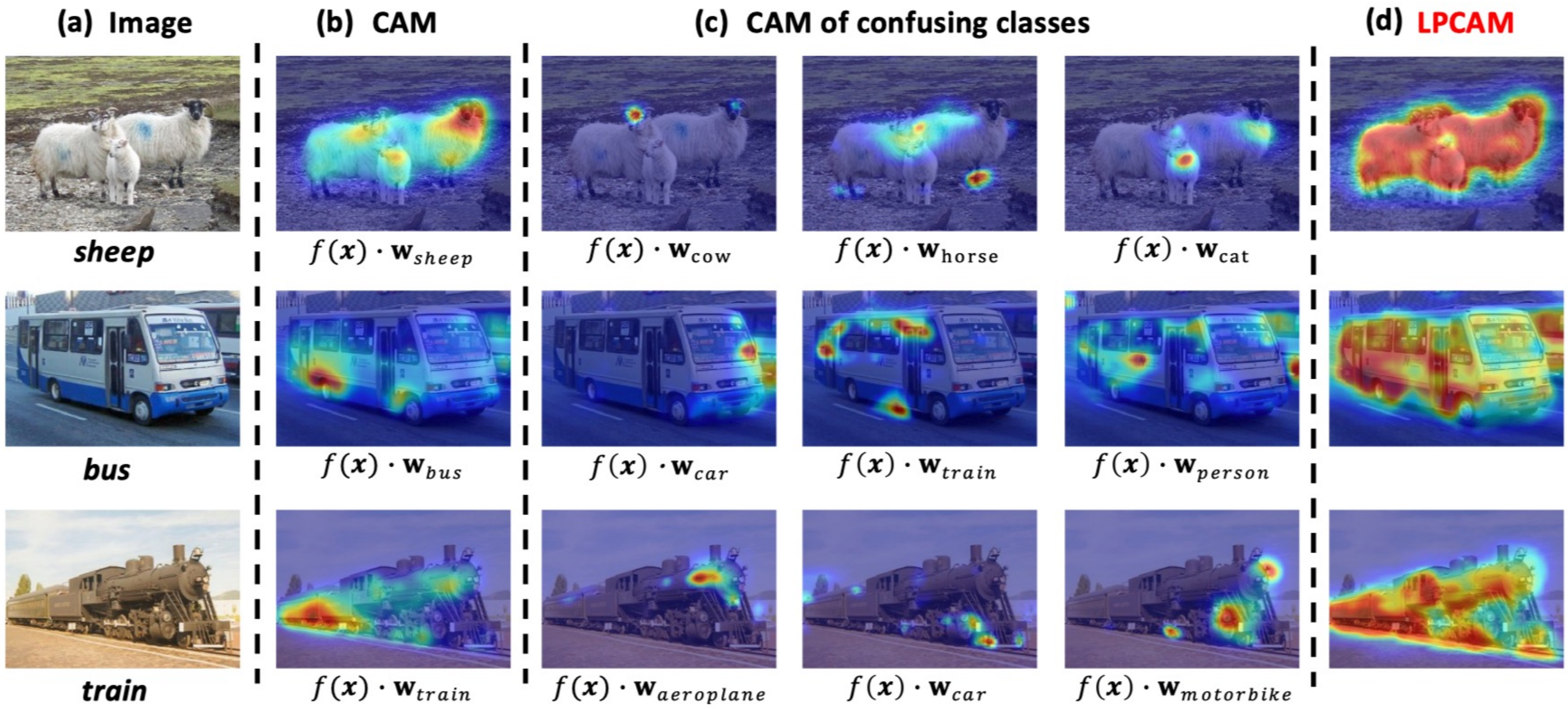
Weakly-Supervised Semantic Segmentation (WSSS)

Problem not solved!



Extracting Class Activation Maps from Non-Discriminative Features as well

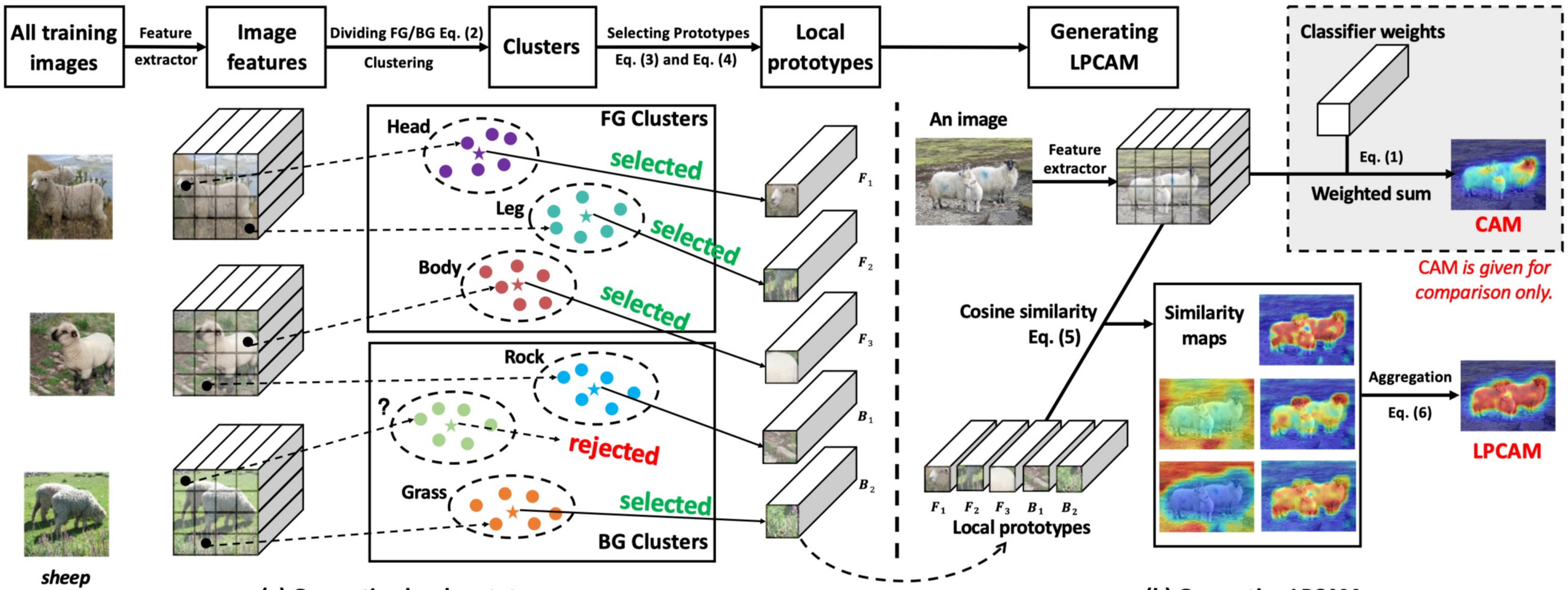
- Motivation: biased classifier



- Question: how to debias?

Extracting Class Activation Maps from Non-Discriminative Features as well

- Solution: use unsupervised clustering to generate non-biased prototypes as classifiers



(a) Generating local prototypes

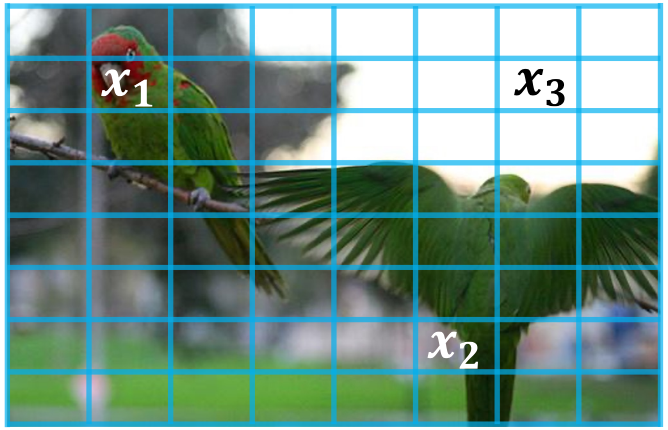
(b) Generating LPCAM

- Question: why this works?

Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

$f(x_3)$=[0, 0, 2]

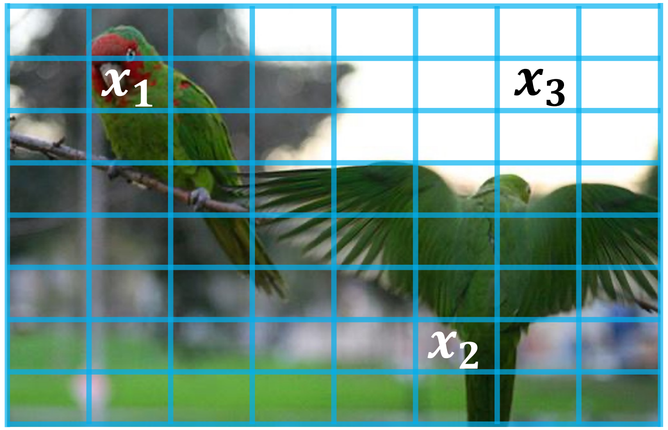


$f(x_1)$=[30, 5, 5]　　$f(x_2)$=[5, 10, 5]

Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

$f(x_3)$=[0, 0, 2]



$f(x_1)$=[30, 5, 5] $f(x_2)$=[5, 10, 5]

Classifier weights
w_{bird} =[4, 1, 1]

CAM Eq. (1)

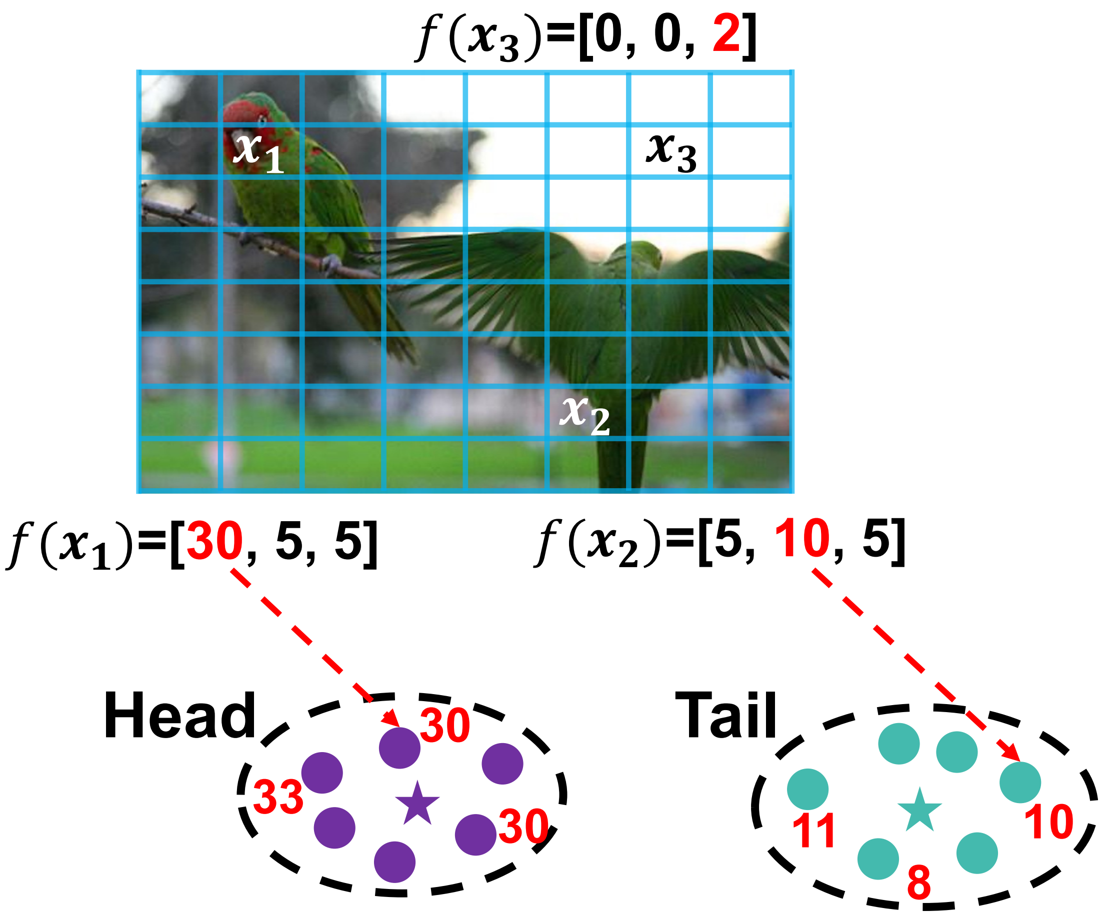
Before Normalization

	x_1	x_2	x_3
CAM	130	35	2

130 is so bigger, which is called “biased”

Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

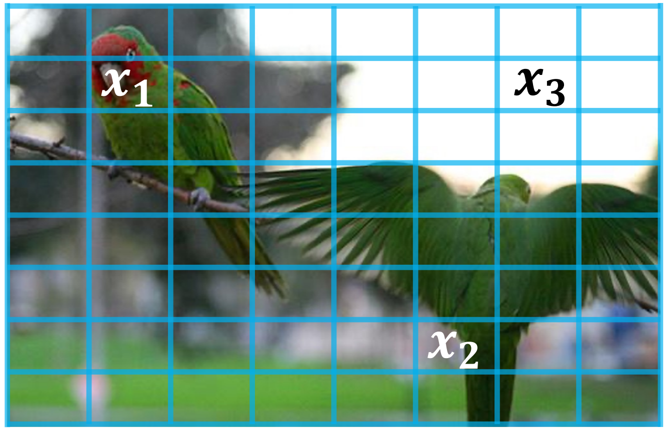


Normalization in each cluster?

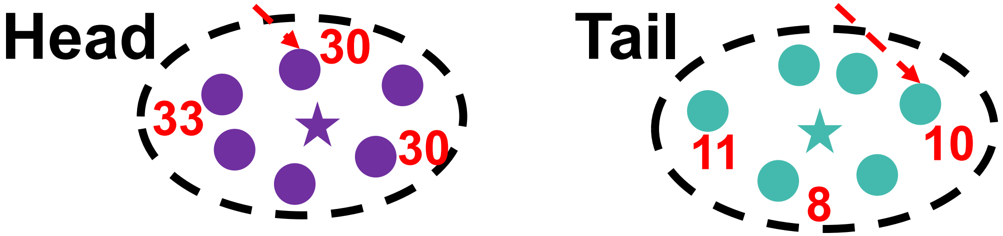
Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

$f(x_3)$=[0, 0, 2]



$f(x_1)$=[30, 5, 5] $f(x_2)$=[5, 10, 5]



Normalization in each cluster?

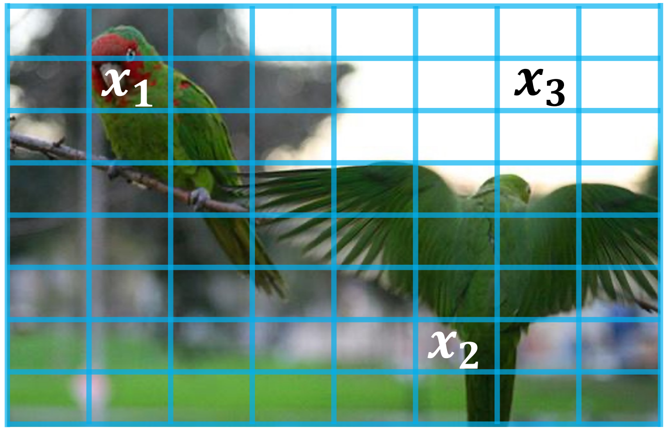
Local prototypes

	x_1	x_2	x_3
head	1.0	0.1	0.0
tail	0.2	0.9	0.1
sky	0.0	0.0	0.9
LPCAM	0.60	0.50	-0.85

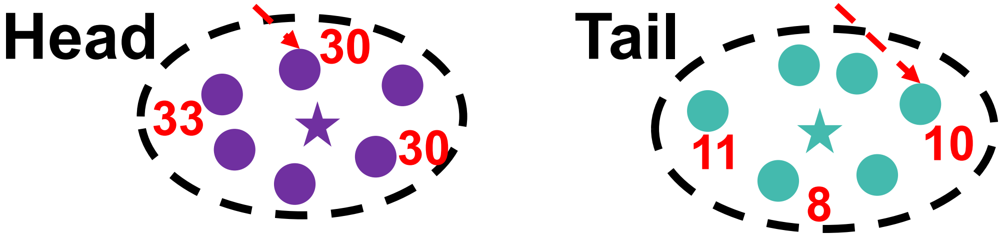
Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

$f(x_3)$=[0, 0, 2]



$f(x_1)$=[30, 5, 5] $f(x_2)$=[5, 10, 5]



Normalization in each cluster?

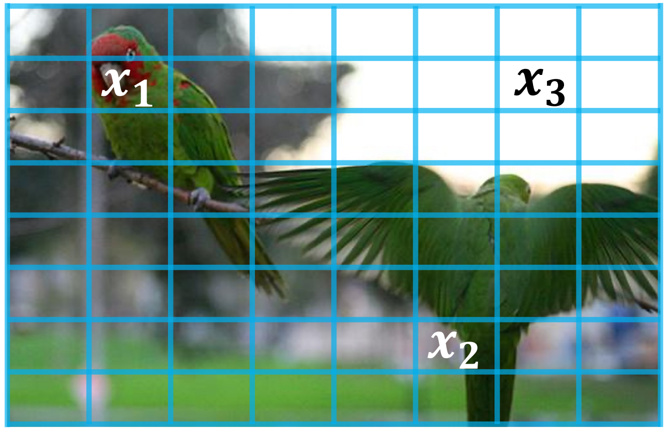
Local prototypes

	x_1	x_2	x_3
head	1.0	0.1	0.0
tail	0.2	0.9	0.1
sky	0.0	0.0	0.9
LPCAM	0.60	0.50	-0.85

Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

$f(x_3)$=[0, 0, 2]

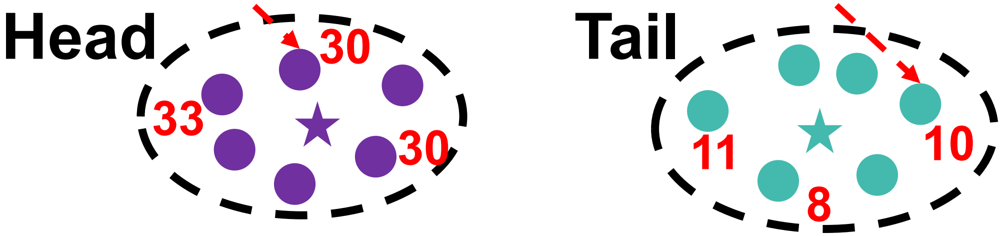


$f(x_1)$=[30, 5, 5]

$f(x_2)$=[5, 10, 5]

Classifier weights
w_{bird} =[4, 1, 1]

CAM Eq. (1)



Before Normalization

	x_1	x_2	x_3
CAM	130	35	2

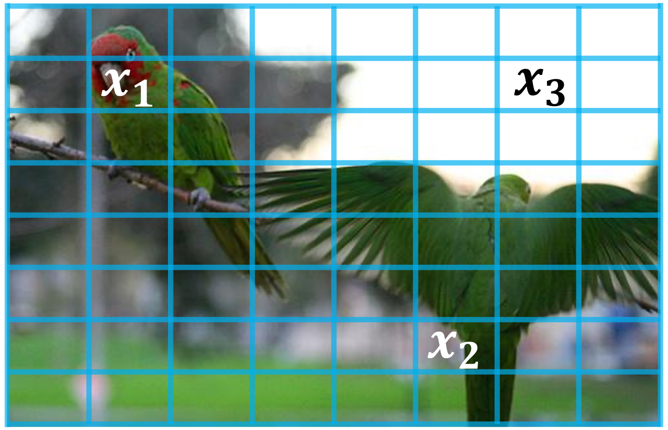
Local prototypes

	x_1	x_2	x_3
head	1.0	0.1	0.0
tail	0.2	0.9	0.1
sky	0.0	0.0	0.9
LPCAM	0.60	0.50	-0.85

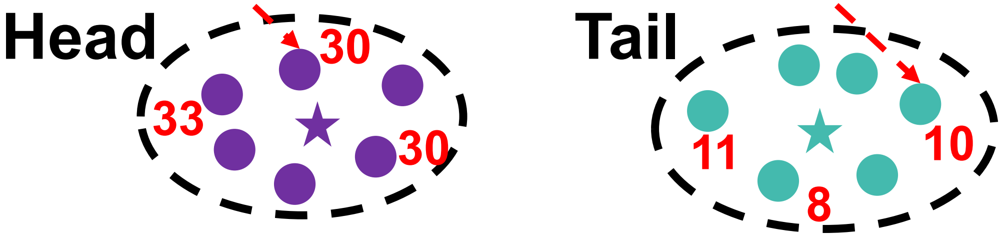
Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)

$f(x_3)$=[0, 0, 2]



$f(x_1)$=[30, 5, 5] $f(x_2)$=[5, 10, 5]



Classifier weights
w_{bird} **=[4, 1, 1]**
CAM Eq. (1)

Before Normalization

	x_1	x_2	x_3
CAM	130	35	2

Local prototypes

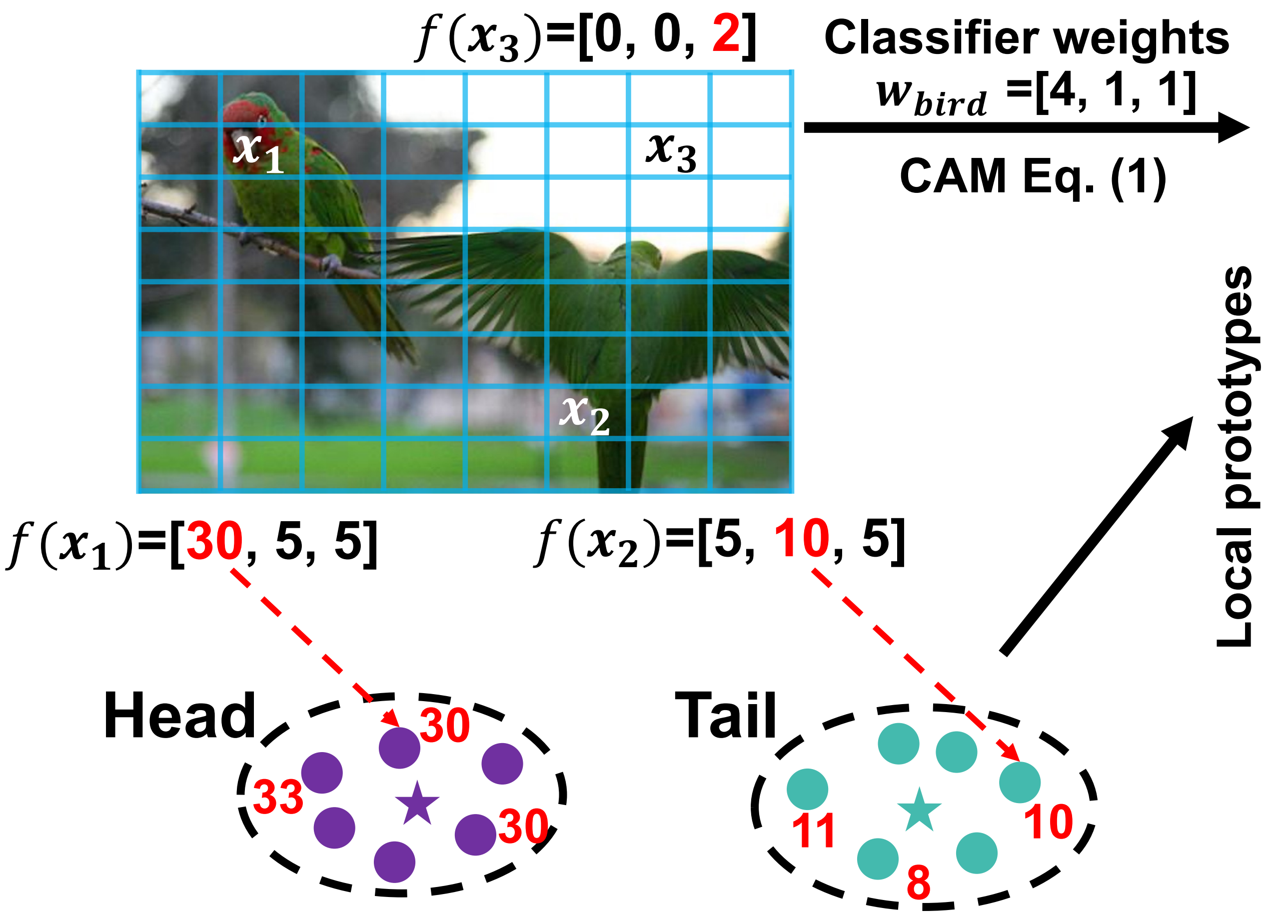
	x_1	x_2	x_3
head	1.0	0.1	0.0
tail	0.2	0.9	0.1
sky	0.0	0.0	0.9
LPCAM	0.60	0.50	-0.85

After Normalization

	x_1	x_2	x_3
CAM	1.00	0.27	0.02
LPCAM	1.00	0.83	0.00

Extracting Class Activation Maps from Non-Discriminative Features as well

- Justification: from supervised biased classifier to unsupervised unbiased classifier (local prototypes)



Before Normalization

	x_1	x_2	x_3
CAM	130	35	2

Local prototypes

	x_1	x_2	x_3
head	1.0	0.1	0.0
tail	0.2	0.9	0.1
sky	0.0	0.0	0.9
LPCAM	0.60	0.50	-0.85

After Normalization

	x_1	x_2	x_3
CAM	1.00	0.27	0.02
LPCAM	1.00	0.83	0.00

biased

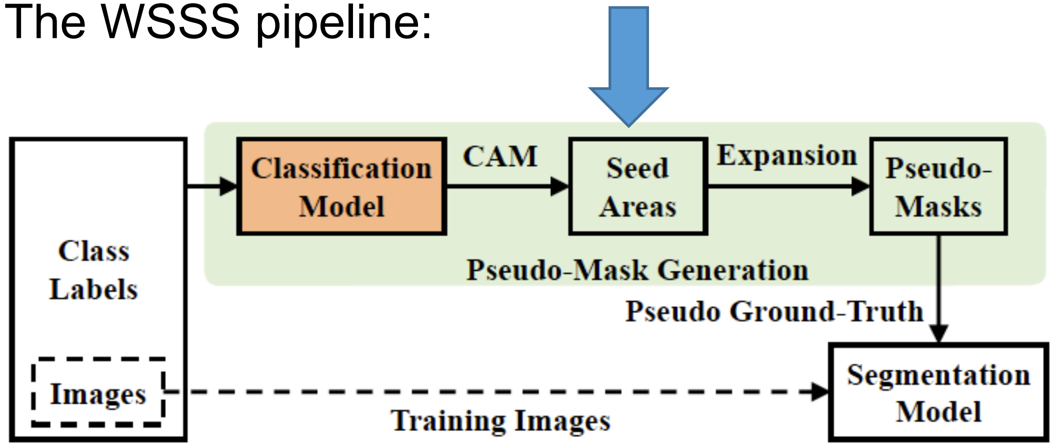
unbiased

Extracting Class Activation Maps from Non-Discriminative Features as well

- Results: LPCAM can be used as improved version of CAM

	Methods	Seed Mask		Pseudo Mask	
		CAM	LPCAM	CAM	LPCAM
VOC	IRN [1]	48.8	54.9+6.1	66.5	71.2+4.7
	EDAM [38]	52.8	54.9+2.1	68.1	69.6+1.5
	MCTformer [44]	61.7	63.5+1.8	69.1	70.8+1.7
	AMN [25]	62.1	65.3+3.2	72.2	72.9+0.7
COCO	IRN [1]	33.1	35.4+2.3	42.5	46.6+4.1
	AMN [25]	40.3	42.5+2.2	46.7	47.7+1.0

The WSSS pipeline:

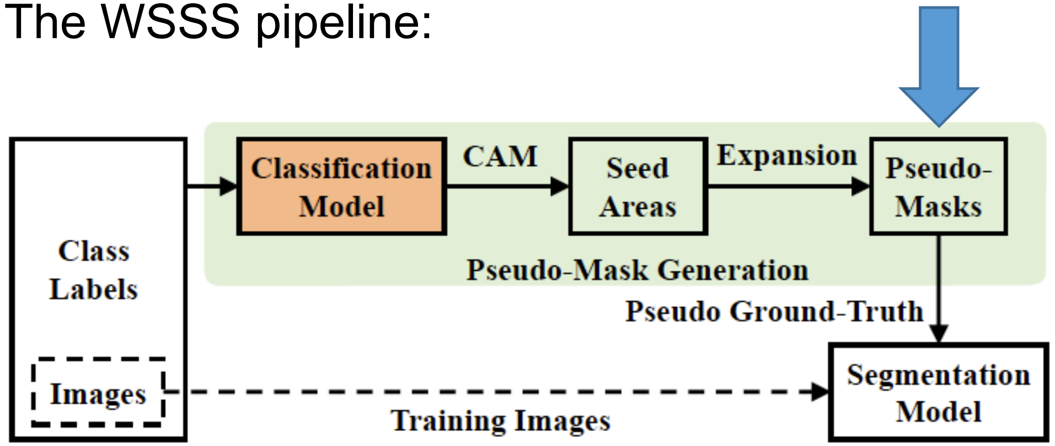


Extracting Class Activation Maps from Non-Discriminative Features as well

- Results: LPCAM can be used as improved version of CAM

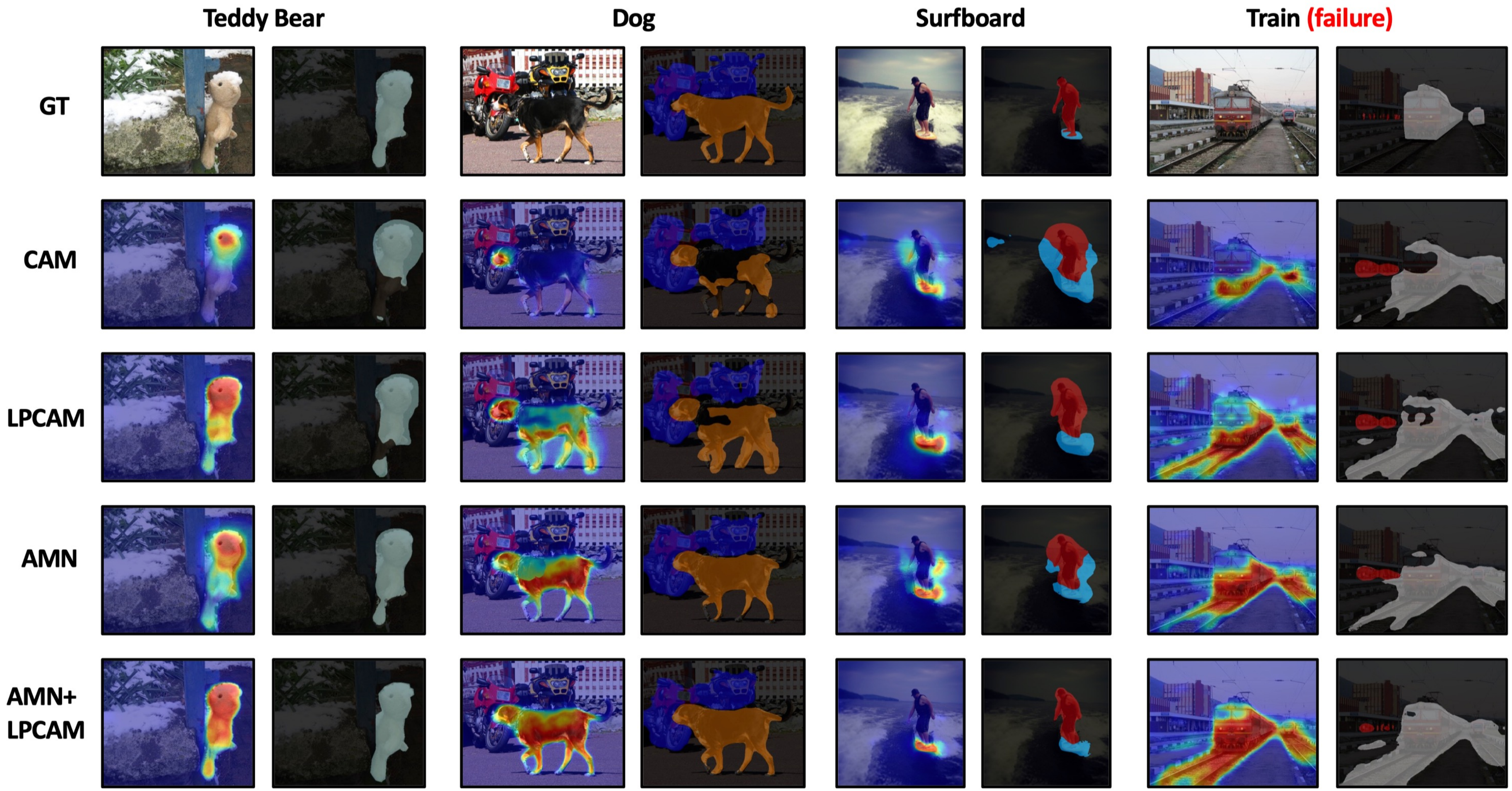
	Methods	Seed Mask		Pseudo Mask	
		CAM	LPCAM	CAM	LPCAM
VOC	IRN [1]	48.8	54.9+6.1	66.5	71.2+4.7
	EDAM [38]	52.8	54.9+2.1	68.1	69.6+1.5
	MCTformer [44]	61.7	63.5+1.8	69.1	70.8+1.7
	AMN [25]	62.1	65.3+3.2	72.2	72.9+0.7
COCO	IRN [1]	33.1	35.4+2.3	42.5	46.6+4.1
	AMN [25]	40.3	42.5+2.2	46.7	47.7+1.0

The WSSS pipeline:



Extracting Class Activation Maps from Non-Discriminative Features as well

- Motivation: biased classifier



Extracting Class Activation Maps from Non-Discriminative Features as well

- Large models released, e.g., SAM (Segment Anything Model)

• represents a rough location of any object or any stuff

Extracting Class Activation Maps from Non-Discriminative Features as well

- Large models released, e.g., SAM (Segment Anything Model)

A simple testing by using an object detector:

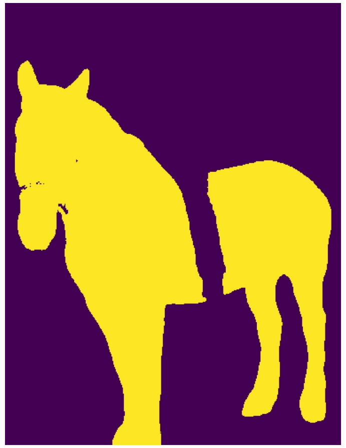
Original image

"Person" bbox

SAM

"Horse" bbox

SAM



VOC: 86.45%!!!

Extracting Class Activation Maps from Non-Discriminative Features as well

- Large models released, e.g., SAM (Segment Anything Model)

A simple testing by using an object detector:

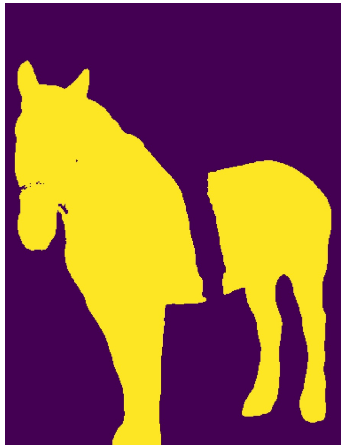
Original image

“Person” bbox

SAM

“Horse” bbox

SAM



VOC: 86.45% !!

	Methods	Seed Mask	
		CAM	LPCAM
VOC	IRN [1]	48.8	54.9 +6.1
	EDAM [38]	52.8	54.9 +2.1
	MCTformer [44]	61.7	63.5 +1.8
	AMN [25]	62.1	65.3 +3.2
COCO	IRN [1]	33.1	35.4 +2.3
	AMN [25]	40.3	42.5 +2.2

Extracting Class Activation Maps from Non-Discriminative Features as well

- Large models released, e.g., SAM (Segment Anything Model)

A simple testing by using an object detector:

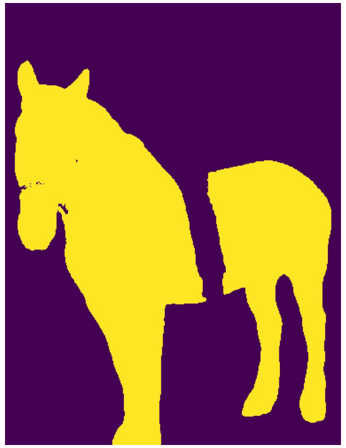
Original image

"Person" bbox

SAM

"Horse" bbox

SAM



Actually,

VOC: 100%!!!

	Methods	Seed Mask	
		CAM	LPCAM
VOC	IRN [1]	48.8	54.9+6.1
	EDAM [38]	52.8	54.9+2.1
	MCTformer [44]	61.7	63.5+1.8
	AMN [25]	62.1	65.3+3.2
COCO	IRN [1]	33.1	35.4+2.3
	AMN [25]	40.3	42.5+2.2

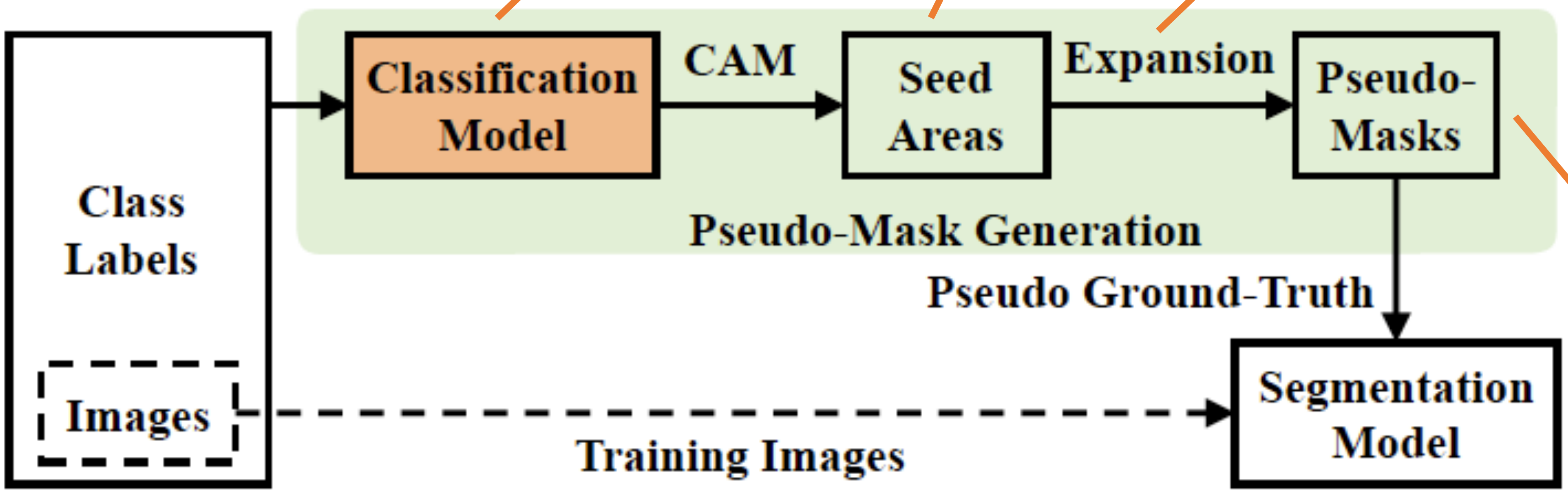
Weakly-Supervised Semantic Segmentation (WSSS)

Problem not solved!

Adaptive thresholding?

Transformer based?

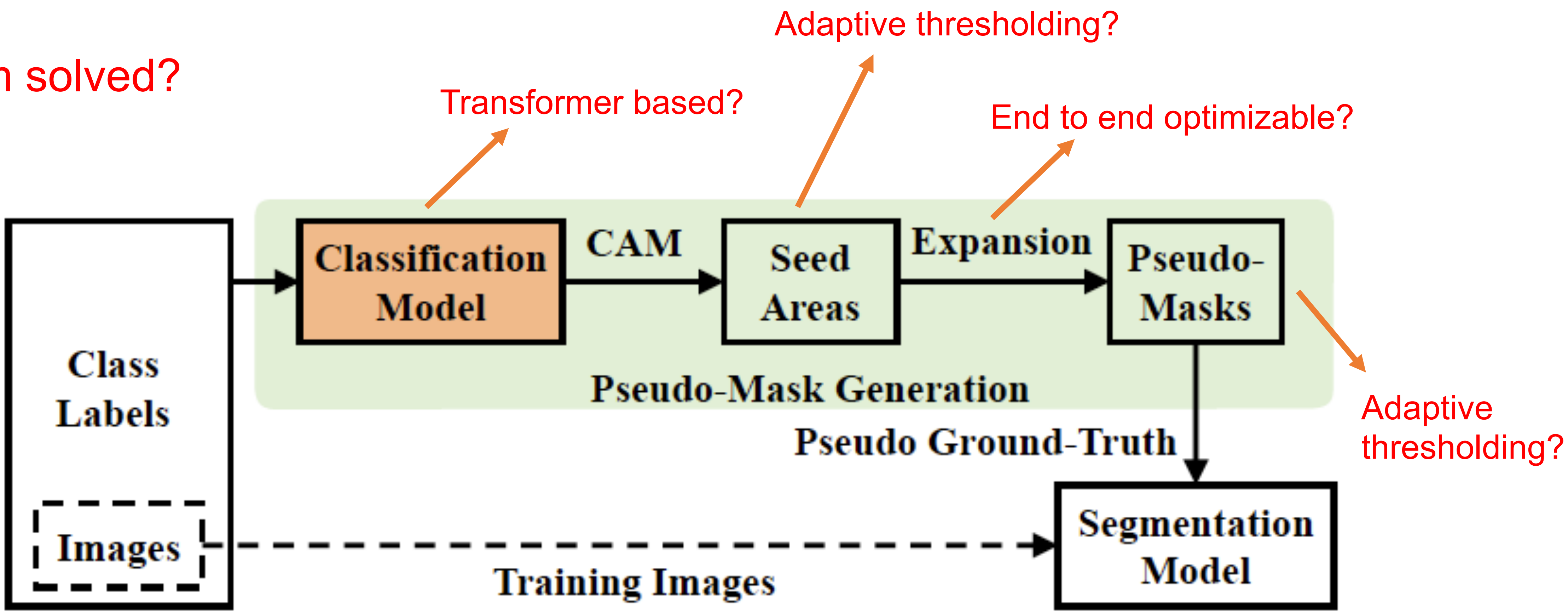
End to end optimizable?



Adaptive thresholding?

Weakly-Supervised Semantic Segmentation (WSSS)

Problem solved?



Weakly-Supervised Semantic Segmentation (WSSS)

Problem solved!

Welcome to my page:

https://qianrusun.com/