Learning from noisy labels & Reinforcement Learning

DivideMix: Learning with Noisy Labels as Semi-supervised Learning

Junnan Li, Richard Socher, Steven C.H. Hoi
Salesforce Research
{junnan.li,rsocher,shoi}@salesforce.com

ICLR 2020

Contents

Introduction

- Existing studies can be roughly divided into two main solutions.
  - Filter Methods (detect potential noisy labels)
    - O2U, Co-Teaching, Co-Teaching+, Decoupling, Abstention
  - Directly Learning
    - Robust loss designing
      - Symmetric Loss (ICCV'19), Normalized Loss (ICML'20)
    - Robust Model
      - MLNT (CVPR'19)
    - Two steps end-to-end approaches
      - 1. detection noise,
      - 2. semi-supervised manner

Methods

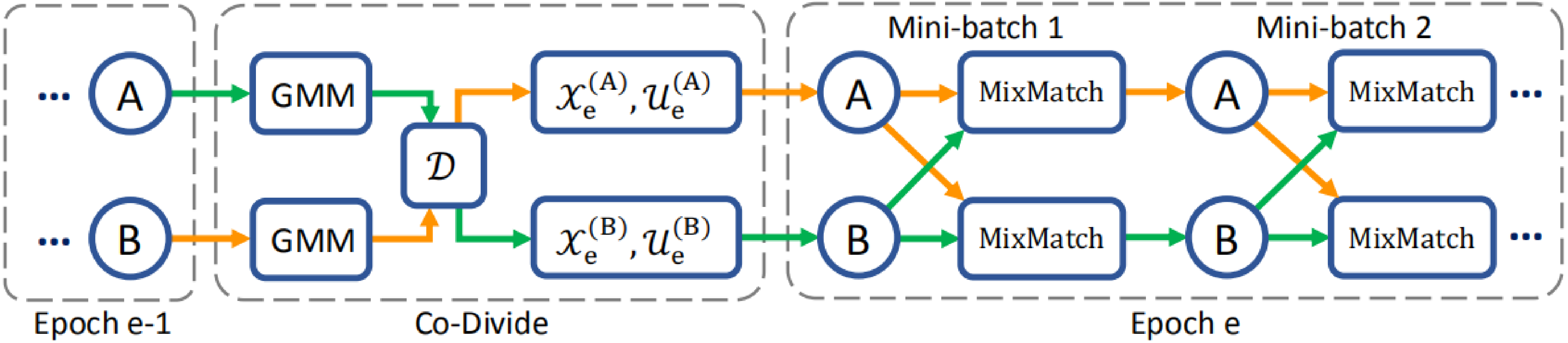


Figure 1: DivideMix trains two networks (A and B) simultaneously. At each epoch, a network models its per-sample loss distribution with a GMM to divide the dataset into a labeled set (mostly clean) and an unlabeled set (mostly noisy), which is then used as training data for the other network (*i.e.* co-divide). At each mini-batch, a network performs semi-supervised training using an improved MixMatch method. We perform label co-refinement on the labeled samples and label co-guessing on the unlabeled samples.

Methods

Algorithm 1: DivideMix. Line 4-8: co-divide; Line 17-18: label co-refinement; Line 20: label co-guessing.

1 **Input:** $\theta^{(1)}$ and $\theta^{(2)}$, training dataset $(\mathcal{X}, \mathcal{Y})$, clean probability threshold τ, number of augmentations M, sharpening temperature T, unsupervised loss weight λ_u, Beta distribution parameter α for MixMatch.

2 $\theta^{(1)}, \theta^{(2)} = \text{WarmUp}(\mathcal{X}, \mathcal{Y}, \theta^{(1)}, \theta^{(2)})$ *// standard training (with confidence penalty)*

3 **while** $e < \text{MaxEpoch}$ **do**

4     $\mathcal{W}^{(2)} = \text{GMM}(\mathcal{X}, \mathcal{Y}, \theta^{(1)})$ *// model per-sample loss with $\theta^{(1)}$ to obtain clean proabability for $\theta^{(2)}$*

5     $\mathcal{W}^{(1)} = \text{GMM}(\mathcal{X}, \mathcal{Y}, \theta^{(2)})$ *// model per-sample loss with $\theta^{(2)}$ to obtain clean proabability for $\theta^{(1)}$*

6   **for** $k = 1, 2$ **do** *// train the two networks one by one*

7         $\mathcal{X}_e^{(k)} = \{(x_i, y_i, w_i) | w_i \geq \tau, \forall (x_i, y_i, w_i) \in (\mathcal{X}, \mathcal{Y}, \mathcal{W}^{(k)})\}$ *// labeled training set for $\theta^{(k)}$*

8         $\mathcal{U}_e^{(k)} = \{x_i | w_i < \tau, \forall (x_i, w_i) \in (\mathcal{X}, \mathcal{W}^{(k)})\}$ *// unlabeled training set for $\theta^{(k)}$*

9   **for** iter $= 1$ **to** num_iters **do**

10   From $\mathcal{X}_e^{(k)}$, draw a mini-batch $\{(x_b, y_b, w_b); b \in (1, ..., B)\}$

11   From $\mathcal{U}_e^{(k)}$, draw a mini-batch $\{u_b; b \in (1, ..., B)\}$

12   **for** $b = 1$ **to** B **do**

13   **for** $m = 1$ **to** M **do**

14                     $\hat{x}_{b,m} = \text{Augment}(x_b)$ *// apply m^{th} round of augmentation to x_b*

15                     $\hat{u}_{b,m} = \text{Augment}(u_b)$ *// apply m^{th} round of augmentation to u_b*

16   **end**

Methods

$p_b = \frac{1}{M}\sum_m \mathrm{p}_{\mathrm{model}}(\hat{x}_{b,m};\theta^{(k)})$ *// average the predictions across augmentations of x_b*

$\bar{y}_b = w_b y_b + (1-w_b)p_b$

// refine ground-truth label guided by the clean probability produced by the other network

$\hat{y}_b = \mathrm{Sharpen}(\bar{y}_b, T)$ *// apply temperature sharpening to the refined label*

$\bar{q}_b = \frac{1}{2M}\sum_m \left(\mathrm{p}_{\mathrm{model}}(\hat{u}_{b,m};\theta^{(1)}) + \mathrm{p}_{\mathrm{model}}(\hat{u}_{b,m};\theta^{(2)})\right)$

// co-guessing: average the predictions from both networks across augmentations of u_b

$q_b = \mathrm{Sharpen}(\bar{q}_b, T)$ *// apply temperature sharpening to the guessed label*

end

$\hat{\mathcal{X}} = \{(\hat{x}_{b,m}, \hat{y}_b); b \in (1,\ldots,B), m \in (1,\ldots,M)\}$ *// augmented labeled mini-batch*

$\hat{\mathcal{U}} = \{(\hat{u}_{b,m}, q_b); b \in (1,\ldots,B), m \in (1,\ldots,M)\}$ *// augmented unlabeled mini-batch*

$\mathcal{L}_{\mathcal{X}}, \mathcal{L}_{\mathcal{U}} = \mathrm{MixMatch}(\hat{\mathcal{X}}, \hat{\mathcal{U}})$ *// apply MixMatch*

$\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_u \mathcal{L}_{\mathcal{U}} + \lambda_r \mathcal{L}_{\mathrm{reg}}$ *// total loss*

$\theta^{(k)} = \mathrm{SGD}(\mathcal{L}, \theta^{(k)})$ *// update model parameters*

end

end

end

Methods

☐ MixMatch with Label Co-Refinement and Co-Guessing

- Refine Label

$$\bar{y}_b = w_b y_b + (1 - w_b) p_b.$$

$$\hat{y}_b = \text{Sharpen}(\bar{y}_b, T) = {\bar{y}_b^c}^{\frac{1}{T}} \Bigg/ \sum_{c=1}^{C} {\bar{y}_b^c}^{\frac{1}{T}}, \text{ for } c = 1, 2, ..., C.$$

- MixMatch

$$\lambda \sim \text{Beta}(\alpha, \alpha),$$
$$\lambda' = \max(\lambda, 1 - \lambda),$$
$$x' = \lambda' x_1 + (1 - \lambda') x_2,$$
$$p' = \lambda' p_1 + (1 - \lambda') p_2.$$

MixMatch transforms $\hat{\mathcal{X}}$ and $\hat{\mathcal{U}}$ into $\mathcal{X}'$ and $\mathcal{U}'$

Method

The total loss

$$\mathcal{L}_{\mathcal{X}} = -\frac{1}{|\mathcal{X}'|} \sum_{x,p \in \mathcal{X}'} \sum_{c} p_c \log(\mathrm{p}^{c}_{\mathrm{model}}(x;\theta)),$$

$$\mathcal{L}_{\mathcal{U}} = \frac{1}{|\mathcal{U}'|} \sum_{x,p \in \mathcal{U}'} \|p - \mathrm{p}_{\mathrm{model}}(x;\theta)\|_2^2 .$$

$$\mathcal{L}_{\mathrm{reg}} = \sum_{c} \pi_c \log\left(\pi_c \middle/ \frac{1}{|\mathcal{X}'| + |\mathcal{U}'|} \sum_{x \in \mathcal{X}' + \mathcal{U}'} \mathrm{p}^{c}_{\mathrm{model}}(x;\theta)\right).$$

$$\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_u \mathcal{L}_{\mathcal{U}} + \lambda_r \mathcal{L}_{\mathrm{reg}}.$$

In our experiments, we set λ_r as 1 and use λ_u to control the strength of the unsupervised loss.

Experiments

Method	Best	Last
Cross-Entropy	85.0	72.3
F-correction (Patrini et al., 2017)	87.2	83.1
M-correction (Arazo et al., 2019)	87.4	86.3
Iterative-CV (Chen et al., 2019)	88.6	88.0
P-correction (Yi & Wu, 2019)	88.5	88.1
Joint-Optim (Tanaka et al., 2018)	88.9	88.4
Meta-Learning (Li et al., 2019)	89.2	88.6
DivideMix	**93.4**	**92.1**

Aysm-CIFAR10

Dataset		CIFAR-10				CIFAR-100			
Method/Noise ratio		20%	50%	80%	90%	20%	50%	80%	90%
Cross-Entropy	Best	86.8	79.4	62.9	42.7	62.0	46.7	19.9	10.1
	Last	82.7	57.9	26.1	16.8	61.8	37.3	8.8	3.5
Bootstrap (Reed et al., 2015)	Best	86.8	79.8	63.3	42.9	62.1	46.6	19.9	10.2
	Last	82.9	58.4	26.8	17.0	62.0	37.9	8.9	3.8
F-correction (Patrini et al., 2017)	Best	86.8	79.8	63.3	42.9	61.5	46.6	19.9	10.2
	Last	83.1	59.4	26.2	18.8	61.4	37.3	9.0	3.4
Co-teaching+* (Yu et al., 2019)	Best	89.5	85.7	67.4	47.9	65.6	51.8	27.9	13.7
	Last	88.2	84.1	45.5	30.1	64.1	45.3	15.5	8.8
Mixup (Zhang et al., 2018)	Best	95.6	87.1	71.6	52.2	67.8	57.3	30.8	14.6
	Last	92.3	77.6	46.7	43.9	66.0	46.6	17.6	8.1
P-correction* (Yi & Wu, 2019)	Best	92.4	89.1	77.5	58.9	69.4	57.5	31.1	15.3
	Last	92.0	88.7	76.5	58.2	68.1	56.4	20.7	8.8
Meta-Learning* (Li et al., 2019)	Best	92.9	89.3	77.4	58.7	68.5	59.2	42.4	19.5
	Last	92.0	88.8	76.1	58.3	67.7	58.0	40.1	14.3
M-correction (Arazo et al., 2019)	Best	94.0	92.0	86.8	69.1	73.9	66.1	48.2	24.3
	Last	93.8	91.9	86.6	68.7	73.4	65.4	47.6	20.5
DivideMix	Best	**96.1**	**94.6**	**93.2**	**76.0**	**77.3**	**74.6**	**60.2**	**31.5**
	Last	**95.7**	**94.4**	**92.9**	**75.4**	**76.9**	**74.2**	**59.6**	**31.0**

Table 1: Comparison with state-of-the-art methods in test accuracy (%) on CIFAR-10 and CIFAR-100 with symmetric noise. Methods marked by * denote re-implementations based on public code.

Sym

Method	Test Accuracy
Cross-Entropy	69.21
F-correction (Patrini et al., 2017)	69.84
M-correction (Arazo et al., 2019)	71.00
Joint-Optim (Tanaka et al., 2018)	72.16
Meta-Cleaner (Zhang et al., 2019)	72.50
Meta-Learning (Li et al., 2019)	73.47
P-correction (Yi & Wu, 2019)	73.49
DivideMix	**74.76**

Table 3: Comparison with state-of-the-art methods in test accuracy (%) on Clothing1M. Results for baselines are copied from original papers.

Method	WebVision		ILSVRC12	
	top1	top5	top1	top5
F-correction (Patrini et al., 2017)	61.12	82.68	57.36	82.36
Decoupling (Malach & Shalev-Shwartz, 2017)	62.54	84.74	58.26	82.26
D2L (Ma et al., 2018)	62.68	84.00	57.80	81.36
MentorNet (Jiang et al., 2018)	63.00	81.40	57.80	79.92
Co-teaching (Han et al., 2018)	63.58	85.20	61.48	84.70
Iterative-CV (Chen et al., 2019)	65.24	85.34	61.60	84.98
DivideMix	**77.32**	**91.64**	**75.20**	**90.84**

Table 4: Comparison with state-of-the-art methods trained on (mini) WebVision dataset. Numbers denote top-1 (top-5) accuracy (%) on the WebVision validation set and the ImageNet ILSVRC12 validation set. Results for baseline methods are copied from Chen et al. (2019).

4.3 Ablation Study

We study the effect of removing different components to provide insights into what makes DivideMix successful. We analyze the results in Table 5 as follows. Appendix C contains additional explanations.

Dataset		CIFAR-10					CIFAR-100			
Noise type		Sym.				Asym.	Sym.			
Methods/Noise ratio		20%	50%	80%	90%	40%	20%	50%	80%	90%
DivideMix	Best	**96.1**	**94.6**	**93.2**	**76.0**	**93.4**	**77.3**	**74.6**	**60.2**	**31.5**
	Last	**95.7**	**94.4**	**92.9**	**75.4**	**92.1**	**76.9**	**74.2**	**59.6**	**31.0**
DivideMix with $\theta^{(1)}$ test	Best	95.2	94.2	93.0	75.5	92.7	75.2	72.8	58.3	29.9
	Last	95.0	93.7	92.4	74.2	91.4	74.8	72.1	57.6	29.2
DivideMix w/o co-training	Best	95.0	94.0	92.6	74.3	91.9	74.8	72.3	56.7	27.7
	Last	94.8	93.3	92.2	73.2	90.6	74.1	71.7	56.3	27.2
DivideMix w/o label refinement	Best	96.0	94.6	93.0	73.7	87.7	76.9	74.2	58.7	26.9
	Last	95.5	94.2	92.7	73.0	86.3	76.4	73.9	58.2	26.3
DivideMix w/o augmentation	Best	95.3	94.1	92.2	73.9	89.5	76.5	73.1	58.2	26.9
	Last	94.9	93.5	91.8	73.0	88.4	76.2	72.6	58.0	26.4
Divide and MixMatch	Best	94.1	92.8	89.7	70.1	86.5	73.7	70.5	55.3	25.0
	Last	93.5	92.3	89.1	68.6	85.2	72.4	69.7	53.9	23.7

Table 5: Ablation study results in terms of test accuracy (%) on CIFAR-10 and CIFAR-100.

EvidentialMix: Learning with Combined Open-set and Closed-set Noisy Labels

Ragav Sachdeva[1], Filipe R. Cordeiro[2], Vasileios Belagiannis[3], Ian Reid[1], and Gustavo Carneiro[1]

[1]University of Adelaide, Adelaide, Australia
[2]Universidade Federal Rural de Pernambuco, Recife, Brazil
[3]Universität Ulm, Ulm, Germany

WACV 2021

Contents

- Open-set noisy labels setting
- Method
- Experiments

Open-set noisy labels [1]

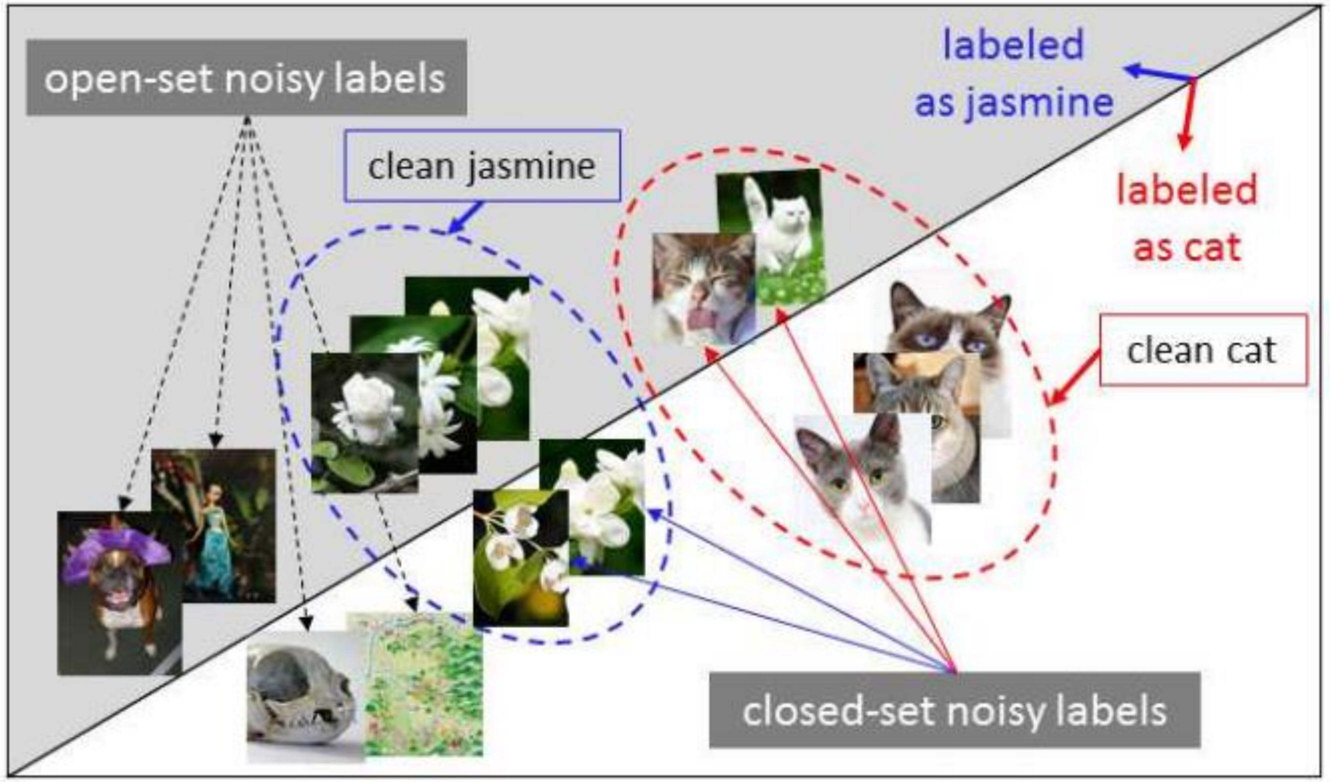
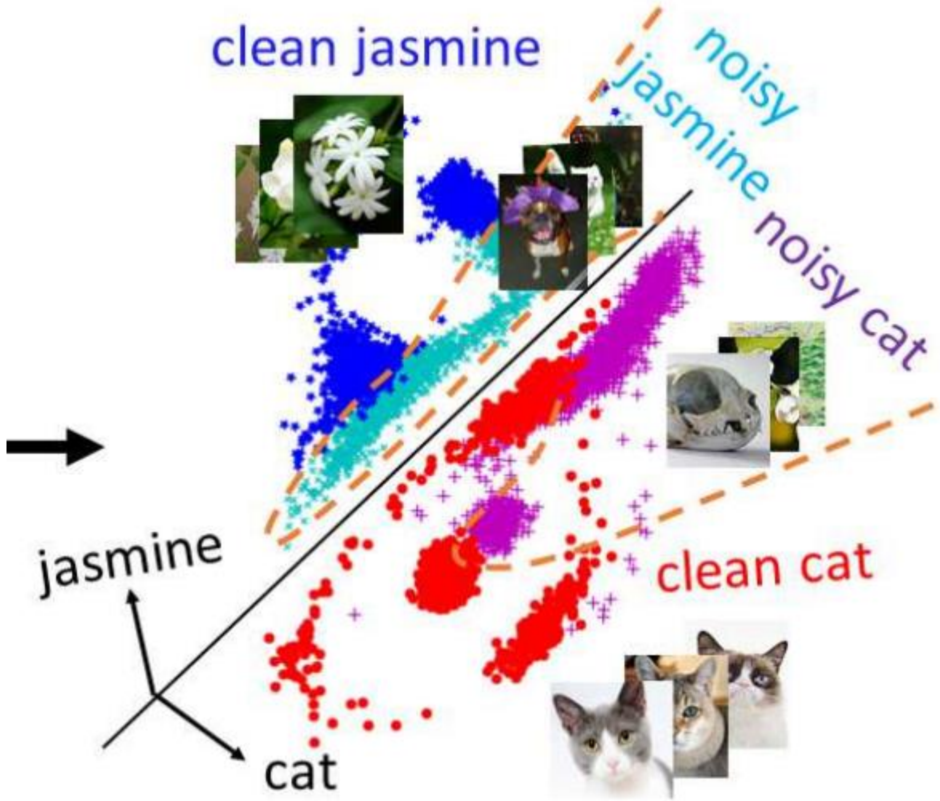


Figure 1. An illustration of closed-set vs open-set noisy labels.



Open-set noisy labels

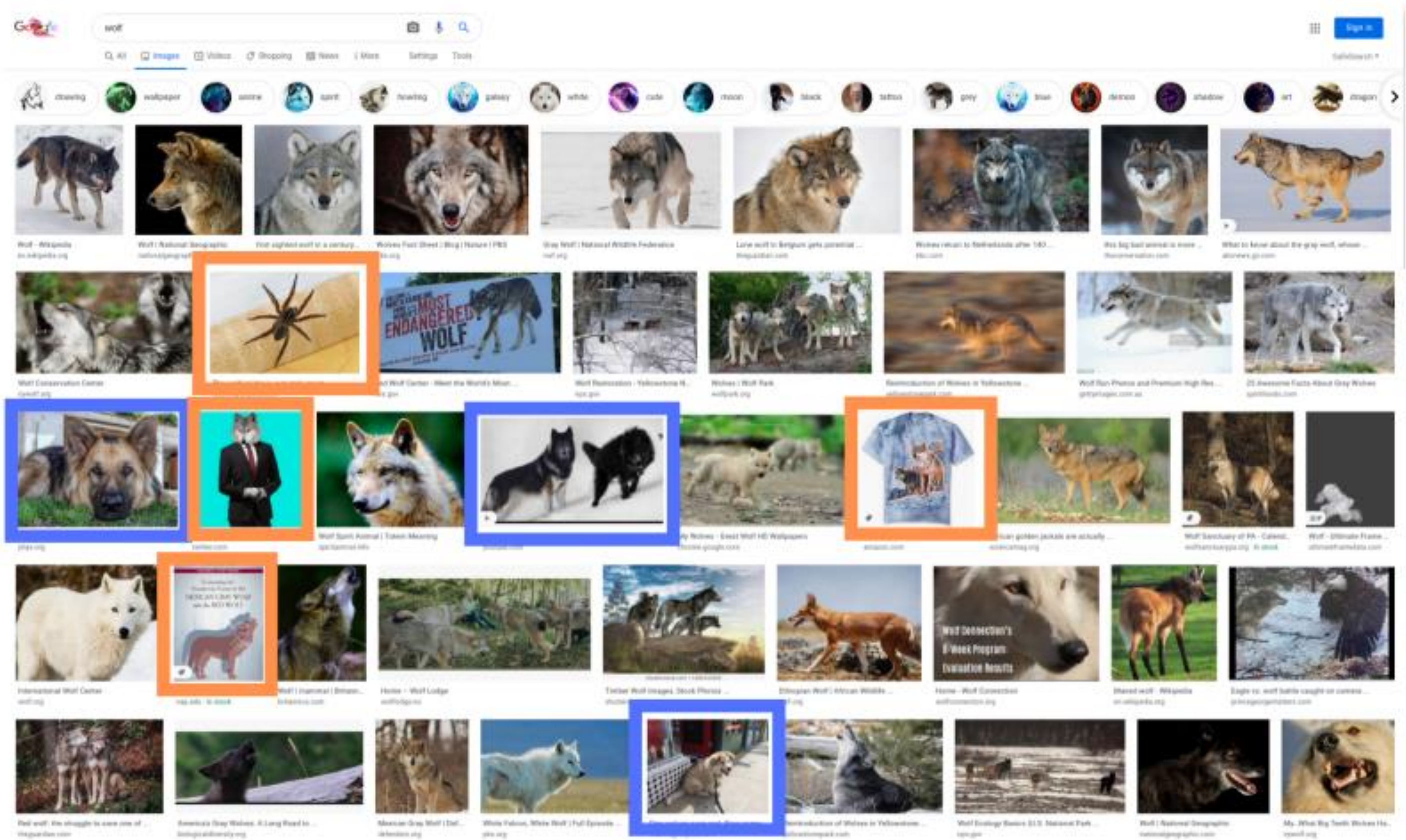
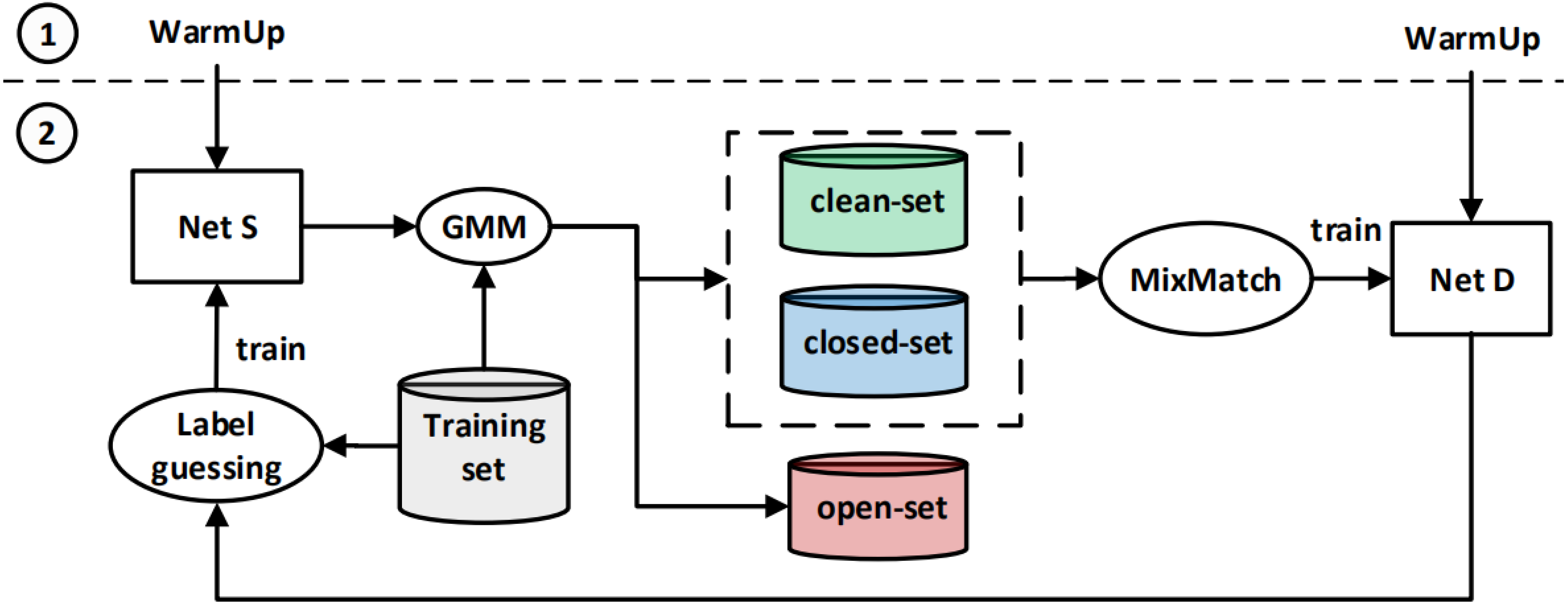


Figure 1: Results of a search engine query to collect data for a wolf-vs-dog binary classifier. The search keyword used here is “wolf”. The images bounded by an orange box are open-set noise (i.e. neither wolf nor dog) and the ones bounded by a blue box are closed-set noise (i.e. labelled as wolf but are actually a dog).

Method


1
WarmUp
WarmUp
2
Net S
GMM
clean-set
closed-set
open-set
MixMatch
train
Net D
train
Label guessing
Training set

Algorithm 1: EvidentialMix (EDM)

Input: $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{|\mathcal{D}|}$, number of augmentations M, temperature sharpening T, loss weights $\lambda^{(\mathcal{U})}$ and $\lambda^{(reg)}$, MixMatch parameter α, number of epochs E.

$f_{\theta^{(D)}}(c|\mathbf{x}), f_{\theta^{(S)}}(c|\mathbf{x})$ = WarmUp($\mathcal{D}$)
while $e < E$ **do**
  $\mathcal{W}, \mathcal{W}^{\text{op}}, \mathcal{W}^{\text{cl}}$ = GMM($\mathcal{D}$, $f_{\theta^{(S)}}(c|\mathbf{x})$)
  `// Train NetD`
  $\mathcal{X} = \{(\mathbf{x}_i, \mathbf{y}_i, w_i) | (\mathbf{x}_i, \mathbf{y}_i, w_i) \in (\mathcal{D}, \mathcal{W}), w_i > \max(w_i^{\text{op}}, w_i^{\text{cl}})\}$
  $\mathcal{U} = \{\mathbf{x}_i | (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}, w_i^{\text{cl}} > \max(w_i, w_i^{\text{op}})\}$
  for *iter=1 to num_iters* **do**
    $\{(\mathbf{x}_b, \mathbf{y}_b, w_b)\}_{b=1}^{B} \subset \mathcal{X}$ `// randomly pick` B `samples from` $\mathcal{X}$
    $\{\mathbf{u}_b\}_{b=1}^{B} \subset \mathcal{U}$ `// randomly pick` B `samples from` $\mathcal{U}$
    for *b=1 to B* **do**
      for *m=1 to M* **do**
        $\hat{\mathbf{x}}_{b,m}$ = DataAugment($\mathbf{x}_b$)
        $\hat{\mathbf{u}}_{b,m}$ = DataAugment($\mathbf{u}_b$)
      end
      for *c=1 to* $|\mathcal{Y}|$ **do**
        $\mathbf{p}_b(c) = \frac{1}{M} \sum_m p_{\theta^{(D)}}(c|\hat{\mathbf{x}}_{b,m})$
        $\mathbf{q}_b(c) = \frac{1}{M} \sum_m p_{\theta^{(D)}}(c|\hat{\mathbf{u}}_{b,m})$
      end
        $\hat{\mathbf{y}}_b = \text{TempSharpen}_T(w_b \mathbf{y}_b + (1 - w_b)\mathbf{p}_b)$
        $\hat{\mathbf{q}}_b = \text{TempSharpen}_T(\mathbf{q}_b)$
    end
    $\hat{\mathcal{X}} = \{(\hat{\mathbf{x}}_{b,m}, \hat{\mathbf{y}}_b)\}_{b \in (1,\dots,B), m \in (1,\dots,M)}$
    $\hat{\mathcal{U}} = \{(\hat{\mathbf{u}}_{b,m}, \hat{\mathbf{q}}_b)\}_{b \in (1,\dots,B), m \in (1,\dots,M)}$
    $\mathcal{X}', \mathcal{U}' = \text{MixMatch}_\alpha(\hat{\mathcal{X}}, \hat{\mathcal{U}})$
    $\theta^{(D)} = \text{SGD}(\mathcal{L}^{(D)}, \theta^{(D)}, \mathcal{X}', \mathcal{U}')$
  end
  `// Train NetS`
  for *i=1 to* $|\mathcal{D}|$ **do**
    $\hat{c}_i = \arg\max_{c \in \mathcal{Y}} \left[(w_i^{\text{cl}}) p_{\theta^{(D)}}(c|\mathbf{x}_i) + (1 - w_i^{\text{cl}}) \mathbf{y}_i(c) \right]$
    $\hat{\mathbf{y}}_i = \text{onehot}(\hat{c}_i)$
  end
  $\theta^{(S)} = SGD(\mathcal{L}^{(S)}, \theta^{(S)}, \{(\mathbf{x}_i, \hat{\mathbf{y}}_i)\}_{i=1}^{|\mathcal{D}|})$
end

Method

- The loss for training Net-D is the same as that of **DivideMix**.

- The loss for training Net-S

$$\mathcal{L}^{(S)} = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|} \ell^{(S)}(\mathbf{x}_i, \mathbf{y}_i, \theta^{(S)}), \qquad \mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{|\mathcal{D}|}$$

$$\ell^{(S)}(\mathbf{x}_i, \mathbf{y}_i, \theta^{(S)}) = \sum_{c=1}^{|\mathcal{Y}|} (\mathbf{y}_i(c) - \alpha_{ic}/S_i)^2 + \frac{\alpha_{ic}(S_i - \alpha_{ic})}{S_i^2(S_i + 1)}, \tag{2}$$

where $\alpha_{ic} = \varphi(f_{\theta^{(S)}}(c|\mathbf{x}_i)) + 1$ for class $c \in \{1, ..., |\mathcal{Y}|\}$, with $\varphi(.)$ representing the ReLU activation function, and $S_i = \sum_{c=1}^{|\mathcal{Y}|} \alpha_{ic}$.

Experiments

ρ		0.3					0.6				
ω		0	0.25	0.5	0.75	1	0	0.25	0.5	0.75	1
EDM (Ours)	Best	**92.9**	**93.8**	**94.5**	**94.8**	**95.3**	**90.6**	**92.9**	**93.4**	**93.7**	94.3
	Last	**91.9**	**93.1**	**94.0**	**94.5**	**95.1**	**89.4**	**91.4**	**92.8**	**93.4**	94.0

	ResNet-101					**9-Layer CNN**				
	CIFAR-10									
	10%	20%	40%	80%	Pair 10%	10%	20%	40%	80%	Pair 10%
Direct Traing	88.31%	83.00%	65.66%	15.91%	88.17%	82.67%	76.42%	56.08%	17.67%	83.83%
Soft Bootstrapping	88.87%	83.20%	69.91%	18.12%	90.08%	82.68%	75.21%	54.55%	17.65%	83.55%
Hard Bootstrapping	89.69%	84.88%	68.90%	15.59%	89.17%	82.96%	75.00%	58.08%	18.18%	84.21%
MentorNet DD	92.80%	91.23%	88.64%	**46.31%**	91.02%	84.78%	80.71%	72.96%	28.19%	85.94%
CurriculumNet	90.59%	84.65%	69.45%	17.95%	90.45%	81.71%	74.02%	57.55%	16.23%	83.62%
Co-Teaching	90.36%	87.26%	82.80%	26.23%	90.77%	85.69%	82.66%	77.42%	22.60%	85.83%
O2U-net (Cycle Length 10)	93.58%	**92.57%**	**90.33%**	37.76%	**94.14%**	87.35%	84.85%	73.34%	33.18%	88.07%
O2U-net (Cycle Length 50)	**93.67%**	91.60%	89.59%	43.41%	93.99%	**87.64%**	**85.24%**	**79.64%**	**34.93%**	**88.22%**

Table 1: Benchmark results of all competing methods and our proposed EDM. Clean data was sampled from CIFAR10, while the open-set noise came from ImageNet32 and CIFAR-100. The total noise in the training data is represented by $\rho \in \{0.3, 0.6\}$, where the closed-set proportion of this noise is $\omega \in \{0, 0.25, 0.5, 0.75, 1\}$ and open-set proportion is $1 - \omega$.

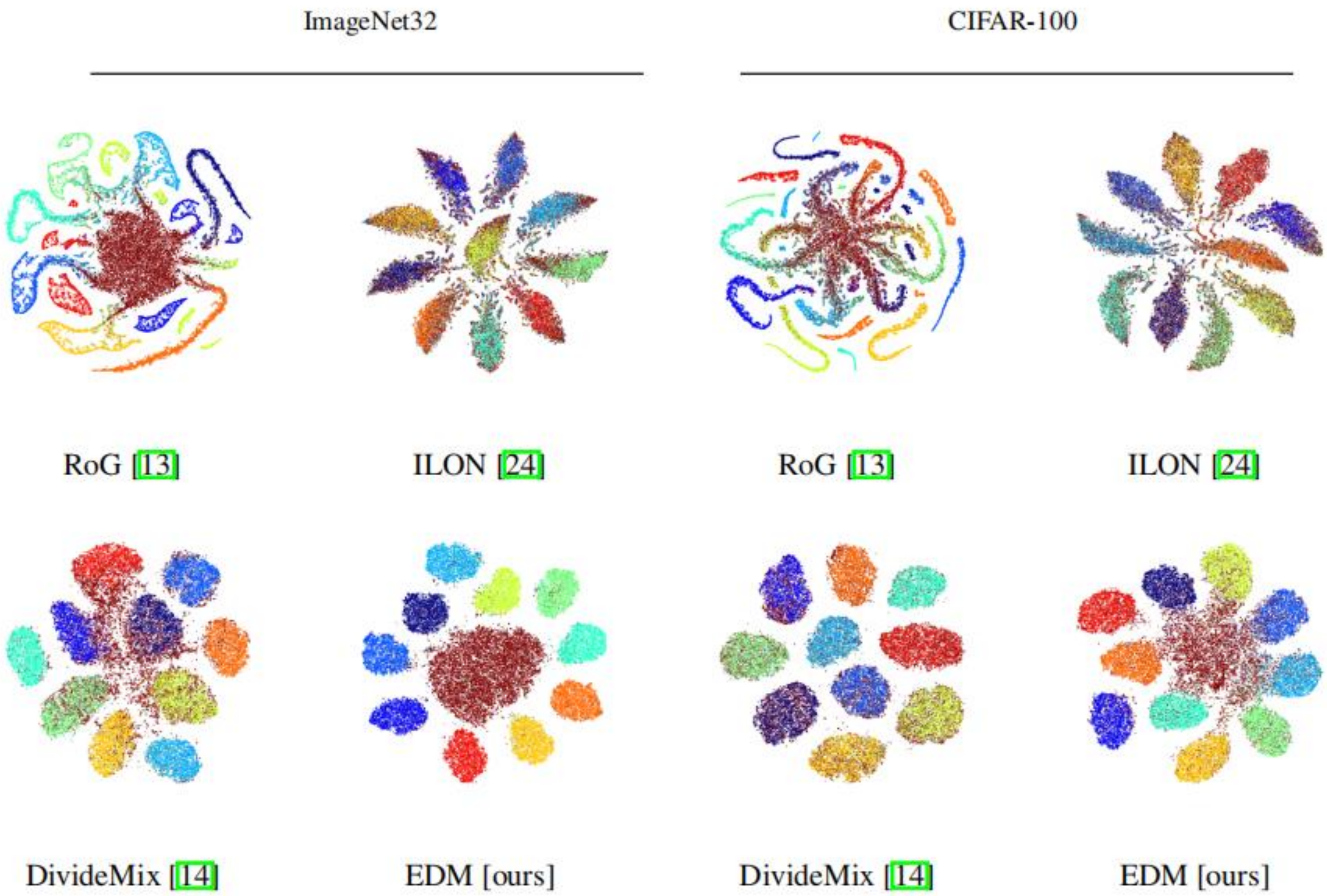
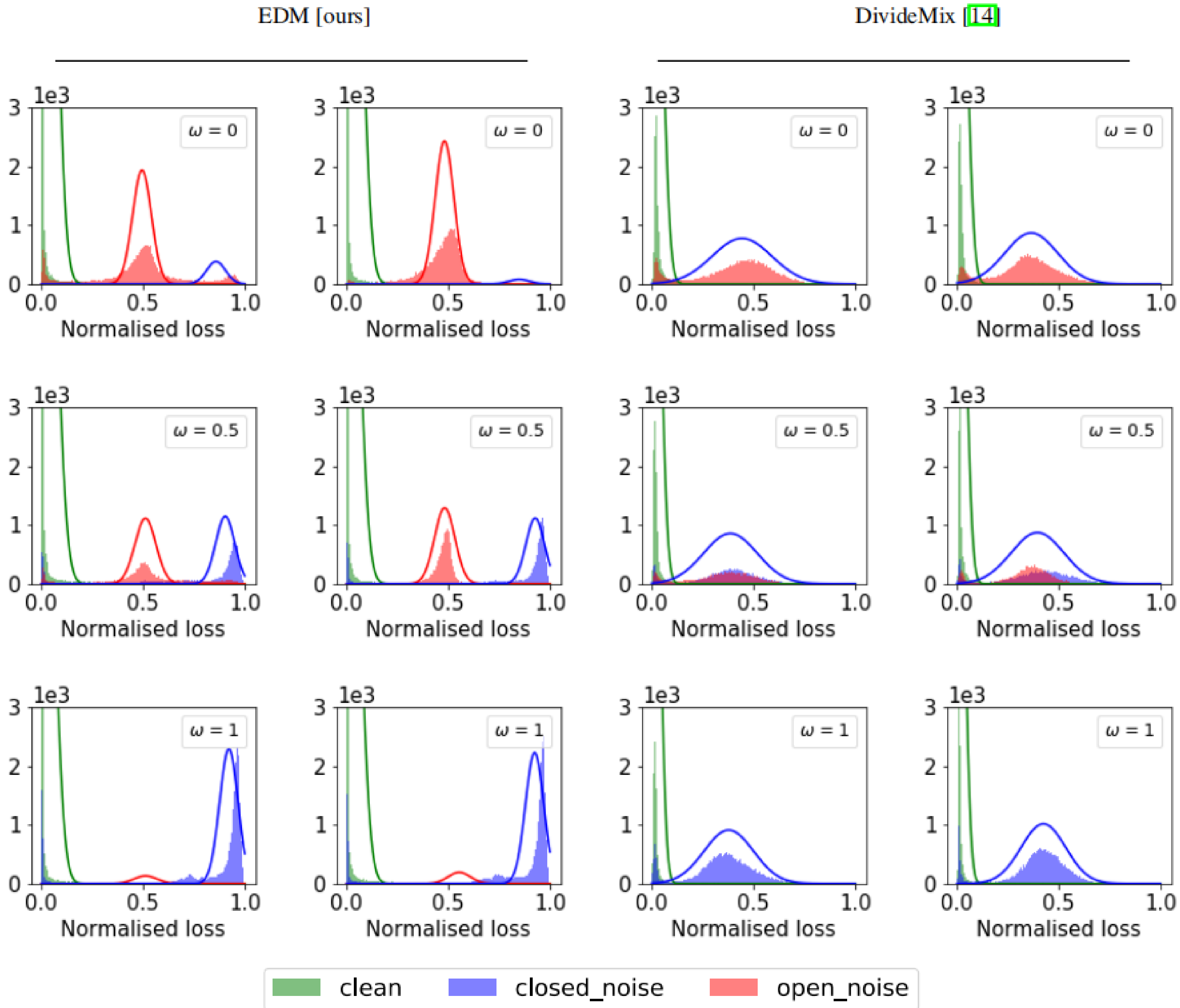


Figure 4: t-SNE plots of the related methods and our proposed EDM, where the total noise rate is $\rho = 0.6$ with the closed-set proportion being $\omega = 0.5$, and CIFAR-100 and ImageNet32 representing open-set data sets. The **brown** samples represent the open-set noise, while the other colours denote the true CIFAR-10 classes.



EDM [ours]
DivideMix [14]
1e3
ω = 0
ω = 0.5
ω = 1
Normalised loss
clean
closed_noise
open_noise

BeBold: Exploration Beyond the Boundary of Explored Regions

Tianjun Zhang[1] **Huazhe Xu**[1] **Xiaolong Wang**[2] **Yi Wu**[3]

Kurt Keutzer[1] **Joseph E. Gonzalez**[1] **Yuandong Tian**[4]

[1]University of California, Berkeley
{tianjunz,huazhe_xu,keutzer,jegonzal}@berkeley.edu

[2]Unveristy of California, San Diego
xiw012@ucsd.edu

[3] Tsinghua University
jxwuyi@gmail.com

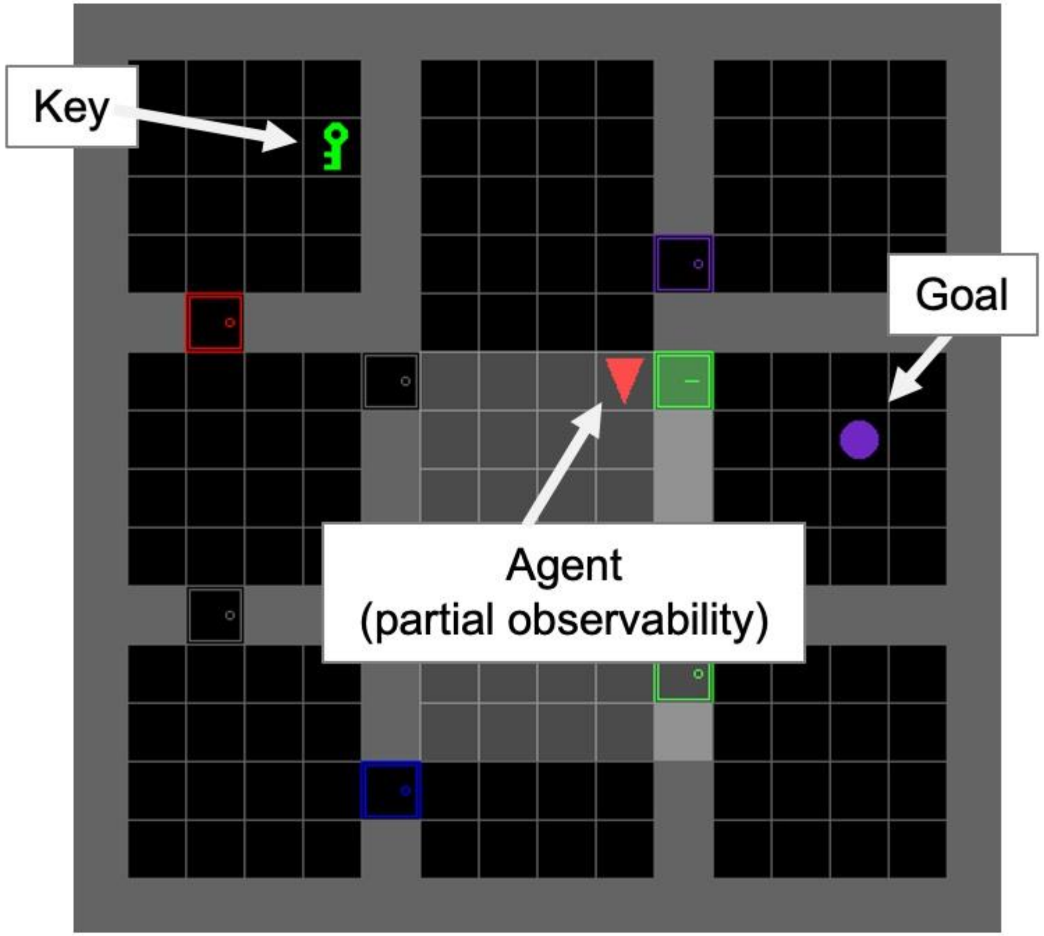
[4]Facebook AI Research
yuandong@fb.com

ICLR'21 Under Review

Contents

- Intrinsic Rewards
- Method
- Experiments

An example



No external reward

when agent wonders around.
when agent picks the key
when agent opens all doors
when agent opens the locked door
...

until the agent reaches the goal

Intrinsic Rewards

- Motivation
  - To motivate agents for exploration before any extrinsic rewards are obtained.
  - Efficient exploration under sparse rewards.

- Intrinsic Rewards
  - Curiosity-driven (CVPR'17)
  - Count-based (NIPS'16)
  - State-diff
    - Finally, state-diff approaches offer rewards if, for each trajectory, representations of consecutive states differ significantly.

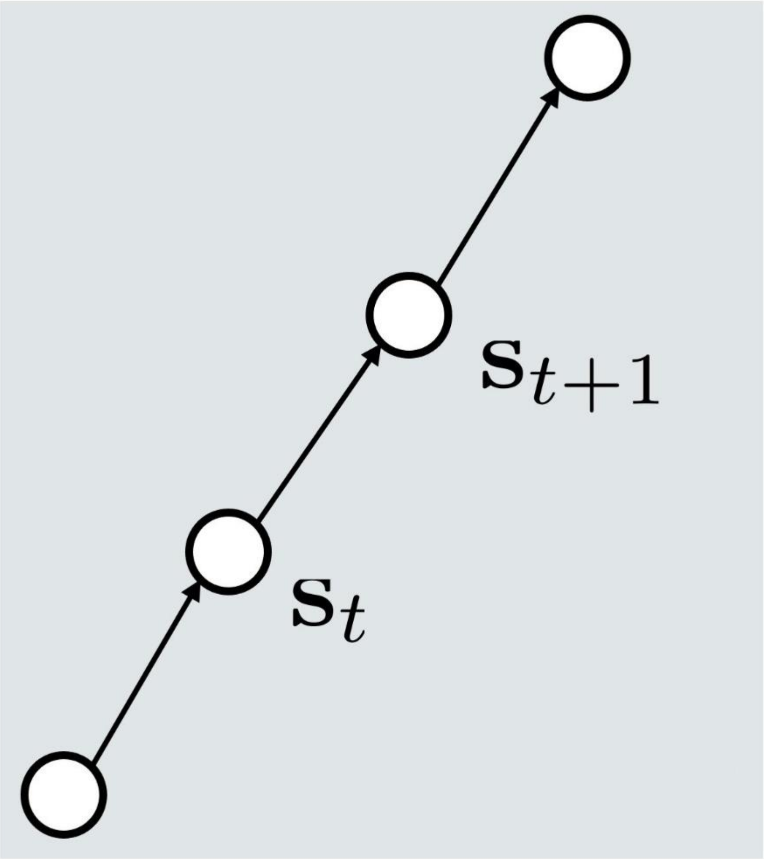
Method

BeBold (Beyond the Boundary of Explored Regions)

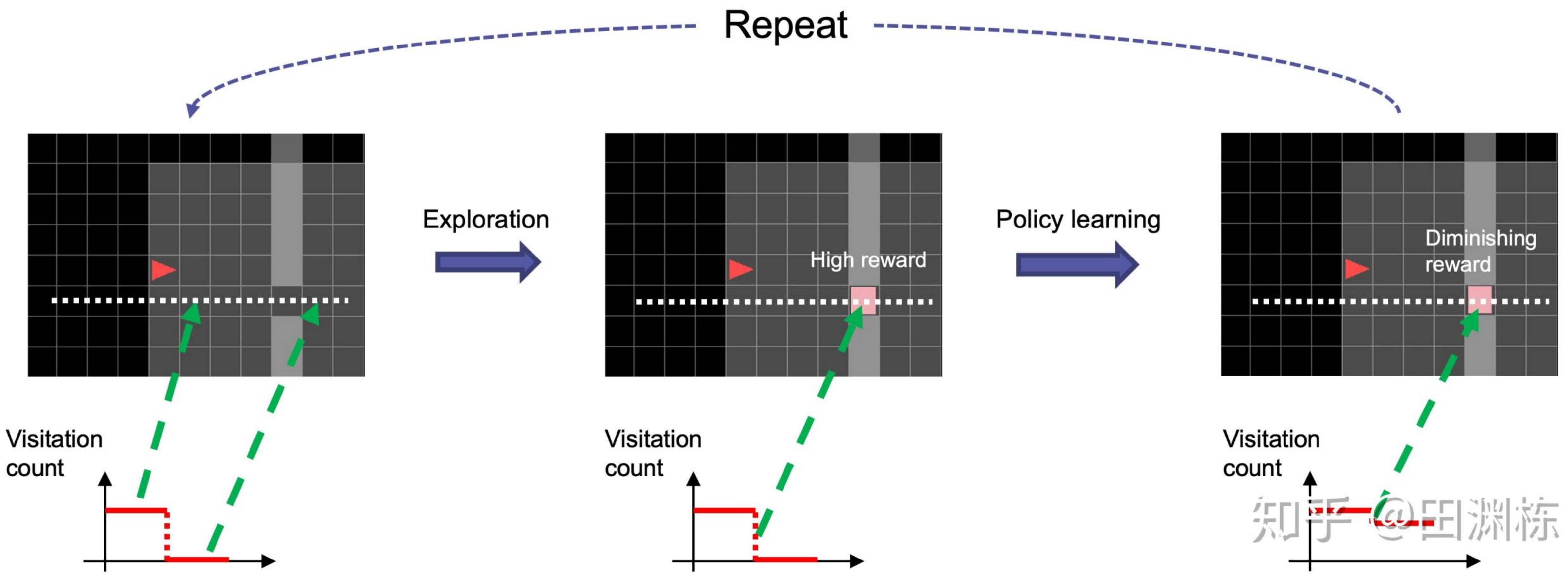
$$r^i(\mathbf{s}_t, \mathbf{a}_t) = \max\left(\frac{1}{N(\mathbf{s}_{t+1})} - \frac{1}{N(\mathbf{s}_t)}, 0\right)$$

Intrinsic Reward

Inverse of visitation counts



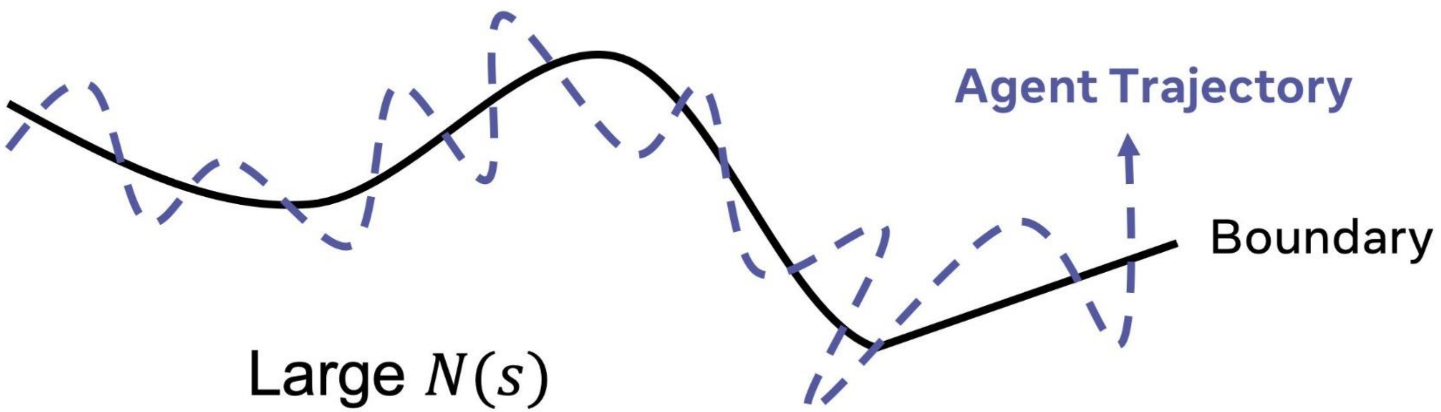
Method



Repeat
Exploration
Policy learning
High reward
Diminishing
reward
Visitation
count
Visitation
count
Visitation
count

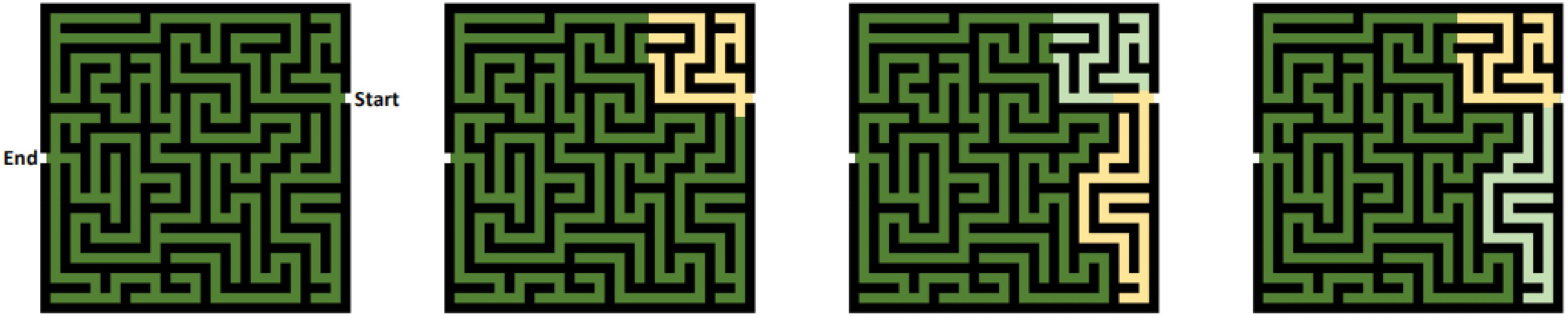
Method

BeBold



$$r^i(\mathbf{s}_t, \mathbf{a}_t) = \max\left(\frac{1}{N(\mathbf{s}_{t+1})} - \frac{1}{N(\mathbf{s}_t)}, 0\right) * \mathbb{1}\{\underline{N_e}(\mathbf{s}_{t+1}) = 1\}$$

RND



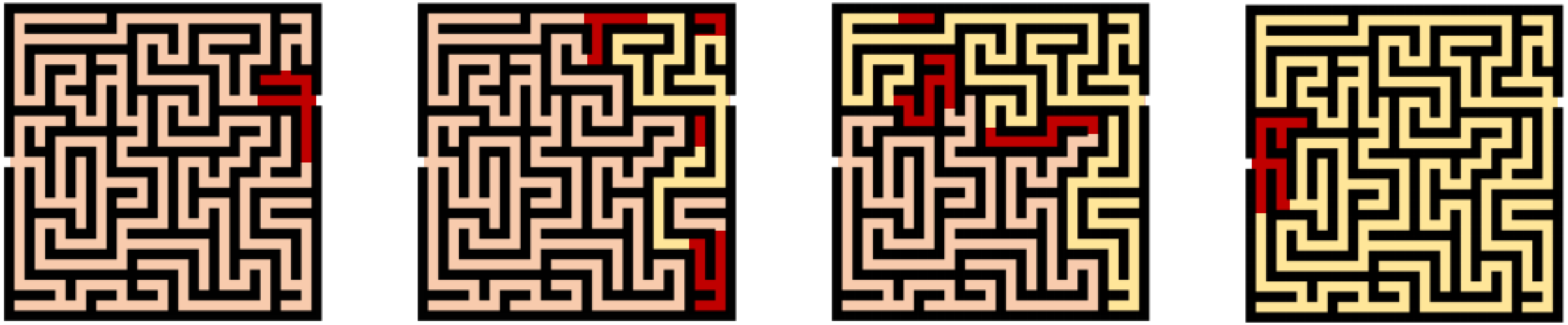
1. RND assigns high IR (dark green) throughout the environment

2. RND temporarily focuses on the upper right corner (yellow)

3. RND by chance starts exploring the bottom right corner heavily, resulting in the IR at top right higher than bottom right

4. RND re-explores the upper right and forgets the bottom right, gets trapped

BeBold



1. BeBold assigns high IR (dark red) near the start and low IR for the rest (light red)

2. BeBold pushes every direction to the frontier of exploration uniformly (yellow)

3. BeBold continuously pushes the exploration frontier

4. BeBold reaches the end of exploration

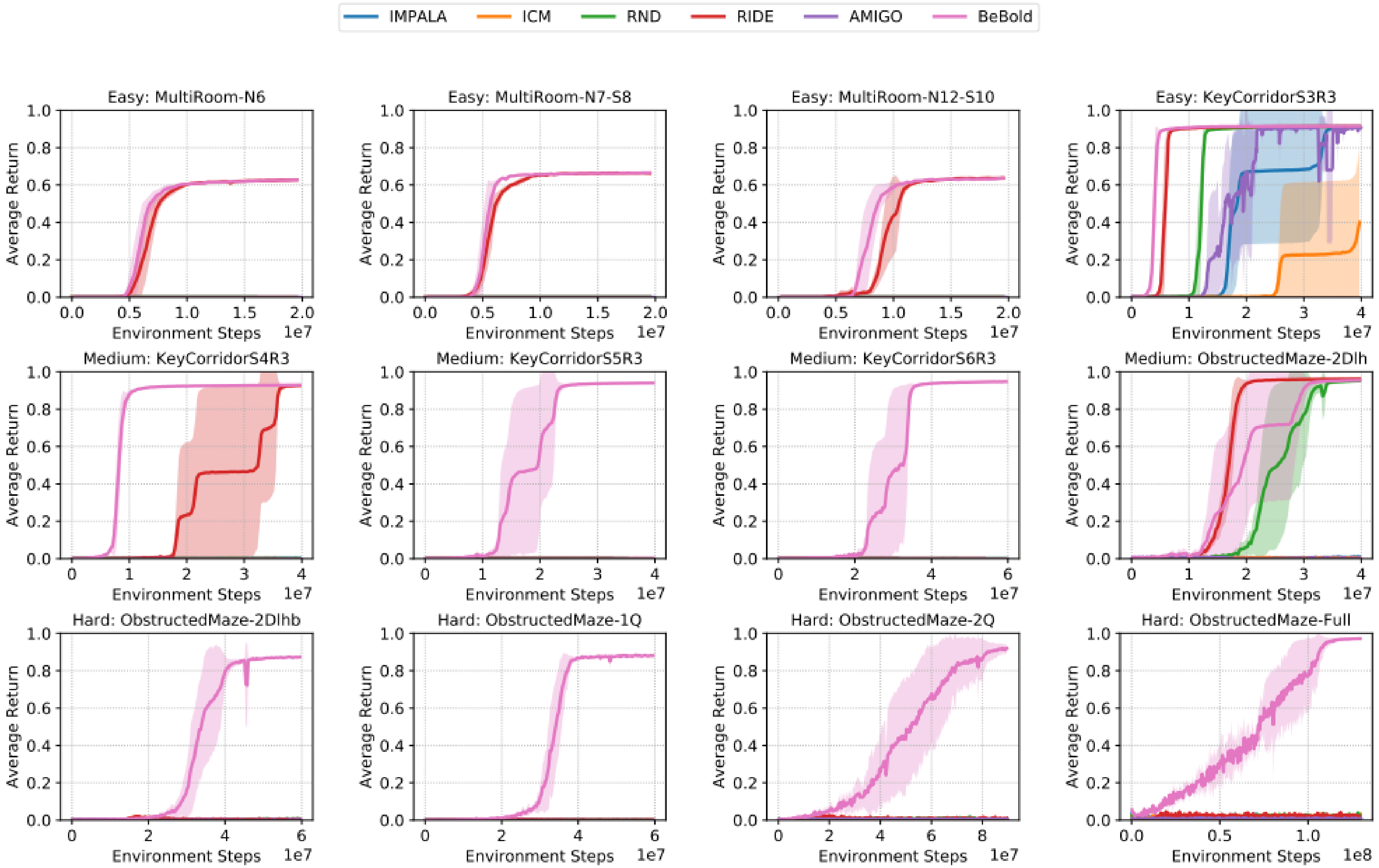


Figure 3: Results for various hard exploration environments from MiniGrid. BeBold successfully solves all the environments while all other baselines only manage to solve two to three relatively easy ones.

Thanks