Dynamic Divide-and-Conquer Adversarial Training For Robust Semantic Segmentation

ICCV 2021

Xiaogang Xu,Hengshuang Zhao

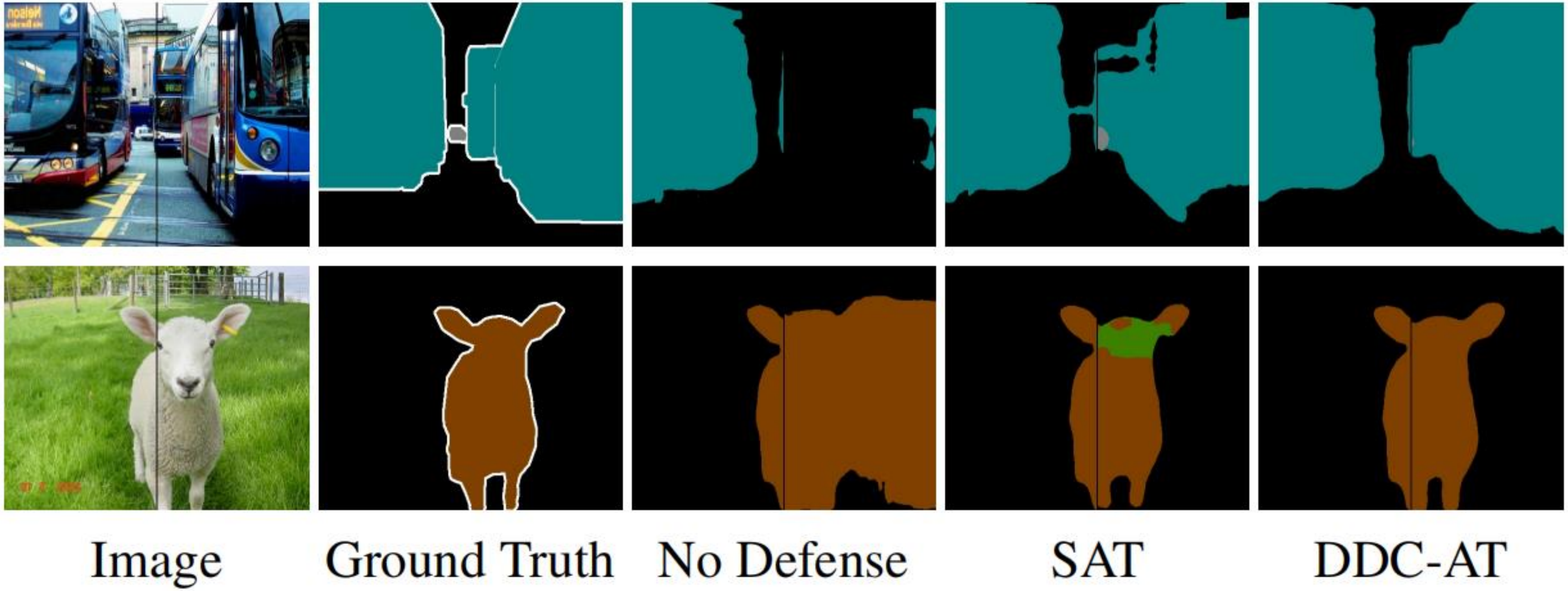


Image
Ground Truth
No Defense
SAT
DDC-AT

论文主要成果概述

文章探索了在语义分割领域通过对抗训练提高模型鲁棒性的方法。

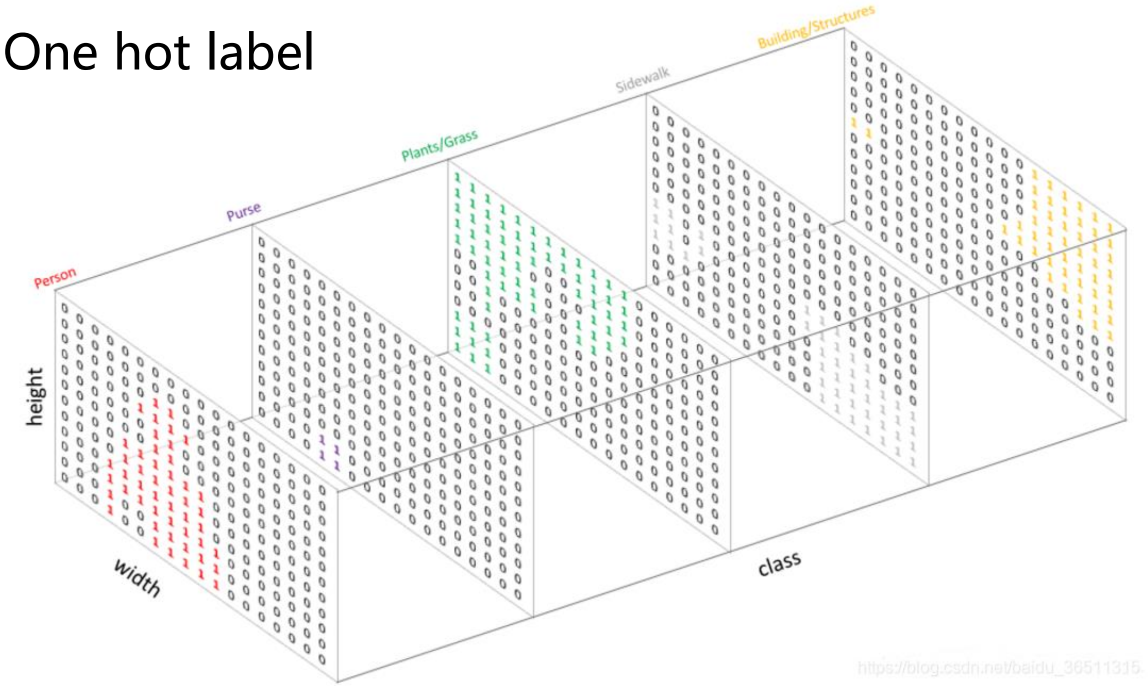
- 优点:
  - 提出了一种语义分割对抗训练的训练框架：SAT
  - 在SAT的基础上进行了改进：DDC-AT
- 不足:
  - 缺少对于时间成本的对比
  - 损失函数由三部分构成，可以再探究一下模型在训练过程中，倾向于通过提升哪一方面的准确率来降低损失值。

语义分割相关背景

常用损失函数： **交叉熵Loss**

$$L = -\sum_{c=1}^{M} y_c log(p_c)$$

One hot label



segmented

1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures

3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	1	1	1	1	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	5	5	5	5	5	5	5
5	5	3	3	3	3	1	1	3	3	5	5	5	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	4	4	5	5	5
4	4	4	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4

Label Map

Standard Adversarial Training（SAT）

Algorithm 1 Standard Adversarial Training

Parameter: clean training set $\mathbf{X}$, segmentation network f, maximum number of training iterations T_{max}, $T \leftarrow 0$

1: **while** $T \neq T_{max}$ **do**

2: Load a mini-batch of data $\mathbf{D}_b = \{x_1^{clean}, ..., x_m^{clean}\}$ from the training set $\mathbf{X}$.

3: Get adversarial samples $\mathbf{A}_b = \{x_1^{adv}, ..., x_m^{adv}\}$ from $\mathbf{D}_b$.

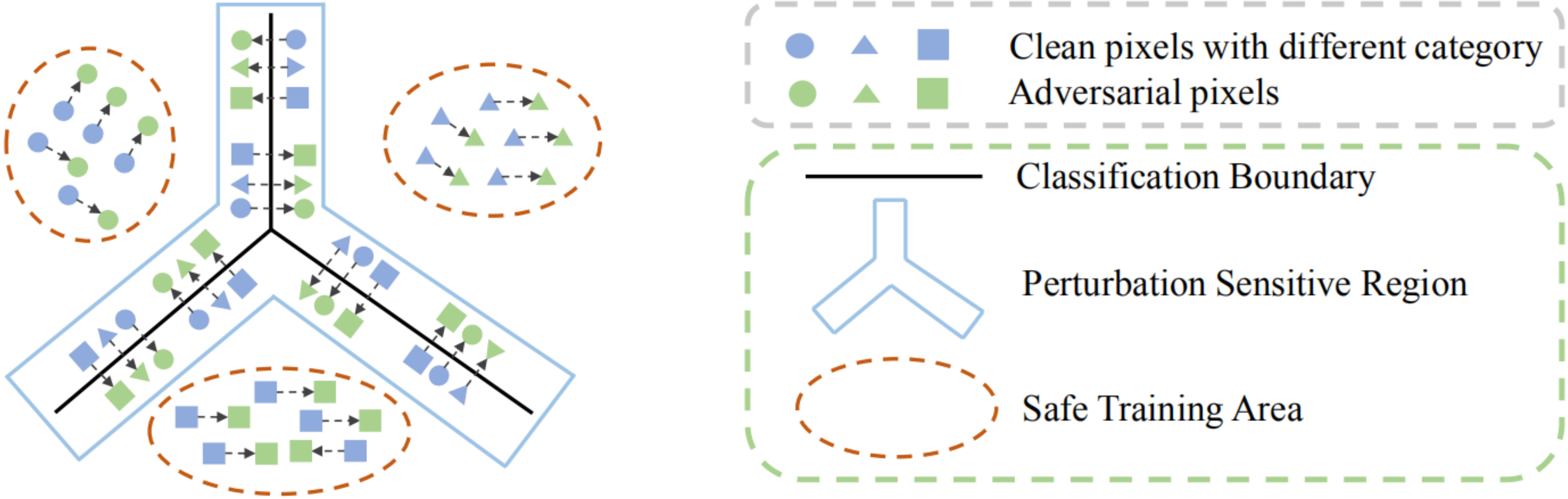
4: Set batch as $\{x_1^{clean}, ..., x_{\lfloor m/2 \rfloor}^{clean}, x_{\lfloor m/2 \rfloor+1}^{adv}, ..., x_m^{adv}\}$ from $\mathbf{D}_b$ and $\mathbf{A}_b$, and compute the loss for this training batch. Update parameters of f. $T \leftarrow T + 1$.

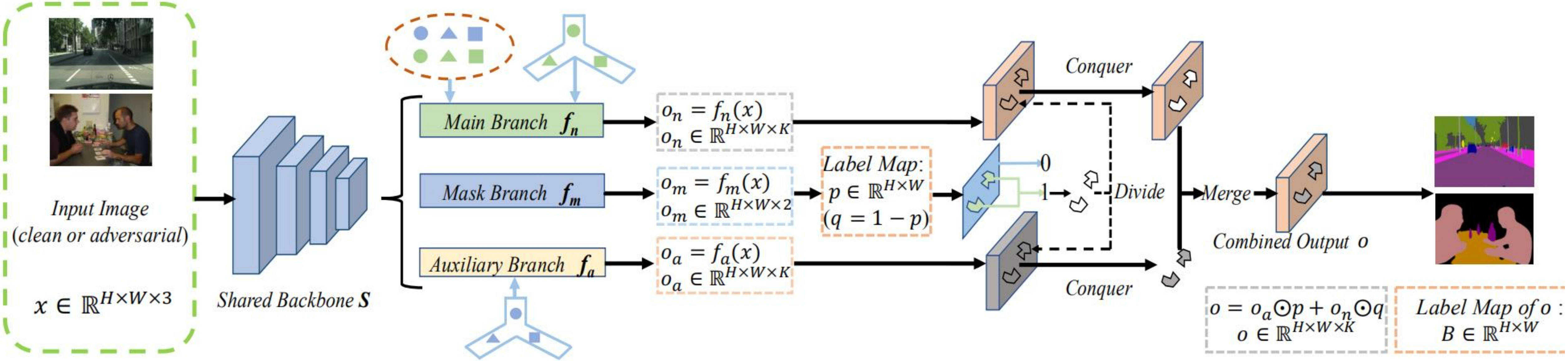
5: **end while**

SAT基础上进行改进：DDC-AT

改进动机：专人专事以降低学习难度

- 根据像素是否具有“边界属性”来对像素进行分类：
  - 像素边界属性的确定：训练一个模型（fm）用于判断像素是否具有边界属性
- 根据像素种类的不同，分别训练出对应种类的模型：
  - 具有边界属性的像素：训练一个模型（fa）用来预测具有边界属性的像素的类别
  - 不具有边界属性的像素：训练一个模型（fn）用来预测不具有边界属性的像素的类别


Clean pixels with different category
Adversarial pixels
Classification Boundary
Perturbation Sensitive Region
Safe Training Area


Input Image
(clean or adversarial)
$x \in \mathbb{R}^{H\times W\times 3}$
Shared Backbone S
Main Branch f_n
Mask Branch f_m
Auxiliary Branch f_a
$o_n = f_n(x)$
$o_n \in \mathbb{R}^{H\times W\times K}$
$o_m = f_m(x)$
$o_m \in \mathbb{R}^{H\times W\times 2}$
$o_a = f_a(x)$
$o_a \in \mathbb{R}^{H\times W\times K}$
Label Map:
$p \in \mathbb{R}^{H\times W}$
$(q = 1 - p)$
0
1
Conquer
Divide
Merge
Conquer
Combined Output o
$o = o_a \odot p + o_n \odot q$
$o \in \mathbb{R}^{H\times W\times K}$
Label Map of o :
$B \in \mathbb{R}^{H\times W}$

DDC-AT 损失函数：

Combined with Eqs. (3) and (4), the overall loss term is

$$\mathcal{L}_{all} = \lambda_1 \mathcal{L}_n + \lambda_2 \mathcal{L}_a + \lambda_3 \mathcal{L}_m, \tag{5}$$

where λ_1, λ_2, and λ_3 are set to 1 in experiments. Overall training procedure is concluded in Alg. 3.

The loss of x for f_n and f_a is written as

$$\begin{aligned} \mathcal{L}_n &= \mathbb{E}\left(-\sum_{i=0}^{K-1}[y(:,:,i)\cdot\log(f_n(x)(:,:,i))]\odot q\right), \\ \mathcal{L}_a &= \mathbb{E}\left(-\sum_{i=0}^{K-1}[y(:,:,i)\cdot\log(f_a(x)(:,:,i))]\odot p\right), \end{aligned} \tag{3}$$

the loss for f_m becomes

$$\mathcal{L}_m = \mathbb{E}\left(-\sum_{i=0}^{1}[\widetilde{M}(:,:,i)\cdot\log(f_m(x)(:,:,i))]\right). \tag{4}$$

Algorithm 2 Algorithm to obtain ground truth (mask label) for training of mask branch f_m

Parameter: clean data x^{clean} with one-hot label y, all-zero matrix $\mathbf{0}$, function $\mathcal{F} = \mathbf{1}[\mathcal{N}]$ ($\mathcal{F}(i,j) = 1$ if $\mathcal{N}(i,j)$ is True)

1: **Obtain** output o_n^{clean}, o_a^{clean}, and o_m^{clean} for x^{clean} from f_n, f_a, and f_m. Label map of o_m^{clean} is p^{clean}.

2: **Compute** $o^{clean} = o_a^{clean} \odot p^{clean} + o_n^{clean} \odot (1 - p^{clean})$, its label map is B^{clean}, $B^{clean}(i,j) \in \{0, 1, ...K-1\}$.

3: **Use** loss $\mathcal{L}(o_n^{clean}, y)$ to generate adversarial examples x^{adv}.

4: **Obtain** output o_n^{adv}, o_a^{adv}, and o_m^{adv} for x^{adv} from f_n, f_a, and f_m. The label map of o_m^{adv} is p^{adv}.

5: **Compute** $o^{adv} = o_a^{adv} \odot p^{adv} + o_n^{adv} \odot (1 - p^{adv})$ with label map B^{adv}, where $B^{adv}(i,j) \in \{0, 1, ...K-1\}$.

6: **Generate** $M^{clean} = \mathbf{1}[B^{clean} \neq B^{adv}]$, $M^{clean} \in \mathbb{R}^{H \times W}$.

7: **Generate** $M^{adv} = \mathbf{0}$ with the same shape of M^{clean}.

8: **return** M^{clean}, M^{adv}, x^{clean}, and x^{adv}.

Algorithm 3 Dynamic divide-and-conquer adversarial training for semantic segmentation networks

Parameter: clean training set $\mathbf{X}$, shared backbone S, main branch f_n, auxiliary branch f_a, mask branch f_m, training batch size m, and maximum training iteration T_{max}, the number of iterations $T \leftarrow 0$

while $T \neq T_{max}$ **do**
  Load a mini-batch of data $\mathbf{D}_b = \{x_1^{clean}, ..., x_b^{clean}\}$ from $\mathbf{X}$ with one-hot labels $\mathbf{Y}_b = \{y_1, ..., y_b\}$.
  Use the current state of network $\{S, f_n, f_a, f_m\}$, $\mathbf{D}_b$, and $\mathbf{Y}_b$ to generate adversarial examples as $\mathbf{A}_b = \{x_1^{adv}, ..., x_b^{adv}\}$, and obtain "mask label" for $\mathbf{D}_b$ and $\mathbf{A}_b$ as $\mathbf{M}_b^{clean} = \{M_1^{clean}, ..., M_b^{clean}\}$ and $\mathbf{M}_b^{adv} = \{M_1^{adv}, ..., M_b^{adv}\}$.
  Compute output from f_m for $\mathbf{D}_b$, and obtain the label map $\{p_1^{clean}, ..., p_b^{clean}\}$.
  Compute output from f_m for $\mathbf{A}_b$, and obtain the label map $\{p_1^{adv}, ..., p_b^{adv}\}$.
  Compute $\{q_1^{clean}, ..., q_b^{clean}\}$ and $\{q_1^{adv}, ..., q_b^{adv}\}$, where $q_i^{clean} = 1 - p_i^{clean}$, $q_i^{adv} = 1 - p_i^{adv}$.
  $\mathbf{T}_b = \{x_1^{clean}, ..., x_{\lfloor b/2 \rfloor}^{clean}, x_{\lfloor b/2 \rfloor+1}^{adv}, ..., x_b^{adv}\}$, $\mathbf{M}_b = \{M_1^{clean}, ..., M_{\lfloor b/2 \rfloor}^{clean}, M_{\lfloor b/2 \rfloor+1}^{adv}, ..., M_b^{adv}\}$,
  $\mathbf{P}_b = \{p_1^{clean}, ..., p_{\lfloor b/2 \rfloor}^{clean}, p_{\lfloor b/2 \rfloor+1}^{adv}, ..., p_b^{adv}\}$, $\mathbf{Q}_b = \{q_1^{clean}, ..., q_{\lfloor b/2 \rfloor}^{clean}, q_{\lfloor b/2 \rfloor+1}^{adv}, ..., q_b^{adv}\}$.
  Compute loss by (3) with $\mathbf{T}_b$, $\mathbf{Y}_b$, $\mathbf{P}_b$ and $\mathbf{Q}_b$. Update weights of network $\{S, f_n, f_a\}$.
  Compute loss by (4) using $\mathbf{T}_b$ and $\mathbf{M}_b$. Update weights of $\{S, f_m\}$. $T \leftarrow T + 1$.
end while

Dynamic:

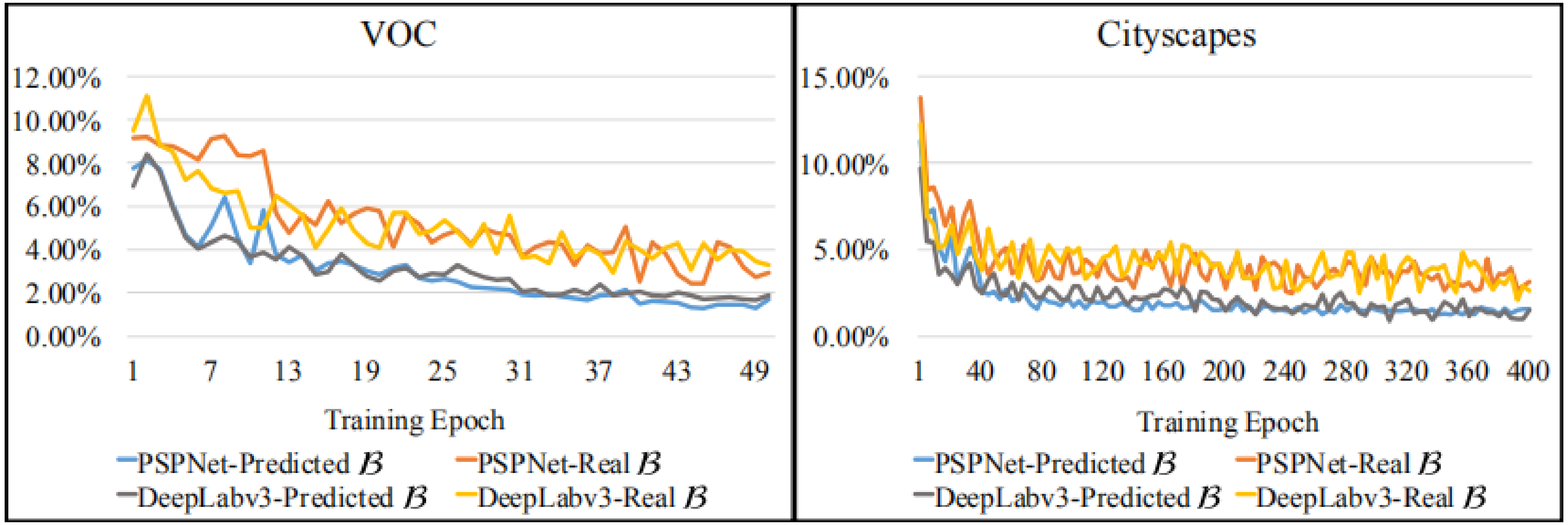


Figure 4. The proportion of predicted/real $\mathcal{B}$ in one clean image with respect to the number of training epoch.

实 验 结 果

实验对比的结论：

- 普通的SAT训练方法就可以有效提高鲁棒性
- DDC-AT相比于SAT，有小幅度的提升

数据集：VOC

数据集：Cityscapes

White Box attack

数据集：VOC

	clean	Model: PSPNet					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	76.9	18.9	7.8	5.4	40.3	3.3	15.7
SAT (Mean)	74.3	68.1	44.5	27.9	59.0	65.5	36.4
DDC-AT (Mean)	76.0	**75.6**	**47.9**	**33.8**	**61.2**	**67.4**	**37.1**
SAT (Std)	0.5	1.8	2.9	3.2	1.4	1.2	4.1
DDC-AT (Std)	0.1	0.5	2.2	4.0	1.1	1.1	1.8
	clean	Model: DeepLabv3					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	77.5	19.6	8.1	5.5	39.3	3.9	16.7
SAT (Mean)	72.7	62.4	43.1	28.8	59.0	66.0	37.0
DDC-AT(Mean)	75.2	**69.9**	**43.6**	**32.3**	**60.4**	**67.1**	**37.8**
SAT (Std)	1.0	0.6	1.9	2.0	0.4	1.2	1.1
DDC-AT (Std)	0.1	1.3	0.5	1.2	0.4	0.4	0.1

数据集：Cityscapes

	clean	Model: PSPNet					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	74.6	26.2	5.5	2.1	35.8	13.8	22.7
SAT (Mean)	69.0	46.7	32.9	25.8	56.0	49.1	45.8
DDC-AT (Mean)	71.7	**50.2**	**34.7**	**28.7**	**57.2**	**50.8**	**46.7**
SAT (Std)	1.0	1.0	0.3	1.0	3.0	1.5	1.8
DDC-AT (Std)	0.1	0.2	0.2	0.3	0.1	0.1	0.1
	clean	Model: DeepLabv3					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	74.8	26.0	5.7	2.3	31.5	13.8	22.6
SAT (Mean)	69.4	46.1	31.8	26.2	56.7	48.4	45.0
DDC-AT (Mean)	71.3	**50.9**	**34.9**	**29.0**	**57.4**	**50.5**	**46.8**
SAT (Std)	1.0	1.0	0.6	0.4	1.7	1.3	0.9
DDC-AT (Std)	0.3	0.4	0.2	0.2	0.2	1.5	0.4

Black Box attack

数据集：VOC

	clean	Model: PSPNet					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	76.9	24.0	10.6	6.0	46.6	15.6	20.9
SAT (Mean)	74.3	56.5	51.3	44.9	64.0	68.5	58.7
DDC-AT (Mean)	76.0	**61.5**	**53.4**	**46.1**	**68.4**	**70.5**	**59.6**
SAT (Std)	0.5	2.9	2.8	4.2	2.1	1.3	3.0
DDC-AT (Std)	0.1	1.7	1.8	3.9	0.3	0.1	0.8
	clean	Model: DeepLabv3					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	77.5	24.6	10.5	7.0	49.1	19.6	20.9
SAT (Mean)	72.7	51.8	51.0	45.0	64.4	68.5	64.5
DDC-AT (Mean)	75.2	**60.4**	**52.6**	**46.0**	**68.7**	**70.6**	**65.9**
SAT (Std)	1.0	3.8	3.7	4.1	1.8	1.6	0.4
DDC-AT (Std)	0.1	5.1	1.8	1.6	1.0	0.5	1.0

数据集：Cityscapes

	clean	Model: PSPNet					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	74.6	28.0	6.9	3.3	35.6	21.1	25.3
SAT (Mean)	69.0	44.4	36.7	30.8	57.7	57.8	56.6
DDC-AT (Mean)	71.7	**50.6**	**37.9**	**32.3**	**58.6**	**58.4**	**57.4**
SAT (Std)	1.0	3.0	3.3	2.6	2.4	2.0	2.5
DDC-AT (Std)	0.1	1.0	1.0	0.2	0.1	0.1	0.3
	clean	Model: DeepLabv3					
		2	4	6	DeepFool	C&W	BIM L_2
No Defense	74.8	29.9	7.6	3.1	35.8	27.3	27.3
SAT (Mean)	69.4	43.2	36.1	31.6	58.4	58.3	57.4
DDC-AT (Mean)	71.3	**47.8**	**37.8**	**32.8**	**59.6**	**59.7**	**59.2**
SAT (Std)	1.0	3.0	3.0	2.3	1.1	1.8	2.3
DDC-AT (Std)	0.3	1.9	1.0	0.3	0.7	0.5	0.2

Table 6. Evaluation under white- and black-box attack on Cityscapes.

	PSPNet (white-box)				PSPNet (black-box)			
	2	DeepFool	C&W	BIM L_2	2	DeepFool	C&W	BIM L_2
multi-task [23]	38.4	40.6	26.3	34.2	40.1	42.4	28.6	35.5
multi-task [30]	30.3	37.6	17.3	25.8	31.4	38.2	23.6	27.3
TS [3]	41.6	54.3	40.4	43.8	43.2	55.7	42.6	49.2
TS [4]	47.9	56.8	44.5	45.2	48.3	57.1	47.2	51.8
DDC-AT	**50.2**	**57.2**	**50.8**	**46.7**	**50.6**	**58.6**	**58.4**	**57.4**
	DeepLabv3 (white-box)				DeepLabv3 (black-box)			
	2	DeepFool	C&W	BIM L_2	2	DeepFool	C&W	BIM L_2
multi-task [23]	37.5	41.2	27.4	35.8	41.6	44.1	32.3	37.9
multi-task [30]	28.9	35.3	16.8	25.5	31.8	38.7	30.2	31.4
TS [3]	42.3	54.0	41.1	42.4	44.5	56.2	44.8	51.5
TS [4]	48.1	55.3	46.5	45.3	50.3	57.6	50.3	53.7
DDC-AT	**50.9**	**57.4**	**50.5**	**46.8**	**47.8**	**59.6**	**59.7**	**59.2**

END