Efficient loss function

Peng Zhang

2021-4-21

内容总览

- 基本概念及内容

  Improved Training Speed, Accuracy, and Data Utilization Through Loss Function Optimization

- 实例分割

  1. Semantic Instance Segmentation with a Discriminative Loss Function
  2. PolarMask: Single Shot Instance Segmentation with Polar Representation

- 场景图解析

  Graphical Contrastive Losses for Scene Graph Parsing

- 场景图生成

  Scene Dynamics: Counterfactual Critic Multi-Agent Training for Scene Graph Generation

基本概念

Loss 是预测值和真实值之映射到某一空间的误差，而损失函数（loss function）是用来**估量模型的预测值f(x)与真实值Y的不一致程度**，它是一个非负实值函数,通常使用L(Y, f(x))来表示，损失函数越小，模型的鲁棒性（Robust）就越好。它是**结构风险函数**重要组成部分，包括了经验风险项和正则项。

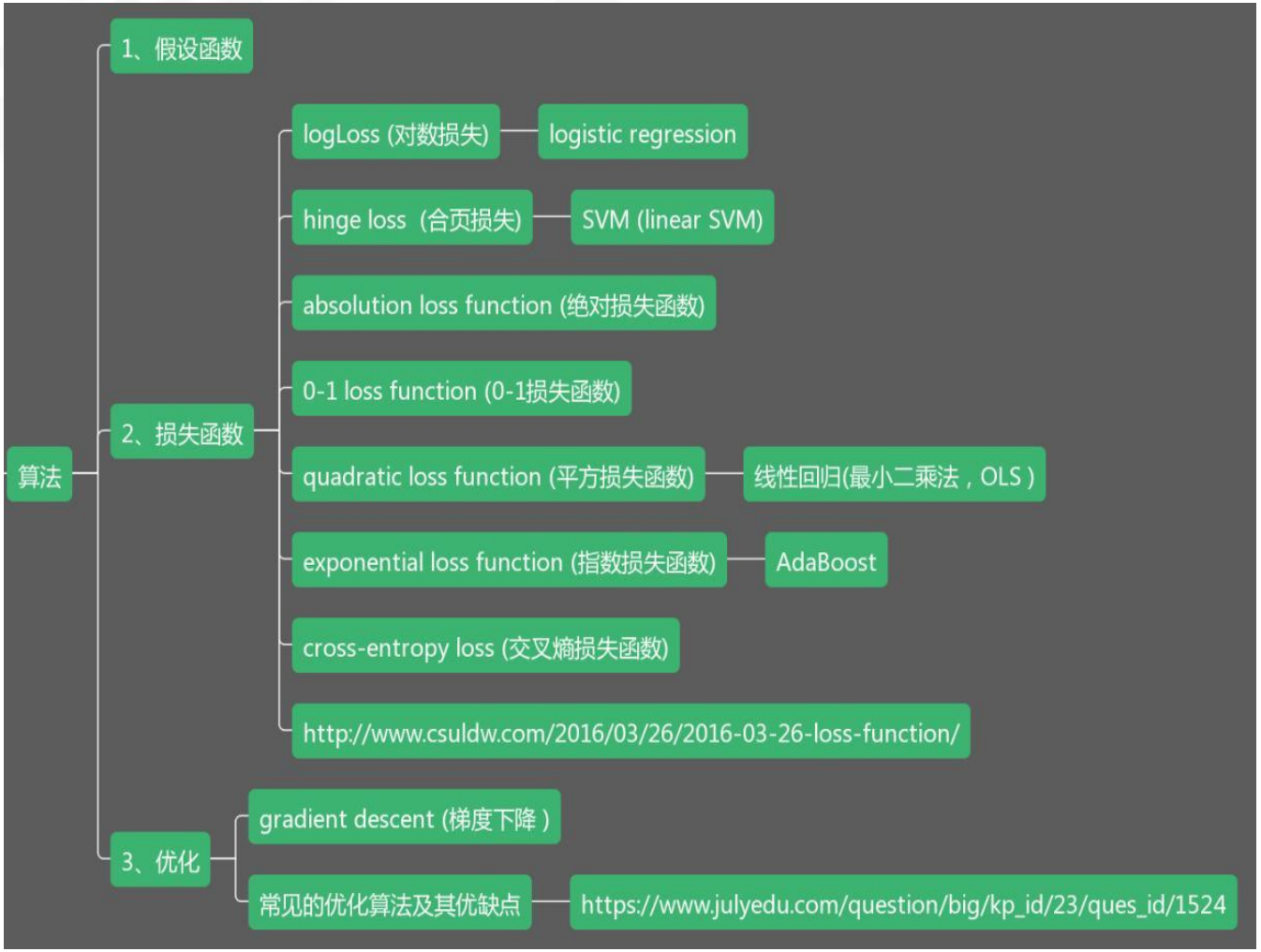
$$\theta^* = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i; \theta)) + \lambda \, \Phi(\theta)$$

损失函数越好，通常模型的性能越好。

损失函数分为**经验风险损失函数**和**结构风险损失函数**。

需要注意过拟合问题

Loss分类


算法
1、假设函数
2、损失函数
logLoss (对数损失)
logistic regression
hinge loss (合页损失)
SVM (linear SVM)
absolution loss function (绝对损失函数)
0-1 loss function (0-1损失函数)
quadratic loss function (平方损失函数)
线性回归(最小二乘法，OLS)
exponential loss function (指数损失函数)
AdaBoost
cross-entropy loss (交叉熵损失函数)
http://www.csuldw.com/2016/03/26/2016-03-26-loss-function/
3、优化
gradient descent (梯度下降)
常见的优化算法及其优缺点
https://www.julyedu.com/question/big/kp_id/23/ques_id/1524

Loss分类（归纳：https://zhuanlan.zhihu.com/p/58883095）

1. **0-1损失函数(zero-one loss)**

0-1损失是指预测值和目标值不相等为1， 否则为0:

$$L(Y, f(X)) = \begin{cases} 1, Y \neq f(X) \\ 0, Y = f(X) \end{cases}$$

特点：

(1)0-1损失函数直接对应分类判断错误的个数，但是它是一个非凸函数，不太适用.

(2)**感知机**就是用的这种损失函数。但是相等这个条件太过严格，因此可以放宽条件，即满足 $|Y - f(x)| < T$ 时认为相等，

$$L(Y, f(X)) = \begin{cases} 1, |Y - f(X)| \geq T \\ 0, |Y = f(X)| < T \end{cases}$$

2. **绝对值损失函数**

绝对值损失函数是计算预测值与目标值的差的绝对值：

$$L(Y, f(x)) = |Y - f(x)|$$

3. **log对数损失函数**

log对数损失函数的标准形式如下：

$$L(Y, P(Y|X)) = -logP(Y|X)$$

特点：

(1) log对数损失函数能非常好的表征概率分布，在很多场景尤其是多分类，如果需要知道结果属于每个类别的置信度，那它非常适合。

(2)健壮性不强，相比于hinge loss对噪声更敏感。

(3)**逻辑回归**的损失函数就是log对数损失函数。

4. **平方损失函数**

平方损失函数标准形式如下：

$$L(Y|f(X)) = \sum_{N}(Y - f(X))^2$$

特点：

(1)经常应用与回归问题

5. **指数损失函数（exponential loss）**

指数损失函数的标准形式如下：

$$L(Y|f(X)) = exp[-yf(x)]$$

特点：

(1)对离群点、噪声非常敏感。经常用在AdaBoost算法中。

Improved Training Speed, Accuracy, and Data Utilization Through Loss Function Optimization

优化抽取数据速度：通过损失函数优化提高训练速度、准确性和数据利用率

Santiago Gonzalez and Risto Miikkulainen
Cognizant Technology Solutions, San Francisco, California, USA
Department of Computer Science,
University of Texas at Austin, Austin, Texas, USA

Arxiv.org 2020

动机：优化抽取数据速度。

主要贡献：提出了Genetic Loss-function Optimization （GLO） 框架来搜索新的损失函数。

创新：包括两个阶段。

•搜索loss函数的形式（遗传算法）

•优化loss函数的系数

•搜索loss函数的形式（遗传算法）：比如 $loss(x,y) = a \cdot x^2 + b \cdot x \cdot y^2 + c$

•优化loss函数的系数：即优化上面例子中a，b，c这三个系数，其中x，y分别表示真实和预测值。 例子：3(5x+y)=15x+3y

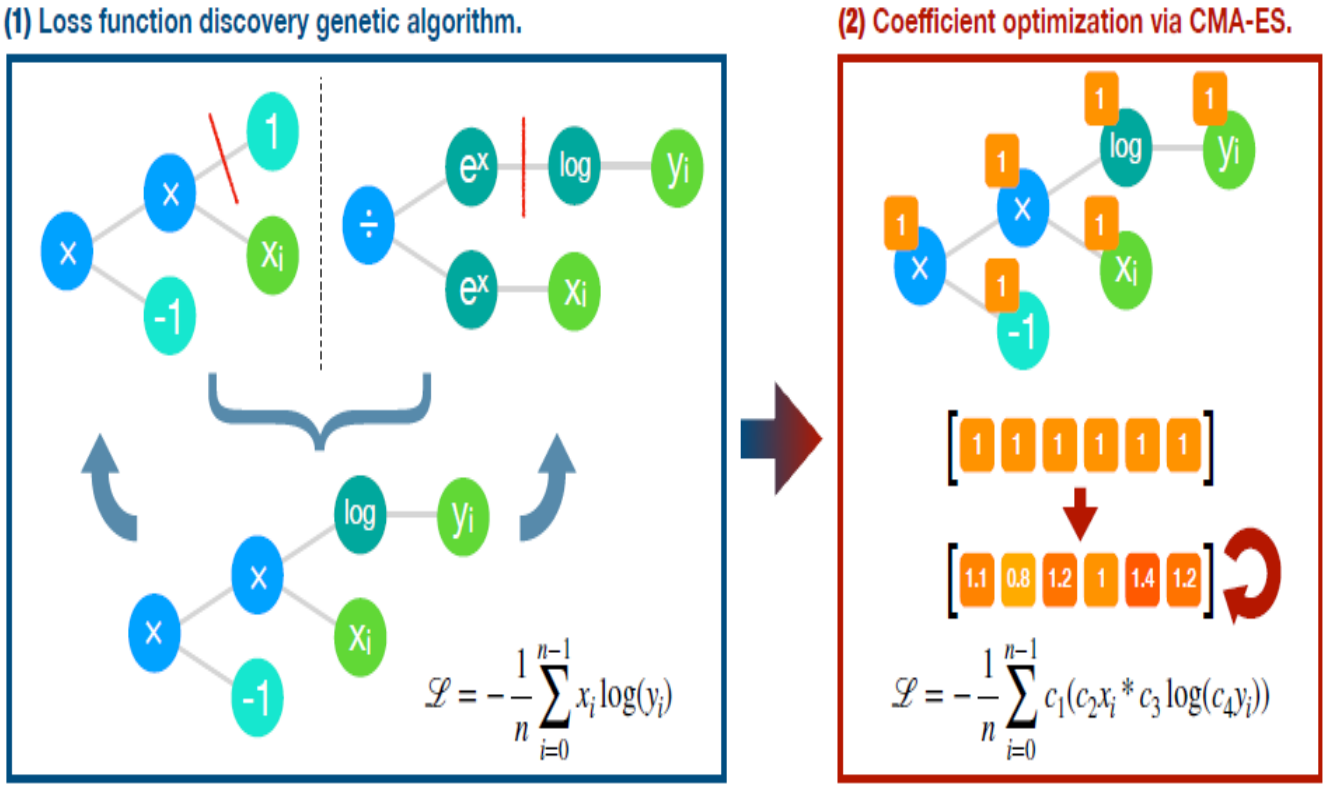


Fig. 1. Genetic Loss Optimization (GLO) overview. A genetic algorithm constructs candidate loss functions as trees. The best loss functions from this set then has its coefficients optimized using CMA-ES. GLO loss functions are able to train models more quickly and more accurately.

实验评估：

$$\mathcal{L}_{\text{Baikal}} = -\frac{1}{n}\sum_{i=0}^{n}\log(y_i) - \frac{x_i}{y_i}$$

使用CMA-ES算法搜索到的系数如下：

$$\mathcal{L}_{\text{BaikalCMA}} = -\frac{1}{n}\sum_{i=0}^{n} c_0\left(c_1 * \log(c_2 * y_i) - c_3\frac{c_4 * x_i}{c_5 * y_i}\right)$$

$$\text{其中} c_0 = 2.7279, c_1 = 0.9863, c_2 = 1.5352, c_3 = -1.1135, c_4 = 1.3716, c_5 = -0.8411$$

Testing accuracy

Loss function	Accuracy
Crossentropy	0.9899
Baikal	0.9933
BaikalCMA	0.9947

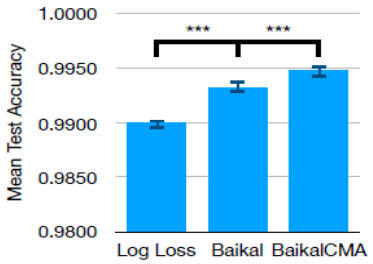


Fig. 2. Mean testing accuracy on MNIST, $n = 10$. Both Baikal and BaikalCMA provide statistically significant improvements to testing accuracy over the cross-entropy loss.

Training speed：

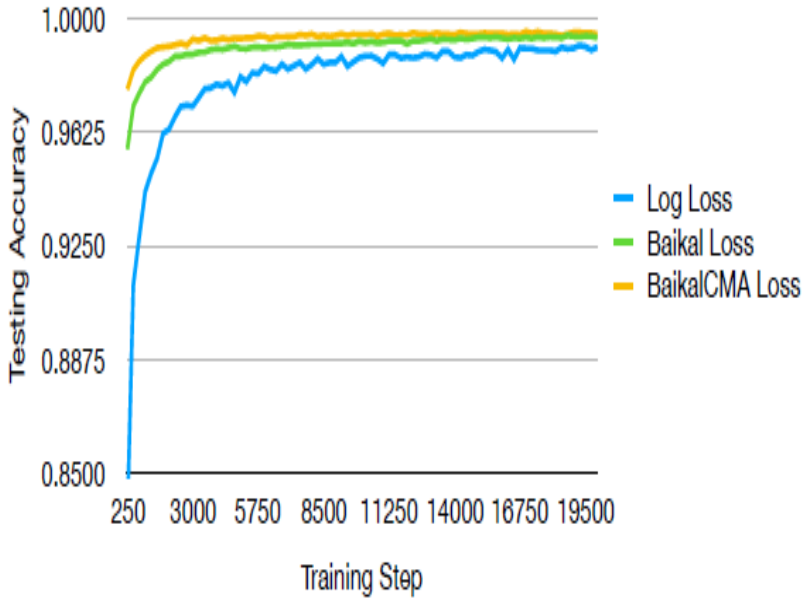


Fig. 3. Training curves for different loss functions on MNIST. Baikal and BaikalCMA result in faster and smoother training compared to the cross-entropy loss.

Training data requirements：

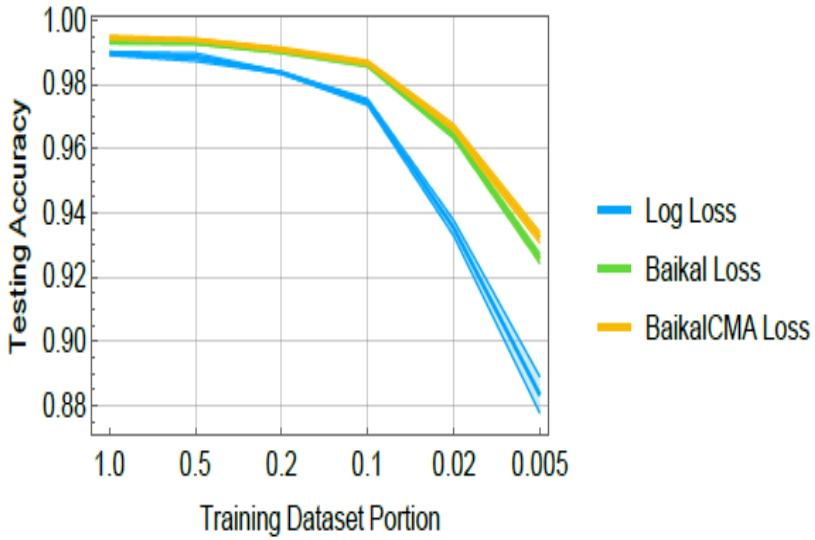


Fig. 4. Sensitivity to different dataset sizes for different loss functions on MNIST. For each size, $n = 5$. Baikal and BaikalCMA increasingly outperform the cross-entropy loss on small datasets, providing evidence of reduced overfitting.

迁移至CIFAR-10的结果

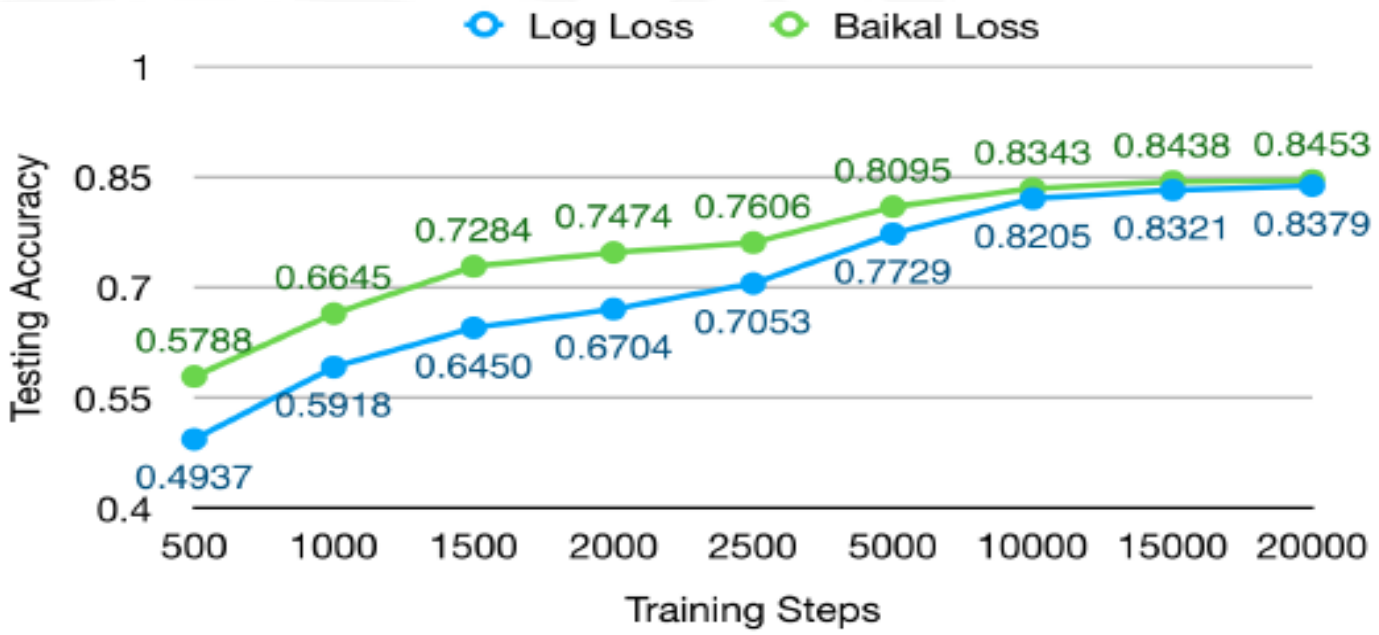


Fig. 5. Testing accuracy across varying training steps on CIFAR-10. The Baikal loss, which has been transferred from MNIST, outperforms the cross-entropy loss on all training durations.

分析：为什么Baikal损失函数更优？

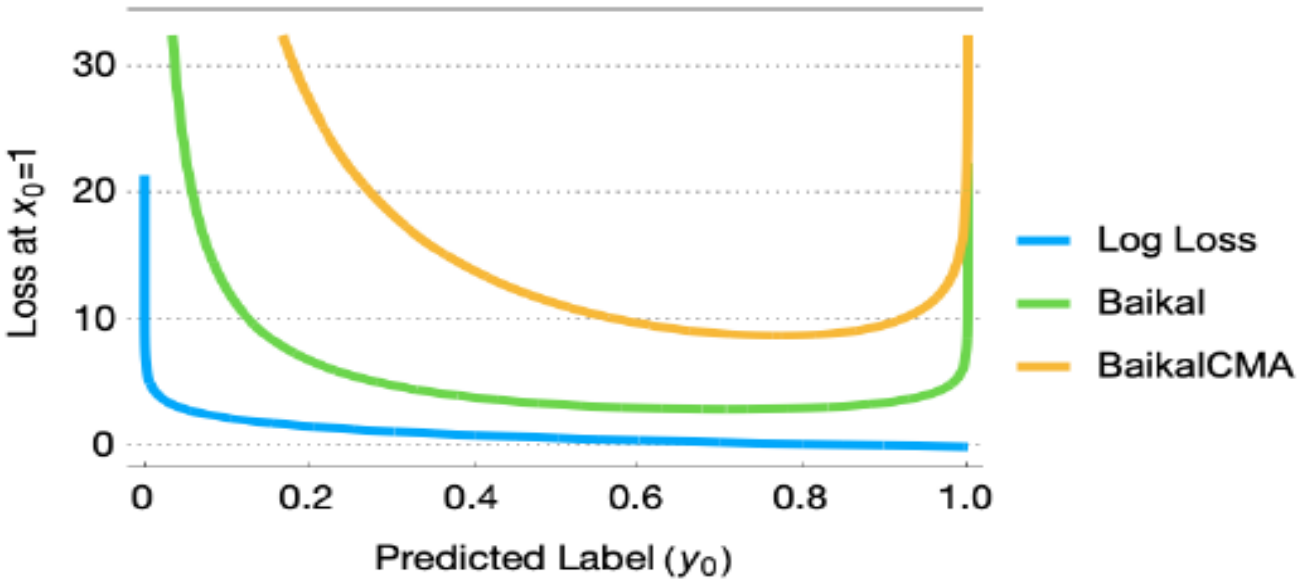


Fig. 6. Binary classification loss functions at $x_0 = 1$. Correct predictions lie on the right side of the graph, and incorrect ones on the left. The log loss decreases monotonically, while Baikal and BaikalCMA present counterintuitive, sharp increases in loss as predictions, approach the true label. This phenomenon provides regularization by preventing the model from being too confident in its predictions.

Semantic Instance Segmentation with a Discriminative Loss Function

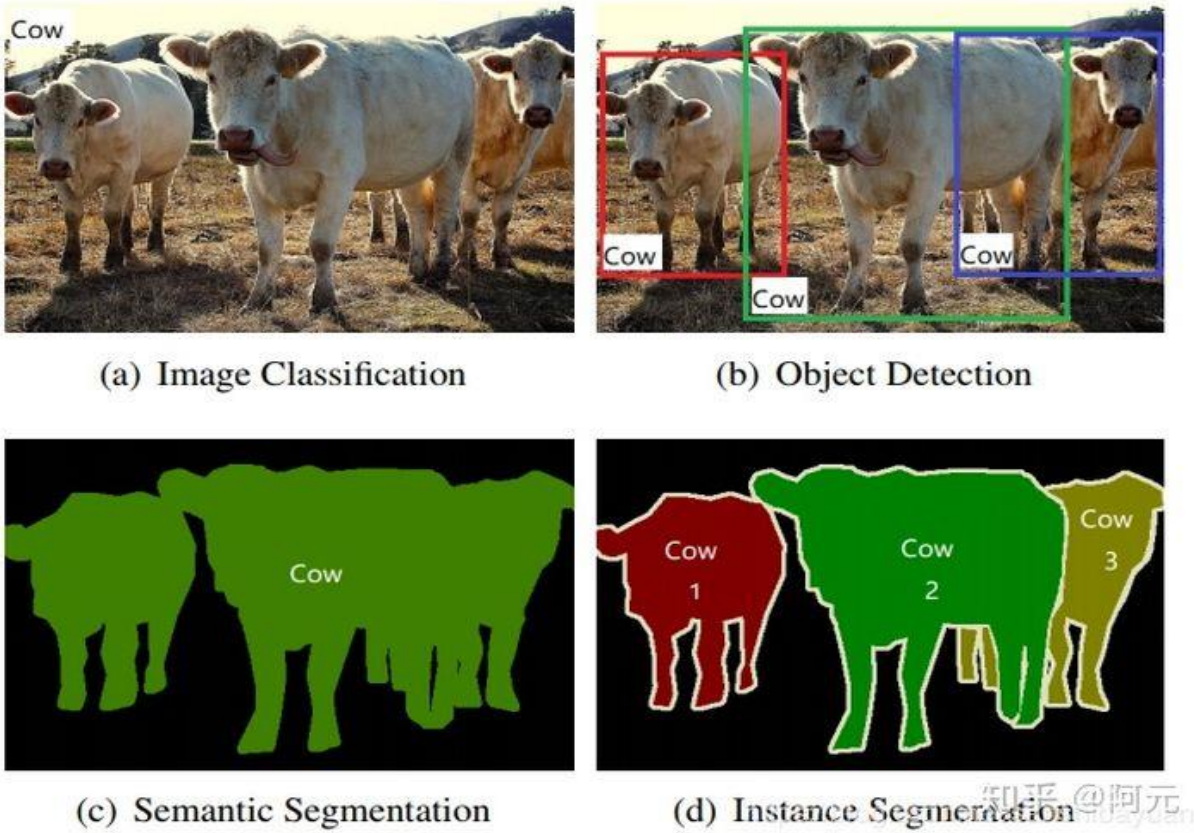
一种基于模糊损失函数的语义实例分割方法

Bert De Brabandere_ Davy Neven_ Luc Van Gool
ESAT-PSI, KU Leuven
firstname.lastname@esat.kuleuven.be

PAPER：https://arxiv.org/abs/1708.02551
CODE：https://github.com/DavyNeven/fastSceneUnderstanding

CVPR 2017

实例分割（Instance Segmentation）



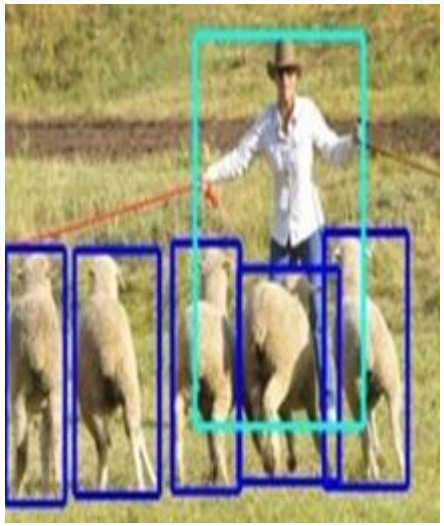
(a) Image Classification (b) Object Detection

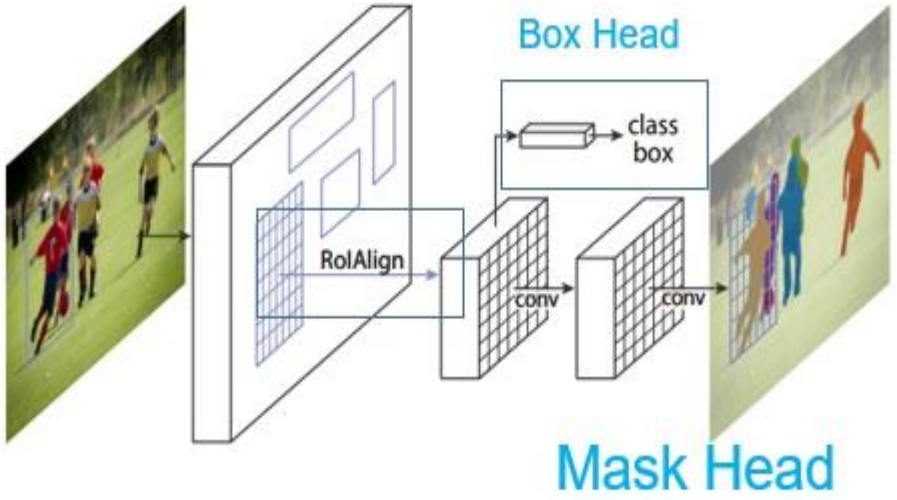
(c) Semantic Segmentation (d) Instance Segmentation

- 自上而下（**Top-Down)**

思路：通过**目标检测**的方法找出实例所在的区域（bounding box），再在检测框内进行**语义分割**，每个分割结果都作为一个不同的**实例**输出。

方法代表：**Mask R-CNN**



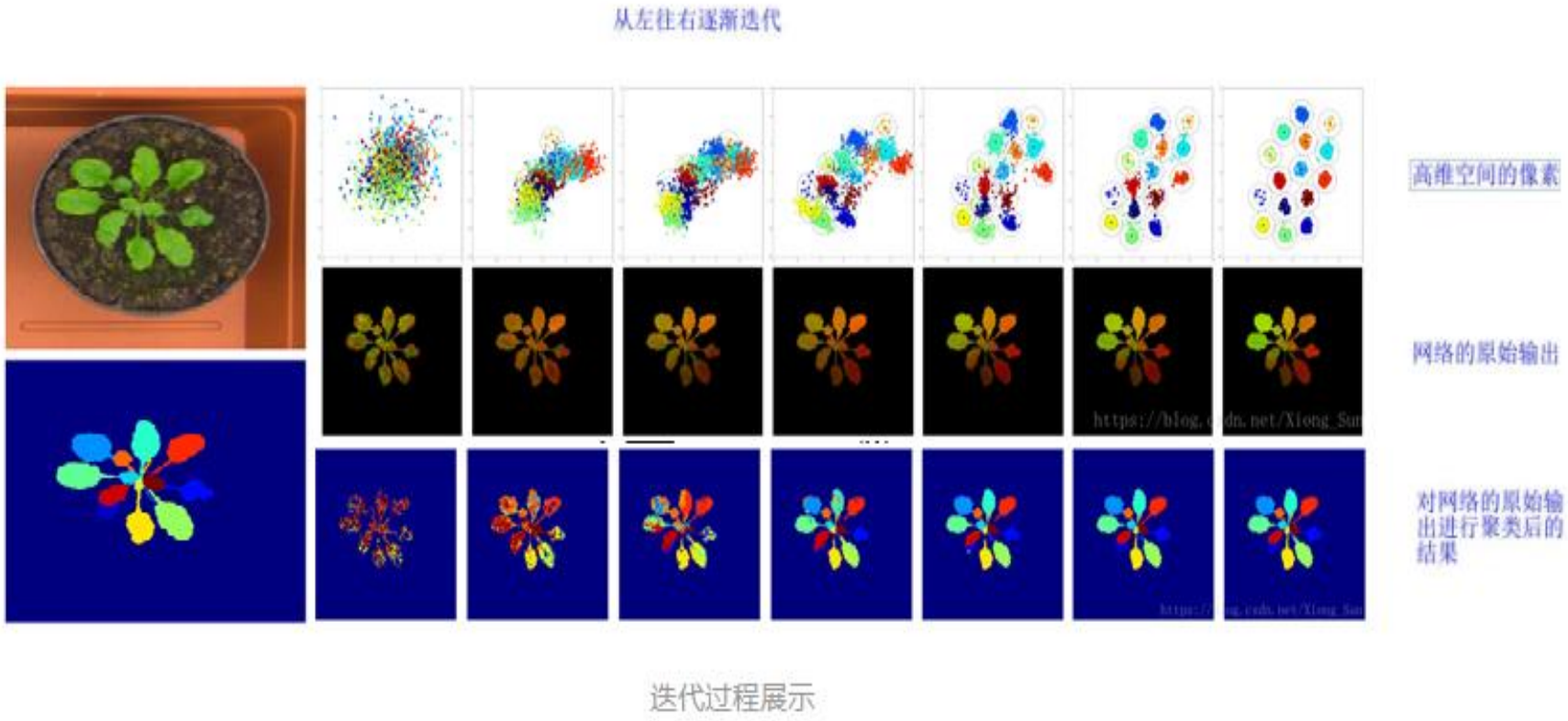


动机：自下而上进行像素级别的**语义分割**，再通过**聚类**、**度量学习**等手段区分不同的实例。

主要贡献：实例嵌入Instance Embedding，通过随机梯度下降使损失最小化。

创新：误差分析。

自下而上（Bottom-Up）：Instance Embedding



判别式损失函数训练网络

$$\mu_c = \frac{1}{N}\sum_{i=1}^{N_c} x_i$$

$$L_{var} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i=1}^{N_c}[\|\mu_c - x_i\| - \delta_{\mathrm{v}}]_+^2 \tag{1}$$

$$L_{dist} = \frac{1}{C(C-1)}\sum_{\substack{c_A=1 \\ c_A\neq c_B}}^{C}\sum_{c_B=1}^{C}[2\delta_{\mathrm{d}} - \|\mu_{c_A} - \mu_{c_B}\|]_+^2 \tag{2}$$

$$L_{reg} = \frac{1}{C}\sum_{c=1}^{C}\|\mu_c\| \tag{3}$$

$$L = \alpha \cdot L_{var} + \beta \cdot L_{dist} + \gamma \cdot L_{reg} \tag{4}$$

In our experiments we set $\alpha = \beta = 1$ and $\gamma = 0.001$.
The loss is minimized by stochastic gradient descent.

实验结果

	SBD	$\lvert DiC \rvert$
RIS+CRF [31]	66.6	1.1
MSU [32]	66.7	2.3
Nottingham [32]	68.3	3.8
Wageningen [44]	71.1	2.2
IPK [25]	74.4	2.6
PRIAn [15]	-	1.3
End-to-end [30]	**84.9**	**0.8**
Ours	84.2	1.0

Table 1. Segmentation and counting performance on the test set of the CVPPP leaf segmentation challenge.

semantic segm.	clustering	AP	AP0.5
resnet38 [40]	mean-shift	21.4	40.2
resnet38 [40]	center threshold	22.9	44.1
ground truth	mean-shift	37.5	58.5
ground truth	center threshold	47.8	77.8

Table 3. Effect of the semantic segmentation and clustering components on the performance of our method on the Cityscapes validation set. We study this by gradually replacing each component with their ground truth counterpart. Row 1 vs row 3: the quality of the semantic segmentation has a big influence on the overall performance. Row 1 vs 2 and row 3 vs 4: the effect of the clustering method is less pronounced but also shows room for improvement.

PolarMask: Single Shot Instance Segmentation with Polar Representation

一种基于极坐标系表示的Single Shot实例分割框架

Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang,
Ding Liang, Chunhua Shen, Ping Luo
The University of Hong Kong Sensetime Group Ltd
Xi'an Jiaotong University， Nanjing University， The University of Adelaide

PAPER：https://arxiv.org/abs/1909.13226
CODE：https://github.com/xieenze/PolarMask

CVPR 2020

动机：提出一种新的极坐标框架PolarMask，基于FCOS（一阶全卷积目标检测），把实例分割统一到了FCN（全卷积网络）的框架下。

创新点：

1. 把两阶段的 ‘先检测再分割’ 实例分割检测框架(如Mask R-CNN)，转换为一阶段框架，同时与目标检测相比计算量没有明显增加。
2. 提出基于极坐标系建模轮廓的实例分割方式，将目标检测的4根射线散发到36根射线。

贡献：与像素级、直角坐标系建模相比的优点。

极坐标建模

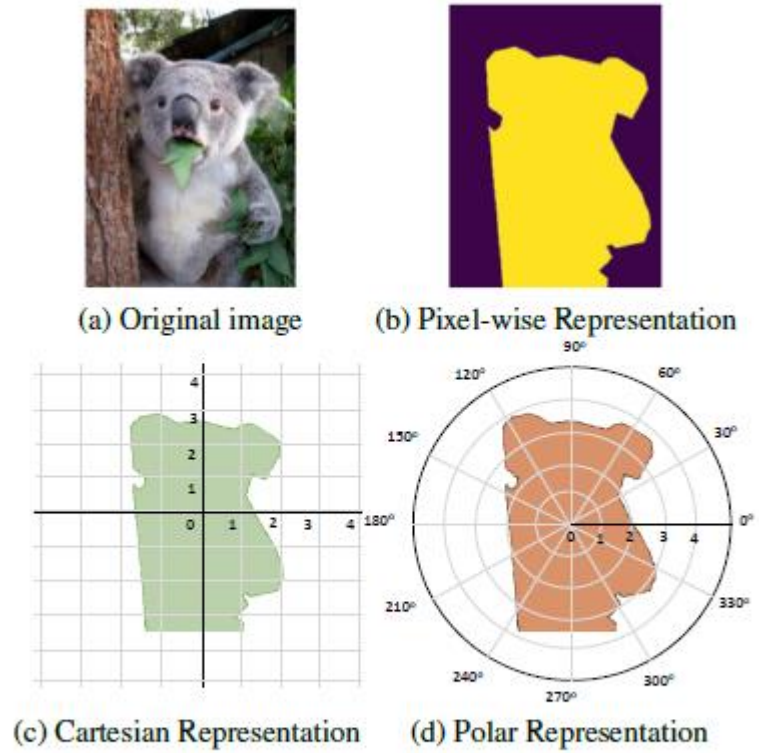


Figure 1 – Instance segmentation with different mask representations. (a) is the original image. (b) is the pixel-wise mask representation. (c) and (d) represent a mask by its contour, in the Cartesian and Polar coordinates, respectively.

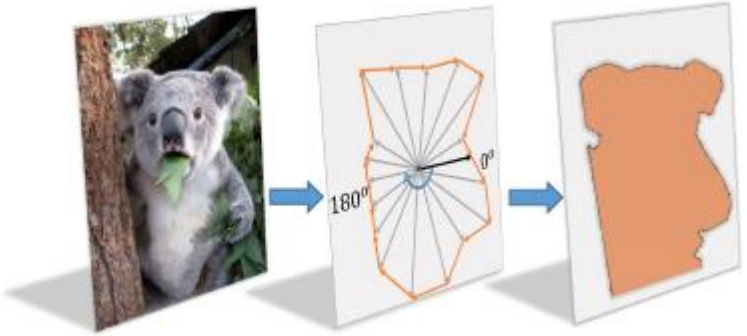


Figure 3 – Mask Assembling. Polar Representation provides a directional angle. The contour points are connected one by one start from 0° (bold line) and assemble the whole contour and mask.

Figure 4 – Polar Centerness. Polar Centerness is used to down-weight such regression tasks as the high diversity of rays' lengths as shown in red lines in the middle plot. These examples are always hard to optimize and produce low-quality masks. During inference, the polar centerness predicted by the network is multiplied to the classification score, thus can down-weight the low-quality masks.

PolarMask网络结构 （Polar CenterNess，Polar IoU Loss）

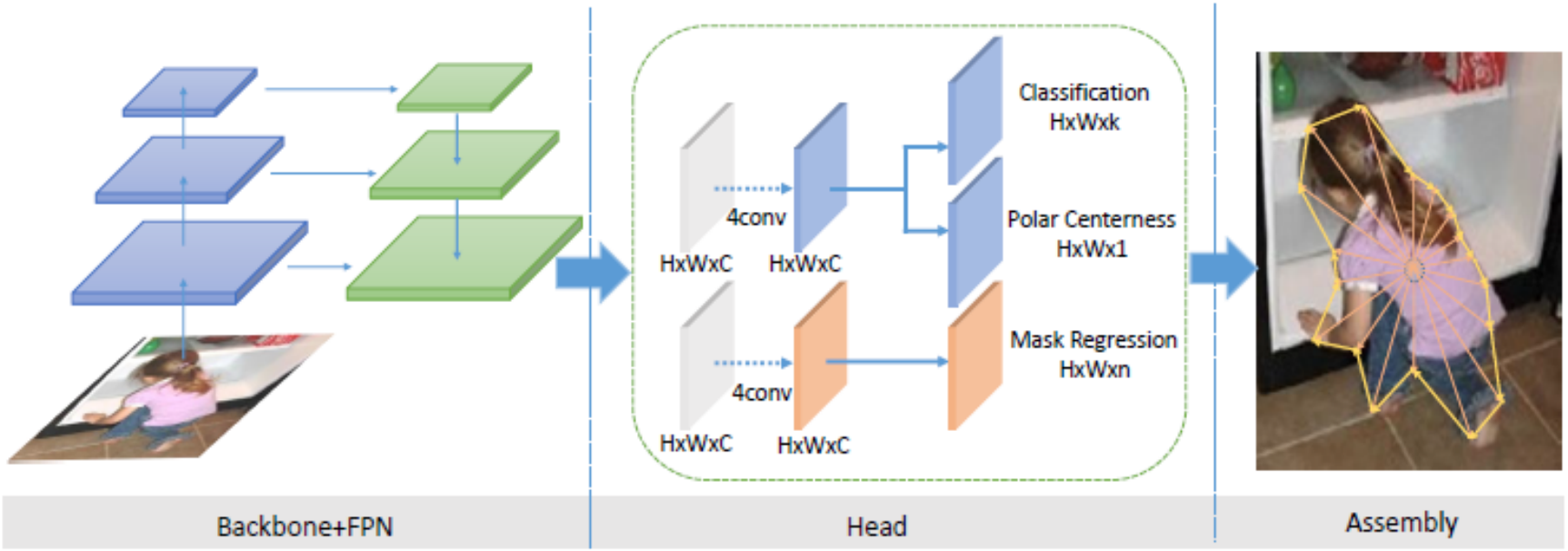


Figure 2 – The overall pipeline of PolarMask. The left part contains the backbone and feature pyramid to extract features of different levels. The middle part is the two heads for classification and polar mask regression. H, W, C are the height, width, channels of feature maps, respectively, and k is the number of categories (e.g., $k = 80$ on the COCO dataset), n is the number of rays (e.g., $n = 36$)

Polar IoU Loss

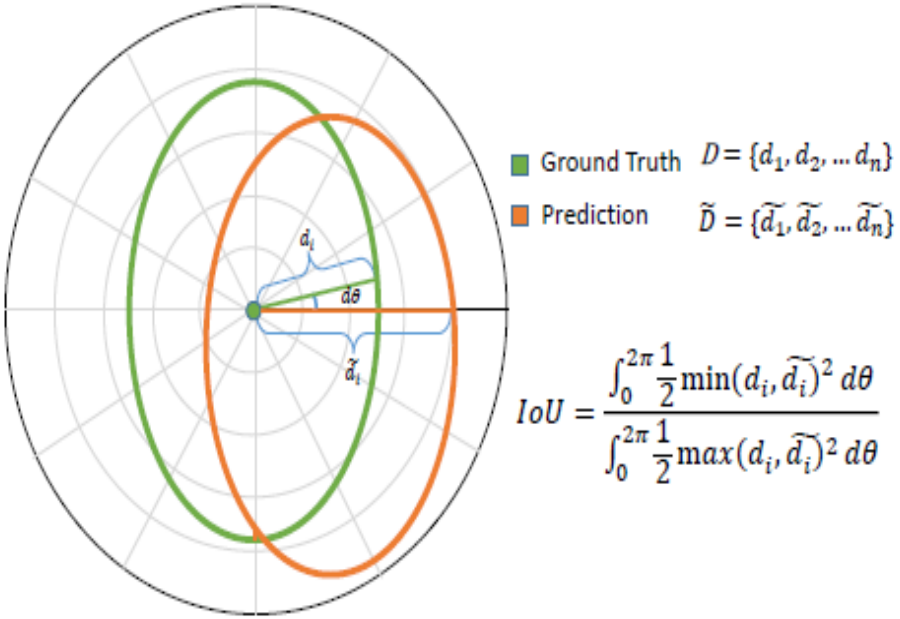


Figure 5 – Mask IoU in Polar Representation. Mask IoU (interaction area over union area) in the polar coordinate can be calculated by integrating the differential IoU area in terms of differential angles.

$$\text{Polar IoU} = \frac{\sum_{i=1}^{n} d_{\min}}{\sum_{i=1}^{n} d_{\max}} \tag{7}$$

Polar IoU Loss is the binary cross entropy (BCE) loss of Polar IoU. Since the optimal IoU is always 1, the loss is actually is negative logarithm of Polar IoU:

$$\text{Polar IoU Loss} = \log \frac{\sum_{i=1}^{n} d_{\max}}{\sum_{i=1}^{n} d_{\min}} \tag{8}$$

实验测试

Figure 6 – Visualization of PolarMask with Smooth-l_1 loss and Polar IoU loss. Polar IoU Loss achieves to regress more accurate contour of instance while Smooth-l_1 Loss exhibits systematic artifacts.

Figure 8 – Results of PolarMask on COCO `test-dev` images with ResNet-101-FPN, achieving 30.4% mask AP (Table 2).

实验结果

method	backbone	epochs	aug	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
two-stage									
MNC [7]	ResNet-101-C4	12	o	24.6	44.3	24.8	4.7	25.9	43.6
FCIS [21]	ResNet-101-C5-dilated	12	o	29.2	49.5	-	7.1	31.3	50.0
Mask R-CNN [15]	ResNeXt-101-FPN	12	o	37.1	60.0	39.4	16.9	39.9	53.5
one-stage									
ExtremeNet [35]	Hourglass-104	100	✓	18.9	44.5	13.7	10.4	20.4	28.3
TensorMask [4]	ResNet-101-FPN	72	✓	37.1	59.3	39.4	17.1	39.1	51.6
YOLACT [2]	ResNet-101-FPN	48	✓	31.2	50.6	32.8	12.1	33.3	47.1
PolarMask	ResNet-101-FPN	12	o	30.4	51.9	31.0	13.4	32.4	42.8
PolarMask	ResNet-101-FPN	24	✓	32.1	53.7	33.1	14.7	33.8	45.3
PolarMask	ResNeXt-101-FPN	12	o	32.9	55.4	33.8	15.5	35.1	46.3
PolarMask	ResNeXt-101-FPN-DCN	24	✓	36.2	59.4	37.7	17.8	37.7	51.5

Table 2 – Instance segmentation mask AP on the COCO `test-dev`. The standard training strategy [14] is training by 12 epochs; and 'aug' means data augmentation, including multi-scale and random crop. ✓ is training with 'aug', o is without 'aug'.

Graphical Contrastive Losses for Scene Graph Parsing

场景图解析：对比损失

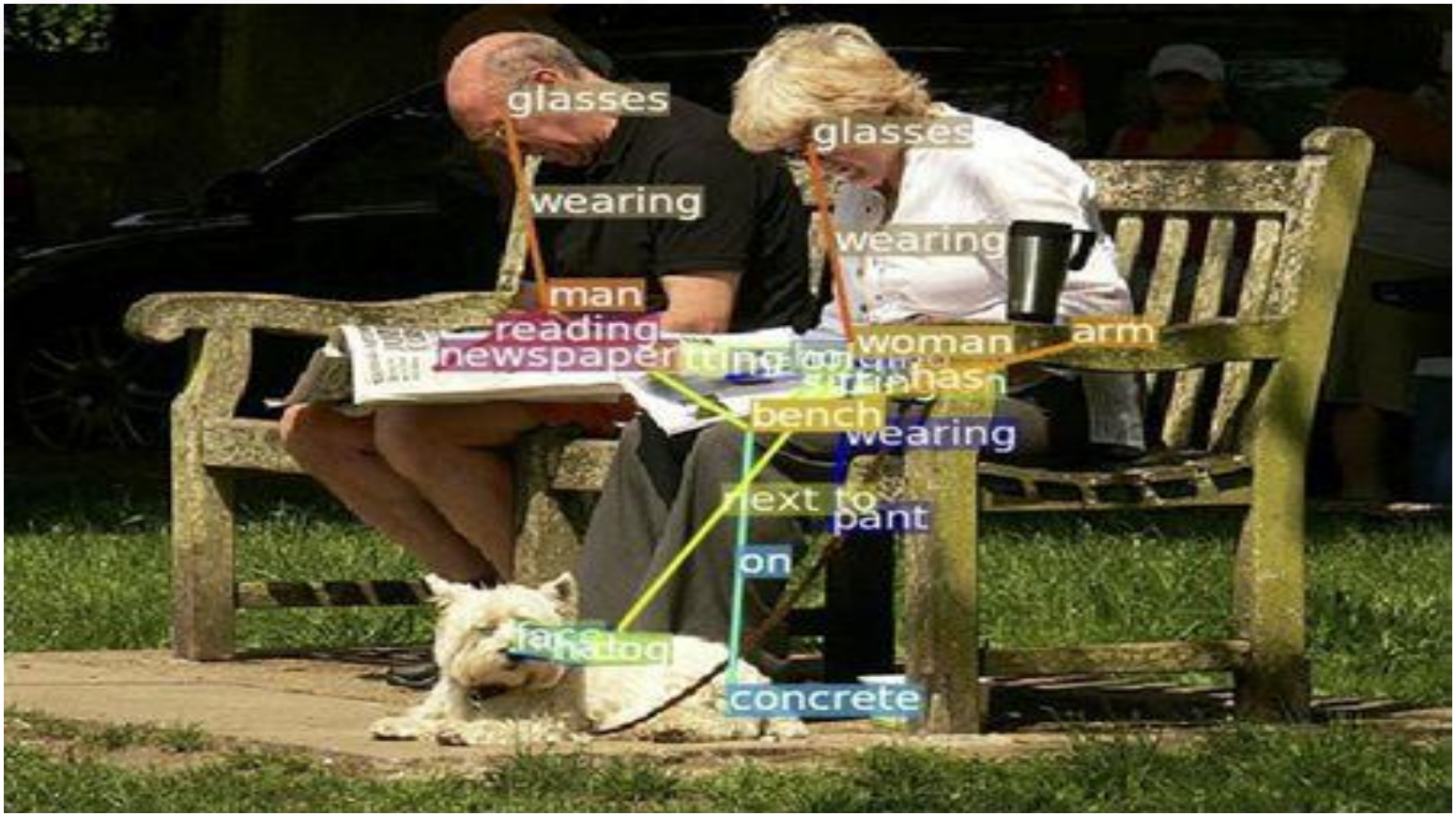
罗格斯大学计算机系博士生——张骥

- 论文链接：https://arxiv.org/abs/1903.02728
- 代码链接：https://github.com/NVIDIA/ContrastiveLosses4VRD

Published as a conference paper at CVPR2019

背景

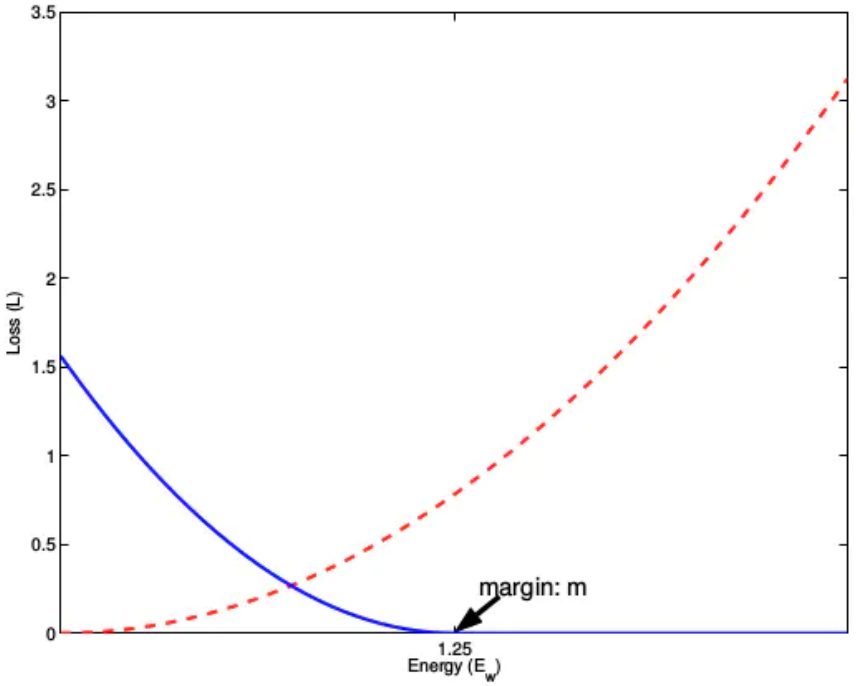
把节点(也就是物体)探测出来不是一个难点，真正的难点在于构建图中的边，也就是物体之间的视觉关系(visual relationship)。



介绍

背景：当两个样本标签相同，即相似y=1，则欧式距离越大损失函数越大，反之，欧式距离越小，损失函数越小；当两样本不相似时y=0，只有后一项，若欧式距离越小反而loss越大；欧式距离越大loss越小。

动机：观察到了边之间存在两种重要的联系，进而针对性地提出三种损失函数来协同地预测视觉关系。

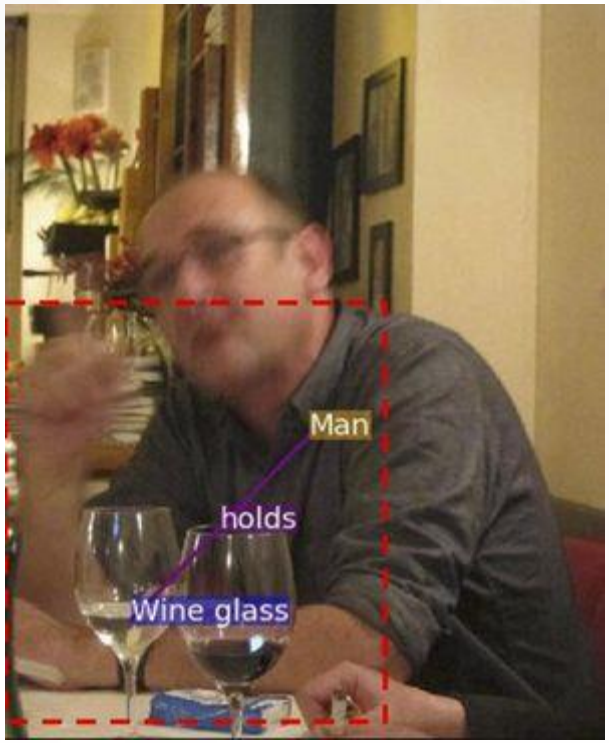


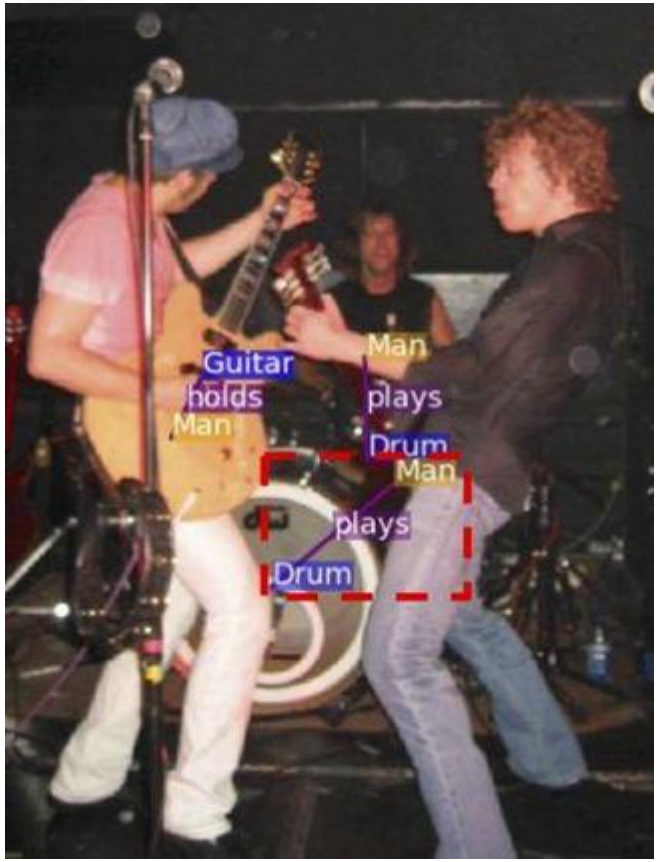
这张图表示的就是损失函数值与样本特征的欧式距离之间的关系，其中红色虚线表示的是相似样本的损失值，蓝色实线表示的不相似样本的损失值。

问题

问题一：客体实例混淆

问题二：邻近关系模糊





解决方案：

三种损失函数：

1. 类别无关损失（**Class AgnosticLoss**）
2. 物体类别相关损失（**Entity Class Aware Loss**）
3. 谓语类别相关损失（**Predicate Class Aware Loss**）

类别无关损失（Class AgnosticLoss）及其目的

该函数的计算分成两步，第一步计算正负样本的置信度差异：

$$m_1^s(i) = \min_{j \in \mathcal{V}_i^+} \Phi(s_i, o_j^+) - \max_{k \in \mathcal{V}_i^-} \Phi(s_i, o_k^-)$$

$$m_1^o(j) = \min_{i \in \mathcal{V}_j^+} \Phi(s_i^+, o_j) - \max_{k \in \mathcal{V}_j^-} \Phi(s_k^-, o_j)$$

第二步是利用第一步的两个差异值来计算一个基于边界的损失：

$$L_1 = \frac{1}{N} \sum_{i=1}^{N} \max(0, \alpha_1 - m_1^s(i)) + \frac{1}{N} \sum_{j=1}^{N} \max(0, \alpha_1 - m_1^o(j))$$

物体类别相关损失（Entity Class Aware Loss）及其目的

$$m_2^s(i,c) = \min_{j \in \mathcal{V}_i^{c+}} \Phi(s_i, o_j^+) - \max_{k \in \mathcal{V}_i^{c-}} \Phi(s_i, o_k^-)$$

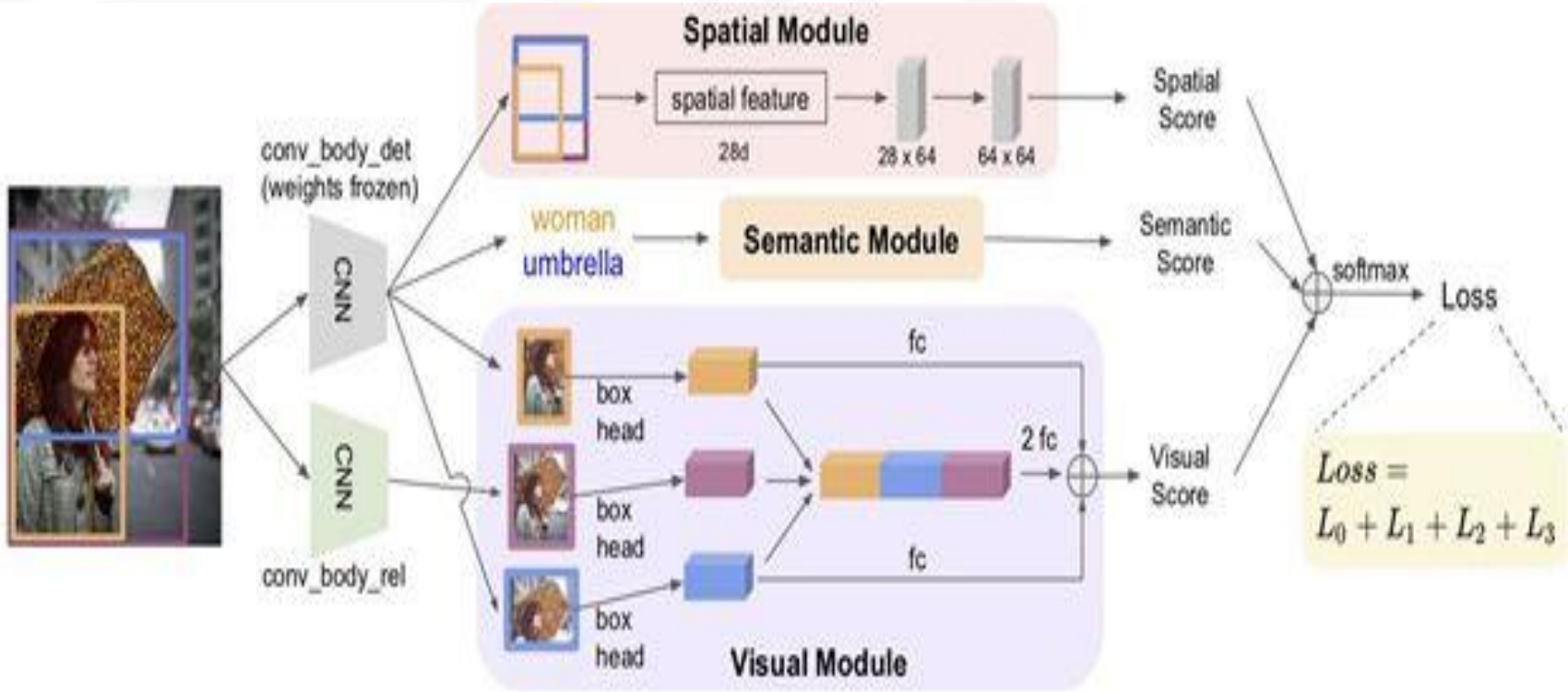
$$m_2^o(j,c) = \min_{i \in \mathcal{V}_j^{c+}} \Phi(s_i^+, o_j) - \max_{k \in \mathcal{V}_j^{c-}} \Phi(s_k^-, o_j)$$

谓语类别相关损失（Predicate Class Aware Loss）及其目的

$$m_3^s(i,e) = \min_{j \in \mathcal{V}_i^{e+}} \Phi(s_i, o_j^+) - \max_{k \in \mathcal{V}_i^{e-}} \Phi(s_i, o_k^-)$$

$$m_3^o(j,e) = \min_{i \in \mathcal{V}_j^{e+}} \Phi(s_i^+, o_j) - \max_{k \in \mathcal{V}_j^{e-}} \Phi(s_k^-, o_j)$$

关系检测网络（RelDN）



实验结果

L_0	L_1	L_2	L_3	R@50	wmAP_{rel}	wmAP_{phr}	score_{wtd}	AP_{rel} per class								
								at	on	holds	plays	interacts_with	wears	inside_of	*under*	*hits*
✓				74.67	34.63	37.89	43.94	32.40	36.51	41.84	36.04	40.43	5.70	44.17	*25.00*	*55.40*
✓	✓			75.06	35.25	38.37	44.46	32.78	36.96	42.93	37.55	43.30	**9.01**	44.15	*100.00*	*50.95*
✓		✓		74.64	35.03	38.18	44.21	32.76	36.82	42.24	37.17	40.47	8.53	44.71	*33.33*	*49.68*
✓			✓	74.88	35.19	38.27	44.36	32.88	36.73	42.38	38.03	43.53	6.71	44.18	*16.67*	*52.06*
✓	✓	✓		75.03	35.38	38.50	44.56	**32.95**	**37.10**	42.82	38.58	43.66	6.79	43.72	*20.00*	*50.24*
✓	✓		✓	**75.30**	35.30	38.27	44.49	32.92	36.73	42.58	38.81	44.13	6.35	42.74	*100.00*	*51.40*
✓		✓	✓	75.00	35.12	38.34	44.39	32.79	36.47	42.31	39.74	41.35	6.11	43.57	*25.00*	*55.12*
✓	✓	✓	✓	74.94	**35.54**	**38.52**	**44.61**	32.92	37.00	**43.09**	**41.04**	**44.16**	7.83	**44.72**	*50.00*	*51.04*

Table 1: Ablation Study on our losses. We report a frequency-balanced wmAP instead of mAP, as the test set is extremely imbalanced and would fluctuate wildly otherwise (see fluctuations in columns "under" and "hits"). We also report score_{wtd}, which is the official OI scoring formula but with wmAP in place of mAP. "Under" and "hits" are not highlighted due to having too few instances.

	R@50	wmAP_{rel}	wmAP_{phr}	score_{wtd}	AP_{rel} per class					AP_{phr} per class				
					at	holds	plays	interacts_with	wears	at	holds	plays	interacts_with	wears
L_0	61.72	25.80	33.15	35.92	14.77	26.34	42.51	21.33	21.03	21.76	35.88	**48.57**	38.74	31.92
$L_0 + L_1 + L_2 + L_3$	**62.65**	**27.37**	**34.58**	**37.31**	**16.18**	**30.39**	**42.73**	**22.40**	**22.14**	**22.67**	**39.60**	48.09	**40.96**	**32.64**

Table 2: Comparison of our model with Graphical Contrastive Loss *vs.* without the loss on 100 images containing the 5 classes that suffer from the two aforementioned confusions, selected via visual inspection on a random set of images.

实验结果（ VRD数据集）

	Relationship		Phrase		Relationship Detection						Phrase Detection					
	free k				k = 1		k = 10		k = 70		k = 1		k = 10		k = 70	
Recall at	50	100	50	100	50	100	50	100	50	100	50	100	50	100	50	100
PPRFCN*[39]	14.41	15.72	19.62	23.75	-	-	-	-	-	-	-	-	-	-	-	-
VTransE*	14.07	15.20	19.42	22.42	-	-	-	-	-	-	-	-	-	-	-	-
SA-Full*[22]	15.80	17.10	17.90	19.50	-	-	-	-	-	-	-	-	-	-	-	-
DR-Net*[3]	17.73	20.88	19.93	23.45	-	-	-	-	-	-	-	-	-	-	-	-
ViP-CNN[11]	17.32	20.01	22.78	27.91	17.32	20.01	-	-	-	-	22.78	27.91	-	-	-	-
VRL[12]	18.19	20.79	21.37	22.60	18.19	20.79	-	-	-	-	21.37	22.60	-	-	-	-
CAI*[43]	20.14	23.39	23.88	25.26	-	-	-	-	-	-	-	-	-	-	-	-
KL distilation[35]	22.68	31.89	26.47	29.76	19.17	21.34	22.56	29.89	22.68	31.89	23.14	24.03	26.47	29.76	26.32	29.43
Zoom-Net[32]	21.37	27.30	29.05	37.34	18.92	21.41	-	-	21.37	27.30	24.82	28.09	-	-	29.05	37.34
CAI + SCA-M[32]	22.34	28.52	29.64	38.39	19.54	22.39	-	-	22.34	28.52	25.21	28.89	-	-	29.64	38.39
RelDN, L_0 only (ImageNet)	21.62	26.12	28.59	35.18	19.57	22.61	21.62	26.12	21.62	26.12	26.39	31.28	28.59	35.18	28.59	35.18
RelDN (ImageNet)	21.52	26.38	28.24	35.44	19.82	22.96	21.52	26.38	21.52	26.38	26.37	31.42	28.24	35.44	28.24	35.44
RelDN, L_0 only (COCO)	26.67	32.55	33.29	41.25	24.30	27.91	26.67	32.55	26.67	32.55	31.09	**36.42**	33.29	41.25	33.29	41.25
RelDN (COCO)	**28.15**	**33.91**	**34.45**	**42.12**	**25.29**	**28.62**	**28.15**	**33.91**	**28.15**	**33.91**	**31.34**	**36.42**	**34.45**	**42.12**	**34.45**	**42.12**

Table 8: Comparison with state-of-the-art on VRD (– means unavailable / unknown). Same with Table 7, L_0 only is the RelDN without our losses. "Free k" means considering k as a hyper-parameter that can be cross-validated.

在VRD数据集上我们看到带losses和不带losses的差别相比VG明显了很多，这是因为VRD的标注相对较完整，整体标注质量也相对较好。

实验效果

L_0 only | with our losses | L_0 only | with our losses

(a) Entity Instance Confusion (b) Proximal Relationship Ambiguity

Figure 5: Example results of RelDN with L_0 only and with our losses. The top row shows RelDN outputs and the bottom row visualizes the learned predicate CNN features of the two models. Red and green boxes highlight the wrong and right outputs (the first row) or feature saliency (the second row). As it shows, our losses force the model to attend to the representative regions that discriminate the correct relationships against unrelated entity pairs, thus is able to disentangle entity instance confusion and proximal relationship ambiguity.

思考及展望

1. 细分谓词：在实际应用中，一个可能更好的构建场景图的方式是把所有谓词分成若干大类，然后每个大类分别用一个模型去学。

2. 空间关系是最难识别的一类，因为相比而言，同一个空间关系所对应的视觉分布要复杂很多。

3. Language Bias（Semantic Module）是符合客观世界分布的，但有其局限性。

Scene Dynamics: Counterfactual Critic Multi-Agent Training for Scene Graph Generation

出自南洋理工大学张含望老师小组， Long Chen, Hanwang Zhang, Jun Xiao, Xiangnan He, Shiliang Pu, Shih-Fu Chang

文章链接：https://arxiv.org/pdf/1805.00065.pdf

ICCV **2019**

背景和动机

目前的关系检测算法大都没有将目标的检测放在graph的层次上思考，但还是没有充分利用graph的特性。

对于目标的识别，现有的算法都是使用每个object的交叉熵之和作为损失函数，但**问题在于这样的前提是认为每个object的重要性是一样的，其实不是的**。

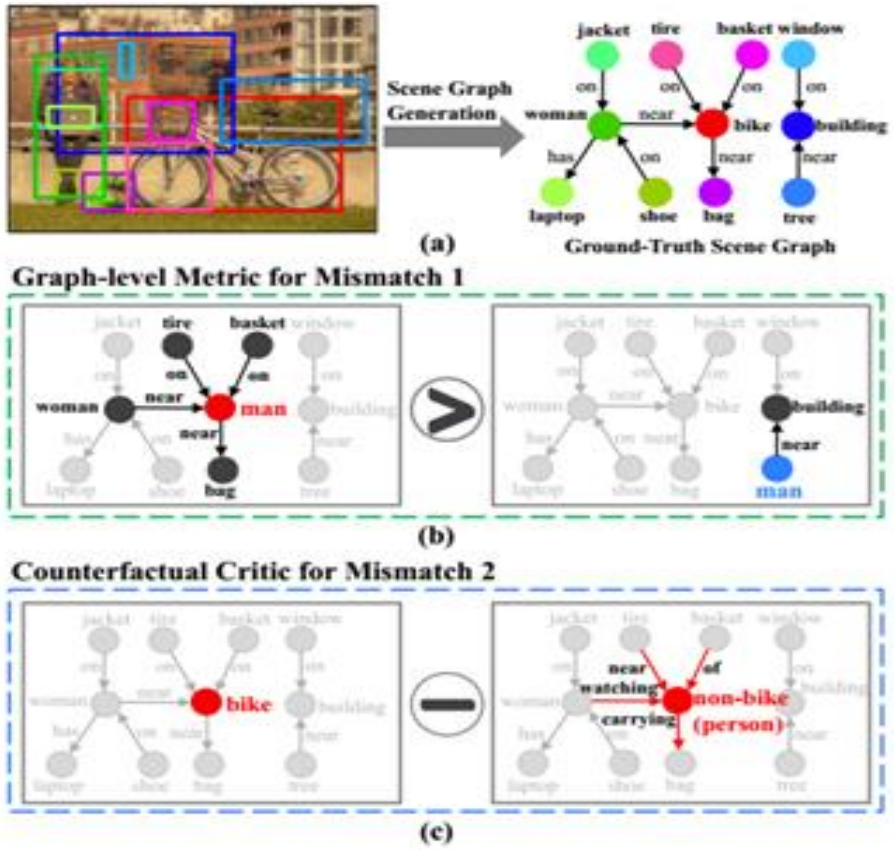


Figure 1. (a) An input image and its ground-truth scene graph, the color of each scene graph node is consistent with the bounding boxes. (b) For mismatch 1, a graph-level metric will penalize the red node more than the blue one, even though both are misclassified as the same object "man". (c) For mismatch 2, the individual reward won by the prediction "bike" of the red node can be identified by excluding the reward from "non-bike" predictions.

创新点

以前在训练集中对每种物体的关系进行统计，找到那些hub node，然后构建损失函数的时候，给它们更大的权重。但本文不是这样简单地的做的，**文章采用graph-level的metric来对检测结果进行评价，如Recall和SPICE**

$$\text{Recall@K} = \#\left[(v_i^T, r_{ij}, v_j^T) == (v_i^{gt}, r_{ij}^{gt}, v_j^{gt})\right]/K.$$

$$\nabla_\theta J \approx \sum_{i=1}^{n} \nabla_\theta \log \pi_i^t(a_i^t|s_i^t;\theta) Q(S^t, A^t)$$

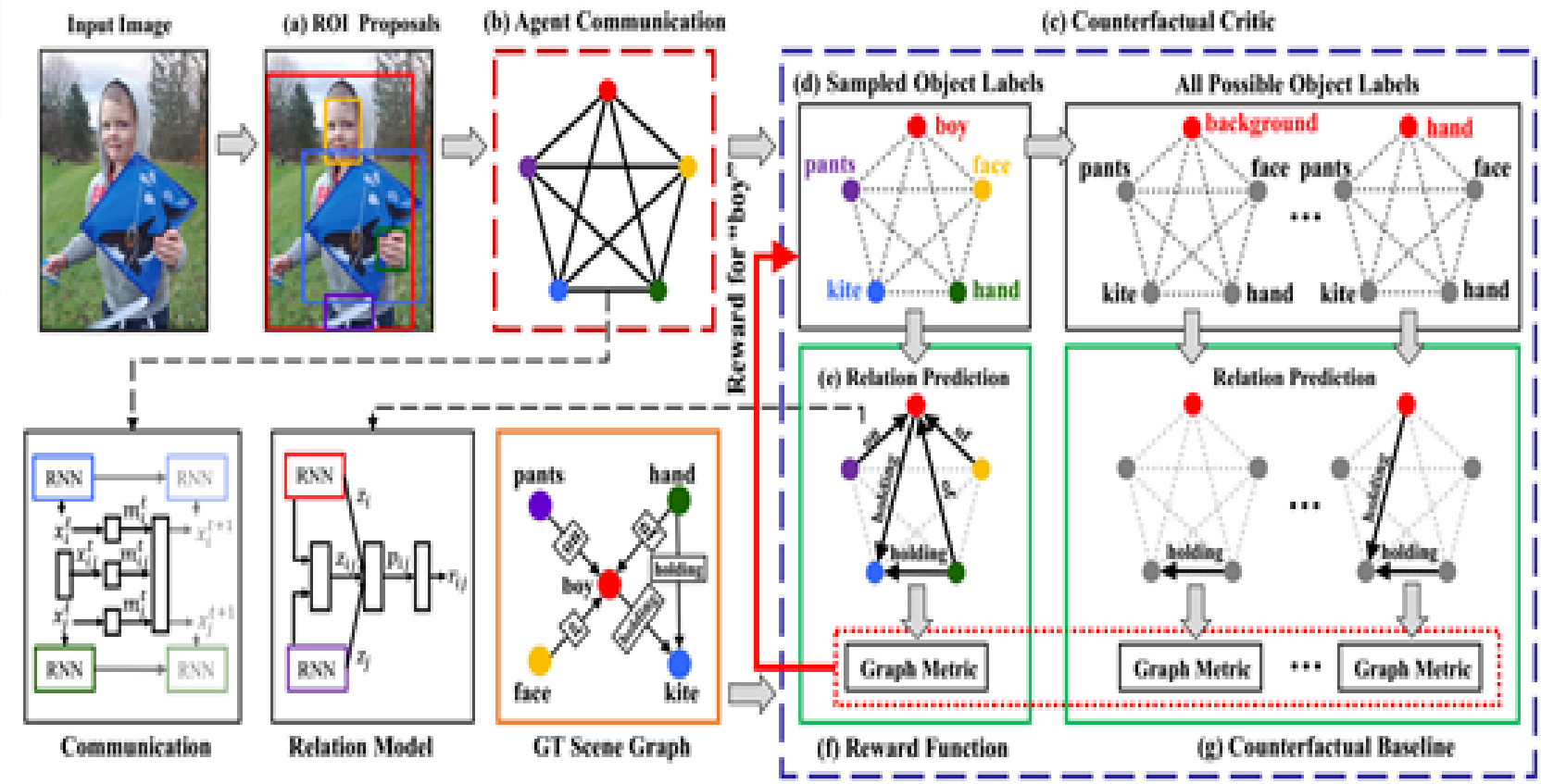
$$\nabla_\theta J \approx \sum_{i=1}^{n} \nabla_\theta \log \boldsymbol{p}_i^T(v_i^T|h_i^T;\theta) R(H^T, \mathcal{V}^T)$$

$$A^i(H^T, \mathcal{V}^T) = R(H^T, \mathcal{V}^T) - \sum \boldsymbol{p}_i^T(\tilde{v}_i^T) R(H^T, (\mathcal{V}_{-i}^T, \tilde{v}_i^T))$$

$$\nabla_\theta J \approx \sum_{i=1}^{n} \nabla_\theta \log \boldsymbol{p}_i^T(v_i^T|h_i^T;\theta) A^i(H^T, \mathcal{V}^T)$$

过程

对于每一个object，将其类别的判断替换成其它的类别，并将每次得到的graph-level metric按分类的概率加权求和。



最终公式：

$$\nabla_\theta J \approx \overbrace{\sum_{i=1}^{n} \nabla_\theta \log \boldsymbol{p}_i^T(v_i^T|h_i^T;\theta) A^i(H^T,\mathcal{V}^T)}^{\text{CMAT}} +$$

$$\underbrace{\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \nabla_\theta \log \boldsymbol{p}_{ij}(r_{ij})}_{\text{XE for relationships}} + \underbrace{\alpha \sum_{i=1}^{n} \nabla_\theta \log \boldsymbol{p}_i^T(v_i^T)}_{\text{XE for objects}},$$

训练细节

1.采用两阶段的训练方法，第一阶段先按照普通的关系检测算法，使用交叉熵训练，之后再加入CMAT进行训练。

2.计算CMAT时，对于每个object如果都将其他类别替换一遍，计算量非常大，因此对于每个object，作者做了实验之后，发现只替换两个概率次大的类别以及background类别，能带来70x的速度提升，而性能损失非常少。

实验结果

		Scene Graph Detection			Scene Graph Classification			Predicate Classification		
	Model	R@20	R@50	R@100	R@20	R@50	R@100	R@20	R@50	R@100
Indep.	VRD [29]	-	0.3	0.5	-	11.8	14.1	-	27.9	35.0
	AssocEmbed [31]	6.5	8.1	8.2	18.2	21.8	22.6	47.9	54.1	55.4
	FREQ [54]	20.1	26.2	30.1	29.3	32.3	32.9	53.6	60.6	62.2
Joint Inference	MSDN* [23]	-	7.0	9.1	-	27.6	29.9	-	53.2	57.9
	IMP† [46]	14.6	20.7	24.5	31.7	34.6	35.4	52.7	59.3	61.3
	TFR [14]	3.4	4.8	6.0	19.6	24.3	26.6	40.1	51.9	58.3
	MOTIFS [54]	21.4	27.2	30.3	32.9	35.8	36.5	58.5	65.2	67.1
	Graph-RCNN [48]	-	11.4	13.7	-	29.6	31.6	-	54.2	59.1
	PISP‡ [12]	-	-	-	-	36.5	38.8	-	65.1	66.9
	CMAT	**22.1**	**27.9**	**31.2**	**35.5**	**38.5**	**39.3**	**60.2**	**66.4**	**68.1**

Table 1. Performance compared with the state-of-the-art methods on the Visual Genome [19] dataset. *: [48] reimplemented MSDN using the same dataset split for fair comparison. †: [54] reimplemented IMP using the same object detection backbone for fair comparison. ‡: PISP is not trained from scratch as it takes the MOTIFS [54] output as input.

总结

后两篇文章侧重点不同：图解析、图生成。
两篇文章相同点：都使用了谓语细分，都使用了VRD数据集。

使用高效的Loss Function对图进行操作：目标检测、生成解析、性能优化等。

效率和精确性上有待提升，还有很大的潜力可挖。

Thanks