An Efficient Pyramid Network for refined edge detection with or without semantic category awareness

TV. Pham[1,2] S. Treuillet[1] Y. Lucas[1] L. Debraux[2]

[1] PRISME, Polytech Orléans Site Galilée, 12 rue de Blois, BP 6744, 45067 Orléans cedex 2, France
[2] Dental Monitoring, 75 rue de Tocqueville, 75017 Paris, France

{p.vantrung, l.debraux}@dental-monitoring.com, {sylvie.treuillet,yves.lucas}@univ-orleans.fr

Abstract

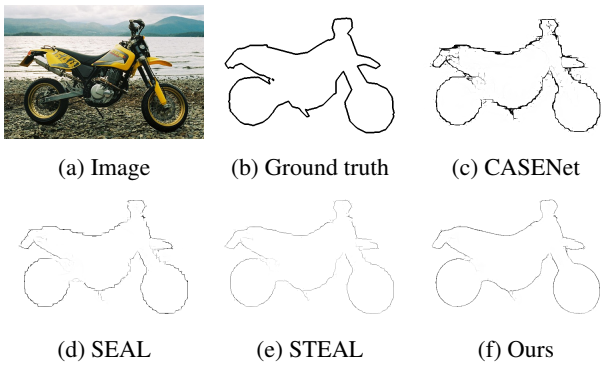
Category-aware semantic edge detection can be seen as an extension of edge detection where edge maps distinguish categories of objects in the scene. Obtaining both precise shape and well categorized boundaries is a challenging double task. This paper presents an improved architecture for semantic edge detection called Edge Pyramid Network (EPN), which effectively combines low-level and high-level features to produce smoother edges. Observing the impact of misalignment and loss functions on training effectiveness, we propose a fine-tuning strategy which incrementally improves edge detection accuracy through alternated training cycles using the weighted negative and the unweighted loss functions. Before applying the fine-tuning strategy, our network already outperforms state-of-the-art semantic edge detection networks on the benchmark datasets SBD and Cityscapes. The performance is further improved using our fine-tuning strategy (+0.85% and +1.78% compared to STEAL on SBD and Cityscapes datasets, respectively). Furthermore, we show that EPN architecture achieves competitive performance (ODS F-measure of .830) against the state-of-the-art category-agnostic edge detection networks on the BSDS500 dataset.

Keywords

Semantic Edge Detection, Feature Pyramid Network, Fine-tuning Strategy.

1 Introduction

For some vision applications, it is necessary to make a very precise edge detection and a semantic classification as well. Category-aware semantic edge detection can be seen as an extension of edge detection where edge maps distinguish categories of objects in the scene. This is a multi-label problem more difficult to solve than binary edge detection in itself. Recently, significant progress has been made with deep learning and CASENet [3] was presented as an effective architecture to achieve both tasks in a single end-to-end deep network. But obtaining both precise shape contours and well categorized contours is a challenging double task due to two major problems : first, misalignment and noise in human annotated edges in the available data sets, secondly imbalance in number of pixels between the class labelled "edge" and the class labelled "non-edge". This latter problem is usually addressed using a weighted cross entropy loss in network training, while the first requires correcting label noise during training. However, we can observe a pernicious side effect in Fig. 1. : the detected edges are thicker than the ground truth ones, i.e. more false positive edges are produced.



(a) Image (b) Ground truth (c) CASENet

(d) SEAL (e) STEAL (f) Ours

FIGURE 1 – From left to right : original image, ground truth, CASENet [3] + NMS edge-thinning, SEAL [4], STEAL [5] and the proposed EPN.

This paper presents an improved architecture for semantic edge detection called Edge Pyramid Network (EPN), which effectively combines low-level and high-level features. Observing that the accuracy of semantic edge detection can be significantly improved by replacing the weighted cross-entropy loss by the unweighted one, we propose a new strategy for fine-tuning to incrementally improve edge detection accuracy and provide thin edges by alternating training cycles with various loss functions, including a weighted-negative loss to address the misalignment. The rest of the paper is organized as follows : section 2 reviews the related work, the architecture design and training strategy of our network are presented in section 3 and experiments and accuracy evaluation against the state-of-the-art algorithms on popular benchmark datasets such as SBD,

Cityscapes and BSDS500 are given in section 4, before to conclude.

2 Related Work

Semantic Edge Detection. Early algorithms address the semantic edge detection task by multi-stages systems. Hariharan *et al.* proposed inverse detectors built on the output of bottom-up contour and object detectors [1]. Bertasius *et al.* introduced a high-for-low approach to predict object boundaries using high-level object from semantic segmentation [13]. Xie *et al.* [7] used a multi-scale and multi-level feature for a rich hierarchical representation learning in their holistically-nested edge detection network (HED). Inspired by the latter, Yu *et al.* [3] developed CASENet, an end-to-end semantic edge detection network, where semantic final edge maps use shared low-level edges extracted from bottom layers of deep convolutional networks VGG [9] and ResNet [10]. A multi-label loss function is proposed to supervise the category-wise edge fused activations. By analyzing the fusion way of CASENet, Hu *et al.* proposed a dynamic feature fusion strategy to learn better weights for each location of the feature map [6]. Recent works argued that misalignment and noise in human annotated edges in the available data sets can degrade edge accuracy and propose to apply correcting strategies during CASENet training. Yu *et al.* considered the edge alignment as latent variable optimization in a probabilistic model to propose SEAL algorithm [4]. Inspired by Non-Maximal Suppression NMS edge-thinning algorithm [17], Acuna *et al.* proposed to add a boundary thinning layer and an active alignment scheme that can be plugged on top of any boundary detection architecture [5].

Weighted and unweighted cross-entropy loss. The imbalance in number of pixels between the classes is usually addressed using a weighted loss function in network training [7] [14] [15] [8] [12] [3]. In HED network, a class-balancing weight is introduced to automatically boost and reduce the positive and negative losses, respectively [7]. Without this weight, HED failed to detect salient edges by producing an embossed edge image, as shown in [12]. In contrast, using unweighted cross-entropy loss for the category-aware edge detection task can make a significant improvement of accuracy, as described in [4]. However, fine-tuning a network using unweighted cross-entropy loss from a model pre-trained on the general image classification task as ImageNet [29] is not trivial, since the loss at the beginning is very large. To address this issue, SEAL [4] used a warm-up model obtained by additionally training a few iterations from an initialized model.

Low-level and high-level features combination. Weighted-fusion layer is a popular technique to combine low-level and high-level side outputs in category-agnostic edge detection networks [7] [14] [8]. This fusion way was then extended in semantic edge detection networks [3] [4] [5] by sharing a concatenation layer where the semantic edge maps use the same low-level edges from bottom layers. Feature Pyramid Network (FPN) [11] provides a relevant way to build image features from all levels via a top-down pathway and lateral connections. This architecture is widely used as a feature extractor in many computer vision applications such as object detection [16] [18], instance segmentation [19] or semantic segmentation [20]. The most advanced work discussed here has inspired our Edge Pyramid Network (EPN) architecture. It integrates but simplifies ResNet and FPN to produce smoother edges. Observing the impact of misalignment and loss functions on training effectiveness, we have developed the new fine-tuning strategy described below to obtain a more precise and finer edge map.

3 Edge Pyramid Network

As previous category-aware edge detection networks [3] [6] [4] [5], the proposed network, Edge Pyramid Network (EPN), produces multiple edge maps corresponding to different object categories.

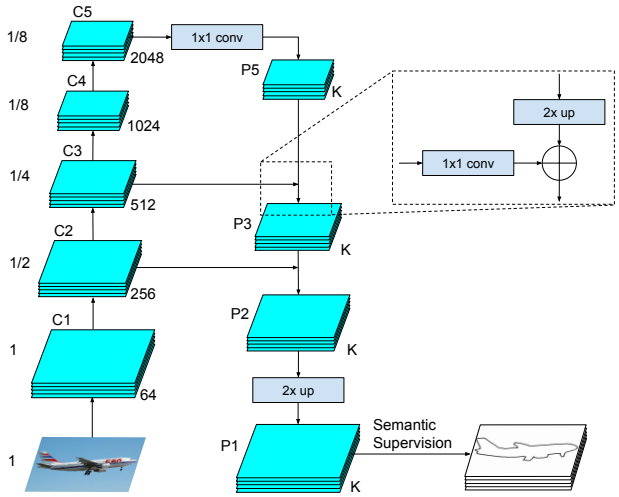


FIGURE 2 – Architecture of Edge Pyramid Network (EPN).

3.1 Network Architecture

EPN architecture is based on ResNet-101 [10] for bottom-up pathway and FPN [11] for top-down pathway and lateral connections.

Bottom-up pathway. We keep only convolution blocks in ResNet-101 and change the stride of the first and the last blocks from 2 to 1, as in CASENet [3]. Therefore, the resolution of features from the first to the fifth blocks are respectively 1, 1/2, 1/4, 1/8, 1/8, as illustrated in Fig.2. We use the output from the last layer of each scale residual block, denoted as $\{C_2, C_3, C_5\}$. In fact, C_1 does not contain semantic information and C_4 has the same resolution as C_5.

Top-down pathway and lateral connections. To extract semantic multi-scale features, we attach a K-channels 1x1 convolution layer on each output of stage's last residual blocks $\{C_2, C_3, C_5\}$, where K is the number of categories.

From the last residual block, the coarsest resolution semantic edge map P_5 is bilinearly upsampled by a factor of 2, and then merged with the lateral connection from C_3 by element-wise addition. Using this process, we obtain others semantic edge maps $\{P_3, P_2, P_1\}$ corresponding to the resolutions of 1/4, 1/2 and 1. Note that no lateral connection from C_1 is included into the pyramid. Therefore, the final semantic edge map P_1 which has the finest resolution is directly upsampled from P_2. Finally, we append a multi-label loss layer to P_1 for the semantic supervision. We do not append a 3x3 convolution layer on each map, as in FPN [11]. More details of the FPN architecture can be found in [11].

3.2 Multi-label unweighted loss

HED [7] is the first category-agnostic edge detection network that uses the weighted cross-entropy loss to deal with the class imbalance issue. The positive and negative losses are weighted respectively by the proportion of edge and non-edge to the total number of pixels of the ground truth. This weighting was then widely used in category-agnostic and category-aware edge detection networks [14] [15] [8] [12] [3]. Inspired by [4], we introduce the multi-label unweighted cross-entropy loss which was successfully used for the category-aware edge detection :

$$l(W) = \sum_k (l_+^k(W) + l_-^k(W)) \tag{1}$$

where

$$\begin{aligned} l_+^k &= -\sum_{j \in Y_+^k} logPr(y_j^k = 1|X;W) \\ l_-^k &= -\sum_{j \in Y_-^k} logPr(y_j^k = 0|X;W) \end{aligned} \tag{2}$$

are the positive and negative loss of the k-th semantic category.

3.3 Multi-label weighted-negative loss

We propose a multi-label weighted-negative loss, inspired by [16] :

$$l(W) = \sum_k (l_+^k(W) + l_{w-}^k(W)) \tag{3}$$

where

$$\begin{aligned} l_+^k &= -\sum_{j \in Y_+^k} logPr(y_j^k = 1|X;W) \\ l_{w-}^k &= -\sum_{j \in Y_-^k} Pr(y_j^k = 0|X;W)^2 logPr(y_j^k = 0|X;W) \end{aligned} \tag{4}$$

The false positive pixels are usually located around thick final edges due to the label misalignment in the ground truth. Therefore, our loss aims to reduce the contribution of negative examples, especially on misclassified non-edge, i.e. false positive pixels. In fact, when a non-edge pixel is misclassified with small $Pr(y = 0|X;W)$, the negative loss is down-weighted. Intuitively, these decreases help the network to automatically focus on edge pixels and to be easily trained without using class weight balancing. In the next section, we describe our training strategy for an effective fine-tuning of the network using the unweighted and weighted-negative cross-entropy losses.

3.4 Fine-tuning Strategy

To iteratively train EPN, we adopt the strategy illustrated in Fig. 3. We denote $M_u^{(i)}$ and $M_{w-}^{(i)}$ as the i-th trained models using the unweighted and weighted-negative cross-entropy losses, respectively.

Initialization step We first initialize the convolution blocks and the K-channels convolution layer on C_5 with the pre-trained model of CASENet [3]. By training our network with the unweighted cross-entropy loss (Eq. (1)), we obtain the first trained model $M_u^{(0)}$. We then apply a 2-steps fine-tuning strategy defined following :

Thick edge detection step By initializing all the layers from unweighted model $M_u^{(i)}$, we fine-tune EPN with the weighted-negative cross-entropy loss. As described in Eq. (3), the contribution of false positive pixels around edge ground truth to the total loss is almost reduced to zero. As a result, the weighted-negative model $M_{w-}^{(i+1)}$ acts as a thick edge detection module which easily accepts edge pixels around ground truth, as shown in the first row of Fig. 3.

Thin edge detection step Similarly to the thick edge detection step, we fine-tune EPN with the unweighted cross-entropy loss from the weighted-negative model $M_{w-}^{(i)}$. Without the modulating factor $Pr(y_j = 0|X;W)^2$, $M_u^{(i)}$ focus on the misclassified non-edge pixels which are almost ignored in $M_{w-}^{(i)}$. In other words, the network attempts to eliminate false positive and find more accurate pixels inside thick edges, as illustrated in the second row of Fig. 3.

From the model obtained in the initialization step $M_u^{(0)}$, we perform alternately and iteratively the thick and thin edge detection steps. This process is composed of successive training cycles $[M_{w-}^{(i)} \rightarrow M_u^{(i)}]$ as follows :

$[M_u^{(0)}] \rightarrow [M_{w-}^{(1)} \rightarrow M_u^{(1)}] \rightarrow [M_{w-}^{(2)} \rightarrow M_u^{(2)}] \rightarrow ...$

The unweighted model $M_u^{(i)}$ is finally chosen as the final model for evaluation, due to its more accurate edge maps.

4 Experiments

In this section, we compare EPN with state-of-the-art semantic edge detection algorithms CASENet [3], SEAL [4] and STEAL [5].

4.1 Datasets and Evaluation Protocol

Semantic Boundary Dataset (SBD) This dataset is composed of 11355 images from PASCAL VOC2011 [1], divided into 8498 training and 2857 test images. This da-

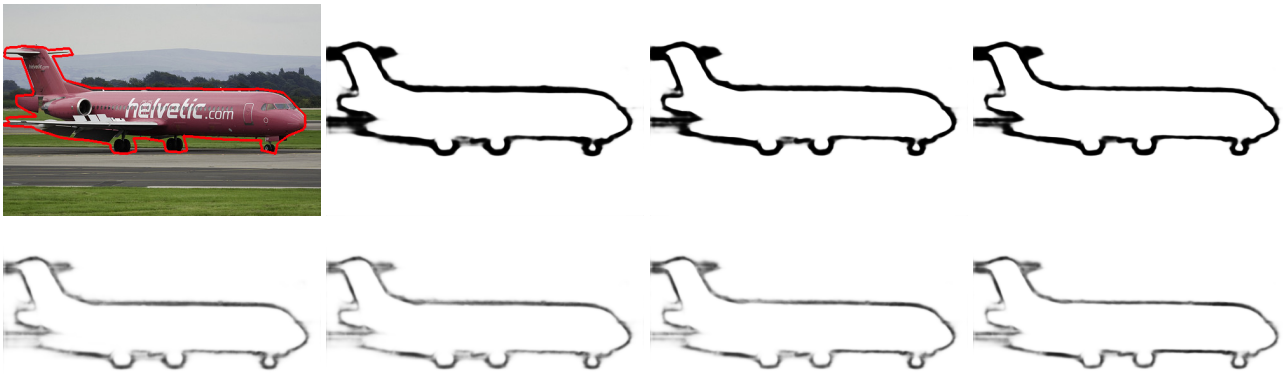


FIGURE 3 – Intermediate results of our fine-tuning strategy. From left to right : original image with the ground truth label corresponding to the category of aeroplane, thick edge maps from $M_{w-}^{(1)}$, $M_{w-}^{(2)}$, $M_{w-}^{(3)}$ models (first row); thin edge maps from $M_u^{(0)}$, $M_u^{(1)}$, $M_u^{(2)}$, $M_u^{(3)}$ models (second row).

taset contains 20 categories for semantic boundary detection. Due to the label misalignment, SEAL [4] re-annotated 1059 test images with high quality labels for more reliable results.

Cityscapes Dataset This dataset consists of 2975 training images, 500 for validation and 1525 for test. The validation images are used for test set because of the unavailability of test labels. Therefore, we use 2975 images for training and 500 others for testing without mixing. In our experiments, the same 19 semantic classes among 30 categories are selected for the benchmark, as others semantic edge detection algorithms [3], [4], [5].

Evaluation Protocol We use the standard performance metric F-measure (MF) at optimal dataset scale (ODS) for evaluation. We report the edge detection accuracy for each class using the evaluation protocol from [4]. We use "Thin" setting in all our experiments and apply a standard non-maximum suppression (NMS) to the edge maps for evaluation, as in other category-agnostic and category-aware edge detection algorithms [7] [15] [8] [5].

4.2 Implementation Details

Data augmentation We follow [4] to generate ground truth labels for training our model. For data augmentation, we resize images with scaling factors 0.5, 0.75, 1.0, 1.25, 1.5 in SBD and keep original resolution in Cityscapes.

Hyper-parameters For training, we choose stochastic gradient descent (SGD) for both SBD and Cityscapes datasets with the same hyper-parameters : batch size (1), learning rate (1e-7), gamma (0.1), iteration size (10), step size (10k), momentum (0.9), weight decay (0.0005), crop size (352x352). We stop training cycles when there is no more improvement of accuracy on the test set. The iteration numbers of fine-tuning steps are empirically set to 22000 and 20000 on SBD and Cityscapes, respectively. Each fine-tuning step takes about a day on a single GeForce GTX 1080 Ti GPU.

Caffe framework We implement our network using Caffe framework. For the multi-label loss, we modify the data input and sigmoid cross-entropy layers from the implementation of RCF [8] for more lightweight data storage compared to [4].

4.3 Results on SBD

Table 1 reports the MF scores of semantic edge detection algorithms on the re-annotated SBD test set under Instance-sensitive (IS) and Non-Instance-sensitive (non-IS) modes, respectively. We additionally evaluate the CASENet model which is fine-tuned with unweighted cross-entropy loss using its trained model on SBD dataset [3], denoted as CASENet-U. Note that we train and evaluate our network under two different modes for fair comparison with SEAL [4] and STEAL [5]. After three training cycles, our model $M_u^{(3)}$ outperforms SEAL and STEAL by 1.9% and 0.85% respectively in MF (ODS). We also report the performance of our network on the original SBD test set following the evaluation protocol from [1], as in Table 2 where our model $M_u^{(3)}$ achieves comparable performance to SEAL (+0.1%), but worse than STEAL (-1.1%). The accuracy of our model degrades compared to the edge alignment algorithms due to the noisy ground truth test labels from the original SBD dataset.

Edge pyramid architecture. In Fig. 4, we show intermediate edge maps extracted from all side outputs. The top-down pathway and lateral connections help the edge map evolves progressively from coarse to fine. The pyramid architecture also makes our thinned edge map after NMS looks smoother relative to the ones obtained by SEAL and STEAL, as illustrated in Fig. 1. We also examine the effect of NMS on thinning edge maps under Non-IS mode. Using NMS increases MF from 66.9 to 67.0 for SEAL, 66.0 to 67.9 for STEAL and 67.0 to 69.0 for EPN ($M_u^{(3)}$).

Additionally, we compare EPN and CASENet-U which is fine-tuned with the unweighted cross-entropy loss. As re-

TABLE 1 – MF scores on the re-annotated SBD test set with Instance-sensitive (IS) and Non-Instance-sensitive (Non-IS) modes following the evaluation protocol from [4].

Mode	Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
IS	CASENet	74.5	59.7	73.4	48.0	67.1	78.6	67.3	76.2	47.5	69.7	36.2	75.7	72.7	61.3	74.8	42.6	71.8	48.9	71.7	54.9	63.6
	CASENet-U	77.3	63.7	75.5	53.0	69.2	80.2	70.9	79.4	50.0	74.3	42.6	78.4	74.4	64.5	77.1	47.9	75.0	51.5	73.8	60.6	67.0
	SEAL	78.0	65.8	76.6	52.4	68.6	80.0	70.4	79.4	50.0	72.8	41.4	78.1	75.0	65.5	78.5	**49.4**	73.3	52.2	73.9	58.1	67.0
	EPN ($M_u^{(3)}$)	**80.8**	**67.7**	**76.8**	**57.6**	**69.6**	**81.4**	**72.6**	**80.3**	**53.4**	**74.6**	**43.5**	**79.9**	**77.7**	**68.0**	**78.7**	49.3	**77.5**	**52.3**	**75.4**	**61.4**	**68.9**
Non-IS	CASENet	74.84	60.17	73.71	47.68	66.69	78.59	66.66	76.23	47.17	69.35	36.23	75.88	72.45	61.78	73.10	43.01	71.23	48.82	71.87	54.93	63.52
	STEAL	80.15	67.80	**77.69**	54.26	**69.54**	**81.48**	71.34	78.97	51.76	73.61	42.82	79.80	76.44	67.68	78.16	**50.43**	75.06	50.99	75.31	59.66	68.15
	EPN ($M_u^{(3)}$)	**81.0**	**68.1**	77.2	**57.6**	69.4	81.4	**72.1**	**80.4**	**52.7**	**74.8**	**43.5**	**80.1**	**77.6**	**68.6**	**78.2**	49.7	**76.9**	**52.2**	**75.9**	**61.5**	**69.0**

TABLE 2 – Results on the original SBD test set following the evaluation protocol from [1].

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
CASENet	83.3	76.0	80.7	63.4	69.2	81.3	74.9	83.2	54.3	74.8	46.4	80.3	80.2	76.6	80.8	53.3	77.2	50.1	75.9	66.8	71.4
CASENet-U	85.1	79.3	84.8	68.1	71.8	83.4	76.6	85.6	58.5	78.7	51.9	83.0	82.3	78.1	84.6	57.7	79.1	52.2	78.2	69.3	74.4
SEAL	84.9	78.6	84.6	66.2	71.3	83.0	76.5	87.2	57.6	77.5	53.0	83.5	82.2	78.3	85.1	58.7	78.9	53.1	77.7	69.7	74.4
STEAL	85.8	80.0	**85.6**	68.4	71.6	**85.7**	**78.1**	**87.5**	**59.1**	**78.5**	**53.7**	**84.8**	**83.4**	**79.5**	**85.3**	**60.2**	79.6	**53.7**	**80.3**	**71.4**	**75.6**
EPN ($M_u^{(3)}$)	**87.3**	**80.2**	84.4	**69.1**	**72.4**	83.8	76.3	84.3	58.9	76.8	51.4	82.3	82.6	78.3	83.4	58.0	**80.6**	51.9	77.7	70.5	74.5

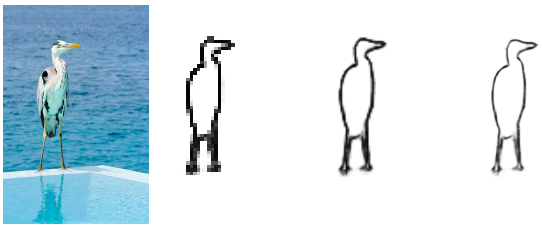


FIGURE 4 – From left to right : original image, intermediate results of EPN corresponding the category of bird from P_5, P_3, P_1. The edge maps are displayed at the same resolution for better view.

ported in Table 1 and 3, our model $M_u^{(0)}$ achieves MF 68.3%, 1.3% higher than CASENet-U. This demonstrates that the feature pyramid in EPN combines better multi-scale features than the fusion way in CASENet when using the unweighted cross-entropy loss.

Fine-tuning strategy. Fig. 3 shows intermediate results of our fine-tuning strategy on the SBD dataset. While the weighted-negative models reduce gradually false positives, the unweighted models make edge maps become crisper after each training cycle. The improvement which can be easily observed on the wing and belly of the airplane illustrates the role of the weighted-negative and unweighted losses in the fine-tuning steps. The increases on the accuracy of intermediate models $M_u^{(i)}$ reported in Table 3 also demonstrate quantitatively the effectiveness of our strategy. Note that our first model $M_u^{(0)}$ already outperforms the alignment learning algorithms SEAL and STEAL.

TABLE 3 – MF scores of our intermediate models on the re-annotated SBD test set.

Mode	Baseline	$M_u^{(0)}$	$M_u^{(1)}$	$M_u^{(2)}$	$M_u^{(3)}$
IS	67.0 (SEAL)	68.3	68.7	68.9	68.9
Non-IS	68.15 (STEAL)	68.2	68.7	68.9	68.9

Weighted cross-entropy loss. We examine the effect of the weighted cross-entropy loss on our architecture. By initializing the convolution blocks of ResNet with the model pre-trained on MS COCO [21], we keep all settings as default and finish the training after 22000 iterations as CASENet. The average accuracy of EPN on the original SBD degrades compared to CASENet. Unlike the unweighted cross-entropy loss, the weighted one makes the network produce more false positive edges which are accumulated through the pyramid architecture due to the label misalignment in the ground truth dataset. In contrast, when using the weighted cross-entropy for the category-agnostic edge detection task on a high quality label dataset as BSDS500, we obtain much better results than CASENet architecture, as described in Section 4.5.

Several variants. We investigated several variants of our network such as addition of feature map from the first stage to the pyramid, keeping the 3x3 convolution layer on each map as in FPN, imposing deep semantic supervision on other side outputs, replacing the initialization model CASENet by CASENet$^-$ or DSN [3]. However, they yielded no improvement.

4.4 Results on Cityscapes

As done for SBD, we also train and evaluate EPN with IS and non-IS modes on the Cityscapes dataset. In the Table 4, EPN achieves better performance compared to SEAL (3.7%) and STEAL (1.78%) after three and five training cycles. Note that our first model $M_u^{(0)}$ gets MF scores 71.1% and 71.5% under IS and non-IS modes, higher than SEAL (69.1%) and STEAL (71.42%), respectively. Compared to SBD, our network gives better results in Cityscapes due to the higher label annotation quality in this dataset.

Matching distance tolerance We evaluate the crispness of predictions by varying the maximum tolerance allowed for matching contours as illustrated in Table 5. When decreasing the matching distance tolerance from 0.0035 to 0.0015, the MF difference between EPN and STEAL increases from 1.8% to 3.6%. This demonstrates that EPN

TABLE 4 – MF scores on the Cityscapes validation set with Instance-sensitive (IS) and Non-Instance-sensitive (Non-IS) modes following the evaluation protocol from [4].

Mode	Method	road	sidewalk	building	wall	fence	pole	t-light	t-sign	veg	terrain	sky	person	rider	car	truck	bus	train	motor	bike	mean
IS	CASENet	86.2	74.9	74.5	47.6	46.5	72.8	70.0	73.3	79.3	57.0	86.5	80.4	66.8	88.3	49.3	64.6	47.8	55.8	71.9	68.1
	SEAL	87.6	77.5	75.9	47.6	46.3	75.5	71.2	75.4	80.9	60.1	87.4	81.5	68.9	88.9	50.2	67.8	44.1	52.7	73.0	69.1
	EPN ($M_u^{(3)}$)	**89.0**	**78.7**	**79.1**	**49.4**	**50.2**	**81.9**	**78.3**	**78.9**	**83.5**	**61.0**	**89.3**	**84.1**	**73.1**	**91.1**	**55.1**	**71.9**	**53.2**	**59.5**	**76.1**	**72.8**
Non-IS	CASENet	87.06	75.95	75.74	46.87	47.74	73.23	72.70	75.65	80.42	57.77	86.69	81.02	67.93	89.10	45.92	68.05	49.63	54.21	73.74	68.92
	STEAL	88.94	78.21	77.75	**50.59**	50.39	75.54	76.31	77.45	82.28	60.19	87.99	82.48	70.18	90.40	53.31	68.50	**53.39**	56.99	76.14	71.42
	EPN ($M_u^{(5)}$)	**89.3**	**79.0**	**79.9**	49.9	**50.7**	**82.2**	**79.6**	**79.2**	**84.2**	**60.7**	**89.8**	**84.0**	**74.1**	**91.1**	**55.3**	**72.5**	50.8	**59.8**	**77.9**	**73.2**

is able to capture more accurate and crisper edges compared to STEAL. As already presented in Fig. 1, our network produces smoother edge map than others.

TABLE 5 – MF with different matching distance tolerances on Cityscapes validation set.

Method	0.0035	0.0025	0.0015
STEAL	71.4	66.8	57.8
EPN	73.2 (+1.8)	69.3 (+2.5)	61.4 (+3.6)

4.5 EPN with Edge Detection on BSDS500

We investigate EPN for the category-agnostic edge detection task. We evaluate our architecture and fine-tuning strategy on BSDS500 dataset [27] which is composed of 200 training, 100 validation and 200 test images.

Implementation details. In this dataset, each image has manually labelled ground truth contours by several annotators. Thus, using the unweighted and weighted-negative losses may omit weak edges in the ground truth labels. Instead, we adopt the annotator-robust loss function from RCF [8], an improvement of the weighted cross-entropy loss, whose η and λ are set to 0.4 and 1.1, respectively. We keep other settings and hyper-parameters as default. The architecture is also adapted to the category-agnostic edge detection task. We add the feature extracted from $C1$ to the pyramid and place the deep supervision on top of all side outputs $\{P_1, P_2, P_3, P_4\}$ in order to better preserve low-level edges from bottom layers. During the data augmentation, we follow the same procedure as described in [7]. Inspired by [15] [25] [12] [8] [26], we employ the PASCAL VOC Context dataset [28] and adapt it to our fine-tuning strategy. Using the same hyper-parameters and loss function, we train alternately and iteratively our network on two datasets. By denoting $M_{VOC}^{(i)}$ and $M_{BSDS}^{(i)}$ as the i-th trained models using PASCAL VOC and BSDS500 datasets respectively, we obtain training cycles as follows :
$[M_{VOC}^{(1)} \rightarrow M_{BSDS}^{(1)}] \rightarrow [M_{VOC}^{(2)} \rightarrow M_{BSDS}^{(2)}] \rightarrow ...$
We start the fine-tuning strategy by the initialized model of ResNet-101 which is pre-trained on ImageNet [29]. Finally, we also use the NMS and the multi-scale testing for evaluation as in other works [25] [12] [8] [26].

Results. Table 6 reports the two standard measures : ODS and OIS (per-image best threshold) of the deep learning based contour detection algorithms on BSDS500 dataset. After four training cycles, our network $M_{BSDS}^{(4)}$ achieves ODS=.830 and OIS=.843 in multi-scale testing, competing with all other networks. The performance of our network is gradually improved through training cycles in single scale prediction (w/o MS) and multi-scale testing (w/ MS), as reported in Table 7.

TABLE 6 – The comparison with contour detection networks on BSDS500 [27] dataset. MS stands for multi-scale testing. VOC stands for training with additional data from PASCAL VOC.

Method	ODS	OIS
N^4-Fields [24]	.753	.769
DeepEdge [23]	.753	.772
DeepContour [22]	.753	.772
HFL [13]	.767	.788
HED [7]	.788	.808
RDS [14]	.792	.810
CEDN [15]	.788	.804
CASENet [3]	.767	.784
CED [25]	.794	.811
Res16x-CED-MS [25]	.810	.829
Res16x-CED-MS-VOC [25]	.822	.840
LPCB [12]	.800	.816
LPCB-VOC [12]	.808	.824
LPCB-MS-VOC [12]	.815	.834
RCF [8]	.806	.823
RCF-ResNet101-VOC [8]	.812	.829
RCF-ResNet101-MS-VOC [8]	.819	.836
BDCN [26]	.806	.826
BDCN-VOC [26]	.820	.838
BDCN-MS-VOC [26]	.828	**.844**
EPN	.810	.823
EPN-VOC ($M_{BSDS}^{(4)}$)	.823	.836
EPN-MS-VOC ($M_{BSDS}^{(4)}$)	**.830**	.843

Only using BSDS500 dataset. Note that if we only use BSDS500 dataset for training without additional data from PASCAL VOC, EPN achieves state-of-the-art results on F-measure ODS (.810) in single scale prediction. Compared to CASENet, our architecture is 5.1% and 3.9% higher on ODS and OIS, respectively. This accounts for the effectiveness of EPN in general edge detection task.

Merging PASCAL VOC and BSDS500 datasets. We also examine the impact of label quality on our network. The BSDS500 dataset which is designed for natural edge detection task has much higher quality of ground truth than

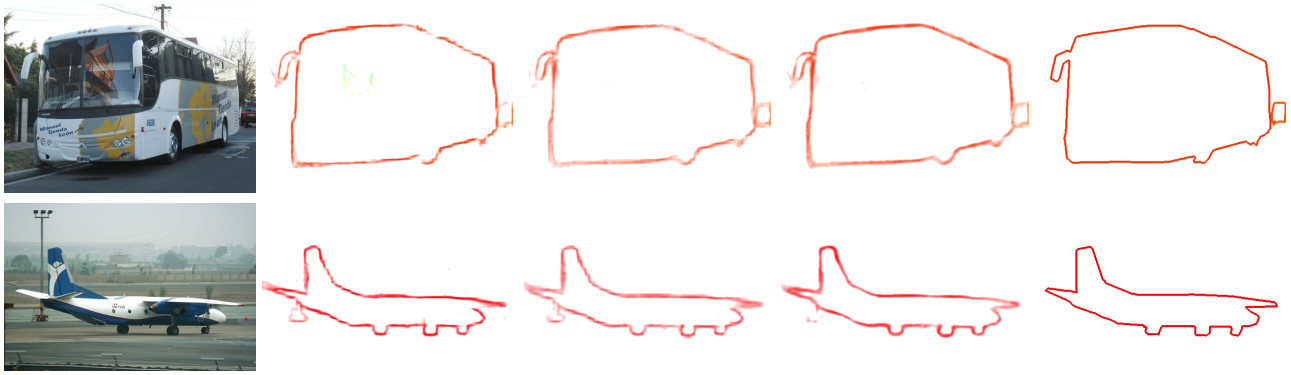


FIGURE 5 – Qualitative results on the SBD dataset. From left to right : original image, SEAL [4], STEAL [5], EPN, ground truth.

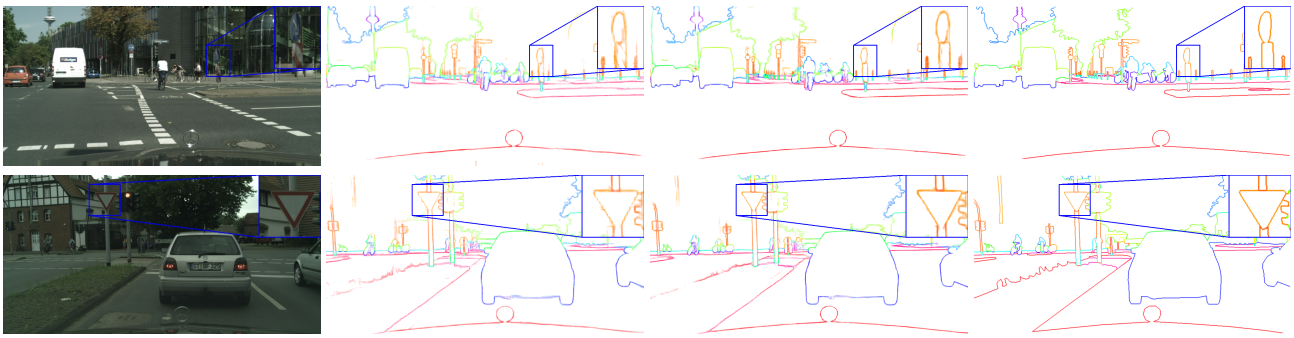


FIGURE 6 – Qualitative results on the Cityscapes validation set. From left to right : original image, STEAL [5], EPN, ground truth.

TABLE 7 – F-measure ODS of intermediate models with and without multi-scale testing on the BSDS500 dataset.

Model	$M^{(1)}_{BSDS}$	$M^{(2)}_{BSDS}$	$M^{(3)}_{BSDS}$	$M^{(4)}_{BSDS}$
w/o MS	.8200	.8217	.8228	.8231
w/ MS	.8275	.8297	.8303	.8298

PASCAL VOC. For this reason, we fine-tune our network with BSDS500 after pre-training with PASCAL VOC. However, when merging both datasets for training, the performance is decreased from .820 ($M^{(1)}_{BSDS}$) to .802 ODS F-measure in single scale prediction. This degradation demonstrates that using weighted cross-entropy loss in EPN architecture do not work well with noisy label data such as PASCAL VOC. This observation was already made on our experiment on SBD dataset as mentioned in Section 4.3.

5 Conclusion

Our original Edge Pyramid Network (EPN) is able to provide very accurate semantic edges by combining low and high level features. The fine-tuning strategy alternatively performing thick and thin edge detection using the weighted-negative and unweighted cross-entropy losses respectively also appears to be quite efficient. Before applying it, our network already produces better results than state-of-the-art semantic edge detection algorithms on SBD and Cityscapes datasets. Using our fine-tuning strategy, the performance is further improved after several training cycles. Additionally, the proposed EPN architecture achieves competitive performance against state-of-the-art algorithms for the category-agnostic edge detection task on the BSDS500 dataset. This demonstrates the effectiveness of the proposed architecture as well as the fine-tuning strategy for category-aware and category-agnostic edge detection tasks.

Références

[1] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. "Semantic contours from inverse detectors". In *ICCV*, 2011.

[2] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. "The Cityscapes dataset for semantic urban scene understanding". In *CVPR*, 2016.

[3] Z. Yu, C. Feng, M.-Y. Liu, and S. Ramalingam. "CASENet : Deep category-aware semantic edge detection". In *CVPR*, 2017.

[4] Z. Yu, W. Liu, Y. Zou, C. Feng, S. Ramalingam, B. Vijaya Kumar, and J. Kautz. "Simultaneous edge align-

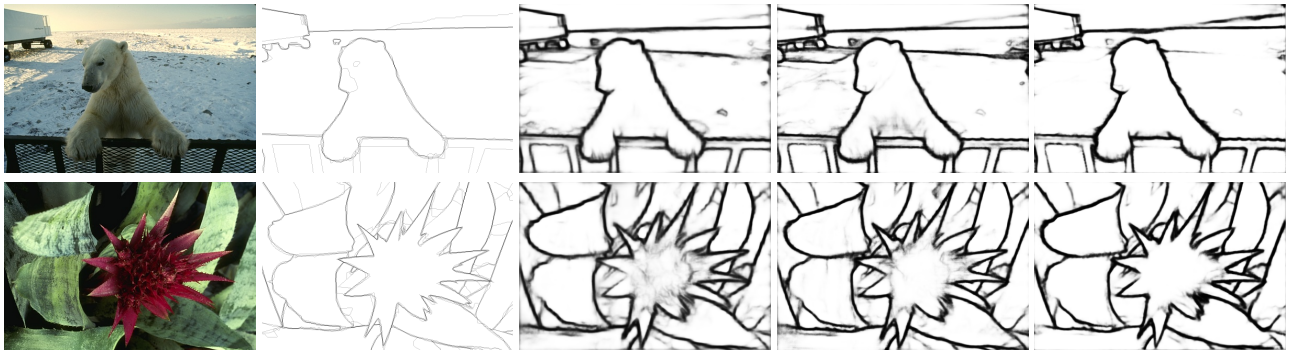


FIGURE 7 – Qualitative results on the BSDS500 test set using single scale. From left to right : original image, ground truth, RCF-ResNet101 [8], BDCN [26], EPN ($M^{(4)}_{BSDS}$).

ment and learning". In *ECCV*, 2018.

[5] D. Acuna, A. Kar, and S. Fidler. "Devil Is in the Edges : Learning Semantic Boundaries from Noisy Annotations". In *CVPR*, 2019.

[6] Y. Hu, Y. Chen, X. Li, and J. Feng. "Dynamic Feature Fusion for Semantic Edge Detection". In *IJCAI*, 2019.

[7] S. Xie and Z. Tu. "Holistically-nested edge detection". In *ICCV*, 2017.

[8] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai. "Richer Convolutional Features for Edge Detection". In *PAMI*, 2019.

[9] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". In *ICLR* , 2015.

[10] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". In *CVPR*, 2016.

[11] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. "Feature pyramid networks for object detection". In *CVPR*, 2017.

[12] R. Deng, C. Shen, S. Liu, H. Wang, and X. Liu. "Learning to predict crisp boundaries". In *ECCV*, 2018.

[13] G. Bertasius, J. Shi, and L. Torresani. "High-for-low, low-for-high : Efficient boundary detection from deep object features and its applications to high-level vision". In *ICCV*, 2015.

[14] Y. Liu and M. S. Lew. "Learning relaxed deep supervision for better edge detection". In *CVPR*, 2016.

[15] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang. "Object contour detection with a fully convolutional encoder-decoder network". In *CVPR* 2016.

[16] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. "Focal loss for dense object detection". In *PAMI*, 2018.

[17] J. Canny. "A computational approach to edge detection". In *PAMI*, 1986

[18] S. Ren, K. He, R. Girshick, and J. Sun. "Faster R-CNN : Towards real-time object detection with region proposal networks". In *NIPS*, 2015.

[19] K. He, G. Gkioxari, P. Dollar, and R. Girshick. "Mask R-CNN". In *ICCV*, 2017.

[20] A. Kirillov, R. Girshick, K. He, and P. Dollar. "Panoptic feature pyramid networks". In *CVPR*, 2019.

[21] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R.B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. "Microsoft COCO : common objects in context". *CoRR, abs/1405.0312*, 2014.

[22] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. "Deepcontour : A deep convolutional feature learned by positivesharing loss for contour detection". In *CVPR*, 2015.

[23] G. Bertasius, J. Shi, and L. Torresani. "Deepedge : A multiscale bifurcated deep network for top-down contour detection". In *CVPR*, 2015.

[24] Y. Ganin and V. Lempitsky. "N4-fields : Neural network nearest neighbor fields for image transforms". In *ACCV*, 2014.

[25] Y. Wang, X. Zhao, Y. Li, and K. Huang. "Deep crisp boundaries : From boundaries to higher-level tasks". In *TIP*, 2018.

[26] J. He, S. Zhang, M. Yang, Y. Shan, T. Huang. "Bi-Directional Cascade Network for Perceptual Edge Detection". In *CVPR*, 2019.

[27] D. R. Martin, C. C. Fowlkes, and J. Malik. "Learning to detect natural image boundaries using local brightness, color, and texture cues". In *PAMI*, 2004.

[28] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. "The role of context for object detection and semantic segmentation in the wild". In *CVPR*, 2014.

[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet : A Large-Scale Hierarchical Image Database". In *CVPR*, 2009.

[30] TV. Pham, Y. Lucas, S.Treuillet, and L. Debraux. "Object Contour Refinement Using Instance Segmentation in Dental Images". In *ACIVS*, 2020.