RINDNet: Edge Detection for Discontinuity in Reflectance, Illumination, Normal and Depth

Mengyang Pu[1]*, Yaping Huang[1,†], Qingji Guan[1], Haibin Ling[2]
[1]Beijing Key Laboratory of Traffic Data Analysis and Mining, Beijing Jiaotong University, China
[2]Department of Computer Science, Stony Brook University, USA
{mengyangpu, yphuang, qjguan}@bjtu.edu.cn; hling@cs.stonybrook.edu

Abstract

As a fundamental building block in computer vision, edges can be categorised into four types according to the discontinuity in surface-Reflectance, Illumination, surface-Normal or Depth. While great progress has been made in detecting generic or individual types of edges, it remains under-explored to comprehensively study all four edge types together. In this paper, we propose a novel neural network solution, RINDNet, to jointly detect all four types of edges. Taking into consideration the distinct attributes of each type of edges and the relationship between them, RINDNet learns effective representations for each of them and works in three stages. In stage I, RINDNet uses a common backbone to extract features shared by all edges. Then in stage II it branches to prepare discriminative features for each edge type by the corresponding decoder. In stage III, an independent decision head for each type aggregates the features from previous stages to predict the initial results. Additionally, an attention module learns attention maps for all types to capture the underlying relations between them, and these maps are combined with initial results to generate the final edge detection results. For training and evaluation, we construct the first public benchmark, BSDS-RIND, with all four types of edges carefully annotated. In our experiments, RINDNet yields promising results in comparison with state-of-the-art methods. Additional analysis is presented in supplementary material.

1. Introduction

Edges play an important role in many vision tasks [33, 40, 43, 46]. While generic edge detection [14, 23, 41, 44] has been extensively studied for decades, specific-edge detection recently attracts increasing amount of efforts due to its practical applications concerning on different types of edges, such as occlusion contours [25, 38, 39] or semantic boundaries [16, 48].

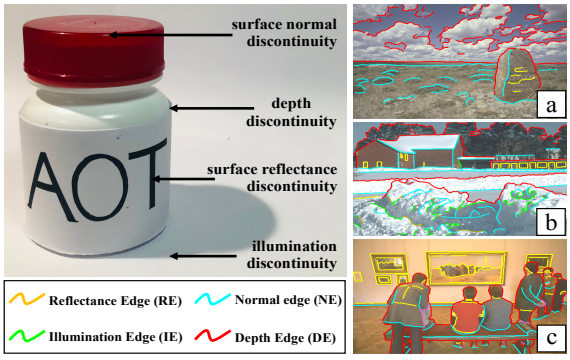


Figure 1. **Left:** Edges are caused by a variety of factors [27, 35]. **Right (a-c):** Samples of the proposed BSDS-RIND dataset.

In his seminal work [27], David Marr summarized four basic ways edges can arise: (1) *surface-reflectance discontinuity*, (2) *illumination discontinuity*, (3) *surface-normal discontinuity*, and (4) *depth discontinuity*, as shown in Fig. 1. Recent studies [33, 40, 43, 46] have shown that the above types of edges are beneficial for downstream tasks. For example, pavement crack detection (reflectance discontinuity) is a critical task for intelligent transportation [46]; shadow edge (illumination discontinuity) detection is a prerequisite for shadow removal and path detection [43]; [33] and [40] show that depth edge and normal edge representation prompt refined normal and sharp depth estimation, respectively. Besides, [18] utilizes four types of cues simultaneously to improve the performance of depth refinement.

Despite their importance, fine-grained edges are still under-explored, especially when compared with generic edges. Generic edge detectors usually treat edges indistinguishably; while existing studies for specific edges focus on individual edge type. By contrast, the four fundamental types of edges, to our best knowledge, have never been explored in an integrated edge detection framework.

In this paper, for the first time, we propose to detect simultaneously the four types of edges, namely *reflectance*

*The work is partially done while the author was with Stony Brook University. † Corresponding author.

edge (RE), *illumination edge* (IE), *normal edge* (NE) and *depth edge* (DE). Although edges share similar patterns in intensity variation in images, they have different physical bases. Specifically, REs and IEs are mainly related to photometric reasons – REs are caused by changes in material appearance (*e.g.*, texture and color), while IEs are produced by changes in illumination (*e.g.*, shadows, light sources and highlights). By contrast, NEs and DEs reflect the geometry changes in object surfaces or depth discontinuity. Considering the correlations and distinctions among all types of edges, we develop a CNN-based solution, named *RINDNet*, for jointly detecting the above four types of edges.

RINDNet works in three stages. In stage I, it extracts general features and spatial cues from a backbone network for all edges. Then, in stage II, it proceeds with four separate decoders. Specifically, low-level features are first integrated under the guidance of high-level hints by Weight Layer (WL), and then fed into RE-Decoder and IE-Decoder to produce features for REs and IEs respectively. At the same time, NE/DE-Decoder take the high-level features as input and explore effective features. After that, these features and accurate spatial cues are forwarded to four decision heads in stage III to predict the initial results. Finally, the attention maps obtained by the Attention Module (AM), which captures the underlying relations between all types, are aggregated with the initial results to generate the final predictions. All these components are differentiable, making RINDNet an end-to-end architecture to jointly optimize the detection of four types of edges.

Training and evaluating edge detectors for all four types of edges request images with all such edges annotated. In this paper, we create the first known such dataset, named *BSDS-RIND*, by carefully labeling images from the BSDS [2] benchmark (see Fig. 1). BSDS-RIND allows the first thorough evaluation of edge detection of all four types. The proposed RINDNet shows clear advantages over previous edge detectors, both quantitatively and qualitatively. The source code, dataset, and benchmark are available at https://github.com/MengyangPu/RINDNet.

With the above efforts, our study is expected to stimulate further research along the line, and benefit more downstream applications with rich edge cues. Our contributions are summarized as follows: (1) We develop a novel end-to-end edge detector, RINDNet, to jointly detect the four types of edges. RINDNet is designed to effectively investigate shared information among different edges (*e.g.*, through feature sharing) and meanwhile flexibly model the distinction between them (*e.g.*, through edge-aware attention). (2) We present the first public benchmark, BSDS-RIND, dedicated to studying simultaneously the four edge types, namely reflectance edge, illumination edge, normal edge and depth edge. (3) In our experiments, the proposed RINDNet shows clear advantages over state of the arts.

2. Related Works

Edge Detection Algorithms. Early edge detectors [5, 19, 42] obtain edges based directly on the analysis of image gradients. By contrast, learning-based methods [11, 21, 28] exploit different low-level features that respond to characteristic changes, then a classifier is trained to generate edges. CNN-based edge detectors [3, 4, 9, 10, 17, 20, 24, 26, 31, 36, 45] do not rely on hand-crafted features and achieve better performance. Combining multi-scale and multi-level features, [14,23,31,44] yield outstanding progresses on generic edge detection. A novel refinement architecture is also proposed in [41] using a top-down backward refinement pathway to generate crisp edges. Recent works [1,47,49,50] pay more attention to special types of edges. In [13] the generic object detector is combined with bottom-up contours to infer object contours. CASENet [48] adopts a nested architecture to address semantic edge detection. For better prediction, DFF [16] learns adaptive weights to generate specific features of each semantic category. For occlusion boundary detection, DOC [39] decomposes the task into occlusion edge classification and occlusion orientation regression, then two sub-networks are used to separately perform the above two tasks. DOOBNet [38] uses an encoder-decoder structure to obtain multi-scale and multi-level features, and shares the backbone features with two branches. OFNet [25] considers the relevance and distinction for the occlusion edge and orientation, thus it shares the occlusion cues between two sub-networks.

Edge Datasets. Many datasets have been proposed for studying edges. BSDS [2] is a popular edge dataset for detecting generic edges containing 500 RGB natural images. Although each image is annotated by multiple users, they usually pay attention to salient edges related to objects. BIPED [29] is created to explore more comprehensive and dense edges, and contains 250 outdoor images. NYUD [37] contains 1,449 RGB-D indoor images, and lacks edge types pertaining to outdoor scenes. Significantly, Multicue [29] considers the interaction between several visual cues (luminance, color, stereo, motion) during boundary detection.

Recently, SBD [13] is presented for detecting semantic contours, using the images from the PASCAL VOC challenge [12]. Cityscapes [7] provides the object or semantic boundaries focusing on road scenes. For reasoning occlusion relationship between objects, the dataset in [34] consists of 200 images, where boundaries are assigned with figure/ground labels. Moreover, PIOD [39] contains 10,000 images, each with two annotations: a binary edge map denotes edge pixels and a continuous-valued occlusion orientation map. The recent dataset in [33] annotates NYUD test set for evaluating the occlusion boundary reconstruction.

Our work is inspired by the above pioneer studies, but makes novel contributions in two aspects: the proposed RINDNet, to the best of our knowledge, is the first edge

detector to jointly detect all four types of edges, and the proposed BSDS-RIND is the first benchmark with all four types of edges annotated.

3. Problem Formulation and Benchmark

3.1. Problem Formulation

Let $X \in \mathbb{R}^{3\times W\times H}$ be an input image with ground-truth labels $\mathcal{E} = \{E^r, E^i, E^n, E^d\}$, where $E^r, E^i, E^n, E^d \in \{0,1\}^{W\times H}$ are binary edge maps indicating the reflectance edges (REs), illumination edges (IEs), surface-normal edges (NEs) and depth edges (DEs), respectively. Our goal is to generate the final predictions $\mathcal{Y} = \{Y^r, Y^i, Y^n, Y^d\}$, where Y^r, Y^i, Y^n, Y^d are the edge maps corresponding to REs, IEs, NEs and DEs, respectively. In our work, we aim to learn a CNN-based edge detector ψ: $\mathcal{Y} = \psi(X)$.

The training of ψ can be done over training images by minimizing some loss functions between $\mathcal{E}$ and $\mathcal{Y}$. Therefore, a set of images with ground-truth labels are required to learn the mapping ψ. We contribute such annotations in this work, and the detailed processes are shown in §3.2.

3.2. Benchmark

One aim of our work is to contribute a first public benchmark, named BSDS-RIND, over BSDS images [2]. The original images contain various complex scenes, which makes it challenging to jointly detect all four types of edges. Fig. 1 (Right) shows some examples of our annotations.

Edge Definitions. It is critical to define four types of edges for the annotation task. Above all, we give the definition of each type and illustrate it with examples.

- ***Reflectance Edges*** **(REs)** usually are caused by the changes in material appearance (*e.g.*, texture and color) of smooth surfaces. Notably, although the edges within paintings in images (see Fig. 1 (c)) could be classified to DEs by human visual system, these edges are assigned as REs since there is no geometric discontinuity.
- ***Illumination Edges*** **(IEs)** are produced by shadows, light sources, highlights, *etc.* (as shown in Fig. 1).
- ***Normal Edges*** **(NEs)** mark the locations of discontinuities in surface orientation and normally arise between parts. As shown in Fig. 1 (b), we take the edges between the building and the ground as an example, the change of depth across these two surfaces is continuous but not smooth, which is caused by the surface-normal discontinuity between them.
- ***Depth Edges*** **(DEs)** are resulted by depth discontinuity, and often coincide with object silhouettes. It is difficult to measure the depth difference (*e.g.*, Fig. 1 (a)), thus the relative depth difference is used to determine whether an edge belongs to DEs. Although there exists the depth changes between windows and walls in the building (Fig. 1 (b)), the small distance ratio is caused by the long distance between them and camera, so such edges are classified to REs rather than DEs.

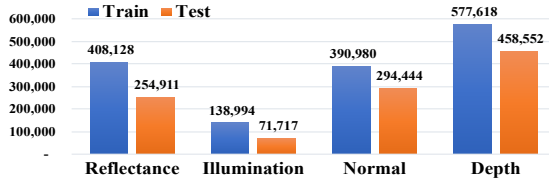


Figure 2. The distribution of pixels for each type of edges on BSDS-RIND training set and testing set.

Annotation Process. The greatest effort for constructing a high-quality edge dataset is devoted to, not surprisingly, manual labeling, checking, and revision. For this task, we manually construct the annotations using ByLabel [32]. Two annotators collaborate to label each image. One annotator first manually labels the edges, and another annotator checks the result and may supplement missing edges. Those edges with labels are added directly to the final dataset if both annotators agree with each other. Ambiguous edges will be revised by both annotators together: the consistent annotations are given after discussion. After iterating several times, we get the final annotations. Moreover, for some edges that are difficult to determine the main factors of their formation, multiple labels are assigned for them. It is only about 53k (2%) pixels with multi-labels in BSDS-RIND. In addition, we use the average Intersection-over-Union (IoU) score to measure agreement between two annotators, and get 0.97, 0.92, 0.93 and 0.95 for REs, IEs, NEs and DEs, respectively. The statistics show good consistency.

With all efforts, finally, a total of 500 images are carefully annotated, leading to a densely annotated dataset, named BSDS-RIND. Then it is split into 300 training images, and 200 testing images, respectively. The total number of pixels for each type on the BSDS-RIND training and testing set are reported in Fig. 2. Significantly, the number of edge pixels in BSDS-RIND are twice as in BSDS. Moreover, edge detection is a pixel-wise task, thus the number of samples provided by BSDS-RIND decently supports learning-based algorithms. More examples and details are given in the supplementary material.

4. RINDNet

In this work, we design an end-to-end network (§4.1), named *RINDNet*, to learn distinctive features for optimizing the detection of four edge types jointly. Fig. 3 shows an overview of our proposed RINDNet, which includes three stages of initial results inference (*i.e.*, extracting common features, preparing distinctive features, and generating initial results) and final predictions integrated by an Attention Module. We also explain the loss functions and training details in §4.2 and §4.3.

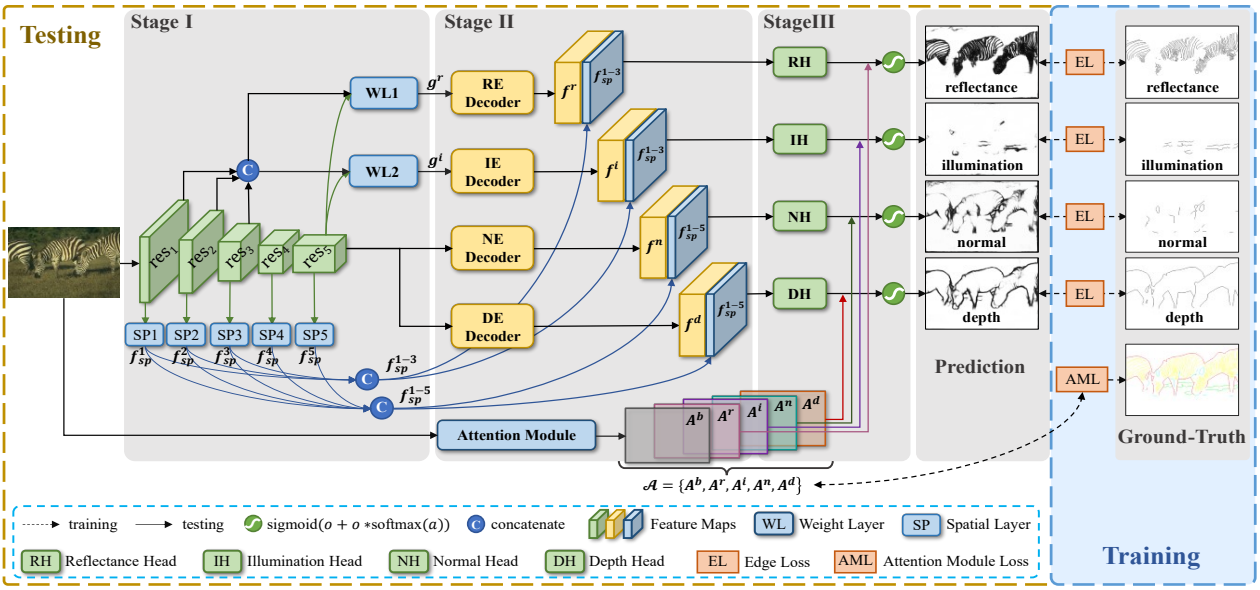


Figure 3. The three-stage architecture of RINDNet. **Stage I**: the input image is fed into a backbone to extract features shared with all edge types. **Stage II**: features across different levels are fused via Weight Layers (WLs), and are forwarded to four decoders in two clusters: RE-Decoder/IE-Decoder and NE-Decoder/DE-Decoder. **Stage III**: four decision heads predict the four types of initial results. In addition, the attention maps learned by the attention module are integrated into the final prediction (A^b used only in training). (Best viewed in color)

4.1. Methodology

Stage I: Extracting Common Features for All Edges. We first use a backbone to extract common features for all edges because these edges share similar patterns in intensity variation in images. The backbone follows the structure of ResNet-50 [15] which is composed of five repetitive building blocks. Specifically, the feature maps from the above five blocks of ResNet-50 [15] are denoted as res_1, res_2, res_3, res_4 and res_5, respectively.

Then, we generate spatial cues from the above features. It is well known that different layers of CNN features encode different levels of appearance/semantic information, and contribute differently to different edge types. Specifically, bottom-layer feature maps res_{1-3} focus more on low-level cues (*e.g.*, color, texture and brightness), while top-layer maps res_{4-5} are in favor of object-aware information. Thus it is beneficial to capture multi-level spatial responses from different layers of feature maps. Given multiple feature maps res_{1-5}, we obtain the spatial response maps:

$$f_{sp}^k = \psi_{\text{sp}}^k(res_k), \quad k \in \{1, 2, 3, 4, 5\} \tag{1}$$

where the spatial responses $f_{sp}^k \in \mathbb{R}^{2\times W\times H}$ are learned by Spatial Layer ψ_{sp}^k which is composed of one convolution layer and one deconvolution layer.

Stage II: Preparing Distinctive Features for REs/IEs and NEs/DEs. Afterwards, RINDNet learns particular features for each edge type separately by the corresponding decoder in stage II. Inspired by [25], we design the Decoder with two streams to recover fine location information, as shown in Fig. 4 (b). Two-stream decoder can work collaboratively and learn more powerful features from different views in the proposed architecture. Although four decoders have the same structure, some special designs are proposed for different types of edges, and we will give the detailed descriptions below. To distinguish each type of edges reasonably and better depict our work, we next cluster the four edge types into two groups, *i.e.*, REs/IEs and NEs/DEs, to prepare features for them respectively.

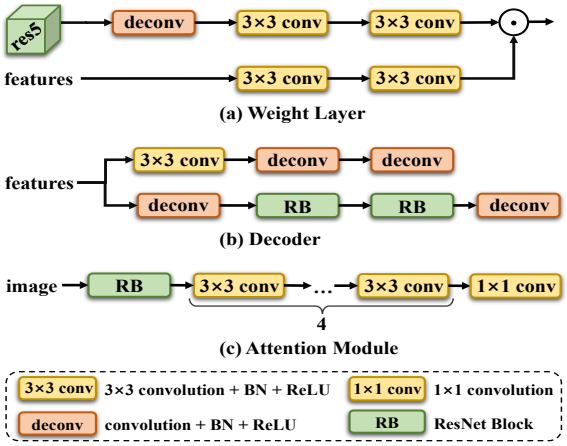


Figure 4. The architectures of (a) Weight Layer, (b) Decoder and (c) Attention Module. $\odot$ is the element-wise multiplication.

REs and IEs. In practice, the low-level features (*e.g.*, res_{1-3}) capture detailed intensity changes that are often reflected in REs and IEs. Besides, REs and IEs are related to the global context and surrounding objects provided by the high-level features (*e.g.*, res_5). Thus, it is desirable that se-

mantic hints may give the felicitous guidance to aware the intensity changes, before forwarding to the Decoder. Moreover, it is notable that simply concatenating the low-level and high-level features may be too computationally expensive due to the increased number of parameters. We therefore propose the Weight Layer (WL) to adaptively fuse the low-level features and high-level hints in a learnable manner, without increasing the dimension of the features.

As shown in Fig. 4 (a), WL contains two paths: the first path receives high-level feature res_5 to recover high resolution through a deconvolution layer, and then two 3×3 convolution layers with Batch Normalization (BN) and ReLU excavate adaptive semantic hints; another path is implemented as two convolution layers with BN and ReLU, which encodes low-level features res_{1-3}. Afterwards, they are fused by element-wise multiplication. Formally, given the low-level features res_{1-3} and high-level hints res_5, we generate the fusion features for REs and IEs individually,

$$\begin{aligned} g^r &= \psi^r_{\text{wl}}\big(res_5, [res_1, res_2, \text{up}(\text{res}_3)]\big), \\ g^i &= \psi^i_{\text{wl}}\big(res_5, [res_1, res_2, \text{up}(\text{res}_3)]\big), \end{aligned} \tag{2}$$

where the WL of REs and IEs are indicated as ψ^r_{wl} and ψ^i_{wl} respectively, g^r/g^i are the fusion features for REs/IEs, and $[\cdot]$ is the concatenation. Note that, the resolution of res_3 is smaller than res_1 and res_2, so one up-sampling operation $\text{up}(\cdot)$ is used on res_3 to increase resolution before feature concatenation. Next, the fusion features are fed into the corresponding Decoder to generate specific features with accurate location information separately for IEs and REs,

$$f^r = \psi^r_{\text{deco}}(g^r), \quad f^i = \psi^i_{\text{deco}}(g^i), \tag{3}$$

where ψ^r_{deco} and ψ^i_{deco} indicate Decode of REs and IEs respectively, and f^r/f^i are decoded feature maps for REs/IEs.

NEs and DEs. Since the high-level features (*e.g.*, res_5) express strong semantic responses that are usually epitomized in NEs and DEs, we utilize res_5 to obtain the particular features for NEs and DEs,

$$f^n = \psi^n_{\text{deco}}(res_5), \quad f^d = \psi^d_{\text{deco}}(res_5), \tag{4}$$

where NE-Decode and DE-Decoder are denoted as ψ^n_{deco} and ψ^d_{deco} respectively, and f^n/f^d are the decoded features of NEs/DEs. Since DEs and NEs commonly share some relevant geometry cues, we share the weights of the second stream of NE-Decoder and DE-Decoder to learn the collaborative geometry cues. At the same time, the first stream of NE-Decoder and DE-Decoder is responsible for learning particular features for REs and DEs, respectively.

Stage III: Generating Initial Results. We predict the initial results for each type of edges by the respective decision head in final stage. The features from previous stages, containing rich location information of edges, can be used to predict edges. Specifically, we concatenate the decoded features f^r/f^i with spatial cues f^{1-3}_{sp} to predict REs/IEs,

$$O^r = \psi^r_{\text{h}}\big([f^r, f^{1-3}_{sp}]\big), \quad O^i = \psi^i_{\text{h}}\big([f^i, f^{1-3}_{sp}]\big), \tag{5}$$

where O^r/O^i are the initial predictions of REs/IEs. The decision heads of REs and IEs, named ψ^r_{h} and ψ^i_{h} respectively, are modeled as a 3×3 convolution layer and a 1×1 convolution layer. Note that REs and IEs do not directly rely on the location cues provided by top-layer, thus spatial cues f^{4-5}_{sp} are not used for them. By contrast, all spatial cues f^{1-5}_{sp} are concatenated with the decoded features to generate initial results for NEs and DEs, respectively,

$$O^n = \psi^n_{\text{h}}\big([f^n, f^{1-5}_{sp}]\big), \quad O^d = \psi^d_{\text{h}}\big([f^d, f^{1-5}_{sp}]\big), \tag{6}$$

where ψ^n_{h} and ψ^d_{h} respectively indicate the decision heads of NEs and DEs, which are composed of three 1×1 convolutional layers to integrate hints at each position. In summary, $\mathcal{O} = \{O^r, O^i, O^n, O^d\}$ denotes the initial result set.

Attention Module. Finally, RINDNet integrates initial results with attention maps obtained by Attention Module (AM) to generate the final results. Since different types of edges are reflected in different locations, when predicting each type of edges, it is necessary to pay more attention to the related locations. Fortunately, the edge annotations provide the label of each location. Accordingly, the proposed AM could infer the spatial relationships between multiple labels with pixel-wise supervision by the attention mechanism. The attention maps could be used to activate the responses of related locations. Formally, given the input image X, AM learns spatial attention maps,

$$\mathcal{A} = \{A^b, A^r, A^i, A^n, A^d\} = \text{softmax}\big(\psi_{\text{att}}(X)\big), \tag{7}$$

where $\mathcal{A}$ is the normalized attention maps by a softmax function, and $A^b, A^r, A^i, A^n, A^d \in [0,1]^{W\times H}$ are the attention maps corresponding to background, REs, IEs, NEs and DEs respectively. Obviously, if a label is tagged to one pixel, the location of this pixel should be assigned with higher attention values. The AM ψ_{att} is achieved by the first building block of ResNet, four 3×3 convolution layers (each layer is followed by ReLU and BN operations), and one 1×1 convolution layers, as shown in Fig. 4 (c).

Finally, the initial results are integrated with the attention maps to generate the final results $\mathcal{Y}$,

$$\mathcal{Y} = \text{sigmoid}\big(\mathcal{O} \odot (1 + A^{\{r,i,n,d\}})\big), \tag{8}$$

where $\odot$ is the element-wise multiplication.

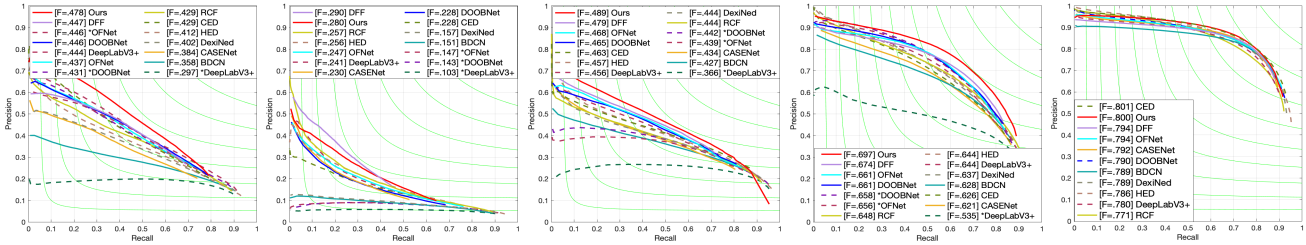
4.2. Loss Function

Edge Loss. We use the loss function presented in [38] to supervise the training of our edge predictions:

$$\mathcal{L}_{\text{e}}(\mathcal{Y}, \mathcal{E}) = \sum_{k\in\{r,i,n,d\}} \ell_{\text{e}}\left(Y^k, E^k\right), \tag{9}$$

Table 1. Quantitative comparison for REs, IEs, NEs, DEs and Average (best viewed in color: "**red**" for best, and "**blue**" for second best).

Method	Reflectance			Illumination			Normal			Depth			Average		
	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP
HED [44]	0.412	0.466	0.343	0.256	0.290	0.167	0.457	0.505	**0.395**	0.644	0.679	0.667	0.442	0.485	**0.393**
CED [41]	0.429	0.473	0.361	0.228	0.286	0.118	0.463	0.501	0.372	0.626	0.655	0.620	0.437	0.479	0.368
RCF [23]	0.429	0.448	0.351	0.257	0.283	**0.173**	0.444	0.503	0.362	0.648	0.679	0.659	0.445	0.478	0.386
BDCN [14]	0.358	0.458	0.252	0.151	0.219	0.078	0.427	0.484	0.334	0.628	0.661	0.581	0.391	0.456	0.311
DexiNed [31]	0.402	0.454	0.315	0.157	0.199	0.082	0.444	0.486	0.364	0.637	0.673	0.645	0.410	0.453	0.352
CASENet [48]	0.384	0.439	0.275	0.230	0.273	0.119	0.434	0.477	0.327	0.621	0.651	0.574	0.417	0.460	0.324
DFF [16]	**0.447**	0.495	0.324	**0.290**	**0.337**	0.151	**0.479**	**0.512**	0.352	**0.674**	**0.699**	0.626	**0.473**	**0.511**	0.363
*DeepLabV3+ [6]	0.297	0.338	0.165	0.103	0.150	0.049	0.366	0.398	0.232	0.535	0.579	0.449	0.325	0.366	0.224
*DOOBNet [38]	0.431	0.489	0.370	0.143	0.210	0.069	0.442	0.490	0.339	0.658	0.689	0.662	0.419	0.470	0.360
*OFNet [25]	0.446	0.483	**0.375**	0.147	0.207	0.071	0.439	0.478	0.325	0.656	0.683	**0.668**	0.422	0.463	0.360
DeepLabV3+ [6]	0.444	0.487	0.356	0.241	**0.291**	0.148	0.456	0.495	0.368	0.644	0.671	0.617	0.446	0.486	0.372
DOOBNet [38]	0.446	**0.503**	0.355	0.228	0.272	0.132	0.465	0.499	0.373	0.661	0.691	0.643	0.450	0.491	0.376
OFNet [25]	0.437	0.483	0.351	0.247	0.277	0.150	0.468	0.498	0.382	0.661	0.687	0.637	0.453	0.486	0.380
RINDNet (Ours)	**0.478**	**0.521**	**0.414**	**0.280**	**0.337**	**0.168**	**0.489**	**0.522**	**0.440**	**0.697**	**0.724**	**0.705**	**0.486**	**0.526**	**0.432**



(a) Reflectance Edges (b) Illumination Edges (c) Normal Edges (d) Depth Edges (e) Generic Edges

Figure 5. Evaluation results on BSDS-RIND for (a) REs, (b) IEs, (c) NEs, (d) DEs and (e) generic edges.

$$\ell_{\mathrm{e}}(Y,E) = -\sum_{i,j}\Big(E_{i,j}\alpha_1\beta^{(1-Y_{i,j})^{\gamma_1}}\log(Y_{i,j}) + (1-E_{i,j})(1-\alpha_1)\beta^{Y^{\gamma_1}}\log(1-Y_{i,j})\Big), \tag{10}$$

where $\mathcal{Y} = \{Y^r, Y^i, Y^n, Y^d\}$ is the final prediction, $\mathcal{E} = \{E^r, E^i, E^n, E^d\}$ is the corresponding ground-truth label, and $E_{i,j}/Y_{i,j}$ are the $(i,j)^{th}$ element of matrix E/Y respectively. Moreover, $\alpha_1 = |E_-|/|E|$ and $1-\alpha_1 = |E_+|/|E|$, where E_- and E_+ denote the non-edge and edge ground truth label sets, respectively. In addition, γ_1 and β are the hyperparameters. We drop the superscript k in Eq. 10 and Eq. 13 for simplicity.

Attention Module Loss. Since the pixel-wise edge annotations provide spatial labels, it is easy to obtain the ground truth of attention. Let $\mathcal{T} = \{T^b, T^r, T^i, T^n, T^d\}$ be the ground-truth label of attention, where T^b specifies the non-edge pixels. $T^b_{i,j} = 1$ if the $(i,j)^{th}$ pixel is located on non-edge/background, otherwise $T^b_{i,j} = 0$. T^r, T^i, T^n, T^d indicate attention labels of REs, IEs, NEs and DEs respectively, which are obtained from $\mathcal{E} = \{E^r, E^i, E^n, E^d\}$,

$$T^k_{i,j} = \begin{cases} E^k_{i,j}, & \text{if } \sum_k E^k_{i,j} = 1, k \in \{r,i,n,d\} \\ 255, & \text{if } \sum_k E^k_{i,j} > 1, k \in \{r,i,n,d\} \end{cases}, \tag{11}$$

where k denotes the type of edges, $T^k_{i,j}$ and $E^k_{i,j}$ indicate the attention label and edge label of the $(i,j)^{th}$ pixel, respectively. The attention label equals the edge label if one pixel is only assigned one type of edge label, or it will be tagged 255 that will be ignored during training if one pixel has multiple types. It is should be noted that multi-labeled edges are used when training four decision heads for each type of edges, and only excluded when training the AM. The loss function $\mathcal{L}_{\mathrm{att}}$ of the AM is formulated as:

$$\mathcal{L}_{\mathrm{att}}(\mathcal{A},\mathcal{T}) = \sum_{k\in\{b,r,i,n,d\}} \ell_{\mathrm{foc}}\left(A^k, T^k\right), \tag{12}$$

$$\ell_{\mathrm{foc}}(A,T) = -\sum_{i,j}\big(T_{i,j}\alpha_2(1-A_{i,j})^{\gamma_2}\log(A_{i,j}) + (1-T_{i,j})(1-\alpha_2)A^{\gamma_2}_{i,j}\log(1-A_{i,j})\big), \tag{13}$$

where ℓ_{foc} indicates the Focal Loss [22] and $\mathcal{A}$ is the output of Attention Module. Note that α_2 and γ_2 are a balancing weight and a focusing parameter, respectively.

Total Loss. Finally, we optimize RINDNet by minimizing the total loss defined as:

$$\mathcal{L} = \lambda\mathcal{L}_{\mathrm{e}} + (1-\lambda)\mathcal{L}_{\mathrm{att}}, \tag{14}$$

where λ is the weight for balancing the two losses.

4.3. Training Details

Our network is implemented using PyTorch [30] and finetuned from a ResNet-50 model pre-trained on ImageNet [8]. Specifically, we adopt the Stochastic Gradient Descent optimizer with momentum=0.9, initial learning rate=10^{-5}, and we decay it by the "poly" policy on every epoch. We

Table 2. Ablation study to verify the effectiveness of each component in our proposed RINDNet.

Method	Reflectance			Illumination			Normal			Depth			Average		
	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP	ODS	OIS	AP
Ours	0.478	0.521	0.414	0.280	0.337	0.168	0.489	0.522	0.440	0.697	0.724	0.705	0.486	0.526	0.432
Ours w/o WL	0.422	0.468	0.357	0.280	0.321	0.180	0.476	0.515	0.425	0.693	0.713	0.700	0.468	0.504	0.416
Ours w/o AM	0.443	0.494	0.338	0.268	0.327	0.139	0.473	0.506	0.378	0.670	0.699	0.649	0.464	0.507	0.376
Ours w/o AM&WL	0.409	0.460	0.316	0.277	0.331	0.178	0.471	0.507	0.389	0.677	0.707	0.662	0.459	0.501	0.386

train the model for 70 epochs on one GPU with a batch size of 4. Moreover, we set $\beta = 4$ and $\gamma_1 = 0.5$ for $\mathcal{L}_{\text{e}}$; $\alpha_2 = 0.5$ and $\gamma_2 = 2$ for $\mathcal{L}_{\text{att}}$; and $\lambda = 0.1$ for total loss. In addition, following [38], we augment our dataset by rotating each image by four different angles of $\{0, 90, 180, 270\}$ degrees. Each image is randomly cropped to 320×320 during training while retaining the original size during testing.

5. Experiment Evaluation

We compare our model with 10 state-of-the-art edge detectors. HED [44], RCF [23], CED [41], DexiNed [31], and BDCN [14] exhibit excellent performance in general edge detection; DeepLabV3+ [6], CASENet [48] and DFF [16] show outstanding accuracy on semantic edge detection; DOOBNet [38] and OFNet [25] yield competitive results for occlusion edge detection. All models are trained on 300 training images and evaluated on 200 test images. In addition, more qualitative results are provided in the supplementary material. We evaluate these models with three metrics introduced by [2]: fixed contour threshold (ODS), per-image best threshold (OIS), and average precision (AP). Moreover, a non-maximum suppression [5] is performed on the predicted edge maps before evaluation.

5.1. Experiments on Four Types of Edges

Comparison with State of the Arts. To adapt existing detectors for four edge types simultaneously, they are modified in two ways: (1) The output $Y \in \{0,1\}^{W \times H}$ is changed to $\mathcal{Y} \in \{0,1\}^{4 \times W \times H}$. In particular, for DeepLabV3+ focusing on segmentation, the output layer of DeepLabV3+ is replaced by an edge path (same as DOOBNet [38] and OFNet [25], containing a sequence of four 3×3 convolution blocks and one 1×1 convolution layer) to predict edge maps. As shown in Table 1, ten compared models are symbolized as HED, RCF, CED, DexiNed, BDCN, CASENet, DFF, *DeepLabV3+, *DOOBNet and *OFNet, respectively. (2) DeepLabV3+, DOOBNet and OFNet only provide one edge prediction branch without structure suitable for multi-class predictions, thus we provide the second modification: the last edge prediction branch is expanded to four, and each branch predicts one type of edges. The modification is similar to the prediction approach of our model and aims to explore the capabilities of these models. They are symbolized as DeepLabV3+, DOOBNet, and OFNet, respectively.

Table 1 and Fig. 5 present the F-measure of the four types of edges and their averages. We observe that the proposed RINDNet outperforms other detectors over most metrics across the dataset. Essentially, [14, 23, 25, 31, 38, 41, 44] are designed for generic edge detection, so the specific features of different edges are not fully explored. Even if we extend the prediction branch of OFNet [25], DOOBNet [38] and DeepLabV3+ [6] to four for learning specific features respectively, the results are still unsatisfactory. DFF [16] could learn the specific cues in a certain extent by introducing the dynamic feature fusion strategy, but the performance is still limited. On the contrary, our proposed RINDNet achieves promising results by extracting the corresponding distinctive features based on different edge attributes.

Table 3. Ablation study on the choices of features or spatial cues from different layers for the proposed RINDNet.

Reference	RE&IE	NE&DE	Average		
			ODS	OIS	AP
Different-Layer Features	res_{1-3}	res_5	0.486	0.526	0.432
	res_{1-3}	res_{1-3}	0.467	0.499	0.422
	res_5	res_{1-3}	0.452	0.482	0.381
	res_5	res_5	0.464	0.489	0.396
Spatial Cues	f_{sp}^{1-3}	f_{sp}^{1-5}	0.486	0.526	0.432
	f_{sp}^{1-3}	f_{sp}^{1-3}	0.472	0.504	0.416
	f_{sp}^{1-5}	f_{sp}^{1-5}	0.478	0.512	0.418
	w/o f_{sp}	w/o f_{sp}	0.478	0.516	0.420

Table 4. Ablation study to verify the effectiveness of Decoder for the proposed RINDNet. SW refers to "Share Weight", 1^{st} and 2^{nd} refer to the first stream and the second stream in Decoder.

RE&IE-Decoder		NE&DE-Decoder		Average		
1^{st}	2^{nd}	1^{st}	2^{nd}	ODS	OIS	AP
√	√	√	w SW	**0.486**	**0.526**	**0.432**
√	×	√	×	0.457	0.492	0.398
√	×	√	w SW	0.476	0.514	0.415
√	√	√	w/o SW	0.474	0.517	0.408

Ablation Study. We first conduct the ablation study to verify the role of Weight Layer (WL) and Attention Module (AM) in RINDNet. In the experiments, each module is removed separately or together to construct multiple variants for evaluation, as shown in Table 2 (rows 2 – 4). Intuitively, WL plays a significant role for REs and IEs. Especially for REs, in terms of ODS, OIS and AP, the WL improves the performance significantly from 42.2%, 46.8% and 35.7% to 47.8%, 52.1% and 41.4%, respectively. Besides, with AM, RINDNet achieves noticeable improvements for all types of edges, as shown in row 1 and row 3 of Table 2. This illustrates the effectiveness of the proposed AM for capturing the distinctions between different edges. Overall, the cooperation of WL and AM allows RINDNet to successfully

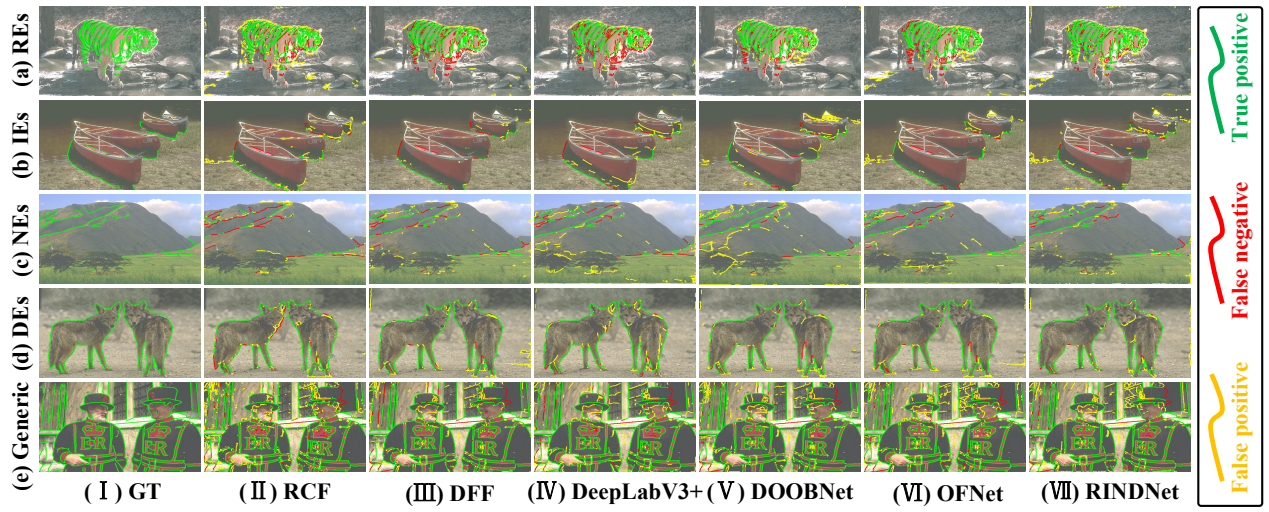


Figure 6. Qualitative comparison for (a) Reflectance Edges, (b) Illumination Edges, (c) Normal Edges, (d) Depth Edges and (e) Generic Edges (best viewed in color: "**green**" for true positive, "**red**" for false negative (missing edges), and "**yellow**" for false positive). We provide visualization results of the top 6 scores in this figure, more results can be found in the **supplementary material**.

capture the specific features of each type of edges and thus delivers remarkable performance gain.

Next, we perform an experiment to verify the effectiveness of different-layer features for detecting REs/IEs and NEs/DEs. As shown in Table 3 (rows 1 – 4), the combination based on edge attributes (row 1) performs better than using other choices (rows 3 – 4). Besides, we also explore the impact of choosing different spatial cues for REs/IEs and NEs/DEs, and report the quantitative results in Table 3 (rows 5 – 7). Similarly, there are average performance drops in different combinations of spatial cues (rows 6 – 7). Basically, ours (row 5) is the reasonable combination that helps achieve the best performance.

We also design careful ablation experiments (Table 4) to study the effectiveness of each stream in Decoder. The two-stream design combined with share weight (referred to as SW) for NEs/DEs performs better together (row 1) than using either of them separately (rows 3 – 4). More detailed results are provided in the supplementary material.

5.2. Experiments on Generic Edges

To fully examine the performance of RINDNet, we modify and test it on generic edges setting (the ground truth of four types of edges are merged to one ground-truth edge map). The outputs of four decision heads are combined and fed into a final decision head ψ_{e} (one 1×1 convolution layer) to predict generic edges: $P = \psi_{\mathrm{e}}\left([Y^r, Y^i, Y^n, Y^d]\right)$. Moreover, the original ground-truth labels of Attention Module (AM) are unavailable in this setting. Thus we take ground-truth labels of generic edges as the supervisions of AM, so that AM could capture the location of generic edges.

We report the quantitative results over generic edges in Table 5 and Fig. 5 (e). Note that CED is pre-trained on HED-BSDS. In contrast, our model is trained from scratch and still achieves competitive results. This confirms the integration capability of our model, especially considering that RINDNet is not specially designed for generic edges. Some qualitative results on BSDS-RIND are shown in the last row in Fig. 6.

Table 5. Comparison of generic edge detection on **BSDS-RIND**.

Method	ODS	OIS	AP
HED [44]	0.786	0.805	**0.834**
CED [41]	**0.801**	**0.814**	**0.824**
RCF [23]	0.771	0.791	0.800
BDCN [14]	0.789	0.803	0.757
DexiNed [31]	0.789	0.805	0.816
CASENet [48]	0.792	0.806	0.786
DFF [16]	0.794	0.806	0.767
DeepLabV3+ [6]	0.780	0.792	0.776
DOOBNet [38]	0.790	0.805	0.809
OFNet [25]	0.794	0.807	0.800
RINDNet (Ours)	**0.800**	**0.811**	0.815

6. Conclusions

In this work, we study edge detection on four types of edges including *Reflectance Edges*, *Illumination Edges*, *Normal Edges* and *Depth Edges*. We propose a novel edge detector RINDNet that, for the first time, simultaneously detects all four types of edges. In addition, we contribute the first public benchmark with four types of edges carefully annotated. Experimental results illustrate that RINDNet yields promising results in comparison with state-of-the-art edge detection algorithms.

Acknowledgements. This work is supported by Fundamental Research Funds for the Central Universities (2019JBZ104) and National Natural Science Foundation of China (61906013, 51827813). The work is partially done while Mengyang Pu was with Stony Brook University.

References

[1] David Acuna, Amlan Kar, and Sanja Fidler. Devil is in the edges: Learning semantic boundaries from noisy annotations. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11075–11083, 2019. 2

[2] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, 2010. 2, 3, 7

[3] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4380–4389, 2015. 2

[4] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In *Int. Conf. Comput. Vis.*, pages 504–512, 2015. 2

[5] John F. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986. 2, 7

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Eur. Conf. Comput. Vis.*, pages 801–818, 2018. 6, 7, 8

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3213–3223, 2016. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 248–255, 2009. 6

[9] Ruoxi Deng and Shengjun Liu. Deep structural contour detection. In *ACM Int. Conf. Multimedia*, pages 304–312, 2020. 2

[10] Ruoxi Deng, Chunhua Shen, Shengjun Liu, Huibing Wang, and Xinru Liu. Learning to predict crisp boundaries. In *Eur. Conf. Comput. Vis.*, pages 562–578, 2018. 2

[11] Piotr Dollár, Zhuowen Tu, and Serge J. Belongie. Supervised learning of edges and object boundaries. In *IEEE Conf. Comput. Vis. Pattern Recog.*, volume 2, pages 1964–1971, 2006. 2

[12] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 2

[13] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *Int. Conf. Comput. Vis.*, pages 991–998, 2011. 2

[14] Jianzhong He, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. Bi-directional cascade network for perceptual edge detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3828–3837, 2019. 1, 2, 6, 7, 8

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 770–778, 2016. 4

[16] Yuan Hu, Yunpeng Chen, Xiang Li, and Jiashi Feng. Dynamic feature fusion for semantic edge detection. In *IJCAI*, pages 782–788, 2019. 1, 2, 6, 7, 8

[17] André Peter Kelm, Vijesh Soorya Rao, and Udo Zölzer. Object contour and edge detection with refinecontournet. In *International Conference on Computer Analysis of Images and Patterns*, pages 246–258. Springer, 2019. 2

[18] Kichang Kim, Akihiko Torii, and Masatoshi Okutomi. Joint estimation of depth, reflectance and illumination for depth refinement. In *Int. Conf. Comput. Vis.*, pages 199–207, 2015. 1

[19] Josef Kittler. On the accuracy of the sobel edge detector. *Image Vis. Comput.*, 1(1):37–42, 1983. 2

[20] Iasonas Kokkinos. Pushing the boundaries of boundary detection using deep learning. *Int. Conf. Learn. Represent.*, 2015. 2

[21] Joseph J. Lim, C. Lawrence Zitnick, and Piotr Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3158–3165, 2013. 2

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Int. Conf. Comput. Vis.*, pages 2980–2988, 2017. 6

[23] Yun Liu, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. Richer convolutional features for edge detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3000–3009, 2017. 1, 2, 6, 7, 8

[24] Yu Liu and Michael S Lew. Learning relaxed deep supervision for better edge detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 231–240, 2016. 2

[25] Rui Lu, Feng Xue, Menghan Zhou, Anlong Ming, and Yu Zhou. Occlusion-shared and feature-separated network for occlusion relationship reasoning. In *Int. Conf. Comput. Vis.*, pages 10343–10352, 2019. 1, 2, 4, 6, 7, 8

[26] Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and Luc Van Gool. Convolutional oriented boundaries. In *Eur. Conf. Comput. Vis.*, pages 580–596. Springer, 2016. 2

[27] David Marr. Vision: A computational investigation into the human representation and processing of visual information, 1982. 1

[28] David R. Martin, Charless C. Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–549, 2004. 2

[29] David A Mély, Junkyung Kim, Mason McGill, Yuliang Guo, and Thomas Serre. A systematic comparison between visual cues for boundary detection. *Vis. Res.*, 120:93–107, 2016. 2

[30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *Adv. Neural Inform. Process. Syst.*, 2017. 6

[31] Xavier Soria Poma, Edgar Riba, and Angel Sappa. Dense extreme inception network: Towards a robust cnn model for

edge detection. In *IEEE Winter Conf. Appl. Comput. Vis.*, pages 1923–1932, 2020. 2, 6, 7, 8

[32] Xuebin Qin, Shida He, Zichen Zhang, Masood Dehghan, and Martin Jagersand. Bylabel: A boundary based semi-automatic image annotation tool. In *IEEE Winter Conf. Appl. Comput. Vis.*, pages 1804–1813, 2018. 3

[33] Michaël Ramamonjisoa, Yuming Du, and Vincent Lepetit. Predicting sharp and accurate occlusion boundaries in monocular depth estimation using displacement fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 14648–14657, 2020. 1, 2

[34] Xiaofeng Ren, Charless C Fowlkes, and Jitendra Malik. Figure/ground assignment in natural images. In *Eur. Conf. Comput. Vis.*, pages 614–627, 2006. 2

[35] Steven Seitz. Edge detection. https://courses.cs.washington.edu/courses/cse455/04wi/lectures/edge.pdf. 1

[36] Wei Shen, Xinggang Wang, Yan Wang, Xiang Bai, and Zhijiang Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3982–3991, 2015. 2

[37] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Eur. Conf. Comput. Vis.*, pages 746–760, 2012. 2

[38] Guoxia Wang, Xiaochuan Wang, Frederick WB Li, and Xiaohui Liang. Doobnet: Deep object occlusion boundary detection from an image. In *ACCV*, pages 686–702. Springer, 2018. 1, 2, 5, 6, 7, 8

[39] Peng Wang and Alan Yuille. Doc: Deep occlusion estimation from a single image. In *Eur. Conf. Comput. Vis.*, pages 545–561, 2016. 1, 2

[40] Xiaolong Wang, David F. Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 539–547, 2015. 1

[41] Yupei Wang, Xin Zhao, and Kaiqi Huang. Deep crisp boundaries. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3892–3900, 2017. 1, 2, 6, 7, 8

[42] Holger Winnemöller, Jan Eric Kyprianidis, and Sven C. Olsen. Xdog: An extended difference-of-gaussians compendium including advanced image stylization. *Comput. Graph.*, 36(6):740–753, 2012. 2

[43] Qi Wu, Wende Zhang, and B. V. K. Vijaya Kumar. Strong shadow removal via patch-based shadow edge detection. In *International Conference on Robotics and Automation*, pages 2177–2182, 2012. 1

[44] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Int. Conf. Comput. Vis.*, pages 1395–1403, 2015. 1, 2, 6, 7, 8

[45] Dan Xu, Wanli Ouyang, Xavier Alameda-Pineda, Elisa Ricci, Xiaogang Wang, and Nicu Sebe. Learning deep structured multi-scale features using attention-gated crfs for contour prediction. In *Adv. Neural Inform. Process. Syst.*, pages 3961–3970, 2017. 2

[46] Fan Yang, Lei Zhang, Sijia Yu, Danil Prokhorov, Xue Mei, and Haibin Ling. Feature pyramid and hierarchical boosting network for pavement crack detection. *IEEE Trans. Intell. Transp. Syst.*, 21(4):1525–1535, 2019. 1

[47] Jimei Yang, Brian Price, Scott Cohen, Honglak Lee, and Ming-Hsuan Yang. Object contour detection with a fully convolutional encoder-decoder network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 193–202, 2016. 2

[48] Zhiding Yu, Chen Feng, Ming-Yu Liu, and Srikumar Ramalingam. Casenet: Deep category-aware semantic edge detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5964–5973, 2017. 1, 2, 6, 7, 8

[49] Zhiding Yu, Weiyang Liu, Yang Zou, Chen Feng, Srikumar Ramalingam, B. V. K. Vijaya Kumar, and Jan Kautz. Simultaneous edge alignment and learning. In *Eur. Conf. Comput. Vis.*, pages 388–404, 2018. 2

[50] Mingmin Zhen, Jinglu Wang, Lei Zhou, Shiwei Li, Tianwei Shen, Jiaxiang Shang, Tian Fang, and Long Quan. Joint semantic segmentation and boundary detection using iterative pyramid contexts. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13666–13675, 2020. 2