Learning Local Contrast for Crisp Edge Detection

Xiao-Nan Fang
Tsinghua Universiy
Beijing, China
fangxn18@mails.tsinghua.edu.cn

Song-Hai Zhang
Tsinghua University
Beijing, China
shz@tsinghua.edu.cn

Abstract

In recent years, the accuracy of edge detection on several benchmarks has been significantly improved by deep-learning-based methods. However, the prediction of deep neural networks is usually blurry and needs further post-processing including non-maximum suppression and morphological thinning. In this paper, we demonstrate that the blurry effect arises from the binary cross entropy loss, and crisp edges could be obtained directly from deep convolutional neural networks. We propose to learn edge maps as the representation of local contrast with a novel local contrast loss. The local contrast is optimized in a stochastic way to focus on specific edge directions. Experiments show that the edge detection network trained with local contrast loss achieves high accuracy comparable to previous methods and dramatically improves the crispness. We also present several applications of the crisp edges, including image completion, image retrieval, sketch generation and video stylization.

Keywords: Please provide 4–6 keywords, separated with comma.

1. Introduction

Edge detection is a basic problem in computer vision. Edges can reveal object boundaries and prominent structures in images, which is meaningful for visual semantics understanding and image editing. Edge detection methods usually output a float value for each pixel which represents the edge confidence. Then the confidence map is binarized with a user-specified threshold, resulting in a two-value edge map. Traditional edge detection methods [16, 3, 18, 25, 29, 31] rely on low-level features and could predict properly aligned *crisp* edges. However, these methods are unlikely to suppress the high-contrast textures while maintaining the low-contrast object boundaries.

The success of convolutional neural networks (CNN) in fundamental computer vision problems such as image recognition [34, 13], object detection [9, 30] and segmentation [4, 12] shows CNN's great capability of encoding visual semantics from large-scale datasets. Recently proposed CNN-based edge detectors [40, 24, 21, 11] significantly improved the accuracy of edge prediction on several benchmarks [1, 27, 26] and achieved performance superior to human. In these prior papers, edge detection is regarded as a per-pixel classification problem, similar to semantic segmentation task [4]. A drawback of these methods is that the output of CNN is blurry. Post-processing steps, including non-maximum suppression and morphological thinning, are required for the computation of precision, recall and F-measure. As pointed out by Huan et al. [15], the mixing operation on feature maps, side outputs as well as upsampling operations lead to thick and blurry edge predictions. CED [37] tried to tackle the problem with U-net [32] structure. Some prior works [6, 15] proposed auxiliary losses and specific network structure modification to predict thin edges, i.e., to improve the crispness. These strategies could reduce the thickness of output edges to some extent, but the accuracy of CNN's raw output is still far from satisfactory.

In this paper, we claim that the CNN architecture can predict crisp edges if it is trained with appropriate loss function. Different from the semantic segmentation problem [4], viewing edge detection as a binary classification problem is not a good formulation because the proportion of the "edge" category is far less than that of the "non-edge" category. During training, a reweighted binary cross entropy (BCE) loss is usually adopted to balance the prediction of the two categories, where "edge" pixels have weights much larger than "non-edge" pixels. In addition, this factor needs to be carefully tuned for different training sets. The emphasis on edge label in the loss function results in thicker and blurrier predictions. Owing to its apparent drawback, we totally discard the BCE loss in this work. Our core observation is that edges should reflect the local contrast at semantic level. We propose a novel local contrast loss to train the edge detection network. An affinity term is used to measure the similarity of neighboring pixel pairs. We desire that pixels of the same type (both edge or both non-edge) should have high similarity, while pixels of different types should have low similarity. Our method does not impose regularization on the side outputs because the low-resolution edge predic-

tions are imprecise and would lead to thick structures after upsampling. Directly minimizing the whole loss function is inefficient, so we introduce a stochastic strategy that optimizes the local contrast for specific edge directions in each training iteration. The most significant difference between our work and previous work is that the BCE loss is removed and the local contrast loss plays an essential role in the optimization process. We also introduce an L_1 regression loss to eliminate the ambiguity. The CNN architecture proposed in CATS [15] is adopted here.

We evaluated our method on two well-known benchmarks: BSDS [1] and NYUDv2 [27]. The results indicate that using the proposed local contrast loss could significantly improve the crispness. Without any post-processing operation, the raw output of our method significantly outperforms previous models that were trained mainly with BCE loss. With standard post-processing process, our method's performance is still comparable to the state-of-the-art approaches, though predicting thinner edges, namely less positive labels, is inherently unhelpful to improve the F-measure. We also used a density measure similar to [36] and variance of edge confidence to evaluate the edge simplicity, showing the improvement on crispness gained from the local contrast loss. Ablation study was conducted to study the effect of side outputs, the choice of balancing parameters and alternative network design.

The CNN-predicted crisp edge map has good visual quality, and is also useful in other scenarios. We show its capability for four applications: image completion, sketch-based image retrieval, sketch generation and video stylization. As proposed in [28], image completion could be separate into two steps: the completion of edge map and the completion of RGB channels with the guidance of edges. We show that the CNN-predicted crisp edge could serve as a better intermediate guidance. Sketches are widely utilized in image retrieval as the input query. The sketch-based image retrieval model [23] learns a joint feature embedding for the sketch domain and image domain, and the similarity is evaluated in the feature space. We show that crisp edge maps extracted from training images could be added into the training sketch set to improve the retrieval accuracy. Generating thin sketch drawings are uneasy due to the difficulty of optimization. We exhibit that the proposed learning framework could be easily adapted to a sketch dataset [19] with additional finetuning. Besides, we show that the crisp edges could improve visual effect of a video stylization algorithm [39].

In summary, our main contributions are:

- We propose that a novel local contrast loss is superior to BCE loss for training deep edge detection network. Our approach achieves high accuracy comparable to state-of-the-art detectors under standard evaluation while significantly improves the edge crispness.
- We propose an efficient and effective optimization scheme to train the CNN model with local contrast loss.
- We propose four applications of the new crisp edge detection model: image completion, image retrieval, sketch generation and video stylization.

2. Related Work

Edge detection was studied in the early years of computer vision as a fundamental problem. Sobel operator [16] and zero-crossing based edge detection [35] are pioneering works. The Canny edge detector [3] is built on the Sobel operator and is more robust due to the bi-threshold design. There are various methods focusing on low-level color and texture features, including Statistical Edges [18], Pb [25] and mean-shift [29]. The resulting edges of those approaches are usually crisp. However, they usually fail to capture high-level semantics and suppress strong textures.

As convolutional neural network (CNN) shows its capability of modeling visual semantics, CNN based edge detectors are developed in recent years. HED [40], COB [24], RCF [21], BDCN [11] are representative methods. These methods adopt VGG-net [34] pretrained on ImageNet [5] as the backbone to extract multi-level semantic features, and use different strategies to fuse these features for edge prediction. In general, hierarchical edge predictions are obtained from different feature layers, and then these side outputs are fused for the final prediction. HED uses the final layer of each stage of VGG to generate edge maps at different scales (from original resolution to $\frac{1}{16} \times \frac{1}{16}$ size), while RCF fuses all feature layers that have the same resolution. COB additionally predicts the contour orientations and constructs a hierarchical boundary map. It can also provide object proposals for object detection and semantic segmentation. BDCN utilizes a more complicated fusion strategy for side outputs combining a deep-to-shallow pass and a shallow-to-deep pass, which can improve the accuracy of detection.

Typically, learning-based methods are aware of high-level semantics, and can predict more accurate results on benchmarks such as Berkeley Segmentation Dataset [1]. The performance could be even better than human annotators. However, these method could only perform well under the standard evaluation protocol, which contains post-processing steps including non-maximum suppression and morphological thinning. The raw output of neural networks is cluttered and can not be directly used for many applications.

Many researchers have investigated how to improve the crispness of edge maps outputted by CNN model. CED [37] tries to tackle this problem by modifying the network structure. It progressively increases the resolution of feature

maps by a backward-refining scheme. Other following works considered to introduce auxiliary loss functions for better crispness supervision. For example, the reciprocal of the dice coefficient is adopted in LPCB [6]. CATS [15] introduces a boundary tracing function and a texture suppression function as a complement of the original weighted BCE loss. These additional functions considers the statistics in a local patch to suppress thick structure and errors on texture regions. A novel unmixing block is introduced in CATS [15] to fuse multi-resolution side outputs. The raw output from CNN are directly compared to ground-truth label to obtain the precision, recall and F-measure without post-processing, which could reflect its crispness. These methods could reduce the thickness of the predicted edges to some extent. As reported in CATS [15], there exists a dramatic performance gap between the raw output and the post-processed edge map (around 0.1 on BSDS [1] and around 0.3 on NYUDv2 [27]). We propose a novel contrast loss that can dramatically improve the edge crispness.

3. Method

Different from prior works that investigated various CNN architectures and feature fusion strategies for edge detection, our work focus on the design of loss function and training strategy. We adopt the RCF-CATS network architecture proposed in [15], which uses the VGG-16 [34] backbone for feature extraction and predicts multi-resolution edge maps by fusing all layers of features. The multi-resolution edge maps are fused with an unmixing block. This block generates weight maps from the multi-resolution side outputs through 3 convolution layers. The final edge prediction is the weighted average of multi-resolution edge maps. For more details about the network architecture, please refer to the prior works [21, 15]. We propose to use a novel local contrast loss, instead of the well-known BCE loss, for network training. Meanwhile we remove the supervision on side outputs because the coarse outputs tend to be blurry and might hinder the crispness of the final edge map.

Fig. 1 illustrates the differences between our method and traditional methods. We take the RCF model [21] as an example. Note that for previous models the multi-resolution side outputs are all supervised by BCE loss. Since we do not regularize these maps, they could be arbitrary latent codes in our model.

3.1. Rethinking the Classification Formulation

The edge detection problem has been widely regarded as a per-pixel binary classification problem. Given the ground-truth edge mask E and non-edge mask N, the error of output probability map O is measured by the weighted binary

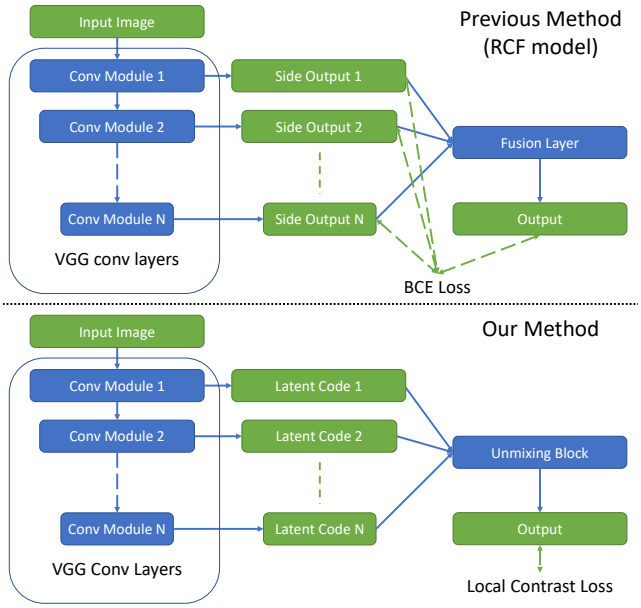


Figure 1. Comparison between traditional deep edge detection method and our crisp edge detection method. We adopt a novel local contrast loss to replace BCE loss and also remove the losses for side outputs that lead to thick structures.

cross entropy loss:

$$L_{bce} = -\sum_{p}(w_1 E(p)\log O(p) + w_2 N(p)\log(1-O(p))), \quad (1)$$

where p represents a pixel. Note that in practice, some pixels labelled by minor annotators are regarded as unknown, so they will not be considered in the loss function. This loss is computed on each pixel independently. However, the edges reflects local structure and the relation in the local patch should be taken into consideration and directly modelled. Since the label distribution is highly unbalanced, the weight w_1 for edge pixels is much higher than w_2. Since it is difficult to find the optimal well-aligned edges, the network tends to predict much more pixels around strong structures as edge in order to decrease the training loss. Another problem is that the several edge maps (side outputs) are predicted from multi-resolution features. The coarse maps could only capture major structures and need to be upsampled using bilinear interpolation. Therefore, the final edge map, no matter how the side outputs are fused, has thick structures in confident region and is blurry in ambiguous region. Auxiliary loss functions proposed in [6, 15] could relieve this problem to some extent. It is because they explicitly model the inter-pixel relation. However, these methods still predict thick edges because they retain the BCE loss as major supervision. To obtain crisp edges from CNN, we totally discard the BCE loss and propose a novel loss

for better supervision. We will first introduce the local contrast on the output of deep neural networks, and derive a local contrast loss for an edge map. Then we provide an efficient and effective strategy to optimize the network with local contrast loss.

3.2. Deep Local Contrast

The basic way to detect edge on an image I is computing the gradient field in the color space or monochrome space. In concrete, to measure the contrast in direction $\mathbf{v} = \overrightarrow{pq}$, where p and q are neighboring pixels, we can estimate the length of directional derivative:

$$|\frac{\partial I}{\partial \mathbf{v}}| = |I(q) - I(p)|. \tag{2}$$

This index reflects the strength of edges at a certain direction. Specifically, the Sobel [16] operator evaluates the edge strength in horizontal and vertical direction.

For edge detection tasks, pixels are divided into two categories: edge and non-edge. Given the output of CNN O, we define the similarity of pixel p and q as:

$$s(p, q) = e^{-|O(p)-O(q)|/\sigma}. \tag{3}$$

We restrict the output $O(p) \in (0, 1)$ by applying a sigmoid layer at the end of the neural network. In our experiments, we set $\sigma = 0.05$. Let E represents the mask of edge pixels and N represents the mask of non-edge pixels. Ideally, pixels belonging to different categories should have lower similarity (namely higher contrast) while those in the same category should have higher similarity (namely lower contrast). Thus we can construct three loss terms for pixel pair (p, q):

$$l_{pos}(p, q) = E(p)E(q)\left(1 - s(p, q)\right), \tag{4}$$

$$l_{neg}(p, q) = N(p)N(q)\left(1 - s(p, q)\right), \tag{5}$$

$$l_{cross}(p, q) = \left(E(p)N(q) + N(p)E(q)\right) s(p, q). \tag{6}$$

The first term l_{pos} is effective when p, q are both edge pixels; the second term l_{neg} is effective when they are non-edge pixels, while the last term supervises the cross-type pairs. Then we could define corresponding losses on the pixel p as the sum of all paired terms in its neighborhood:

$$\hat{l}_{pos}(p) = \sum_{q \in \mathcal{N}_{s_1}(p)} l_{pos}(p, q), \tag{7}$$

$$\hat{l}_{neg}(p) = \sum_{q \in \mathcal{N}_{s_2}(p)} l_{neg}(p, q), \tag{8}$$

$$\hat{l}_{cross}(p) = \sum_{q \in \mathcal{N}_{s_3}(p)} l_{cross}(p, q). \tag{9}$$

Here $\mathcal{N}_s(p)$ represents an $s \times s$ patch centered at p. We adopt different neighborhood patch sizes s_1, s_2, s_3 for computing $\hat{l}_{pos}, \hat{l}_{neq}, \hat{l}_{cross}$. Since the edges are sparse in the image, we choose smaller s_1 to focus on local connections. To suppress large textured region, the choice of s_2 should be relatively large. Intuitively, setting a large patch size could improve the learning of image semantics, but the local details could be neglected. We set the sizes as $s_1 = 3, s_2 = 19, s_3 = 11$ for our model, and we will discuss the choice of patch size in experimental section. Each contrast loss is then summed up across the image and normalized into $[0, 1]$ by dividing it by the number of effective pairs.

The local contrast term does not specify the category label (0 or 1). Therefore, we introduce L_1 loss for regularization. Let G represent the ground-truth edge map, the reconstruction loss is defined as:

$$L_{rec} = \sum_p |O(p) - G(p)|. \tag{10}$$

We use different notations for ground-truth map G and E because there could be multiple annotations for one image. We would discuss the definition of G and E in the experimental section.

3.3. Optimization Scheme

Similar to the tracing loss proposed in [15], the loss functions in Eq. 7, 8 and 9 consider the edge distribution in the whole patch. One could traverse all neighboring pixel pairs in the image and calculate the contrast term on these pairs. However, this loss function requires large amounts of computation and is uneasy to optimize. We provide a simple approach to optimize the objective by estimating the whole contrast with one paired contrast in the neighborhood. We randomly choose a direction vector v, and compute all neighboring pixel pairs (p, q) satisfying $q = p + v$ on the edge map. Note that the neighborhood size of $l_{pos}, l_{neg}, l_{cross}$ are different, so we sample three different directions for the three terms independently. Then we derive the stochastic loss function:

$$L_{pos} = \frac{\sum_p l_{pos}(p, p + v_1)}{\sum_p E(p)E(p + v_1)}, \tag{11}$$

$$L_{neg} = \frac{\sum_p l_{neg}(p, p + v_2)}{\sum_p N(p)N(p + v_2)}, \tag{12}$$

$$L_{cross} = \frac{\sum_p l_{cross}(p, p + v_3)}{\sum_p \left(E(p)N(p + v_3) + N(p)E(p + v_3)\right)}. \tag{13}$$

The total loss function used in practice is:

$$L = \lambda_p L_{pos} + \lambda_n L_{neg} + \lambda_c L_{cross} + \lambda_r L_{rec}. \tag{14}$$

The offset vector $v_i, i \in \{1, 2, 3\}$ is uniformly sampled in the $s_i \times s_i$ region. Note that the vector v_i represents the normal direction of a group of edges, meaning that in each iteration the model learns to predict edges at a certain direction. The step size of v_i could also vary so that image structures at different scales could be taken into consideration. The model would learn the edges at all directions and multiple scales after enough training iterations, equivalent to the minimization of the total contrast loss term described in Eq. 7, 8 and 9. We will display the effectiveness of the stochastic optimization strategy in ablation study.

4. Experiment

4.1. Datasets and Settings

We trained and evaluated our local-contrast-based model on two popular edge detection benchmarks: BSDS [1] and NYUDv2 [27]. The VGG-16 parameters [34] pretrained on ImageNet [5] was adopted as initialization for feature-extraction module. The learning rate was set as 0.0001 initially and was decreased by a factor of 0.1 every 10 epochs. The model was trained over 30 epochs. We adopted standard data augmentation including scaling, rotation and horizontal flipping.

For BSDS dataset, 300 images are used for training and the test set contains 200 images. There are several ground-truth edge maps created by different annotators. These edge maps are averaged and the result is denoted by Avg. The pixels labeled by enough annotators are considered as confident positive samples, and we do not calculate the loss on other ambiguous pixels. In concrete, the edge mask E, non-edge mask N for contrast loss and the guidance G for reconstruction loss are defined as:

$$E(p) = \begin{cases} 1, Avg(p) > 0.33, \\ 0, \text{otherwise}. \end{cases} \tag{15}$$

$$N(p) = \begin{cases} 1, Avg(p) = 0, \\ 0, \text{otherwise}. \end{cases} \tag{16}$$

$$G(p) = 1 - N(p). \tag{17}$$

We view all pixels labelled by at least one annotator as positive for the calculation of L_1 loss to encourage the prediction of positive label. However, pixels with little confidence are not calculated in the local contrast loss function, where $E(p) = N(p) = 0$. We set the weighting factors $\lambda_p = 0.5$, $\lambda_n = 0.5$, $\lambda_c = 1$ and $\lambda_r = 1$ for BSDS dataset.

The NYUDv2 training set contains 795 images and the corresponding test set contains 654 images. We trained and evaluated our model using RGB images only. We set the weighting factors $\lambda_p = 0$, $\lambda_n = 1$, $\lambda_c = 1$ and $\lambda_r = 1$ for the experiment because we observe that the output of NYUDv2 model is thicker than that of BSDS model and it is more important to suppress texture regions. The setting of learning rate is the same as BSDS. Since there is only one ground-truth edge annotation in this dataset, we directly set

$$E(p) = G(p) = Avg(p), N(p) = 1 - Avg(p), \tag{18}$$

4.2. Standard Evaluation

The standard evaluation protocol (SEval) has been widely adopted in prior works. It contains a non-maximum suppression step and a morphological thinning step before matching. The predicted edge map is evaluated by matching the ground-truth labels. The threshold for tolerant matching distance is set as 0.0075 for BSDS and 0.011 for NYUDv2 as previous works did. The output map is binarized under different thresholds in $(0, 1)$. Precision, recall and F-measure are calculated for each binary map. The optimal dataset scale (ODS) and optimal image scale (OIS) value are computed for both datasets.

Table 1 reports the ODS and OIS F-measure of different methods on two datasets. Our method achieves accuracy comparable to state-of-the-art methods such as CATS [15] on standard evaluation, indicating that the local contrast loss provides adequate guidance for the edge detection task. We need to emphasize that increasing the thickness of edge map is helpful to improve the F-measure, especially the OIS metric, as more edges are predicted with different confidence. However, it would hinder the visual quality of the edge map and those type of edge map is difficult to be used in other applications. Fig. 2 and 3 exhibit some qualitative edge results. Additional result of our method is provided in the supplementary material.

4.3. Crispness Evaluation

The crispness-emphasized evaluation (CEval) is proposed by Huan et al. [15], in which the raw output of CNN is directly matched with ground-truth labels to obtain precision and recall, without any post-processing. We follow this setting and report the ODS and OIS F-measure in Table 2. Our method achieves significant improvement under CEval, which could also be easily observed in Fig. 2 and 3. The CEval accuracy on BSDS dataset gets much closer to the SEval accuracy, which means that the raw outputs may be directly used in many applications.

We also introduce another density metric, similar to the sparsity term proposed in [36], to evaluate the crispness of the edge map. We compute the average "density of the edge map by taking the sum in 5×5 patches. The edge density metric is defined as:

$$D = \frac{\sum_p \left(O(p) \sum_{q \in \mathcal{N}_5(p)} O(q) \right)}{\sum_p O(p)}. \tag{19}$$

Previous deep-learning-based methods tend to predict ambiguous values in $(0, 1)$ using multi-scale fusion strategy.

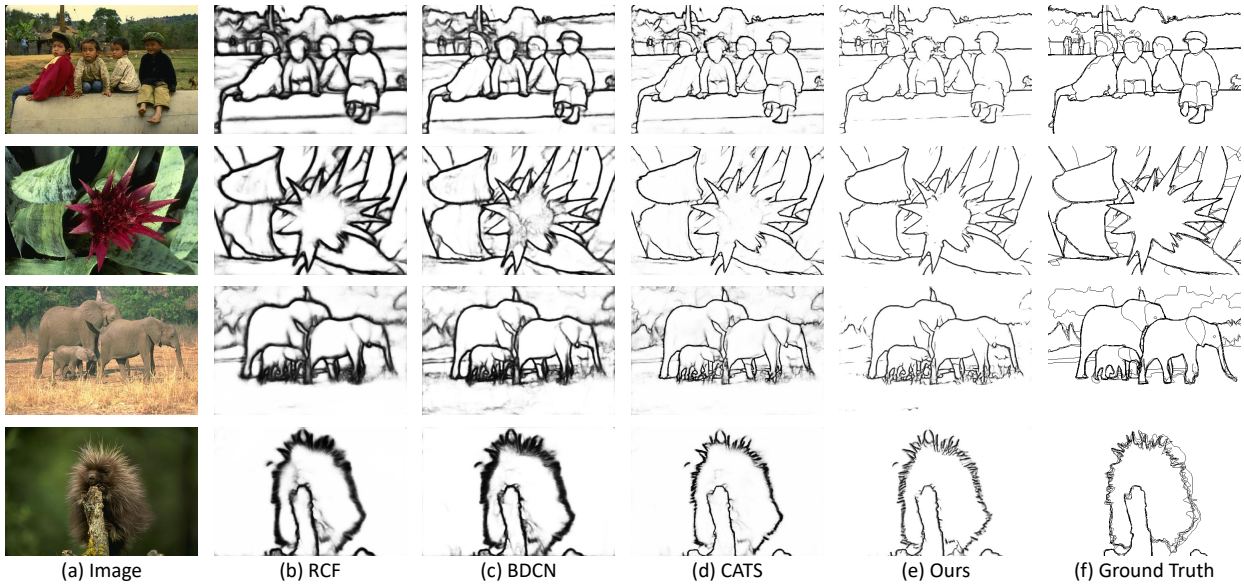


Figure 2. Qualitative comparison on BSDS dataset [1]. Our methods is compared with prior works including RCF [21], BDCN [11] and CATS [15].

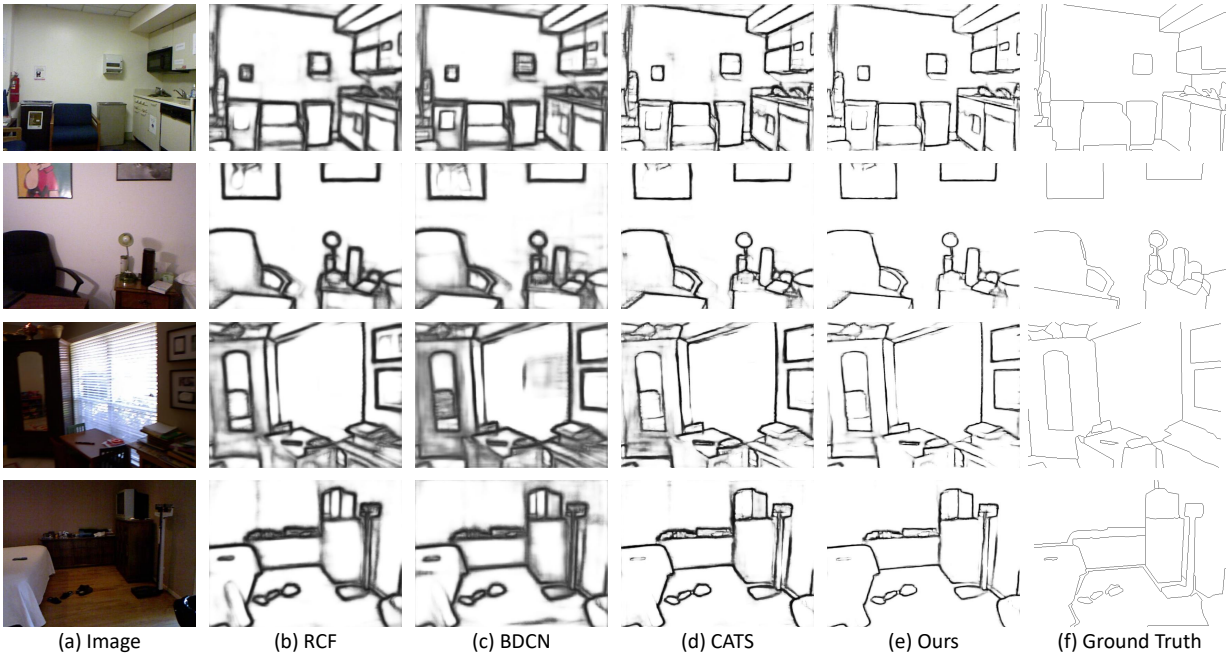


Figure 3. Qualitative comparison on NYUDv2 dataset [27]. Our methods is compared with prior works including RCF [21], BDCN [11] and CATS [15].

It can improve the performance under F-measure at the expense of visual quality and simplicity. We compute the variance of non-zero pixels to reveal the distribution of edge confidence.

The result in Table 3 shows that our method produces thinner edges with lower density. The high variance of our method indicates that the proposed method tend to make more confident predictions (either 0 or 1) compared with RCF and CATS-RCF.

4.4. Ablation Study

In this subsection, we investigate the influence of network design, loss function and side outputs. The ablation study is conducted on BSDS and the result is presented in Table 4.

Without the unmixing block (namely the final convolu-

Table 1. Standard evaluation on BSDS dataset [1] and NYUDv2 dataset [27]. We report and compare the ODS and OIS F-measure for the post-processed edge map.

Method	BSDS		NYUDv2	
	ODS	OIS	ODS	OIS
Human	0.803	0.803	-	-
OEF [10]	0.746	0.770	-	-
N^4-Fields [8]	0.753	0.769	-	-
DeepContour [33]	0.757	0.776	-	-
HFL [2]	0.767	0.788	-	-
CEDN [42]	0.788	0.804	-	-
DeepBoundary [17]	0.789	0.811	-	-
COB [24]	0.793	0.820	-	-
CED [37]	0.794	0.811	-	-
AMH-Net [41]	0.798	0.829	-	-
DCD [20]	0.799	0.817	-	-
LPCB [6]	0.800	0.816	0.739	0.754
HED [40]	0.793	0.811	0.722	0.737
RCF [21]	0.799	0.815	0.745	0.759
BDCN [11]	0.807	0.822	0.748	0.762
CATS-RCF [15]	0.805	0.822	0.752	0.765
Ours	0.805	0.818	0.749	0.761

Table 2. Crispness-emphasized evaluation on BSDS dataset [1] and NYUDv2 dataset [27]. We report and compare the ODS and OIS F-measure for raw edge map.

Method	BSDS		NYUDv2	
	ODS	OIS	ODS	OIS
HED	0.588	0.608	0.387	0.404
RCF	0.585	0.604	0.398	0.413
BDCN	0.636	0.650	0.426	0.450
CATS-RCF	0.705	0.716	0.474	0.488
Ours	0.766	0.776	0.557	0.568

Table 3. Results of edge density and confidence variance on BSDS dataset [1] and NYUDv2 dataset [27]. These two metrics are computed on raw edge maps.

Dataset	Method	Density	Variance
BSDS	RCF	13.34	0.069
	CATS-RCF	11.66	0.083
	Ours	11.18	0.110
NYUDv2	CATS-RCF	13.80	0.090
	Ours	13.04	0.097

tional fusion layers) introduced by [15], the network turns back to the RCF architecture, which would bring a slight drop on SEval accuracy. Meanwhile, the performance gap under CEval indicates that the unmixing block is essential

Table 4. Ablation study on BSDS dataset [1]. We evaluated alternative network structure and optimization strategy. When removing the unmixing block, the model returns to the original RCF architecture. We removed the L_1 loss, the contrast loss l_{pos} between positive pairs, added the side output loss to assess their influence. We tried identical neighborhood sizes for different types of pixels and found that it is inferior to our final settings. We also show that optimizing the contrast loss at random directions is better than calculating the whole contrast in an iteration. The output edge map was processed under standard evaluation (SEval) and crispness-emphasized evaluation (CEval).

Method	SEval		CEval	
	ODS	OIS	ODS	OIS
w/o unmixing block	0.803	0.814	0.683	0.685
w/o L_1 loss	0.800	0.809	0.658	0.658
w/ side output loss	0.799	0.811	0.680	0.681
w/o l_{pos}	0.805	0.822	0.705	0.716
$s_1 = s_2 = s_3 = 5$	0.803	0.815	0.652	0.652
w/o random dir.	0.803	0.812	0.677	0.677
ours	0.805	0.818	0.766	0.776

for thin edge predictions, which is consistent with the finding in [15]. Removing the L_1 loss would make the optimization more difficult, and also decrease the F-measure. The supervision on side outputs (i.e., multi-resolution intermediate edge maps)is proven to be unnecessary because the result on coarse level is inaccurate, and the local contrast loss is not a good supervision for the coarse output. Removing the similarity term l_{pos} for positive pixel pairs can improve the SEval accuracy, but would bring lots of spiny structure in the prediction and lower the CEval accuracy. Fig. 4 shows two examples of this effect. The choice of neighborhood size is also important for edge crispness. We test identical neighborhood size $s_1 = s_2 = s_3 = 5$ and find that the CEval accuracy would drop a lot. Sampling random directions when calculating the local contrast can not only accelerate the loss computation, but also improve the CEval accuracy. We believe that it is because in each iteration the network can focus on certain edge directions, making the optimization easier.

5. Application

5.1. Edge-Guided Image Inpainting

Image inpainting aims to faithfully complete some part of the image that is corrupted or manually erased. A feasible approach is to predict the completed edge map as a guidance for the completion of color content, as introduced in [28]. We maintain the general framework of the Edge-Connect method and evaluate several edge alternatives for guidance, including Canny [3], RCF [21] and our crisp edges. The network has supervision both on the out-

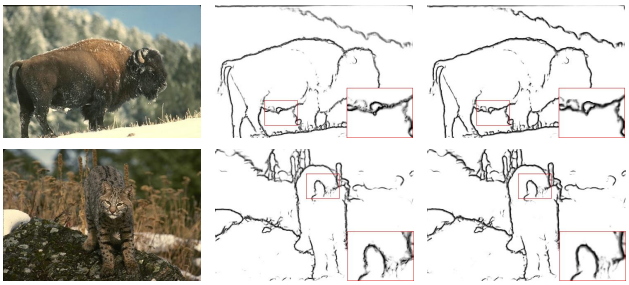


Figure 4. Comparison of using loss term l_{pos} for positive pairs or not during training. Without l_{pos}, the network will predict spiny structures with lower visual quality.

Table 5. Results of edge-guided image inpainting. We used Edge-Connect model as the inpainting network and trained it with different types of edges as intermediate guidance, including Canny (used in prior paper), RCF and our crisp edge. We report four metrics on image quality: PSNR, SSIM, MAE and FID.

Edge Used	PSNR	SSIM	MAE	FID
Canny (original)	30.468	0.9400	0.0190	1.301
RCF	30.473	0.9395	0.0196	1.973
Ours	31.305	0.9474	0.0179	1.226

Table 6. Sketch-based image retrieval on TUBerlin dataset. The original training sketch set was augmented by edge maps from RCF or our proposed method. We report retrieval accuracy (percentage) using raw features and using $32, 64, 128$ bits hash code.

Training data	Raw	32 bits	64 bits	128 bits
Original	83.063	80.016	81.238	81.970
+ RCF edge	82.588	80.044	82.026	82.393
+ Our edge	84.817	83.073	84.082	83.764

Table 7. Density evaluation for different sketch generation method. We evaluated on the test set proposed in PhotoSketching (PS) [19]. Our proposed model can generate much thinner edges.

Method	PS	Ours Pretrained	Ours Finetuned
Density	16.22	11.58	11.68

put image and the intermediate edge map. The facial image dataset Celeba [22] is utilized to train the inpainting model separately.

Table 5 reports four image quality measures including peak signal to noise ratio (PSNR), structural similarity (SSIM) [38], mean absolute error (MAE) and Frechet Inception Distance (FID) [14]. We found that the crisp edge is a better intermediate form for image completion task, as it conveys much semantic information and is easier to be learned by neural networks. Fig. 5 shows several examples

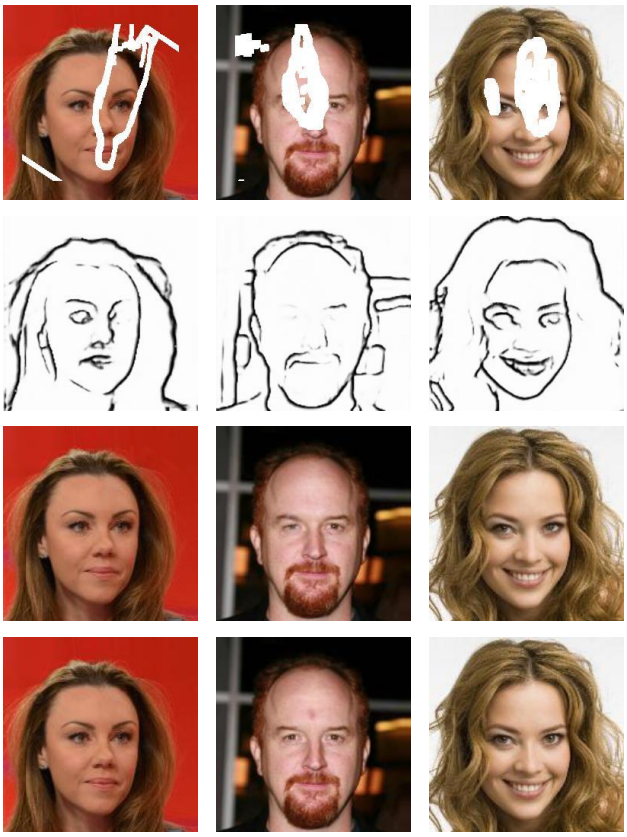
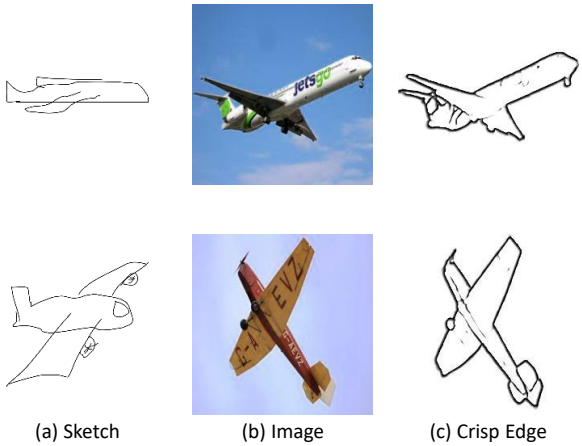


Figure 5. Examples of image completion using Edge-Connect and CNN predicted crisp edges. First row: input image with irregular mask; second row: predicted edges; third row: inpainted image guided by edges from the first stage network; last row: ground-truth image.

of the completed edge map and the completed image.

5.2. Sketch-Based Image Retrieval

Retrieving images from large database using sketch has been widely studied. The main-stream approaches learn a joint feature embedding for data from image domain and sketch domain. Then the feature similarity between query sketch and image in the database is computed for retrieval. Collecting hand-drawn sketches requires lots of time and effort. A simple way to gather training data is to utilize automatic generated edge maps. We used the output of our crisp edge model pretrained on BSDS [1] as augmented data to retrain the DASE model proposed in [23]. Examples of original and augmented data are displayed in Fig. 6. We computed the edge maps for all training images and added them to the training sketch set. Let x_i be the feature vector of an image or sketch and y_i the corresponding category label. A learnable feature center c_j is assigned to the j-th



(a) Sketch (b) Image (c) Crisp Edge

Figure 6. Data augmentation in sketch-based image retrieval. In TUBerlin dataset, the sketches (a) and images (b) are unpaired. We extracted the crisp edge maps (c) from the training images using the proposed model, and add them into the training sketch set to boost the retrieval performance.

category. The DASE loss is defined as

$$l(x_i, y_i) = -\log \frac{e^{-m^2||x_i - c_{y_i}||^2}}{e^{-m^2||x_i - c_{y_i}||^2} + \sum_{j \neq y_i} e^{-||x_i - c_j||^2}}. \tag{20}$$

The parameter m is gradually increased from 1 to 4 during training to ensure that the feature vector x_i would find its nearest neighbor that belongs to class y_i. The crisp edges, serving as pseudo sketch data, however, do not need to achieve such high classification confidence, so we let $m = 1$ for these training edges in the whole training process.

Table 6 illustrates the retrieval accuracy on TUBerlin [7] dataset. For each object category, we reserved 10 sketches for query. The retrieval is conducted by nearest neighbor search in the feature space. The feature vector could be mapped into binary hash codes to reduce the memory storage. We set the length of hash code as $32, 64, 128$ bits and report the retrieval accuracy separately. Adding the crisp edges into the training sketch set made apparent improvement on the accuracy under all settings. The accuracy of 128 bits hashing is slightly lower than that of 64 bits hashing, possibly because of over-fitting in the hashing step.

5.3. Sketch Generation

Sketch is a simple form for human to depict objects or scenes. PhotoSketching [19] is a CNN-based method for sketch generation. The deep sketch generation model proposed in [19] is trained with adversarial loss and a modified L_1 loss that measures the minimum distance among several sketches drawn by annotators. We provide an easy way to transfer our crisp edge detection model to a crisp sketch generation model. We adopted the our model pretrained on BSDS and finetuned it on the sketch dataset proposed by [19], which contains 800 training images, 100 validation images and 100 test images. We only use the local contrast loss for fintuning ($\lambda_p = 0, \lambda_n = 1, \lambda_c = 1, \lambda_r = 0$) because the network has been well initialized.

Fig. 7 displays several examples of the new sketch generation approach. The PhotoSketching model [19] can only predict thick structures due to the limitation of their loss function design. Human usually pay more attention to detail inside the main object, while the edge detection network pretrained on BSDS dataset focuses on boundaries. The finetuning process can ensure that our model predicts edges visually similar to man-made sketches. Our model predicts much thinner edges, so that more detailed structures can be extracted. Table 7 reports the density metric defined in Eq. 19 and verifies our model's superiority.

5.4. Video Stylization

Edges are important cues in stylization applications, because the stylization process needs to simplify the image content and emphasize important structures. We show the use of our crisp edge detection method for a real-time video abstraction algorithm [39]. The original paper adopted the Difference of Gaussian (DoG) algorithm to extract edges for rendering. In Fig. 8, we compare different cartoonization styles with original DoG edges, RCF edges [21] and our crisp edges. The crisp edges can better illustrate the important image structures. The sequential results could be found in the supplementary material.

6. Conclusion

In this paper, we propose a novel formulation for the deep-learning-based edge detection problem. We introduce the concept of local contrast and design a local contrast loss for network supervision. The local contrast loss is calculated on neighboring pixel pairs, and aims to maximize the similarity between pixels within the same category while minimize the cross-type similarity. We also propose an effective stochastic method to optimize the total local contrast by sampling several edge directions in each training iteration. Using the novel loss function and training strategy, we could obtain crisp edges directly from CNN architecture. The predicted edges still achieves high accuracy comparable to the state-of-the-art methods under the standard evaluation protocol and significantly improve the accuracy under crispness-emphasized evaluation.

The raw output of neural network can be directly used without any post-processing steps. We display four possible applications using the new crisp edge detection network, including image completion via edge prediction and guidance, sketch-based image retrieval, sketch generation and video stylization.

In the future, we would like to explore other scenarios

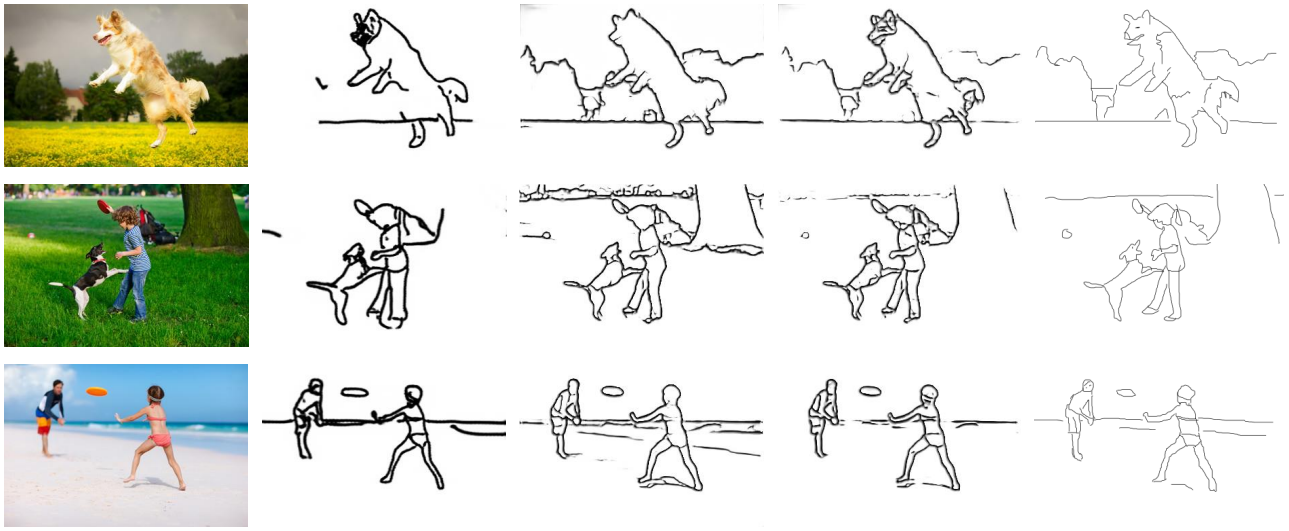


Figure 7. Sketch generation from images. The PhotoSketching method [19] generates thick sketches and might neglect thin structures. Our crisp edge detection model finetuned on sketch dataset can better predicts important structures in the image and our result is closer to human drawing.

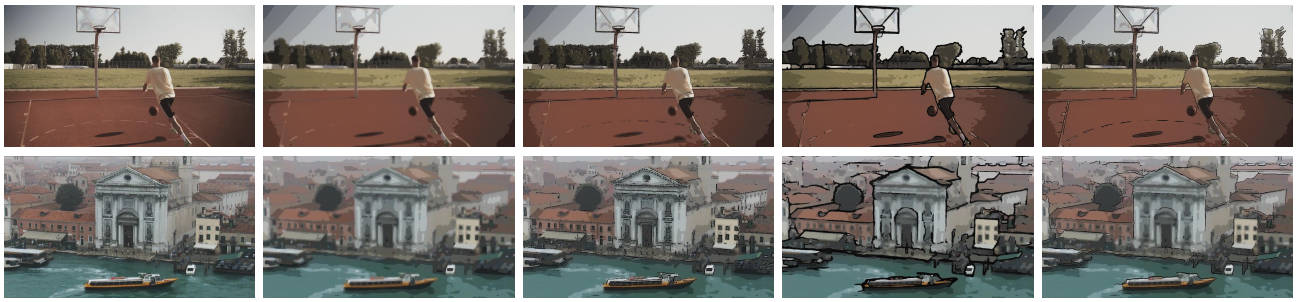


Figure 8. Crisp edges for video cartoonization. We apply the real-time video abstraction algorithm with several types of edges. Edges predicted by our method can emphasize the prominent image structures with better visual effects.

where the local contrast supervision could be adopted, for example, in semantic segmentation and image stylization. It would also be interesting to investigate more powerful loss functions for edge detection task and the generation of other thin-scale structures, such as line art.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011. 1, 2, 3, 5, 6, 7, 8

[2] G. Bertasius, J. Shi, and L. Torresani. High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision. In *Proceedings of the IEEE international conference on computer vision*, pages 504–512, 2015. 7

[3] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986. 1, 2, 7

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 1

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 5

[6] R. Deng, C. Shen, S. Liu, H. Wang, and X. Liu. Learning to predict crisp boundaries. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 562–578, 2018. 1, 3, 7

[7] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4):44:1–44:10, 2012. 9

[8] Y. Ganin and V. Lempitsky. $\mathcal{N}^4$-fields: Neural network nearest neighbor fields for image transforms. In *Asian conference*

on computer vision, pages 536–551. Springer, 2014. 7

[9] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[10] S. Hallman and C. C. Fowlkes. Oriented edge forests for boundary detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1732–1740, 2015. 7

[11] J. He, S. Zhang, M. Yang, Y. Shan, and T. Huang. Bi-directional cascade network for perceptual edge detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3828–3837, 2019. 1, 2, 6, 7

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[14] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 8

[15] L. Huan, N. Xue, X. Zheng, W. He, J. Gong, and G.-S. Xia. Unmixing convolutional features for crisp edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 1, 2, 3, 4, 5, 6, 7

[16] J. Kittler. On the accuracy of the sobel edge detector. *Image and Vision Computing*, 1(1):37–42, 1983. 1, 2, 4

[17] I. Kokkinos. Pushing the boundaries of boundary detection using deep learning. *arXiv preprint arXiv:1511.07386*, 2015. 7

[18] S. Konishi, A. L. Yuille, J. M. Coughlan, and S. C. Zhu. Statistical edge detection: Learning and evaluating edge cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1):57–74, 2003. 1, 2

[19] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan. Photo-sketching: Inferring contour drawings from images. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1403–1412. IEEE, 2019. 2, 8, 9, 10

[20] Y. Liao, S. Fu, X. Lu, C. Zhang, and Z. Tang. Deep-learning-based object-level contour detection with ccg and crf optimization. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 859–864. IEEE, 2017. 7

[21] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai. Richer convolutional features for edge detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3000–3009, 2017. 1, 2, 3, 6, 7, 9

[22] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 8

[23] P. Lu, G. Huang, H. Lin, W. Yang, G. Guo, and Y. Fu. Domain-aware se network for sketch-based image retrieval with multiplicative euclidean margin softmax. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3418–3426, 2021. 2, 8

[24] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. V. Gool. Convolutional oriented boundaries. In *European conference on computer vision*, pages 580–596. Springer, 2016. 1, 2, 7

[25] D. R. Martin, C. C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE transactions on pattern analysis and machine intelligence*, 26(5):530–549, 2004. 1, 2

[26] D. A. Mély, J. Kim, M. McGill, Y. Guo, and T. Serre. A systematic comparison between visual cues for boundary detection. *Vision research*, 120:93–107, 2016. 1

[27] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 1, 2, 3, 5, 6, 7

[28] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. *arXiv preprint arXiv:1901.00212*, 2019. 2, 7

[29] S. Paris. Edge-preserving smoothing and mean-shift segmentation of video streams. In *European Conference on Computer Vision*, pages 460–473. Springer, 2008. 1, 2

[30] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1

[31] X. Ren and L. Bo. Discriminatively trained sparse code gradients for contour detection. In *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 1*, pages 584–592, 2012. 1

[32] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1

[33] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3982–3991, 2015. 7

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 2, 3, 5

[35] V. Torre and T. A. Poggio. On edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):147–163, 1986. 2

[36] M. Wang, X.-N. Fang, G.-W. Yang, A. Shamir, and S.-M. Hu. Prominent structures for video analysis and editing. *IEEE Transactions on Visualization and Computer Graphics*, 27(7):3305–3317, 2021. 2, 5

[37] Y. Wang, X. Zhao, and K. Huang. Deep crisp boundaries. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3892–3900, 2017. 1, 2, 7

[38] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 8

[39] H. Winnemöller, S. C. Olsen, and B. Gooch. Real-time video abstraction. *ACM Transactions On Graphics (TOG)*, 25(3):1221–1226, 2006. 2, 9

[40] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015. 1, 2, 7

[41] D. Xu, W. Ouyang, X. Alameda-Pineda, E. Ricci, X. Wang, and N. Sebe. Learning deep structured multi-scale features using attention-gated crfs for contour prediction. *Advances in neural information processing systems*, 30, 2017. 7

[42] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang. Object contour detection with a fully convolutional encoder-decoder network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 193–202, 2016. 7