Tiny and Efficient Model for the Edge Detection Generalization

Xavier Soria
National University of Chimborazo, Ecuador
xavier.soria@unach.edu.ec

Yachuan Li
China University of Petroleum (East China), China
liyachuan@s.upc.edu.cn

Mohammad Rouhani
INRIA Paris, France
mohammad.rouhani@inria.fr

Angel D. Sappa[1,2]
[1]ESPOL Polytechnic University, Ecuador
[2]Computer Vision Center, Spain
asappa@espol.edu.ec & asappa@cvc.uab.es

Abstract

Most high-level computer vision tasks rely on low-level image operations as their initial processes. Operations such as edge detection, image enhancement, and super-resolution, provide the foundations for higher level image analysis. In this work we address the edge detection considering three main objectives: simplicity, efficiency, and generalization since current state-of-the-art (SOTA) edge detection models are increased in complexity for better accuracy. To achieve this, we present Tiny and Efficient Edge Detector (TEED), a light convolutional neural network with only $58K$ parameters, less than 0.2% of the state-of-the-art models. Training on the BIPED dataset takes ***less than 30 minutes****, with each epoch requiring* ***less than 5 minutes****. Our proposed model is easy to train and it quickly converges within very first few epochs, while the predicted edge-maps are crisp and of high quality. Additionally, we propose a new dataset to test the generalization of edge detection, which comprises samples from popular images used in edge detection and image segmentation. The source code is available in* `https://github.com/xavysp/TEED`.

1. Introduction

Large scale Deep Learning (DL) models are frequently used in many computer vision applications, as documented in [53, 22]. However, for low level tasks such as image enhancement, super-resolution [51] and edge detection [50], more efficient and lightweight models are necessary as these steps are preliminary to higher level image analysis. Therefore, edge detection models should come with low computational cost and latency. For these reasons, classical edge detectors like Sobel [37] or Canny [3] are still widely used in many applications. However, recent deep learning architectures with over 10 million parameters have been proposed to outperform state-of-the-art approaches in various benchmarks [48, 33]. While these models are powerful, they come with a significant computational expense.

In order to reduce the computational cost, new procedures for training the DL models have emerged, allowing the utilization of lightweight models through careful dataset selection. According to [16, 24, 40], the standard datasets for edge detection, like BSDS [1], are originally introduced for image segmentation; although they have edge level annotations some of their ground truth comes with wrong annotations [16, 41]. Having this problem in mind, the new DL training procedure uses BIPED dataset instead of BSDS, which avoids tedious setting for transfer learning, and also reduces the training and testing time; we refer to this procedure as Training from the Scratch (TFS). The edge-maps generated from these different training procedures are compared in Fig. 1, where the results from TEED use the TFS procedure. Note that BSDS dataset was not used in training of TEED but our model is capable of generalize the edge detection. In this manuscript the edge detection generalization is the capacity of the learning algorithm to predict most edges from an arbitrary image—any image even gray-scale that comes from a visible wavelength band.

The proposal, named Tiny and Efficient Edge Detector (TEED) is capable of predicting thinner and clearer edge-maps. Compared to the SOTA models [14], [42], and [33] TEED stands out due to being remarkably **simple**, eliminating the need for transfer learning or exhaustive hyperparameter tuning. Additionally, TEED is highly **efficient**, demonstrating rapid convergence and producing superior results both quantitatively and qualitatively. Our model generates robust results while it can handle various scenes and different input types, whether in color or grayscale. To assess the TEED's **generalization** ability, we have prepared a new test dataset by selecting images based on a *frequency criteria* from commonly used datasets in edge detection,

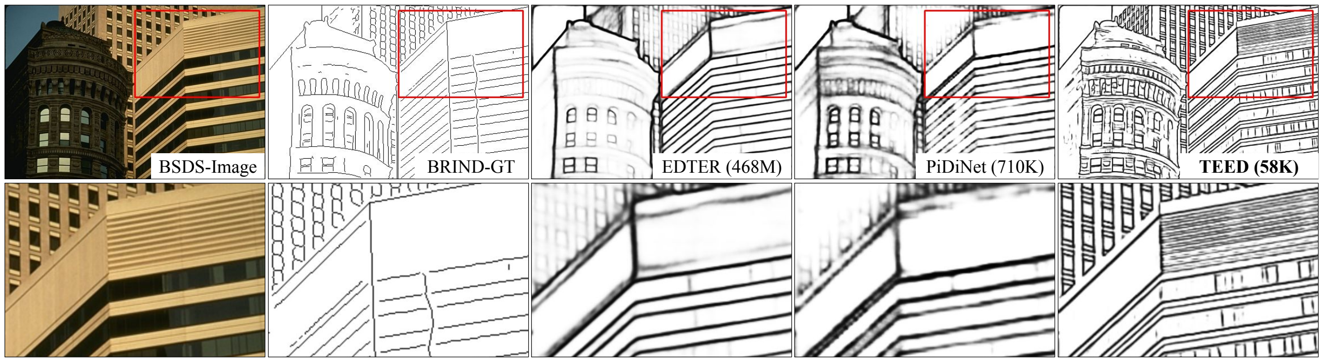

Figure 1: Edges from our proposal (TEED) and the state-of-the-art models. EDTER [33] and PiDiNet [42] have been trained in BSDS500[1] following the standard training procedure. TEED has been trained **from the scratch with BIPED** [41] with a reduced hyper-parameters tuning.

segmentation, and other low-level processing tasks. We named this collection of images the Unified Dataset for Edge Detection (UDED).

Overall, we present five main contributions: (i) TEED: a simple but robust CNN model with only 58K parameters; (ii) dfuse: a new, efficient fusion module inspired from Co-Fusion in CATS [17] (while coFusion has around 40K parameters, dfuse has less than 1K parameters); (iii) a dloss function for efficient and rapid training convergence; (iv) UDED: a new dataset to test edge detection generalization, which includes the ground truths annotated through human perceptual edges; and finally, (v) a fair quantitative and qualitative comparison with SOTA models that have less than 1M parameters in UDED. To evaluate the robustness in UDED dataset, a downstream task (sketch image retrieval) is used, this validation also compare the standard and new DL based approaches.

The remainder of this paper is organized as follows: Section 2 reviews related work and discusses the parameter requirements of each approach. Section 3 elaborates on the proposed architecture. Section 4 describes the dataset and the evaluation procedure. Section 5 presents experimental results; and finally, conclusions are given in Section 6.

2. Literature Review

Edge detection is widely used from the low to high level image analysis (e.g., medical image segmentation [8], sketch-based image retrieval [38]). For a comprehensive review, we refer readers to [55, 2, 28], and [50].

In recent years different deep learning based approaches have been proposed for tackling the edge detection problem. They can be classified into two categories. The first category includes approaches that train their models mainly with BSDS500 [1], NYUD [13], and PASCAL-Context [31] databases, without applying a validation process on the given annotations. This leads to tedious additional steps before and during training. The second category includes approaches that split the problem up into edge, contour, and boundary detection, as also outlined in BSDS300 [29] and thoroughly explained in [16] and [30]. Subsequently, BIPED [40] and BRIND [32] showed that a model trained from scratch on a curated dataset for edge detection could accurately predict over 80% of edges in a given scene. Our approach is aligned with this latter methodology. Additionally, we suggest that an efficient yet lightweight model, after training, can predict over 80% of edges in any image dataset considered for evaluation.

With recent developments in datasets proposed for DL model training, such as those in [41, 32], the next step is to find a dataset that encompasses images from various scenarios for comprehensive evaluation. To this end, our paper introduces a small yet diverse dataset for edge evaluation. In addition, various metrics are considered to ensure a fair comparison, as presented in Section 5.

2.1. Edge Fusion Methods

Fusion of multi-scale features to generate edges is as important as feature extraction in edge detection tasks. The earliest methods involve reshaping the multi-scale feature matrix into the size of the final result, and then computing the weighted sum as the final result. This fusion method, though simple, is very effective and widely adopted by different works such as HED [48], RCF [26], BDCN [14]. However, this fusion approach has two drawbacks: firstly, the fine branches lack global semantic information, and secondly, features in the same channel share the same weight and have equal importance in channel fusion.

In order to address the aforementioned drawbacks, Deng et al. ([7, 6]) used the decoder structure of U-Net [34] to gradually incorporate global information into the shallow features. However, recent research [49] suggests that semantic information gradually decays as it is fused downward in U-Net structures, diminishing its guiding effect. To simultaneously preserve multi-scale features and generate pixel-level weight matrices, recent works such as FCL [49] generate a pixel-level weight matrix for each scale feature during multi-scale feature generation. CATS [17], on the other hand, splices multi-scale features and combines spatial and channel information to alleviates edge localization ambiguity. While generating crisp edges with the contribution of tracing loss and a context-aware fusion block (coFusion), it can lead to suppress the nearest neighbor edges. To overcome this limitation, the Double Fusion module is proposed in the current work, which improve efficiently the procedure of coFusion with fewer parameters.

2.2. Loss Functions

To improve the performance of edge detection, researchers have proposed various loss functions to optimize the learning process. HED [48] is a seminal work in this field that introduced Weighted Cross-Entropy (WCE) as a loss function for the end-to-end supervised learning. However, it is well known that WCE suffers from multiple annotation inconsistencies in the BSDS500 dataset [26]. In order to address this problem, subsequent studies (e.g., [26, 14, 42]) propose the WCE+ loss function by ignoring the disputed pixels while measuring WCE.

In recent years, researchers have gradually identified issues with the WCE+ loss function. Due to the significant disparity between the number of edge and non-edge pixels, backpropagation gradients tend to assign larger weights to edge pixels, leading to blurry edges. The problem of imbalanced positive and negative samples is further aggravated by WCE+ where controversial edges are ignored. Therefore, several alternative approaches have been proposed to achieve crisper edge detection by refining the loss function. For instance, [7] uses a combination of Dice coefficient and Cross-Entropy. [6] goes a step further by incorporating the structural differences between the output and the ground truth using SSIM [46]. Lastly, [17] optimizes the loss function by dividing the image into three categories: edge, confusing, and non-edge pixels.

Although the use of these loss functions has significantly improved edge detection, a crucial issue is ignored: the edge fusion module pays different levels of attention to edge-maps predicted in the preliminary stages. In the current work we address this problem by employing different loss functions to monitor the main architecture and the fusion module (dfuse for TEED) separately, enabling them to capture information at various levels.

3. Proposed Model

In this section, we present the proposed Tiny and Efficient Edge Detector (TEED) architecture in detail. We begin by introducing the backbone architecture, followed by USNet from DexiNed [41]. Then, we present the Edge fusion module, termed Double Fusion (dfuse), as well as a proposed loss function, named Double Loss (dloss). All components united in TEED provide simplicity, efficiency, and edge detection generalization, making this new approach an effective and efficient edge detection model that reduces training and testing time as well as computational cost.

3.1. TEED Backbone Architecture

Since the inception of ResNet [15], Xception [4], EfficientNet [43] architectures powered by dense and skip connections have achieved impressive results improving forward and backward operations from the shallower to the deeper CNN layers. These advantages are plausible in many computer vision tasks. Following these successes, DexiNed [40] and LDC [39] use similar architectures for edge detection. The result is a model trained from scratch that still achieves state-of-the-art accuracy. The TEED backbone architecture presented in Fig. 2 is based on LDC [39]. It consists of three green blocks (i.e., Block: x2 in Fig. 2), each has two standard CNN layers. We did not consider the VGG16 architecture [36] due to its lack of skip-connections, which can reduce the efficacy of edge detection in deeper layers. In TEED we only use $58K$ parameters, which is far lower than DexiNed and LDC, $35M$ and $674K$ parameters respectively. We reached this compact network by reducing the number of convolutional layers and skipping Batch Normalisation (BN). With this reduction also falls the accuracy, to overcome this drawbacks, we consider a new activation function, which is proposed in [45], $\zeta(h) = smish(h) = h \cdot tanh[\ln(1 + sigmoid(h))]$, where h is the feature map of the respective layer in TEED. The $smish$ is capable of reducing the lack of efficient training optimization effect without BN. Besides, according to our empirical observations, this non-linearity function improve the RELU function used in the SOTA models.

Overall, every block of TEED has 2 convolutional layers followed by ζ—$[conv1 + smish + conv2 + smish]$. The three backbone blocks have 16, 32, and 48 layers, respectively. Then, the output of the first block h_1 and the second block h_2 are combined by skip connection $skip1$ through the summation operation $(h_1 + h_2)$. This skip-connection is applied immediately after the max-pooling operation. Another skip connection, $skip2$, is used to fuse h_2 and the sub-outputs of block 3 as follow $(Block3.1 + h2)/2$; this is similar to the skip connection in [40] and [39]. The outputs of these three blocks feed USNet, as indicated by gray arrows pointing downwards in Fig. 2. The convolutional layers used for skip-connections have 1×1 kernel size.

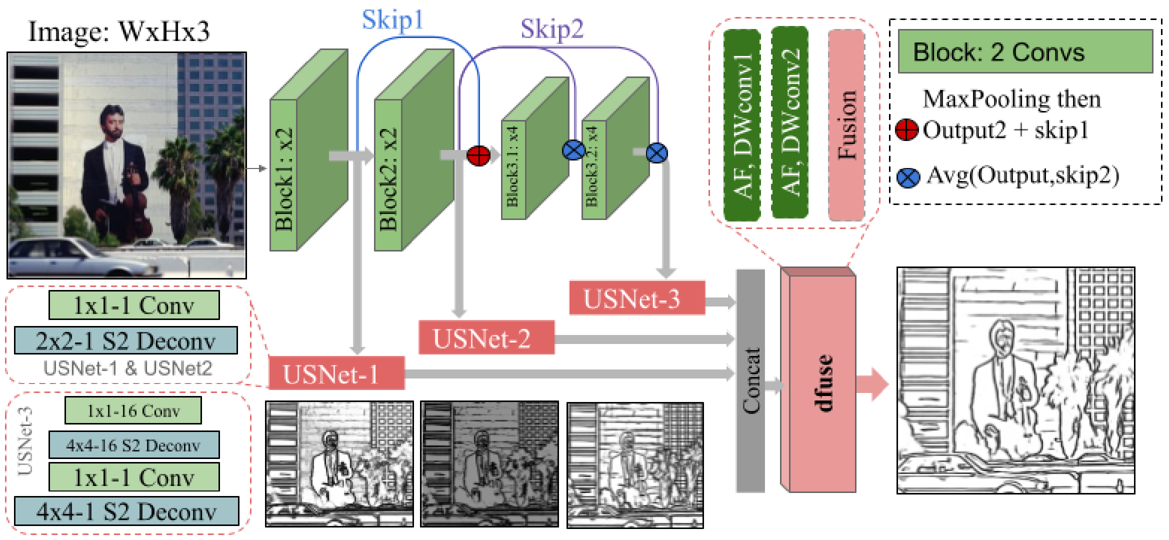

Figure 2: TEED architecture.

3.2. USNet

The USNet module of TEED is similar to DexiNed [40], with a slight modification: i) we use Xavier initialization in all of USNet layers, ii) we use the same activation function of the backbone (ζ). The outputs of USNet are the edge-map predictions, $\hat{y}_i$, with the same width and height of the input image x, where $x \in \mathbb{R}^{W\times H\times 3}$. As illustrated in Fig. 2, USNet is composed of 1 convolutional layer (Conv) followed by activation ζ and 1 deconvolutional layer (Deconv). The kernel size and number of filters can be seen in the bottom left of Fig. 2; for instance, ”Deconv $2\times 2-1$ $S2$” is a deconvolutional layer with a kernel size of 2×2, 1 filter, and the upscale process $\times 2$. The USNet-2 has the same architecture as the USNet-1. The size of the output from block 3-2 is $\times 4$ down-sampled; hence, USNet-3 has two Conv and 2 Deconv layers as shown in the bottom left of Fig. 2. Finally, the edge-map predictions from the respective USNet modules is $\hat{y}_i = \sigma(USNet(h_i))$.

3.3. Double Fusion

The edge fusion module in an end-to-end edge detection learning model is typically a CNN or a set of layers that merges edge-maps generated in different scale levels of the backbone network. This module in TEED is named **dfuse** as illustrated in Fig. 2; the predicted edge-map from this module is denoted by $\hat{y}_{dfuse}$. This module is inspired from CATS [17]. In terms of efficiency, TEED+dfuse comes with just 58K parameters, much less than 99K parameters in TEED+coFusion.

The dfuse is composed of two Depth-wise convolutional layers (DWConv), since this approach applies a single convolutional filter to each input channel [11], which increments the receptive field but reduce the cost of computation. This module does not use Softmax activation nor group normalization; instead, it employs the $Smish$ activation function [45] (ζ) to regularize the weight maps during training. The kernel size of the DWConv is $3\times 3-24$, which reduces the spatial dimensions of dfuse. Note that the activation function ζ is applied before the DWConv layers. The output from the DWConv layers is fused twice:

$$h_{dfuse} = \zeta(ewa(DWc_1(\hat{Y}) + DWc_2(\hat{H}_{dfuse}))), \quad \text{where,} \quad \hat{y}_{dfuse} = \sigma(h_{dfuse}), \tag{1}$$

and σ is the sigmoid function. The output of USNet is denoted as $\hat{Y} = [\hat{y}_1, \hat{y}_3, \hat{y}_3]$ and the feature map of DWConv is $\hat{H}_{dfuse} \in \mathbb{R}^{W\times H\times 24}$. As indicated in eq. (1) two fusions are applied in *dfuse*: the first one is the sum of DWConv1 and DWConv2, where the second fusion is applied through element-wise addition (ewa) of the first fusion, followed by the activation function ζ. At this stage, $\hat{y}_{dfuse}$ has the same size as the respective y.

3.4. Double Loss

We introduce an approach to measure the error on the training dataset of pairs (x, y), where y is the ground truth edge map of image x. The loss function considered for our end to end training on edge detection is the weighted cross entropy L_{wce}, which was originally proposed by HED [48] and slightly modified later in BDCN [14]. L_{wce} helps in detecting as many edges as possible. However, if the detected edges are absent in the ground truth, the resulting conflict is reflected in the form of artifacts or noise in the detected edge space. If a model is trained using a ground truth that is visually annotated by humans, L_{wce} may show this drawback; see PiDiNet [42] result in Fig. 1.

In order to overcome this problem, CATS [17] proposes a new loss function called tracing loss L_{trcg}, which is a combination of L_{wce}, boundary tracing fusion and texture suppression function. This loss function leads to faster convergence of the model during training, compared to L_{wce}, while the predicted edge-maps are clearer and thinner. However, since the ground truth is generated through human visual judgment, some edges may be omitted in the predicted edge-map. As L_{trcg} is relying more on a given ground truth (compared to L_{wce}) some true edges may be excluded. In addition, deploying a tiny model such as TEED could result in losing the generalization ability.

To address these drawbacks, TEED employs L_{wce} for comparing the ground-truth y with the outputs of USNets $\{\hat{y_i}\}_1^3$ as well as L_{trcg} for $\hat{y}_{dfuse}$. Therefore, in the one hand, dfuse module is fed with more detected edges thanks to L_{wce}, on the other hand L_{trcg} controls the prediction in the dfuse module, $\hat{y}_{dfuse}$. Moreover, the structure of dfuse is developed to reduce the drawbacks of L_{trcg} and leverage the benefits of L_{wce}. The global loss (dloss) can be summarized as follow:

$$L_{dloss} = \sum_{i=1}^{3} L_{wce}(\hat{y}_i, y) + L_{trcg}(\hat{y}_{dfuse}, y). \quad (2)$$

4. Edge Detection Datasets and Evaluation

This section presents the datasets used for training and assessing the models discussed in Sec. 5 together with a brief overview of the metrics employed for evaluation.

4.1. Datasets for Training TEED

Due to the impressive results of the recently released dataset for edge detection, Barcelona Images for Perceptual Edge Detection (BIPEDv2) [40, 41] referred to as BIPED in this manuscript, we trained all models with this dataset. **BIPED** was firstly introduced in [40]. It contains 250 images in high definition (720 × 1280): 50 images of which were selected by the authors for testing and the rest for training and validation. The data augmentation procedure used in LDC [39] is implemented for TEED.

4.2. Dataset for Testing Edge Generalization

The well known datasets BSDS500 [1], PASCAL-Context [31], and NYUD [13] are usually considered for training and evaluating edge detection methods. However, based on our understanding and analysis of the literature [24, 16, 41], this assumption is wrong, as these datasets are not intended for edge detection. The ground truths in these datasets are prepared for boundary detection and/or image segmentation [25]. Only BIPED [41] and BRIND [32] have been proposed to tackle the edge detection problem.

Hence, in order to test whether a trained model can generalize by detecting edges on images from different scenes, a new dataset is proposed: Unified Dataset for Edge Detection (UDED). This UDED is created with the purpose of reducing the evaluation procedure by selecting images from datasets focused on low- and mid-level tasks, and carefully annotating perceptual edges. The proposed UDED contains 30 images selected from: BIPED [40], BSDS500 [1], BSDS300 [29], DIV2K [20], WIREFRAME [19], CID [10], CITYSCAPES [5], ADE20K [54], MDBD [30], NYUD [13], THANGKA [27], PASCAL-Context [31], SET14 and URBAN100 [18], and the cameraman image. The image selection process consists on computing the Inter-Quartile Range (IQR) intensity value on all the images, images larger than 720 × 720 pixels were not considered. Then, the images are sorted by this IQR value and 30 of them uniformly selected from that sorted list—around 2 images per dataset have been selected.

Finally, since edge detection is a low-level process, required for other tasks (e.g., image super-resolution guidance [12] and sketch-based image retrieval [52]), we propose to conduct an application-oriented evaluation that validates the generalization of UDED.

4.3. Metrics for Quantitative Evaluation

In the current work, we focus on the most commonly used metrics for quantitatively evaluating edge detection methods: Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS) [29]. Additionally, Peak Signal to Noise Ratio (PSNR), Mean Square Error (MSE), and Mean Absolute Error (MAE) are also considered for the quantitative comparison as they have recently been suggested in [35, 9, 44, 21].

5. Experiments

In general, an edge detector based on CNN is validated through training and testing in different dataset (e.g., BSDS [1], NYUDv2 [13]). Our manuscript proposes a new methodology for the edge detection model evaluation. We suggest evaluating it only in a dataset especially prepared for edge detection. This dataset should be designed in a way that is efficient and gives results in a short period of time, which is not possible in the test set of the state of the art, nowadays. Therefore, we propose evaluating on the UDED dataset, presented in Sec. 4. To validate the effectiveness of UDED, a downstream task for sketch image retrieval is considered [52, 38]. Implementation details are given below, followed by an ablation study. Later, we present both quantitative and qualitative results using the UDED dataset. Finally, TEED is validated in sketch image retrieval.

5.1. Implementation Details

TEED is implemented in PyTorch and trained on an NVIDIA 3090 GPU. TEED training is based on Adam optimizer [23], a batch size of 8, an initial learning rate of $8e-4$ changed to $8e-5$ at epoch 5, a weight decay of $2e-4$. Validation results at epoch 6 are reported. Despite the fact that BIPED provides binary annotations, due to the interpolations when edge maps are built floating point values appear; hence we apply a transformation to edge annotations y by adding 0.2 to values greater than 0.1, and then clipping the results to [0,1], similar to LDC [39]. A Lenovo Yoga C740-15IML laptop with an Intel i5-10210U processor is used to report the FPS values.

5.2. Ablation Study

In this section various components of the TEED model are analyzed, using BIPED [41] as training data. Table 1 shows the different configurations of TEED, starting in the first column with $B2$, as showing in Fig. 2, TEED is composed of three blocks, and $B2$ represents just 2 blocks of the proposal and $B3$ corresponds to the results when all blocks are considered. The table presents information about the number of parameters ($\#P$), loss functions used in training (L_{trcg} proposed for CAST [17] and L_{wce} the weighted cross entropy loss), fusion modules (coF), activation functions (including $Relu$, $Tanh$, and $Smish$). Column $Conv$ corresponds to the standard convolution layer used in the $dfuse$, and the column $DWConv$ corresponds to the Depth Wise Convolution used on the fusion module. The last two columns correspond to the standard metrics used for the edge detection quantitative evaluation.

Table 1 is divided into two sections. The top section shows results of TEED using the first configuration. For instance, the first row in the top section shows the results of TEED using only the loss function from CATS with the same fusion module (coFusion), and the Smish activation function in the TEED backbone. The resulting ODS is 0.810, and this version of TEED has 99K parameters. Starting from the third row in the top section of Table 1, we begin using the dfuse module, which reduces the number of parameters by 40K; by using DWConv, we reduce the number of parameters from 60K to 58K for TEED. Overall, we can see that using dloss, dfuse, and the Smish activation function in the TEED model contribute to both efficiency and accuracy.

5.3. Quantitative Results

Based on the quantitative results from DexiNed [41] and LDC [39], we trained all models with BIPED. Table 2 presents results from TEED and the state-of-the-art models with less than **1M parameters**; all these models are trained with BIPED and evaluated with UDED. The approaches considered in the comparison are as follow: the block 2 of BDCN [14]—BDCN-B2; three versions of PiDiNet [42]—the standard PiDiNet, PidiNed-Small, and PiDiNet-tiny-L; TIN [47]; LDC [39]; Canny edge detector [3] and TEEDup—we up-scale the input image (x), to 1.5 before feeding the model. Results from DexiNed edge detection model [41] are provided just for reference. This table shows ODS and OIS, the last epoch used for evaluation, number of parameters (#P), time of training till reaching the last epoch (Train-time), Frames Per Second (FPS), Mean Square Error (MSE), Mean Absolute Error (MAE), and Peak-Signal-to-Noise-Ratio (PSNR). The results in ODS and OIS are from edge-maps after applying NMS. Results from MSE, MAE, and PSNR are before applying NMS, this process also let us know which edge-map has less artifacts or noises.

As shown in the table, TEED and TEEDup achieve the best results in all evaluation criteria with only 6 epochs and a training time of less than 30 minutes. In contrast, all other approaches require between 10 and 53 hours of training to achieve similar performance. It should be noted that TEED is the architecture with less number of parameters. The second smaller architecture (PiDiNet-tiny-L [42]) requires 30 hours of training, while TEED needs only **30 minutes**.

5.4. Qualitative Results

Figure 3 presents six images from UDED dataset for the perceptual judgement. The images used in UDED come from different datasets and the selection procedure is detailed in Sec 4. We can see that from the third column, the images used in the comparison are challenging; for instance, the image from the last column has a large number of edges annotated in the ground truth, and most of the models used for comparison predict edge-maps with noise. In conclusion, it can be said that the compact TEED architecture excels at detecting as many edges as possible. TEED is able to obtain superior edge-maps compared to state-of-the-art approaches in all the images shown in Fig. 3; actually, not only more edges are detected but also **thinner** and **cleaner** edge-maps are obtained.

5.5. Discussion

Since the 1980s, the edge detection evaluation on a set of images with corresponding ground truths, has been challenging [55, 50]. This is because edge maps are not the ultimate goal, but rather they are used for higher level tasks [41] of computer vision and image processing. For addressing this issue, we propose the UDED dataset, which includes a small yet diverse set of images with different intensities to evaluate the performance under various scenarios.

In order to validate our dataset, we consider a subsequent use of the edge-maps generated from TEED. Testing predicted edge-maps in some application is an effective way to validate an edge detector. Hence, we consider sketch based image retrieval [52, 38] to compare the edge-map contribution in the first and last versions of QMUL-chair and QMUL-shoe datasets; results are depicted in Table 3 and 4. These results show that reaching the best performance in

B2	B3	#P	Train-data	L_{wce}	L_{trcg}	coF	DF	Conv	DWConv	Relu	Tanh	Smish	ODS	OIS
	✓	$99K$	BIPED		✓	✓		✓				✓	.810	.842
	✓	$99K$	BIPED	✓	✓	✓		✓				✓	.821	**.854**
	✓	58K	BIPED	✓	✓		Hap		✓			✓	.814	.84
	✓	$60K$	BIPED	✓	✓		EWA	✓				✓	.825	.851
	✓	58K	BIPED	✓	✓		EWA		✓	✓			.816	.827
	✓	58K	BIPED	✓	✓		EWA		✓		✓		.815	.837
✓		**17K**	BIPED	✓	✓		EWA		✓			✓	.796	.823
	✓	58K	BIPED	✓	✓		EWA		✓			✓	**.828**	.842

Table 1: Detailed ablation study of TEED backbone, Double-Loss and Double-Fusion.

Method	↓Epoch	↓#P	↓Train-time	↑FPS	↑ODS	↑OIS	↓MSE	↓MAE	↑PSNR
Canny [3]	–	–	–	392.5	.742	.743	—	—	—
DexiNed [41]	*11*	*35M*	*∼20 hours*	*.34*	*.815*	*.826*	*.095*	*.149*	*10.799*
PiDiNet [42]	20	710K	∼53 hours	.67	.812	.824	.126	.194	9.49
LDC [39]	16	674K	∼10 hours	2.57	.817	.838	.084	.134	11.268
BDCN-B2 [14]	20	268K	–	1.97	.821	.839	.136	.205	9.229
TIN [47]	1.6M	244K	∼14 Hours	1.2	.803	.827	.094	.16	10.734
PiDiNet-small [42]	20	184K	∼40 hours	—	.821	.834	.133	.202	9.174
PiDiNet-tiny-L [42]	20	73K	∼30 hours	1.59	.821	.834	.136	.21	9.182
TEED (Ours)	**6**	**58K**	**∼30 min**	**2.6**	.828	.842	.073	**.107**	11.965
TEEDup (Ours)	**6**	**58K**	**∼30 min**	1.94	**.834**	**.847**	**.071**	**.107**	**12.05**

Table 2: Results when models are trained with BIPED [41] but evaluated with the proposed UDED dataset.

		QMUL-Shoe [52]		QMUL-Chair [52]	
Method	#P	Top1	Top10	Top1	Top10
TripletSN [52]	—	.3913	.8783	.6907	.9794
BDCN-BSDS	16.3M	.3913	.8348	.6391	.9896
BDCN-BIPED	16.3M	.513	.8869	.8144	.9896
PiDiNet-BIPED	710K	.5043	.8347	.8041	**1**
TEED-BIPED	**58K**	**.5217**	**.8957**	**.835**	**1**

Table 3: Sketch image retrieval results in QMUL Shoe and Chair datasets [52]. BDCN-BSDS is the model trained with BSDS [16], BDCN-BIPED stands for the model trained with BIPED [41].

		QMUL-ShoeV2 [38]		QMUL-ChairV2 [38]	
Method	#P	Top1	Top10	Top1	Top10
HOLEF [38]	—	**.6174**	.9478	.8144	.9588
BDCN-BSDS	16.3M	.3826	.8869	.6804	.9896
BDCN-BIPED	16.3M	.5565	.8956	**.8969**	.9896
PiDiNet-BIPED	710K	.5304	.8869	.8762	**1**
TEED-BIPED	**58K**	.5565	**.9565**	.8865	**1**

Table 4: Sketch image retrieval results in the updated QMUL shoe and chair datasets reported in HOLEF [38].

the most widely used datasets does not guarantee its effectiveness in the subsequent tasks. TEED achieves the best results in sketch image retrieval task using only 58K parameters, which is less than 9% of PiDNet's [42] parameters and less than 0.3% of BDCN's parameters. The results suggest that TEED has a strong generalization capability, as it performs well on new datasets like QMUL-chair and QMUL-shoe, even though it was only trained on BIPED.

6. Conclusions

This paper presents TEED, a deep learning-based edge detector that produces edge-maps that mimics human visual perception. The performance of TEED is evaluated using various metrics and a subsequent task, sketch-based image retrieval, and compared with SOTA edge detectors. Our results demonstrate that TEED outperforms other edge detectors in terms of accuracy, while requiring significantly fewer parameters and being easy to train within few epochs. Additionally, our experiments show that the proposed UDED dataset is a useful tool for validating edge detectors across different scenarios. Overall, the results suggest that TEED is a highly effective and efficient edge detector that could be used in a wide range of computer vision applications.

7. Acknowledgements

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-22-1-0261; and partially supported by the Grant PID2021-128945NB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe"; the "CERCA Programme / Generalitat de Catalunya"; and the ESPOL project CIDIS-12-2022.

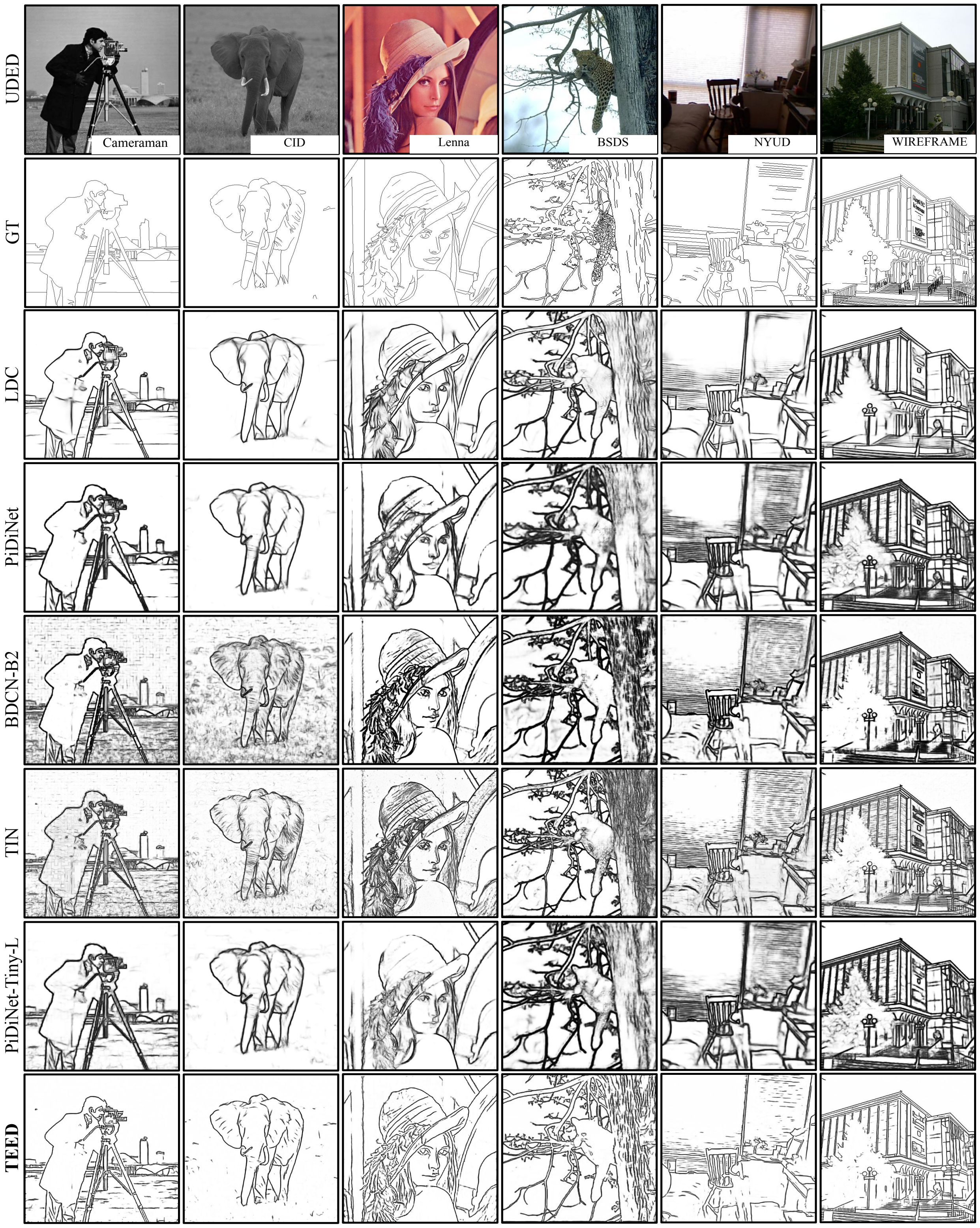

Figure 3: Edge-maps predicted with the lightweight SOTA models and TEED—images from UDED dataset.

References

[1] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 1, 2, 5

[2] M. Basu. Gaussian-based edge-detection methods-a survey. *Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 32(3), 2002. 2

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

[4] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 3

[5] 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 *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[6] Ruoxi Deng and Shengjun Liu. Deep structural contour detection. In *Proceedings of the 28th ACM international conference on multimedia*, pages 304–312, 2020. 2, 3

[7] Ruoxi Deng, Chunhua Shen, Shengjun Liu, Huibing Wang, and Xinru Liu. Learning to predict crisp boundaries. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 562–578, 2018. 2, 3

[8] Xiaogang Du, Yinyin Nie, Fuhai Wang, Tao Lei, Song Wang, and Xuejun Zhang. Al-net: Asymmetric lightweight network for medical image segmentation. *Frontiers in Signal Processing*, 2, 2022. 2

[9] SERT Eser and AVCI Derya. A new edge detection approach via neutrosophy based on maximum norm entropy. *Expert Systems with Applications*, 115:499–511, 2019. 5

[10] Cosmin Grigorescu, Nicolai Petkov, and Michel A Westenberg. Contour detection based on nonclassical receptive field inhibition. *Transactions on Image Processing*, 12(7):729–739, 2003. 5

[11] Yunhui Guo, Yandong Li, Liqiang Wang, and Tajana Rosing. Depthwise convolution is all you need for learning multiple visual domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8368–8375, 2019. 4

[12] Honey Gupta and Kaushik Mitra. Pyramidal edge-maps and attention based guided thermal super-resolution. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 698–715. Springer, 2020. 5

[13] Saurabh Gupta, Pablo Arbelaez, and Jitendra Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *Conference on Computer Vision and Pattern Recognition*, June 2013. 2, 5

[14] Jianzhong He, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. Bdcn: Bi-directional cascade network for perceptual edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):100–113, 2022. 1, 2, 3, 4, 6, 7

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[16] Xiaodi Hou, Alan Yuille, and Christof Koch. Boundary detection benchmarking: Beyond f-measures. In *Conference on Computer Vision and Pattern Recognition*, June 2013. 1, 2, 5, 7

[17] Linxi Huan, Nan Xue, Xianwei Zheng, Wei He, Jianya Gong, and Gui-Song Xia. Unmixing convolutional features for crisp edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6602–6609, 2022. 2, 3, 4, 5, 6

[18] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015. 5

[19] Kun Huang, Yifan Wang, Zihan Zhou, Tianjiao Ding, Shenghua Gao, and Yi Ma. Learning to parse wireframes in images of man-made environments. In *CVPR*, June 2018. 5

[20] Andrey Ignatov, Radu Timofte, et al. Pirm challenge on perceptual image enhancement on smartphones: report. In *European Conference on Computer Vision (ECCV) Workshops*, January 2019. 5

[21] Junfeng Jing, Shenjuan Liu, Gang Wang, Weichuan Zhang, and Changming Sun. Recent advances on image edge detection: A comprehensive review. *Neurocomputing*, 2022. 5

[22] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022. 1

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6

[24] Ou Li and Peng-Lang Shui. Noise-robust color edge detection using anisotropic morphological directional derivative matrix. *Signal Processing*, 165:90–103, 2019. 1, 5

[25] Ou Li and Peng-Lang Shui. Noise-robust color edge detection using anisotropic morphological directional derivative matrix. *Signal Processing*, 165:90–103, 2019. 5

[26] Y. Liu, M. Cheng, X. Hu, J. Bian, L. Zhang, X. Bai, and J. Tang. Richer convolutional features for edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1939–1946, 2019. 2, 3

[27] Yanchun Ma, Yongjian Liu, Qing Xie, Shengwu Xiong, Lihua Bai, and Anshu Hu. A tibetan thangka data set and relative tasks. *Image and Vision Computing*, 108:104125, 2021. 5

[28] Raman Maini and Himanshu Aggarwal. Study and comparison of various image edge detection techniques. *International Journal of Image Processing*, 3(1), 2009. 2

[29] D.R. Martin, C.C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and

texture cues. *Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004. 2, 5

[30] David A Mély, Junkyung Kim, Mason McGill, Yuliang Guo, and Thomas Serre. A systematic comparison between visual cues for boundary detection. *Vision Research*, 120, 2016. 2, 5

[31] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *Conference on Computer Vision and Pattern Recognition*, 2014. 2, 5

[32] Mengyang Pu, Yaping Huang, Qingji Guan, and Haibin Ling. Rindnet: Edge detection for discontinuity in reflectance, illumination, normal and depth. In *International Conference on Computer Vision*, pages 6879–6888, 2021. 2, 5

[33] Mengyang Pu, Yaping Huang, Yuming Liu, Qingji Guan, and Haibin Ling. Edter: Edge detection with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1402–1412, June 2022. 1, 2

[34] Olaf Ronneberger, Philipp Fischer, and Thomas 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. 2

[35] Diana Sadykova and Alex Pappachen James. Quality assessment metrics for edge detection and edge-aware filtering: A tutorial review. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2366–2369. IEEE, 2017. 5

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[37] Irwin Sobel. Camera models and machine perception. Technical report, Computer Science Department, Technion, 1972. 1

[38] Jifei Song, Qian Yu, Yi-Zhe Song, Tao Xiang, and Timothy M Hospedales. Deep spatial-semantic attention for fine-grained sketch-based image retrieval. In *Proceedings of the IEEE international conference on computer vision*, pages 5551–5560, 2017. 2, 5, 6, 7

[39] Xavier Soria, Gonzalo Pomboza-Junez, and Angel Domingo Sappa. Ldc: Lightweight dense cnn for edge detection. *IEEE Access*, 10:68281–68290, 2022. 3, 5, 6, 7

[40] Xavier Soria, Edgar Riba, and Angel Sappa. Dense extreme inception network: Towards a robust cnn model for edge detection. In *Winter Conference on Applications of Computer Vision*, 2020. 1, 2, 3, 4, 5

[41] Xavier Soria, Angel Sappa, Patricio Humanante, and Arash Akbarinia. Dense extreme inception network for edge detection. *Pattern Recognition*, 139:109461, 2023. 1, 2, 3, 5, 6, 7

[42] Zhuo Su, Wenzhe Liu, Zitong Yu, Dewen Hu, Qing Liao, Qi Tian, Matti Pietikäinen, and Li Liu. Pixel difference networks for efficient edge detection. In *International Conference on Computer Vision (ICCV)*, pages 5097–5107, 2021. 1, 2, 3, 5, 6, 7

[43] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 3

[44] Nazish Tariq, Rostam Affendi Hamzah, Theam Foo Ng, Shir Li Wang, and Haidi Ibrahim. Quality assessment methods to evaluate the performance of edge detection algorithms for digital image: A systematic literature review. *IEEE Access*, 9:87763–87776, 2021. 5

[45] Xueliang Wang, Honge Ren, and Achuan Wang. Smish: A novel activation function for deep learning methods. *Electronics*, 11(4):540, 2022. 3, 4

[46] Zhou 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. 3

[47] Jan Kristanto Wibisono and Hsueh-Ming Hang. Traditional method inspired deep neural network for edge detection. In *IEEE International Conference on Image Processing (ICIP)*, pages 678–682, 2020. 6, 7

[48] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015. 1, 2, 3, 4

[49] Wenjie Xuan, Shaoli Huang, Juhua Liu, and Bo Du. Fcl-net: Towards accurate edge detection via fine-scale corrective learning. *Neural Networks*, 145:248–259, 2022. 2, 3

[50] Daipeng Yang, Bo Peng, Zaid Al-Huda, Asad Malik, and Donghai Zhai. An overview of edge and object contour detection. *Neurocomputing*, 2022. 1, 2, 6

[51] Wenming Yang, Xuechen Zhang, Yapeng Tian, Wei Wang, Jing-Hao Xue, and Qingmin Liao. Deep learning for single image super-resolution: A brief review. *IEEE Transactions on Multimedia*, 21(12):3106–3121, 2019. 1

[52] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Chen-Change Loy. Sketch me that shoe. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 799–807, 2016. 5, 6, 7

[53] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *CVPR*, pages 12104–12113, 2022. 1

[54] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127:302–321, 2019. 5

[55] Djemel Ziou, Salvatore Tabbone, et al. Edge detection techniques-an overview. *Pattern Recognition and Image Analysis C/C of Raspoznavaniye Obrazov I Analiz Izobrazhenii*, 8, 1998. 2, 6