Efficient Transformer-based Edge Detector

1st Jinghuai Jie, 2nd Qifan Wang, 3rd Junmin Wu, 4th Yan Guo*,5th Baojian Hua*
University of Science and Technology of China, Hefei, China
Suzhou Institute for Advanced Research, USTC, Suzhou, China
jhjie@mail.ustc.edu.cn,wangqifan@mail.ustc.edu.cn,jmwu@ustc.edu.cn, guoyan@ustc.edu.cn,bjhua@ustc.edu.cn

***Abstract*—Edge detection is a core component in a wide range of vision tasks, and is expected to be both efficient and accurate to identify boundaries and edges in images. And it has been a hot research field for long period. Currently, vision transformers are playing an increasingly prominent role in various downstream tasks, and the SOTA edge detector EDTER employs vision transformer as its encoder. However, ViTs are also known for their computational burden due to the large amount of parameters, leading to higher processing latency than lightweight CNNs. Recently, EfficientFormerV2, which has a novel network with low latency and high parameter efficiency, has proved that transformer-based network could outperform CNN-based network in both accuracy and efficiency. Inspired by this architecture, this paper proposes a novel edge detector EFED with EfficientFormerV2 as the encoder, and an efficient Multi-Level Aggregation decoder SMLA to extract both local and global features. Extensive experiments are conducted on two widely employed datasets, BSDS500 and NYUDv2, demonstrating that compared with EDTER, our detector not only improves the throughput by 10 times, but also achieves competitive accuracy. With single scale input on BSDS500 dataset, our EFED model achieves ODS F-measure and OIS F-measure of 82.4% and 84.2%, while for EDTER the corresponding values are 82.4% and 84.1%, respectively.**

***Index Terms*—edge detection, EfficientformerV2, TFED, SMLA**

I. Introduction

Edge detection is a fundamental computer vision problem as it is basis for a wide variety of applications, such as object detection [1], image segmentation [2], and object tracking [3]. Edge detection aims to extract object boundaries and visually salient edges both accurately and speedily.

Edge detectors have developed from traditional methods to deep learning algorithms. Intuitively, intense variation in color and other visual cues indicates the existence of edges. Therefore, traditional methods [4,5] mostly obtain edges based on local and low-level characteristics such as color and texture. However, the inability to capture high level and global semantic information is the drawback of traditional methods, resulting low detection precision. To obtain appropriate representation of both high and low level visual cues, convolutional neural networks (CNNs) based methods are proposed and significant progress has been made [6,7], since the hierarchical structure of CNN are good at learning global semantic features. On the other hand, while CNN enlarges the receptive fields and grasp global features, some essential fine details are inevitably and gradually lost.

Since its first introduction in 2020, Vision Transformers (ViTs) [8,9] have been playing a significant role in various vision tasks and inspired many follow-up works to further improve the model architecture. A transformer-based edge detector, Edge Detection TransformER (EDTER)[10], has achieved State-of-the-Art result on BSDS500[11], NYUDv2[12], and Multicue[13] datasets. In transformer, Multi Head Self Attention (MHSA) is the essential mechanism to effectively model spatial dependencies and enable global receptive field. However, the cost of MSHA is quadratic computation complexity with respect to the number of tokens (resolution). As a result, transformer-based architectures are more computation intensive and have higher latency compared to widely adopted CNNs networks. For example, the training of EDTER takes about 26.4 hours (15.1 for Stage I and 11.3 for Stage II), and inference runs at 2.2 FPS on a V100 with single scale inputs.

To address the problem, one research direction is to reduce the quadratic computation complexity of the attention mechanism. Swin [9] and subsequent works [14, 15] propose window-based attention so that the receptive field is constrained to a pre-defined window size, which reduces the computation complexity to be linear to resolution. Another direction is to combine lightweight CNN and attention mechanism to form a hybrid architecture, which can naturally avoid performing MHSA on high resolution and save computations. Recently, EfficientFormerV2[16] has further improved EfficientFormer[17], comprehensively studies mobile-friendly design choices and introduce novel changes, producing a vision transformer model as small and fast as MobileNetV2 while obtaining better performance. Therefore, EfficientFormerV2 has the potential to serve as an efficient backbone in various downstream tasks.

In this work, we employ EfficientFormerV2 as the backbone encoder for edge detection task with high precision and performance. We also introduce a multi-level aggregation decoder to extract both global semantics and local cues. Compared with EDTER, our inference throughput (25.51 FPS on RTX 4090 GPU with single scale inputs) is over 10 times higher than that of EDTER (2.2 FPS on V100 GPU with single scale inputs).

Our contributions can be summarized in three folds: (1) We introduce EFED, which employs EfficientFormerV2 as the backbone encoder, and enables speedy and precise edge detection. (2) We propose a simple yet efficient SMLA decoder to efficiently integrate both local and global features and extract rich feature information. (3) We conduct extensive experiments on widely used edge detection benchmarks, BSDS500 and

* Corresponding authors: guoyan@ustc.edu.cn; bjhua@ustc.edu.cn.

NYUDv2, demonstrating the competitive performance and high efficiency of our model when compared to state-of-the-art methods. We also conduct experiments to test the adaptability and flexibility of our architecture, and four variants of different sizes are designed and tested.

II. Related Work

As a fundamental vision task, edge detection has been extensively studied over years. In the following, we highlight related works mainly from two aspects: edge detectors and vision transformers.

A. Edge Detectors

Early edge detectors rely on local information and analyze image gradients to detect intense feature change and extract edges, with Canny [6] as typical representatives. These methods are quite efficient but with obvious drawbacks, since they are unable to obtain global semantic features. With the development of deep learning algorithms, convolutional neural networks (CNNs) have been successfully introduced in edge detection tasks, since the intrinsic multi-level structure are able to gradually attain global information. DeepEdge [3] exploits object-aware cues for contour detection. HED [18] supervises side output layers to learn rich hierarchical features, whose outstanding performance boosts the development of edge detection with CNN. RCF [19] is another milestone work of edge detection, and it combines hierarchical features from all convolutional layers. To achieve effective results, BDCN [20] is another representative work, which outperforms previous works by utilizing layer-specific supervision inferred from a bi-directional cascade structure. More recently, EDTER [10] accomplishes the edge detection task using ViT [12], which employs two-stage architecture to extract global and local feature, considerably improves the accuracy, but at the cost of high computational burden. To reduce computational expense, PiDiNet [21] integrates the traditional edge detection operators into a CNN model, that is, pixel difference convolution, to extract edge-related features rather than employing pre-trained networks, achieving considerably reduction in the model size while keeping competitive accuracy. UAED[29] introduces an uncertainty aware edge detector, which employs uncertainty to investigate the subjectivity and ambiguity of diverse annotations, and enables the network to concentrate on the important pixels.

B. Vision Transformers

Since 2020, vision transformers have been more and more widely used in various vision tasks. ViT[8] directly uses the transformer to process sequences of image patches and achieves the state-of-the-art, which encourages further improvement in applications of vision transformers. Later researches includes hierarchical design, injecting locality, or exploring different types of token mixing, etc.

With its advantageous performance, one research direction is the efficient deployment of ViTs. For reducing the computation complexity of ViTs, many works propose new modules and architecture design, while others eliminate redundancies in attention mechanism. Similar to CNNs, various optimization methods such as architecture search, pruning, and quantization are also explored for ViTs. However, there are still major obstacles to render transformers more efficient[23,24]. For example, even the quadratic computation complexity could be reduced by regularizing the span, the reshaping and indexing operations may not be supported on resource constrained devices. Based on such consideration, EfficientFormer[17] analyzes the network architecture and operators used in ViT-based models, identifies inefficient designs, and then introduces a dimension-consistent pure transformer as a design paradigm. Later, EfficientFormerV2[16] introduces a novel fine-grained joint search strategy for transformer models that can find efficient architectures by optimizing latency and number of parameters simultaneously. Experiments show that EfficientFormerV2 achieves higher accuracy than MobileNetV2 while keeping similar latency.

III. Edge Detection with EFED

The overall framework of the proposed EFED is illustrated in Fig. 1. It is clear that EFED has a lightweight structure, with EfficientFormerV2 as the encoder, and a simple Feature Pyramid Network and Multi-level aggregation structure as the decoder. This section introduces the details of the proposed EFED network: firstly, a brief overview of the encoder; secondly, the design of the SMLA decoder; and finally, the loss function.

A. EfficientFormerV2

EfficientFormerV2 employs a 4-stage hierarchical design which obtains feature sizes in $\left\{\frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}\right\}$ of the input resolution. Similar to its predecessor [17], EfficientFormerV2 starts with a small kernel convolution stem. to embed input image instead of using inefficient embedding of non-overlapping patches,

$$X_{i|_{i=1},j|_{j+1}}^{B,C_j|_{j+1},\frac{H}{4},\frac{W}{4}} = stem(X_0^{B,3,H,W}) \tag{1}$$

where B denotes the batch size, C refers to channel dimension (also represents the width of the network), H and W are the height and width of the feature, X_j is the feature in stage j, $j \in \{1,2,3,4\}$, and $i$indicates the $i-th$ layer. The first two stages capture local information on high resolutions; thus a unified Feed Forward Network (FFN) is employed,

$$X_{i+i,j}^{B,C_j,\frac{H}{2^{j+1}},\frac{W}{2^{j+1}}} = S_{i,j}\dot{F}FN^{C_j,E_{i,j}}(X_{i,j}) + X_{i,j} \tag{2}$$

where $S_{i,j}$ is a learnable layer scale and the FFN is constructed by two properties: stage width C_j and a per-block expansion ratio. Each FFN is residual connected. In the last two stages, both local FFN and global MHSA blocks are used. Therefore, on top of Eqn.2, global blocks are defined as:

$$X_{i+i,j}^{B,C_j,\frac{H}{2^{j+1}},\frac{W}{2^{j+1}}} = S_{i,j}\dot{M}HSA(Proj(X_{i,j}) + X_{i,j} \tag{3}$$

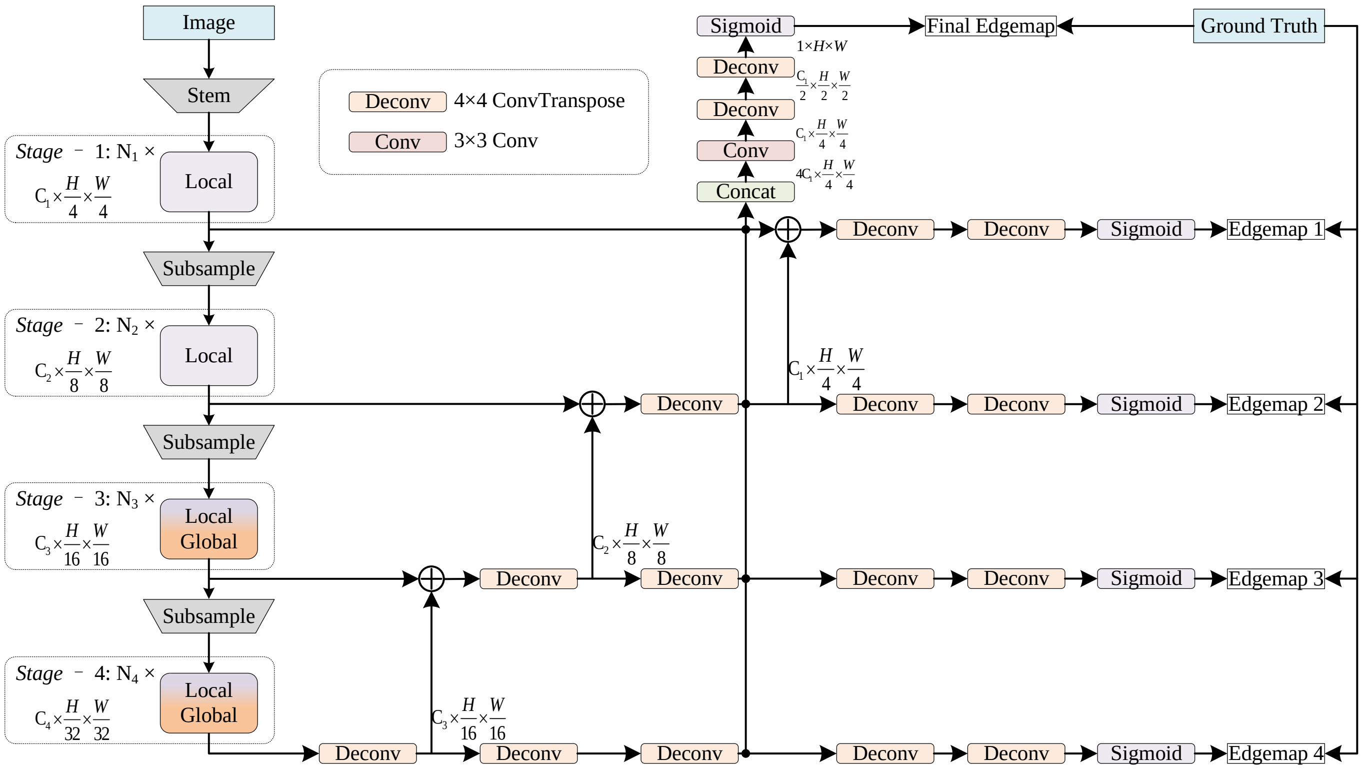

Fig. 1. The overall framework of our proposed EFED. EFED mainly consists of two modules: encoder and decoder. The EfficientFormerV2 is used as encoder to extract the features from different levels. The decoder, Simple Multi-Level Aggregation(SMLA) fuses the detail and semantic features from high level to low level. For each level, a side edgemap is generated, as well as a primary edgemap, all of which are used to compute the loss and produce the final edgemap.

where Queries (Q), Keys (K), and Values (V) are projected from input features through linear layers Q, K, V ←Proj($X_{i,j}$), and

$$MHSA(Q, K, V) = Softmax(Q\dot{K}^T + ab)\dot{V} \tag{4}$$

with ab as a learnable attention bias for position encoding.

B. Decoder

Multi-level feature aggregation is crucial for detecting precise and thin edges, which could lead to generation of rich semantic information. The well known and widely used Feature Pyramid Network could effectively integrate features. Taking into account of both efficiency and effectiveness, and inspired by the multi-level feature aggregation in vision tasks [10,19], we propose a Simple Multi-Level Aggregation (SMLA) decoder, as illustrated in Fig. 1. The FPN and MLA design enables the decoder to learn richer and more informative feature representation, thus improve the overall performance of the model.

Specifically, we perform a deconvolution(4x4 transpose convolution) operation, on the Stage-4 output feature map $F_4 \in \mathbb{R}^{C_4\times\frac{H}{32}\times\frac{W}{32}}$ to upsample it to $F_4' \in \mathbb{R}^{C_3\times\frac{H}{16}\times\frac{W}{16}}$. We then apply two deconvolution operations on F_4' to generate $F_4''' \in \mathbb{R}^{C_1\times\frac{H}{4}\times\frac{W}{4}}$. For the Stage-3 output feature map F_3, we add it with F_4' and perform a deconvolution operation to generate $F_3' \in \mathbb{R}^{C_2\times\frac{H}{8}\times\frac{W}{8}}$. We further apply a deconvolution operation on F_3' to generate $F_3'' \in \mathbb{R}^{C_1\times\frac{H}{4}\times\frac{W}{4}}$. For the Stage-2 output feature map F_2, we add it with F_3' and perform a deconvolution operation to generate $F_2' \in \mathbb{R}^{C_1\times\frac{H}{4}\times\frac{W}{4}}$. Finally, we add the Stage-1 output feature map F_1 with F_2' to generate $F_1' \in \mathbb{R}^{C_1\times\frac{H}{4}\times\frac{W}{4}}$. We concatenate F_1, F_2', F_3'', and F_4'' and apply a convolution operation to generate $F \in \mathbb{R}^{C_1\times\frac{H}{4}\times\frac{W}{4}}$. Lastly, we perform two deconvolution operations and a sigmoid operation on the feature maps F, F_1, F_2', F_3'', and F_4''' to generate the primary edge map $E \in \mathbb{R}^{1\times H\times W}$, and the side edge maps $S_1, S_2, S_3, S_4 \in \mathbb{R}^{1\times H\times W}$.

C. Loss Function

We employ the loss function proposed in [18] for all the five edge maps. Given an edge map E and the corresponding ground truth Y, the loss is computed as follows:

$$\begin{aligned}\ell(E, Y) = -\sum_{i,j}&(Y_{i,j}\alpha\log{(E_{i,j})}\\ &+(1-Y_{i,j})(1-\alpha)\log{(1-E_{i,j})}),\end{aligned} \tag{5}$$

where $E_{i,j}$ and $Y_{i,j}$ are the $(i,j)^{th}$ element of matrix E and Y, respectively. $\alpha = \frac{|Y^-|}{|Y^-|+|Y^+|}$ represents the percentage of negative pixel samples, with $|\cdot|$ denoting the number of pixels. Since BSDS500 dataset is annotated by multiple annotators, firstly, the multiple annotations should be normalized into edge probability maps within the range of [0, 1]. Then, if the probability of a pixel is greater than a threshold value η, it is indicated as a positive sample; otherwise, it is labeled as a negative sample.

For the primary edge map, denoted as $\mathcal{E}$, and four edge maps, denoted as $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, and $\mathcal{S}_4$, the loss is calculated separately according to Eq. 5. And the overall loss function is as follows:

$$\mathcal{L} = \mathcal{L}_{\mathcal{E}} + \lambda \mathcal{L}_{\mathcal{S}} = \ell(\mathcal{E}, Y) + \lambda \sum_{k=1}^{4} \ell\left(\mathcal{S}_k, Y\right), \qquad (6)$$

$\mathcal{L}_{\mathcal{E}}$ and $\mathcal{L}_{\mathcal{S}}$ represent the losses for the primary edge map $\mathcal{E}$ and the side edge maps $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4$, respectively. Meanwhile, λ denotes the weight that balances $\mathcal{L}_{\mathcal{E}}$ and $\mathcal{L}_{\mathcal{S}}$. Based on previous works and our experimental observations, we set λ to 0.4.

IV. Experiments

A. Implementation Details

The proposed network is implemented using PyTorch library on RTX 4090 GPU. In detail, EFED uses the pre-trained weights of EfficientFormerV2 to initialize the encoder and is trained for 40k iterations using the AdamW optimizer. A cosine decay learning rate scheduler is employed, and the first 15k iterations warm up the learning rate in a linear manner, and the remaining iterations are decayed according to the scheduler. The initial learning rate is 0 and a preset learning rate is set to 6e-5. For BSDS500, the batch size is set to 8, and for NYUDv2, the batch size is set to 4. The momentum and weight decay of the optimizer are set to 0.9 and 0.001, respectively.

The training of the EFED-L model (27.18 MB) takes about 4 hours on RTX 4090, far more efficient than Transformer-based model EDTER (468.84MB), which takes 26.4 hours on V100. The inference throughput is 25.51 FPS on RTX 4090, ten times the throughput of EDTER on V100 (2.2 FPS). The GPU memory requirement is about 6GB, nearly 1/5 of EDTER(29GB).

When evaluating, standard non-maximum suppression (NMS) is applied to thin detected edges, and both Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS) F-score are reported.

B. Datasets

Two datasets are employed for evaluation include: 1)Widely used BSDS500, which has 200 training, 100 validation and 200 test images; following previous works, we train our model on the data consisting of augmented BSDS and VOC Context dataset; and 2) NYU Depth (NYUD) dataset, which contains 1449 RGB and HHA image pairs, with train (381 images), validation (414 images), and test sets (654 images). For BSDS500, we set the threshold η to 0.3 to select positive samples. For NYUDv2, there is no need to set the threshold η since only one annotation exists per picture. As for the maximum allowed tolerance distance between the detected edge and ground truth, following the examples of previous works, for BSDS500 it is 0.0075 and for NYUDv2 it is 0.011.

TABLE I

Results on BSDS500 testing set. † means training with extra PASCAL VOC data, and ‡ is the multi-scale testing. It is worth noting that with single scale input, the OIS of our model is 0.842, exceeding that of EDTER, 0.841; and with VOC extra data for training, the OIS of our model is equal to that of EDTER.

	Method	Pub.'Year	ODS	OIS
Traditional	Canny	PAMI'86	0.611	0.676
	gPb-UCM	PAMI'10	0.729	0.755
	SCG	NeurIPS'12	0.739	0.758
	SE	PAMI'14	0.743	0.764
	OEF	CVPR'15	0.746	0.770
CNN-based	DeepEdge	CVPR'15	0.753	0.772
	DeepContour	CVPR'15	0.757	0.776
	HED	ICCV'15	0.788	0.808
	Deep Boundary†‡	ICLR'15	0.789	0.811
	CEDN	CVPR'16	0.788	0.804
	RDS	CVPR'16	0.792	0.810
	AMH-Net	NeurIPS'17	0.798	0.829
	RCF†‡	CVPR'17	0.811	0.830
	CED†	CVPR'17	0.815	0.833
	LPCB†‡	ECCV'18	0.815	0.834
	BDCN†‡	CVPR'19	0.828	0.844
	DSCD†‡	ACMMM'20	0.822	0.859
	PiDiNet†	ICCV'21	0.807	0.823
	UAED†‡	CVPR'23	0.844	0.864
	PEdger-large†	ACMMM'23	0.823	0.841
Transformer-based	EDTER	CVPR'22	0.824	0.841
	EDTER†		0.832	0.847
	EDTER‡		0.840	0.858
	EDTER†‡		0.848	0.865
	ETED-L	Ours	0.824	0.842
	ETED-L†		0.829	0.847
	ETED-L‡		0.836	0.854
	ETED-L†‡		0.842	0.860

C. Comparison with State-Of-The-Art

We compare our model with previous works on both datasets from various aspects.

On BSDS500 dataset. We compare our L model with *traditional detectors* such as Canny, gPb-UCM, SCG, SE and OEF, and *CNN-based detector* such as DeepEdge, DeepContour, HED, Deep Boundary, CEDN, RDS, AMH-Net, RCF, CED, LPCB, BDCN, DSCD, PiDiNet, UAED and PEdger, and *transformer-based detector* EDTER. The results are summarized in Table I. We notice that our L model, trained on the BSDS500 dataset, achieves an OIS of 0.842 with single-scale inputs, exceeding the current SOTA EDTER. With additional training data and multi-scale testing (following the settings of RCF, CED, BDCN, etc.), our method achieves 0.842 (ODS), 0.860 (OIS), which is superior to most of existing edge detectors, inferior to only EDTER and UAED. Nonetheless, with singe-scale input the inference speed of EFED is superior to both EDTER(2.2FPS on V100)and UAED(17FPS on RTX3090). Several qualitative results of challenging samples in the testing set of BSDS500 are presented in Fig. 2. The generated outputs exhibit clear and exact edge predictions, further validating the efficacy of our method. Fig. 3 shows Precision-Recall curves of all methods on BSDS500, further validating the effectiveness of EFED.

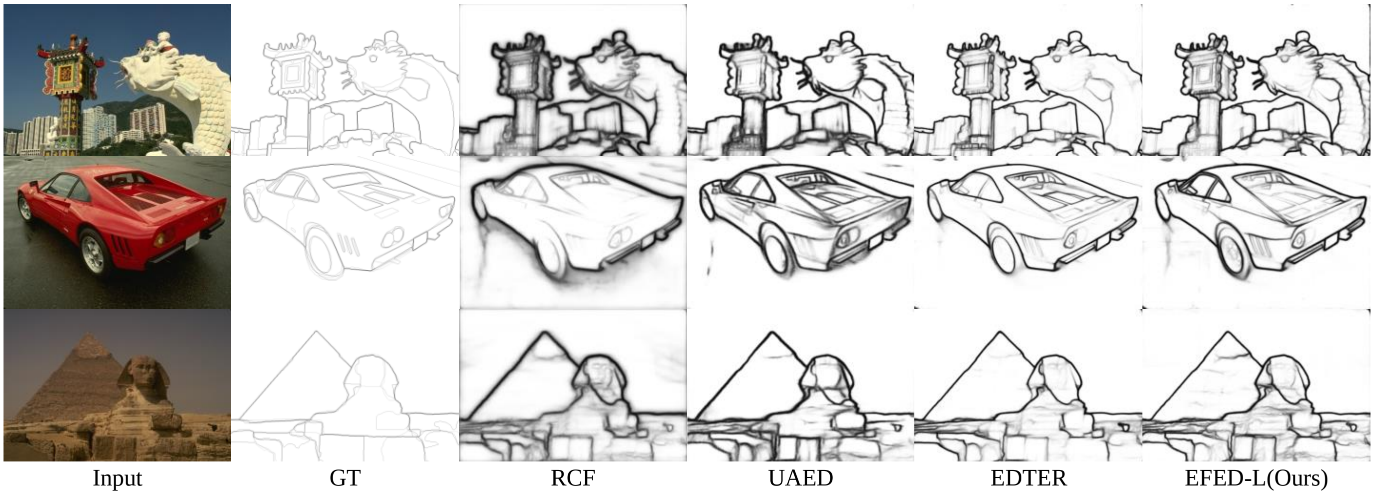

Fig. 2. Qualitative comparisons on three challenging samples in the testing set of BSDS500.

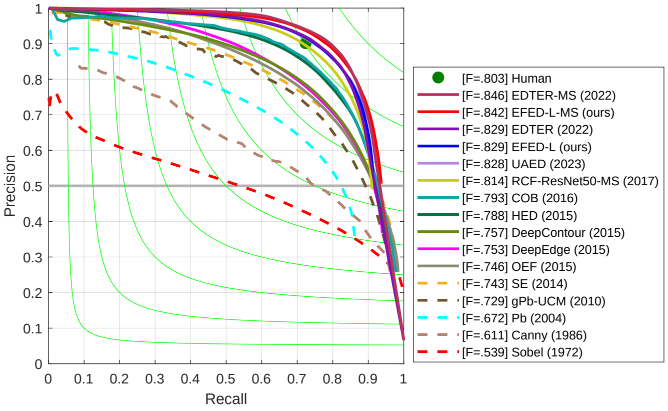

Fig. 3. The precision-recall curves on BSDS500.

TABLE II
QUANTITATIVE COMPARISONS ON NYUDV2. ALL RESULTS ARE COMPUTED WITH A SINGLE SCALE INPUT.THE BEST TWO RESULTS ARE HIGHLIGHTED IN **RED** AND **BLUE**, RESPECTIVELY.

	Method	Pub.' Year	ODS	OIS
Traditional	gPb-UCM	PAMI'10	0.632	0.661
	gPb+NG	CVPR'13	0.687	0.716
	SE	PAMI'14	0.695	0.708
	SE+NG+	ECCV'14	0.706	0.734
	OEF	CVPR'15	0.651	0.667
	SemiContour	CVPR'16	0.680	0.700
CNN-based	HED	ICCV'15	0.720	0.734
	RCF	CVPR'17	0.729	0.742
	AMH-Net	NeurIPS'17	0.744	0.758
	LPCB	ECCV'18	0.739	0.754
	BDCN	CVPR19	**0.748**	0.763
	PiDiNet	ICCV'21	0.733	0.747
	PEdger	ACMMM'23	0.742	0.757
TF	EDTER	CVPR'22	**0.774**	**0.789**
	EFET-L	Ours	0.744	**0.768**

On NYUDv2 dataset. As for NYUDv2 dataset, we conduct experiments on RGB images and compare our L model against the state-of-the-art methods including *traditional detectors* gPb-ucm, gPb+NG,SE, SE+NG+, OEF, SemiContour, *CNN-based detector* HED, RCF , AMH-Net, LPCB, BDCN, PiDiNet, and and *transformer-based detector* EDTER. All results are based on single-scale input. Table II shows the quantitative results of our method and other competitors. Our method achieves the second best score of 0.768 of OIS with single scale input .

D. Scalability Tests

In order to adapt to different application scenarios, we design four variants with different model size, and conduct scalability experiments on them. As for the EFED variants, the configuration settings of the encoder in the L, S0, S1, and S2 models are consistent with the L, S0, S1, and S2 variants of EfficientFormerV2. Extensive experiments are conducted to study the scalability and throughput of EFED variants. The result is shown in Table III. The models are all trained using the BSDS500 training and validation sets and evaluated with the BSDS500 test set. We also test the ODS and OIS with extra training data. As expected, when the size of our model decreases, the ODS and OIS will decrease accordingly, and at the same time the throughput and parameters increases.

TABLE III
SCALABILITY EXPERIMENTS

Variants	***ODS/OIS***	***ODS/OIS†***	***Parameters***	***Throughput***
S0	0.782/0.809	0.799/0.823	3.71M	48.9FPS
S1	0.806/0.826	0.815/0.833	6.37M	42.86FPS
S2	0.817/0.834	0.823/0.840	13.08M	32.15FPS
L	0.824/0.842	0.829/0.847	27.18M	25.51FPS

E. Ablation Study

We perform our ablation study on BSDS dataset to verify the effectiveness of our proposed decoder. We first compare the effect of different up-sampling methods,namely, the 1*1 convolution kernel with bilinear interpolation,the 3*3 convolution kernel with bilinear interpolation, as well as 4*4 transpose convolution; then the effect of bottom-up path is also verified. From the quantitative results shown in Table IV, it is clear that decoder with transpose convolution achieves best performance. Even though transpose convolution introduces more parameters than the other two, its throughput is the second of the three.

TABLE IV
ABLATION STUDY ON UPSAMPLING

Upsample	***ODS***	***OIS***	***Parameters***	***Throughput***
1*1conv+bilinear	0.815	0.833	25.68M	25.59FPS
3*3conv+bilinear	0.821	0.838	26.45M	25.00FPS
ConvTranspose	0.824	0.842	27.18M	25.51FPS

Next, we carry out the ablation study of bottom-up path which is commonly used in previous works[10,20]. To be more computationally efficient, EFED gives up bottom-up path. And the results in Table V. shows that the presence of bottom-up path has no obvious positive effect on either ODS or OIS, however, it degrades the throughput from 25.51 FPS to 24.45 FPS.

TABLE V
ABLATION STUDY ON BOTTOM-UP PATH

	ODS	***OIS***	***Parameters***	***Throughput***
-	0.824	0.842	27.18M	25.51FPS
Bottom-Up Path	0.824	0.842	30.00M	24.45FPS

CONCLUSION

In this paper, we propose a novel efficient transformer-based edge detection framework, namely EFED. By introducing EffcientFormerV2 as the encoder, EFED is able to capture multi-level features with accuracy and efficiency. Moreover, EFED employs a simple yet powerful Multi-Level Aggregation (SMLA) decoder to explore high-resolution representations. Besides, Feature Pyramid Network (FPN) incorporates global and local contexts to better predict the edge results. Experimental results illustrate that EFED yields competitive results in comparison with state-of-the-arts.

REFERENCES

[1] Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. ”Faster r-cnn: Towards real-time object detection with region proposal networks.” Advances in neural information processing systems 28 (2015). Hu, Zeyu, Mingmin Zhen, Xuyang Bai, Hongbo Fu, and Chiew-lan Tai.

[2] Hu, Zeyu, Mingmin Zhen, Xuyang Bai, Hongbo Fu, and Chiew-lan Tai. ”Jsenet: Joint semantic segmentation and edge detection network for 3d point clouds.” In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16, pp. 222-239.

[3] Wang, Qiang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. ”Fast online object tracking and segmentation: A unifying approach.” In Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition, 2019, pp. 1328-1338.

[4] Canny, John. ”A computational approach to edge detection.” IEEE Transactions on pattern analysis and machine intelligence, 1986, 679-698.

[5] Dollar, Piotr, Zhuowen Tu, and Serge Belongie. ”Supervised learning of edges and object boundaries.” In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06), vol. 2, pp. 1964-1971. IEEE, 2006.

[6] Bertasius, Gedas, Jianbo Shi, and Lorenzo Torresani. ”Deepedge: A multi-scale bifurcated deep network for top-down contour detection.” In Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 4380-4389.

[7] Bertasius, Gedas, Jianbo Shi, and Lorenzo Torresani. ”High-for-low and low-for-high: Efficient boundary detection from deep object features and its applications to high-level vision.” In Proceedings of the IEEE international conference on computer vision, 2015, pp. 504-512.

[8] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani et al. ”An image is worth 16x16 words: Transformers for image recognition at scale.” 2020, arXiv preprint arXiv:2010.11929.

[9] Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. ”Swin transformer: Hierarchical vision transformer using shifted windows.” In Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 10012-10022.

[10] Pu, Mengyang, 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, 2022, pp. 1402-1412.

[11] Arbelaez, Pablo, Michael Maire, Charless Fowlkes, and Jitendra Malik. ”Contour detection and hierarchical image segmentation.” IEEE transactions on pattern analysis and machine intelligence 33, no. 5 2010, pp. 898-916.

[12] Silberman, Nathan, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. ”Indoor segmentation and support inference from rgbd images.” In Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12, pp. 746-760.

[13] Mély, David A., Junkyung Kim, Mason McGill, Yuliang Guo, and Thomas Serre. ”A systematic comparison between visual cues for boundary detection.” Vision research 120, 2016, pp. 93-107.

[14] Dong, Xiaoyi, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. ”Cswin transformer: A general vision transformer backbone with cross-shaped windows.” In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 12124-12134.

[15] Liu, Ze, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning et al. ”Swin transformer v2: Scaling up capacity and resolution.” In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 12009-12019.

[16] Li, Yanyu, Ju Hu, Yang Wen, Georgios Evangelidis, Kamyar Salahi, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. ”Rethinking vision transformers for mobilenet size and speed.” In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 16889-16900.

[17] Li, Yanyu, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. ”Efficientformer: Vision transformers at mobilenet speed.” Advances in Neural Information Processing Systems 35, 2022, pp. 12934-12949.

[18] Xie, Saining, and Zhuowen Tu. ”Holistically-nested edge detection.” In Proceedings of the IEEE international conference on computer vision, 2015, pp. 1395-1403.

[19] Liu, Yun, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. ”Richer convolutional features for edge detection.” In Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 3000-3009.

[20] He, Jianzhong, Shiliang Zhang, Ming Yang, Yanhu Shan, and Tiejun Huang. ”Bi-directional cascade network for perceptual edge detection.” In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 3828-3837.

[21] Su, Zhuo, Wenzhe Liu, Zitong Yu, Dewen Hu, Qing Liao, Qi Tian, Matti Pietikäinen, and Li Liu. ”Pixel difference networks for efficient edge detection.” In Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 5117-5127.

[22] Wang, Wenhai, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. ”Pyramid vision transformer: A versatile backbone for dense prediction without convolutions.” In Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 568-578.

[23] Jin, Qing, Jian Ren, Oliver J. Woodford, Jiazhuo Wang, Geng Yuan, Yanzhi Wang, and Sergey Tulyakov. ”Teachers do more than teach: Compressing image-to-image models.” In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13600-13611.

[24] Jin, Qing, Jian Ren, Richard Zhuang, Sumant Hanumante, Zhengang Li, Zhiyu Chen, Yanzhi Wang, Kaiyuan Yang, and Sergey Tulyakov. ”F8net: Fixed-point 8-bit only multiplication for network quantization.” arXiv preprint arXiv:2202.05239 (2022).

[25] Yu, Changqian, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. ”Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation.” International Journal of Computer Vision 129, 2021, 3051-3068.

[26] Li, Hanchao, Pengfei Xiong, Haoqiang Fan, and Jian Sun. ”Dfanet: Deep feature aggregation for real-time semantic segmentation.” In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 9522-9531.

[27] Huang, Zilong, Yunchao Wei, Xinggang Wang, Wenyu Liu, Thomas S. Huang, and Humphrey Shi. ”Alignseg: Feature-aligned segmentation networks.” IEEE Transactions on Pattern Analysis and Machine Intelligence 44, no. 1, 2021, 550-557.

[28] Peng, Juncai, Yi Liu, Shiyu Tang, Yuying Hao, Lutao Chu, Guowei Chen, Zewu Wu et al. ”Pp-liteseg: A superior real-time semantic segmentation model.” arXiv preprint arXiv:2204.02681 (2022).

[29] Zhou, Caixia, Yaping Huang, Mengyang Pu, Qingji Guan, Li Huang, and Haibin Ling. ”The Treasure Beneath Multiple Annotations: An Uncertainty-aware Edge Detector.” In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 15507-15517.