ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240518

EdgeNAT: Transformer for Efficient Edge Detection

Jinghuai Jie[a], Yan Guo[a,*], Guixing Wu[a], Junmin Wu[a] and Baojian Hua[a,]**

[a]University of Science and Technology of China

Abstract. Transformers, renowned for their powerful feature extraction capabilities, have played an increasingly prominent role in various vision tasks. Especially, recent advancements present transformer with hierarchical structures such as *Dilated Neighborhood Attention Transformer* (DiNAT), demonstrating outstanding ability to efficiently capture both global and local features. However, transformers' application in edge detection has not been fully exploited. In this paper, we propose *EdgeNAT*, a one-stage transformer-based edge detector with DiNAT as the encoder, capable of extracting object boundaries and meaningful edges both accurately and efficiently. On the one hand, EdgeNAT captures global contextual information and detailed local cues with DiNAT, on the other hand, it enhances feature representation with a novel SCAF-MLA decoder by utilizing both inter-spatial and inter-channel relationships of feature maps. Extensive experiments on multiple datasets show that our method achieves state-of-the-art performance on both RGB and depth images. Notably, on the widely used BSDS500 dataset, our L model achieves impressive performances, with ODS F-measure and OIS F-measure of 86.0%, 87.6% for multi-scale input,and 84.9%, and 86.3% for single-scale input, surpassing the current state-of-the-art EDTER by 1.2%, 1.1%, 1.7%, and 1.6%, respectively. Moreover, as for throughput, our approach runs at 20.87 FPS on RTX 4090 GPU with single-scale input. Code: https://github.com/jhjie/EdgeNAT.

1 Introduction

Edge detection is fundamental for various computer vision tasks[44, 19, 36]. The primary objective of edge detection is to precisely extract object boundaries and visually salient edges from input images. As illustrated in Figure 1, inherent challenges of this task include the presence of distant objects, blurring boundary in complex backgrounds, intense color variations within an objects, etc. Therefore, it requires appropriate representation of not only local features like color and texture, but also global semantic information to suppress noise as well as to distinguish object boundary from complex background.

Traditional edge extraction methods[21, 6] mostly rely on local information, such as variation of color and texture. CNN-based deep learning based edge detectors can learn global and semantic features[4, 5] with expansion of receptive field, but are likely to lose detail information. To preserve both intricate local information as well as global context, former deep learning detectors [41, 25] employ multi-level aggregation to effectively integrate global features and local details. To mitigate the limitations in the absence of a hierarchical structure in ViT[11], the first transformer based edge detector EDTER[31] implements a two-stage approach to obtain and combine global features and local details, which demonstrates superior edge detection ability than CNN-based detectors. However, the computational burden known for vision transformer is exacerbated by EDTER's two-stage design.

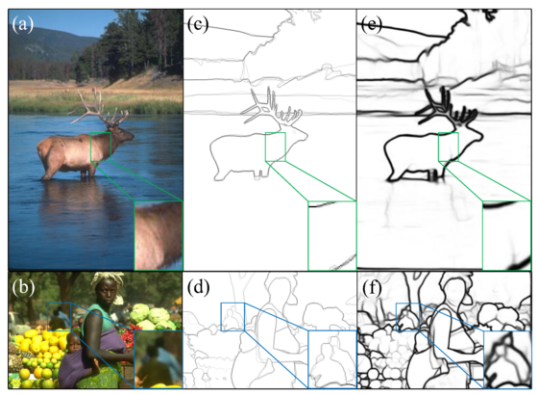

Figure 1. Illustration of the predictions of our EdgeNAT. (a, b): Input images from BSDS500. (c, d): the corresponding groundtruth. (e, f): Detected edges by EdgeNAT. (e) shows EdgeNAT doesn't extracts an edge in the neck area of the deer on the presence intense color variation.(f) shows EdgeNAT accurately extracts edges of distant and blurred objects.

Recently, DiNAT[15], an improved hierarchical transformer combining both neighbor attention and dilated neighbor attention, has exemplified significant progress in various vision tasks. Since DiNAT is able to preserve locality, maintain translation equivariance, expand the receptive field exponentially, and capture longer-range inter-dependencies, edge detector based on it (Figure 2) could abandon the two-stage design, significantly improving throughput. Furthermore, to take better usage of rich channels information in the feature map generated by transformer-based encoder, we introduce a novel decoder, Spatial and Channel Attention Fusion-Multi-Level Aggregation (SCAF-MLA). The Spatial and Channel Attention Fusion Module (SCAFM) of the decoder integrates both spatial and channel attention concurrently. As a whole, our detector is capable of extracting local detail information at lower levels, which is beneficial to the detection of edges associated with distant blurred objects, and extracting global semantic feature information at higher levels, which is beneficial to mitigating excessive noise within the object and to distinguishing inconspicuous edges.

With elaborate design, our models exhibit excellent capability of generating accurate and crisp edge maps.To verify the scalability of our edge detector, we further propose five versions of models with varying sizes, following DiNAT's configuration. Our contributions in this paper can be summarized as follows: (1) We introduce Ed-

* Corresponding Author. Email: guoyan@ustc.edu.cn.
** Corresponding Author. Email: bjhua@ustc.edu.cn.

geNAT, a one-stage Transformer-based edge detector, which enables local and global features extraction, leading to speedy and precise edge detection. (2) We propose an innovative feature fusion module, SCAFM, to enhance the feature representation generated by the encoder. We further design the SCAF-MLA decoder based on it. (3) Extensive experiments conducted on widely recognized edge detection benchmarks, such as BSDS500 and NYUDv2, demonstrate the superior performance and high efficiency of our model when compared to state-of-the-art methods. Adaptability and flexibility of our architecture are also verified on five variants of our model.

2 Related Work

Edge Detection. Early edge detectors[21, 6] mainly rely on local features, like significant variation of color, texture and intensity to detect edges. Machine learning-based methods[23, 29] employ handcrafted low-level features to train classifiers and achieve impressive performance compared to earlier approaches. Such methods are always ignorant of global information and semantic boundaries. CNN-based deep learning techniques are able to expand receptive fields to capture global features, thus yield remarkable progress in edge detection. DeepEdge[4] employs a multi-scale CNN to classify edge candidate points extracted by Canny edge detector. Recent methods have further enhanced edge detection by exploiting hierarchical and multi-scale feature maps CNN encoders produce. [25, 41] learn rich hierarchical features by supervising the layers at each level, leading to improved detection performance. BDCN[17], on the other hand, achieves greater accuracy through a bidirectional feature processing structure. PiDiNet[34] introduces pixel-differential convolutional integration into the CNN model. EDTER[31] is the first attempt to introduce Vision Transformer (ViT)[11] for edge detection tasks. To capture multi-scale features, EDTER proposes a two-stage architecture to remedy the lack of hierarchical structure in ViT. The first stage focuses on global feature while the second stage focuses on local features. Features learned in both stages are fused, resulting in significant improvements in performance and achieving SOTA in edge detection task. PEdger[13] enhances edge detection performance by leveraging information obtained from different training moments and heterogeneous structures. UAED[46] investigates the subjectivity and ambiguity of different annotations through uncertainty based on the fact that dataset labels have multiple annotations.

Vision Transformer. Since the introduction of ViT[11], transformers have been widely used in vision field[3, 27, 35]. After years of development, Transformer with multi-scale hierarchical structure are playing increasingly important role in downstream vision tasks. Swin Transformer[26] proposes Window Self Attention (WSA) and Shift Window Self Attention (SWSA), with SWSA expanding the receptive field, enabling it to capture both local and global features. NAT[16] proposes Neighborhood Attention (NA), the first efficient and scalable sliding window attention mechanism, which restricts self-attention to localised windows and preserves translation equivariance. DiNAT[15] extends NA to Dilated Neighborhood Attention (DiNA), which expands receptive fields exponentially and thus captures long-range inter dependency and global features. Besides, Neighborhood Attention Extension (NATTEN)[16] is developed to better implement NA and DiNA as an extension to PyTorch with an efficient CUDA kernel.

Feature Fusion Module. The feature fusion module is commonly used in edge detection and other vision tasks to strengthen feature representations, which is crucial to improving the accuracy. SENet[18] investigates channel relationships and introduces a novel architectural unit, the Squeeze-and-Excitation (SE) block, enhances global feature extraction by computing channel attention using global average pooling. CBAM[39] employs both global average pooling and global maximum pooling to compute attention maps on two separate dimensions, namely, channel attention and spatial attention, the latter being overlooked by SENet. CBAM is able to extract informative features by blending cross-channel and spatial information together. ECA[37] proposes a local cross channel interaction strategy implemented via 1D convolution and a method to adaptively select kernel size of 1D convolution. PP-LiteSeg[30] introduces UAFM, a feature fusion module that leverages channel attention or spatial attention to enrich the representation of fused features, with spatial and channel attention modules exploiting inter-spatial and inter-channel relationships of the input features.

3 EdgeNAT

Figure 2 illustrates the overall framework of EdgeNAT, a one-stage end-to-end edge detector. DiNAT is employed as the encoder since it exhibits exceptional performance in preserving locality, maintaining translation equivariance, expanding receptive field, and capturing long-range dependencies,etc. SCAF-MLA, a novel decoder with SCAFM to exploit both spatial and channel features from feature maps, is introduced to effectively facilitate feature fusion. We further improve the performance of SCAF-MLA by pre-fusion, that is, for the fusing operation, the feature channels of each layer are reduced to the number of channels in first level of the encoder, denoted as C in Figure 2, rather than to 1.

3.1 Review Dilated Neighborhood Attention Transformer

Below is a brief introduction on DiNAT, encoder of our network, following the work presented in [15].

To begin with, DiNAT employs two 3 × 3 convolutional layers with a stride of 2 as a tokenizer to obtain a feature map with a resolution of one-fourth of the input image. Additionally, DiNAT utilizes a single 3 × 3 convolutional layer with a stride of 2 for downsampling between hierarchical levels, reducing the spatial resolution by half while doubling the number of channels.The resulting feature maps are thus of sizes $\frac{h}{4} \times \frac{w}{4} \times c$, $\frac{h}{8} \times \frac{w}{8} \times 2c$, $\frac{h}{16} \times \frac{w}{16} \times 4c$ and $\frac{h}{32} \times \frac{w}{32} \times 8c$.

DiNAT adopts a straightforward stacking of DiNA layers, following a similar structural pattern as other commonly used Transformers. For simplicity, we keep notations limited to single dimensional NA and DiNA . Given input $X \in \mathbb{R}^{n \times d}$, whose rows are d-dimensional token vectors, and query and key linear projections of X, Q and K, and relative positional biases between any two tokens i and j, $B(i, j)$, δ-dilated neighborhood attention weights for the i-th token with neighborhood size k, $\mathbf{A}_i^{(k,\delta)}$, is defined as the matrix multiplication of the i-th token's query projection, and its k nearest neighboring tokens' key projections with dilation value δ:

$$\mathbf{A}_i^{(k,\delta)} = \begin{bmatrix} Q_i K^T_{\rho_1^\delta(i)} + B_{\left(i,\rho_1^\delta(i)\right)} \\ Q_i K^T_{\rho_2^\delta(i)} + B_{\left(i,\rho_2^\delta(i)\right)} \\ \vdots \\ Q_i K^T_{\rho_k^\delta(i)} + B_{\left(i,\rho_k^\delta(i)\right)} \end{bmatrix}, \tag{1}$$

where $\rho_j^\delta(i)$ denotes i's j-th nearest neighbor with dilation value δ. The δ-dilated neighboring values for the i-th token is similarly de-

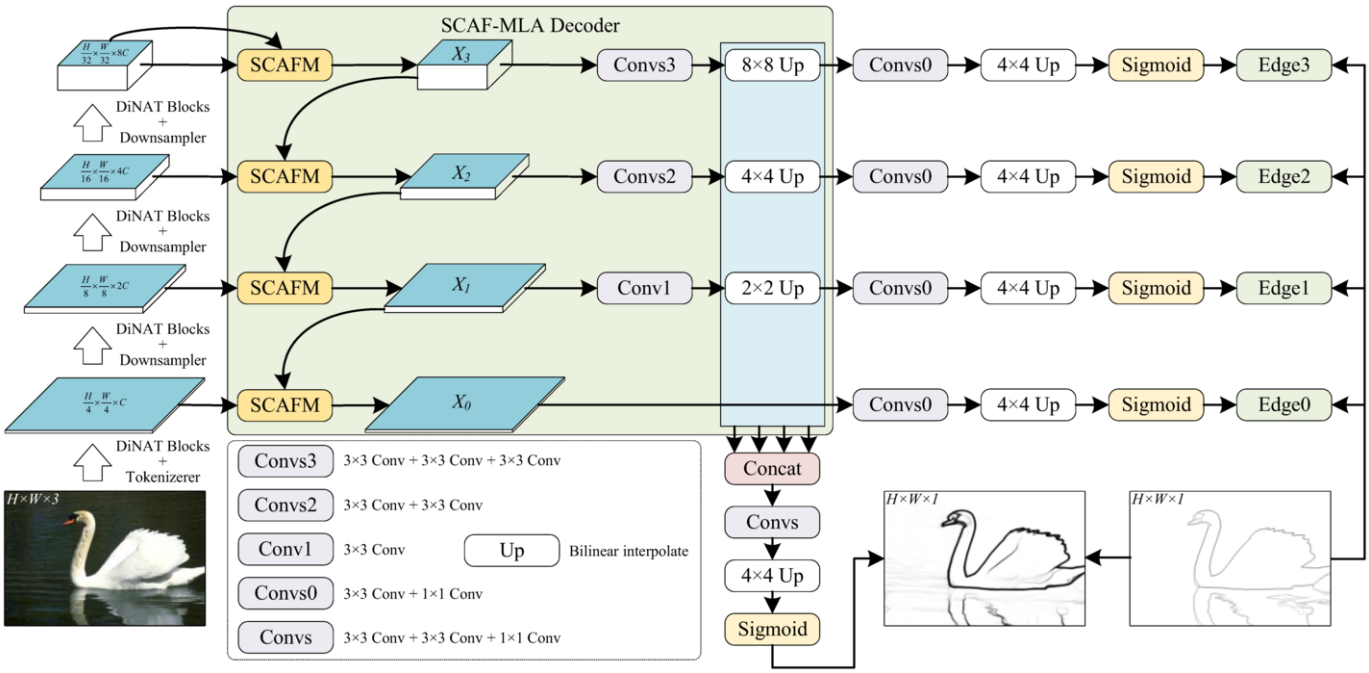

Figure 2. The overall framework of our proposed EdgeNAT. SCAFM is illustrated in Figure 3.

fined with neighborhood size k, $\mathbf{V}_i^{(k,\delta)}$:

$$\mathbf{V}_i^{(k,\delta)} = \left[\begin{array}{cccc} V^T_{\rho_1^\delta(i)} & V^T_{\rho_2^\delta(i)} & \cdots & V^T_{\rho_k^\delta(i)} \end{array} \right]^T, \tag{2}$$

where V is a linear projection of X. DiNA output for the i-th token neighborhood size k with dilation value δ is then defined as:

$$\mathrm{DiNA}_k^\delta(i) = \mathrm{softmax}\left(\frac{\mathbf{A}_i^{(k,\delta)}}{\sqrt{d_k}} \right) \mathbf{V}_i^{(k,\delta)}, \tag{3}$$

where $\sqrt{d}$ is the scaling parameter, and d is the embedding dimension. This operation is repeated for every pixel in the feature map.

Summary of DiNAT configuartions and dilation values will be provided in the supplementary material.

3.2 SCAF-MLA Decoder

Decoders play a critical role in various vision tasks. Taking inspiration from multilevel feature fusion techniques employed in vision tasks, we propose a novel decoder, SCAF-MLA, to effectively utilize numerous channels in the feature maps output from transformer-based encoder. SCAF-MLA enables the supervision on multiple levels, and learns rich hierarchical features, thus enhances the performance of edge detection. Besides, SCAF-MLA Decoder is more computationally efficient, without the commonly employed PPM[45] and bottom-up path[17, 31], while experimental results demonstrate that our designed decoder achieves more superior performance.

SCAFM.

Inspired by UAFM[30] in multi-level features fusing, we propose the Spatial and Channel Attention Fusion Module (SCAFM) as the main component of the SCAF-MLA. SCAFM is designed to extract both spatial and channel features, concurrently preserving the distinctive attributes of the current level while capturing higher-level features. The architecture of SCAFM is depicted in Figure 3. SCAFM consists of a spatial attention module (SAM) and a channel attention module (CAM) to compute inter-spatial and inter-channel weights, denoted as α_{Sp} and α_{Ch}, respectively. Specifically, the upper-level feature is denoted as F_{high} and the current-level feature as F_{low}. To begin with, bilinear interpolation is employed to upsample F_{high} to the same size as F_{low}. Subsequently, convolutional operations is utilized to increase the channels of F_{low} to match those of F_{high}, denoted as F_{conv}. For α_{Sp}, we start by performing mean and max operations along the channel dimension on F_{up} and F_{conv}, resulting in the generation of four features, each with a dimension of $\mathbb{R}^{1\times H\times W}$. Subsequently, these four features are concatenated and processed through convolutional and sigmoid operations, yielding $\alpha_{\mathrm{Sp}} \in \mathbb{R}^{1\times H\times W}$. This process can be represented by thefollowing equations:

$$\begin{aligned} F_{up} &= Up(F_{high}), \\ F_{conv} &= Conv(F_{low}), \\ \alpha_{\mathrm{Sp}} &= Sigmoid(Conv(Cat(Mean(F_{up}), \\ &\quad Max(F_{up}), Mean(F_{conv}), \\ &\quad Max(F_{conv})))), \end{aligned} \tag{4}$$

Regarding α_{Ch}, average pooling and max pooling operations are applied on F_{up} and F_{conv}, generating four features with dimensions $\mathbb{R}^{C\times 1\times 1}$. Then, these features are concatenated and subjected to convolutional and sigmoid operations, generating $\alpha_{\mathrm{Ch}} \in \mathbb{R}^{C\times 1\times 1}$, described as:

$$\begin{aligned} \alpha_{\mathrm{Ch}} &= Sigmoid(Conv(Cat(AvgPool(F_{up}), \\ &\quad MaxPool(F_{up}), AvgPool(F_{conv}), \\ &\quad MaxPool(F_{conv})))), \end{aligned} \tag{5}$$

The input features are then fused with the generated weights α_{Sp} and α_{Ch} through multiplication and addition operations, resulting

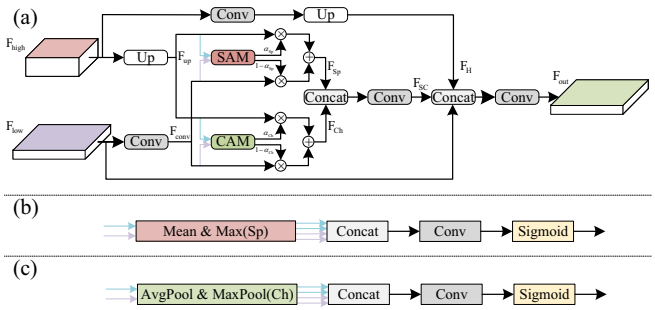

Figure 3. (a) The detailed architecture of the SCAFM. (b) The Spatial Attention Module(SAM). (c) The Channel Attention Module(CAM).

in features F_{Sp} and F_{Ch}. Subsequently, these features are concatenated and convolved, generating $F_{SC} \in \mathbb{R}^{C\times H\times W}$. Then, we perform convolutions on F_{high} followed by upsampling, generating features F_H with dimensions $\mathbb{R}^{C\times H\times W}$. Afterwards, F_H, F_{SC}, and F_{low} are concatenated and convolved to obtain the fused feature $F_{out} \in \mathbb{R}^{C\times H\times W}$. The aforementioned process can be described as:

$$\begin{aligned} F_{up} =& Up(Conv(F_{high})),\\ F_{Sp} =& F_{up}\cdot\alpha_{\text{Sp}} + F_{conv}\cdot(1-\alpha_{\text{Sp}}),\\ F_{Ch} =& F_{up}\cdot\alpha_{\text{Ch}} + F_{conv}\cdot(1-\alpha_{\text{Ch}}),\\ F_{SC} =& Conv(Cat(F_{Sp}, F_{Ch})),\\ F_{out} =& Conv(Cat(F_{SC}, F_H, F_{low})), \end{aligned} \tag{6}$$

Pre-fusion. Most previous detectors[25, 34] fuse feature maps from different layers only after reducing their channels to 1, resulting in insufficient feature integration. Inspried by EDTER[31], which fuses the feature maps with a larger number of channels, we apply one, two, and three 3×3 convolutions to the feature maps X_1, X_2, and X_3 outputted by SCAFM respectively, reducing their channels to match that of X_0, rather than reducing to 1. We then use bilinear interpolation to upsample X_1, X_2, and X_3 to match X_0. Subsequently, we perform a concatenation operation on these four feature maps, and further reduce the channels to 1 using two 3×3 convolutions and one 1×1 convolution. Finally, we upsample the 1-channel feature map using bilinear interpolation and compute the sigmoid function to obtain the edge map $E \in \mathbb{R}^{1\times H\times W}$.

3.3 Loss Function

We employ the loss function proposed in [41] for the 4 side edge maps and 1 primary edge map. Given an edge map E and its corresponding ground truth Y, the loss function is computed as follows:

$$\begin{aligned} \ell(E,Y) = -\sum_{i,j}&(Y_{i,j}\alpha\log(E_{i,j})\\ &+(1-Y_{i,j})(1-\alpha)\log(1-E_{i,j})), \end{aligned} \tag{7}$$

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 $|Y^+|$ and $|Y^-|$ denoting the number of positive and negative sample pixels, respectively. Since BSDS500 dataset is annotated by multiple annotators, we first normalize the multiple annotations 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 labeled as a positive sample; otherwise, it is labeled as a negative sample.

After the concatenation operation of the four feature maps output from our decoder, two 3×3 convolutions and one 1×1 convolution are applied to reduce the dimension of the concatenated feature maps. Similarly, a 3×3 convolution and a 1×1 convolution are applied to reduce the dimension of the four side feature maps. Subsequently, a sigmoid operation is performed on each of these reduced-dimensional feature maps to generate primary edge map and four side edge maps, denoted as $\mathcal{E}$, $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, and $\mathcal{S}_4$. We calculate the loss for both primary edge map and side edge maps to introduce additional supervision. To sum up, the overall loss function is as follows:

$$\mathcal{L} = \mathcal{L}^{\mathcal{E}} + \lambda\mathcal{L}^{\mathcal{S}} = \ell(\mathcal{E},Y) + \lambda\sum_{k=1}^{4}\ell\left(\mathcal{S}^k, Y\right), \tag{8}$$

$\mathcal{L}^{\mathcal{E}}$ 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 [31] and our experimental observations, we set λ to 0.4.

4 Experiments

4.1 Datasets

Two mainstream datasets are used to evaluate our proposed EdgeNAT, namely, BSDS500 and NYUDv2.

BSDS500[2] consists of 500 RGB images, with 200 for training, 100 for validation, and 200 for testing. Similar to [41, 25], the dataset is augmented to 28,800 images by flipping, scaling, and rotating. PASCAL VOC Context dataset[12] is used as additional training data and its 10,103 training images are also augmented to 20,206 by flipping, as in most previous works[25, 17]. The model is pre-trained with the augmented PACSAL VOC Context dataset and then fine-tuned with the 300 training and validation images of BSDS500 dataset, and is evaluated on 200 testing images.

NYUDv2[33] consists of 1449 labeled pairs of aligned RGB and depth images, with 381 training images, 414 validation images, and 654 testing images. As in [41, 25],the training and validation sets are combined and augmented to train the model.

4.2 Implementation Details

Our EdgeNAT is implemented with PyTorch and is based on mmsegmentation[7] and NATTEN[16]. We use the pre-trained weights of DiNAT[15] to initialize EdgeNAT's transformer blocks. To generate binary edge maps, for BSDS500, we set the threshold η to 0.3 to select positive samples. For NYUDv2, there is only one annotation per picture, so there is no need to set the threshold η.

We use the AdamW optimizer and train for 40k iterations using a cosine decay learning rate scheduler, where the first 15k iterations warm up the learning rate in a linear manner, and the remaining ones are decayed according to the scheduler. The initial learning rate is 0 and a preset learning rate is set to 6e-5. For BSDS500, we set its batch size to 8, and for NYUDv2, we set its batch size to 4.

All experiments were conducted on RTX 4090 GPU. The training of the L model of EdgeNAT (472.38MB) takes 6 hours, far more efficient than Transformer-based model EDTER (468.84MB)[31], which takes 26.4 hours. The inference runs at 20.87 FPS on RTX 4090, nearly ten times the speed of EDTER on V100 (2.2 FPS). During training, since our model is a one-stage edge detection model, for

320×320 images, the GPU memory requirement is about 20GB, 2/3 of EDTER(29GB).

Optimal Dataset Scale (ODS) and Optimal Image Scale (OIS) are two metrics for all datasets. Before evaluation, we perform non-maximum suppression on the predicted edge maps. For the maximum allowed tolerance distance between the detected edge and ground truth, we set it to 0.0075 for BSDS500 and to 0.011 for NYUDv2 as in previous works.

4.3 Ablation Study

Ablation experiments are performed on the BSDS500 data set to verify the effectiveness of our proposed decoder. Specifically, we first compare the effect of pre-fusion (reduce the channels of feature map to C) and final-fusion (reduce the channels of feature map to 1); then the effect of bottom-up path is also verified. From the quantitative results shown in Table 1, it is clear that regardless of pre-fusion or final-fusion, Bottom-up Path has negative effects on edge detection performance, indicating it is not suitable for DiNAT. For edge detection models with relatively large number of feature map channels, pre-fusion without PPM will be a better choice.

ODS / OIS	Final-fusion	Pre-fusion
Bottom-up Path	0.838 / 0.852	0.839 / 0.856
-	0.838 / 0.853	0.840 / 0.856

Table 1. Ablation study of the effectiveness of the pre-fusion and bottom-up path on BSDS500. All results are computed with a single-scale input without additional training data.

PPM	SCAFM	ODS	OIS
✗	✗	0.837	0.853
✓	✗	0.836	0.851
✗	✓	0.843	0.859
✓	✓	0.841	0.856

Table 2. Ablation study on the effectiveness of PPM and SCAFM. All results are computed with a single-scale input without additional training data.

Next, the effectiveness of PPM[45] and our proposed SCAFM are verified and compared. The quantitative results shown in Table 2 demonstrates that SCAFM works best without PPM, achieving best ODS and OIS score, 84.3% and 85.9% respectively. In summary, we will use the SCAF-MLA decoder without Bottom-up Path and PPM for the next experiments.

4.4 Network Scalability

EdgeNAT-L has a relatively large amount of parameters (472.38MB). In order to adapt to different application scenarios, we conduct scalability experiments on different model sizes. The configuration settings of the encoder of the L, S0, S1, S2, and S3 variants of our EdgeNAT are the same as those of the Large, Mini, Tiny, Small, and Base versions of the DiNAT[15]. Extensive experiments are conducted to study the scalability and throughput of EdgeNAT variants. The result is shown in Figure 4. The models are all trained using the BSDS500 training and validation sets with or without PASCAL VOC, and evaluated with the BSDS500 test set. As expected, when

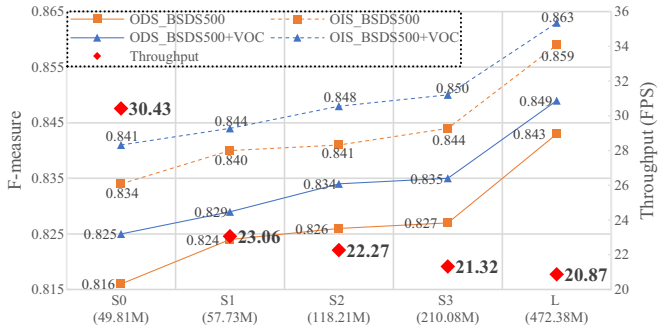

Figure 4. Exploration on the scalability of EdgeNAT. Bottom row shows the number of parameters for each model. The models are trained with or without PASCAL VOC dataset. All results are computed with a single-scale input.

the size of our model decreases, the ODS and OIS will decrease accordingly, and the throughput increases.

It is worth noting that the processing speed of the S0 model is much higher than that of other models. This should be contributed to the fact that its third level has only 6 layers, while the others models have 18 layers. The ODS and OIS of the L model are much higher than other models, due to the fact that the encoder is pre-trained on ImageNet-22K, while encoder of other models are pre-trained on ImageNet-1K. The results of multi-scale input experiment of S0, S1, S2, and S3 models as well as their visualization results will be provided in the supplementary material.

4.5 Comparison with State-of-the-arts

On BSDS500 dataset. We compare our L model with *traditional detectors* such as Canny[6], gPb-UCM[1], SCG[40], SE[10] and OEF[14], and *CNN-based detector* such as DeepEdge[4], DeepContour[32], HED[41], Deep Boundary[22], CEDN[43], RDS[24], AMH-Net[42], RCF[25], CED[38], LPCB[9], BDCN[17], DSCD[8], PiDiNet[34], UAED[46] and PEdger[13], and *transformer-based detector* such as EDTER[31]. The results are summarized in Table 3 and Figure 6, respectively. We notice that our L model, trained on the BSDS500 dataset, achieves an ODS of 84.3% with single-scale inputs, outperforming all competing detectors. Furthermore, when employing multi-scale inputs, our method achieves an even higher ODS of 85.5%. By utilizing additional training data and adopting multi-scale input (following the configurations of RCF, EDTER, *etc.*), our method attains 86.0%(ODS), 87.6%(OIS), which clearly demonstrate the superiority of our method over all existing state-of-the-art edge detectors. Several qualitative results are presented in Figure 5. It can be observed that our proposed EdgeNAT demonstrates a distinct advantage in terms of prediction quality. The generated outputs exhibit clear and exact edge predictions, further validating the efficacy of our method.

On NYUDv2 dataset. We conduct experiments on three types of inputs (RGB, HHA, and RGB-HHA). The RGB-HHA results are obtained by averaging the edge detections from RGB and HHA. We compare our L model with deep learning-based detectors, including HED[41], COB[28], RCF[25], AMH-Net[42], LPCB[9], BDCN[17], PiDiNet[34], PEdger[13], and EDTER[31]. All results are based on single-scale inputs. The results are shown in Table 4. It can be observed that our L model achieveds ODS of 78.9%, 72.6%, and 79.4% for RGB, HHA, and RGB-HHA, respectively, surpassing the second-best method by 1.5%, 0.9% and 1.0%, respectively.

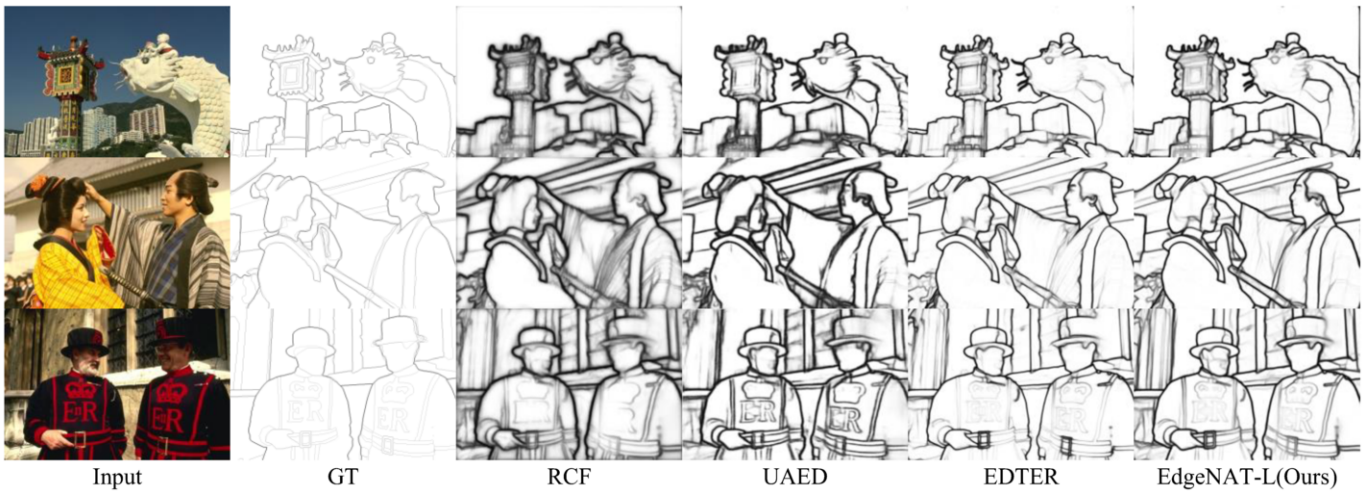

Figure 5. Qualitative comparisons on three challenging samples in the testing set of BSDS500. It is interesting to notice that in the third example, the edge of the hat on the right is completed by our L model, though the hat edge is hard to distinguish even for human eyes. This unprecedented phenomenon demonstrates our model has better global semantic understanding than previous works.

	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
	EdgeNAT-L	Ours	0.843	0.859
	EdgeNAT-L†		0.849	0.863
	EdgeNAT-L‡		**0.855**	**0.870**
	EdgeNAT-L†‡		**0.860**	**0.876**

Table 3. Results on BSDS500 testing set. The best two results are highlighted in **red** and **blue**, respectively, and same for other tables. † means training with extra PASCAL VOC data, and ‡ is the multi-scale testing.

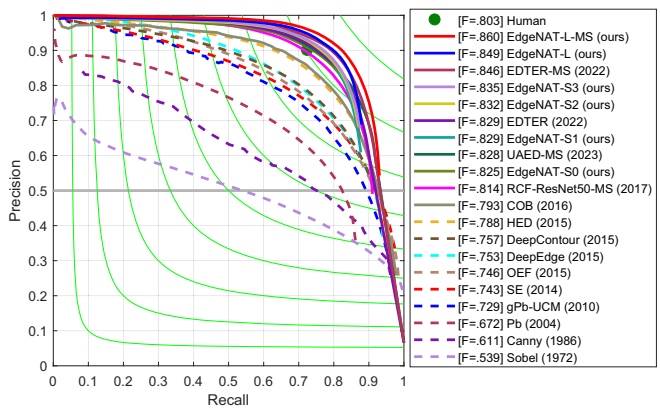

Figure 6. The precision-recall curves on BSDS500.

Furthermore, our approach also attains the highest OIS among all the evaluated methods. The results of our other models, and the precision-recall curves, will be presented in the supplementary material.

Method	RGB		HHA		RGB-HHA	
	ODS	OIS	ODS	OIS	ODS	OIS
HED	0.720	0.734	0.682	0.695	0.746	0.761
COB	-	-	-	-	**0.784**	**0.805**
RCF	0.729	0.742	0.705	0.715	0.757	0.771
AMH-Net	0.744	0.758	**0.717**	**0.729**	0.771	0.786
LPCB	0.739	0.754	0.707	0.719	0.762	0.778
BDCN	0.748	0.763	0.707	0.719	0.765	0.781
PiDiNet	0.733	0.747	0.715	0.728	0.756	0.773
PEdger	0.742	0.757	-	-	-	-
EDTER	**0.774**	**0.789**	0.703	0.718	0.780	0.797
EdgeNAT-L	**0.789**	**0.803**	**0.726**	**0.741**	**0.794**	**0.808**

Table 4. Quantitative comparisons on NYUDv2. All results are computed with a single scale input.

5 Conclusion

Our contributions are summarized as follows: firstly, we introduce DiNAT as the encoder, which enables our proposed edge detector not only more accurate than current SOTA EDTER, but also ten times faster than it. Secondly, we propose SCAFM, a module that concatenates spatial attention and channel attention, to generate richer and more accurate feature representation for the decoder. Thirdly, we design five version of models with different parameter sizes to adapt to complex and diverse application scenarios and conduct extensive experiments on the BSDS500 and NYUDv2 datasets, demonstrating that EdgeNAT achieves superiority in both efficiency and accuracy. Our supplementary material is available [20].

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.

[2] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik. Contour Detection and Hierarchical Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, May 2011. doi: 10.1109/TPAMI.2010.161.

[3] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

[4] G. Bertasius, J. Shi, and L. 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*, pages 4380–4389, 2015.

[5] 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.

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

[7] M. Contributors. Mmsegmentation: Openmmlab semantic segmentation toolbox and benchmark, 2020.

[8] R. Deng and S. Liu. Deep structural contour detection. In *Proceedings of the 28th ACM international conference on multimedia*, pages 304–312, 2020.

[9] 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.

[10] P. Dollár and C. L. Zitnick. Fast edge detection using structured forests. *IEEE transactions on pattern analysis and machine intelligence*, 37(8): 1558–1570, 2014.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[13] Y. Fu and X. Guo. Practical edge detection via robust collaborative learning. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 2526–2534, 2023.

[14] 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.

[15] A. Hassani and H. Shi. Dilated neighborhood attention transformer. *arXiv preprint arXiv:2209.15001*, 2022.

[16] A. Hassani, S. Walton, J. Li, S. Li, and H. Shi. Neighborhood attention transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6185–6194, 2023.

[17] 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.

[18] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[19] Z. Hu, M. Zhen, X. Bai, H. Fu, and C.-l. 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*, pages 222–239. Springer, 2020.

[20] J. Jie, Y. Guo, G. Wu, J. Wu, and B. Hua. Edgenat: Transformer for efficient edge detection, 2024. URL https://arxiv.org/abs/2408.10527.

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

[22] I. Kokkinos. Pushing the boundaries of boundary detection using deep learning. In *4th International Conference on Learning Representations, ICLR 2016*, 2016.

[23] J. J. Lim, C. L. Zitnick, and P. Dollár. Sketch tokens: A learned mid-level representation for contour and object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3158–3165, 2013.

[24] Y. Liu and M. S. Lew. Learning relaxed deep supervision for better edge detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 231–240, 2016.

[25] 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.

[26] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[27] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022.

[28] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool. Convolutional oriented boundaries. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 580–596. Springer, 2016.

[29] D. Martin, 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, May 2004. ISSN 1939-3539. doi: 10.1109/TPAMI.2004.1273918. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[30] J. Peng, Y. Liu, S. Tang, Y. Hao, L. Chu, G. Chen, Z. Wu, Z. Chen, Z. Yu, Y. Du, et al. Pp-liteseg: A superior real-time semantic segmentation model. *arXiv preprint arXiv:2204.02681*, 2022.

[31] M. Pu, Y. Huang, Y. Liu, Q. Guan, and H. Ling. Edter: Edge detection with transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1402–1412, 2022.

[32] 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.

[33] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *Computer Vision – ECCV 2012*, pages 746–760. Springer, 2012.

[34] Z. Su, W. Liu, Z. Yu, D. Hu, Q. Liao, Q. Tian, M. Pietikäinen, and L. Liu. Pixel difference networks for efficient edge detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5117–5127, 2021.

[35] S. Walton, A. Hassani, X. Xu, Z. Wang, and H. Shi. Stylenat: Giving each head a new perspective. *arXiv preprint arXiv:2211.05770*, 2022.

[36] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 1328–1338, 2019.

[37] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[38] 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.

[39] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[40] R. Xiaofeng and L. Bo. Discriminatively trained sparse code gradients for contour detection. *Advances in neural information processing systems*, 25, 2012.

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

[42] 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 pro-*

cessing systems, 30, 2017.

[43] 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.

[44] Z. Yu, R. Huang, W. Byeon, S. Liu, G. Liu, T. Breuel, A. Anandkumar, and J. Kautz. Coupled segmentation and edge learning via dynamic graph propagation. *Advances in Neural Information Processing Systems*, 34:4919–4932, 2021.

[45] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[46] C. Zhou, Y. Huang, M. Pu, Q. Guan, L. Huang, and H. Ling. The treasure beneath multiple annotations: An uncertainty-aware edge detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15507–15517, 2023.