Three Birds One Stone: A Unified Framework for Salient Object Segmentation, Edge Detection and Skeleton Extraction

Qibin Hou[1] Jiangjiang Liu[1] Ming-Ming Cheng[1] Ali Borji[2] Philip H.S. Torr[3]

[1]Nankai University [2]University of Central Florida [3]University of Oxford
andrewhoux@gmail.com, cmm@nankai.edu.cn

Abstract. In this paper, we aim at solving pixel-wise binary problems, including salient object segmentation, skeleton extraction, and edge detection, by introducing a unified architecture. Previous works have proposed tailored methods for solving each of the three tasks independently. Here, we show that these tasks share some similarities that can be exploited for developing a unified framework. In particular, we introduce a horizontal cascade, each component of which is densely connected to the outputs of previous component. Stringing these components together allows us to effectively exploit features across different levels hierarchically to effectively address the multiple pixel-wise binary regression tasks. To assess the performance of our proposed network on these tasks, we carry out exhaustive evaluations on multiple representative datasets. Although these tasks are inherently very different, we show that our unified approach performs very well on all of them and works far better than current single-purpose state-of-the-art methods. All the code in this paper will be publicly available.

1 Introduction

Convolutional neural networks (CNNs) have been widely used in most fundamental computer vision tasks (e.g., semantic segmentation [1, 2], edge detection [3], salient object segmentation [4,5], skeleton extraction [6,7].) and have achieved unprecedented performance on many tasks. To date, most of the existing methods are designed only for a single task because different tasks often favor different types of features. Their design criterion is single-purpose, greatly restricting their applicability to other tasks [8].

In this paper, our goal is to present a unified framework for solving three important binary problems, including salient object segmentation, edge detection, and skeleton extraction. In Fig. 1, we illustrate a 2D space representing features that these tasks favor. Specifically, salient object segmentation, as addressed in many existing works [4, 5, 9], requires the ability to extract homogeneous regions and hence relies more on high-level features (Figs. 2c and 2f). Edge detection aims at detecting accurate boundaries, thus it needs more low-level features to sharpen the coarse edge maps produced by deeper layers [3, 10] (Figs. 2d and 2e). Skeleton extraction [6, 7], on the other hand, prefers high-level semantic information to detect scale-variant (either thick or thin) skeletons. From the standpoint of the network architecture, in spite of three different tasks, all of them require multi-level features in varying degrees (See Fig. 1). Consequently, a

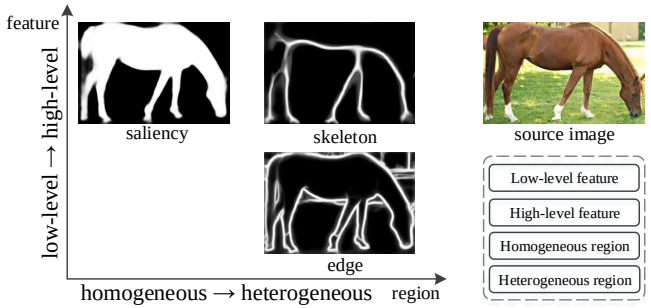

Fig. 1. Preferred features of different tasks in our work. On the right side, we show the source image and two dimensions of features favored by different tasks. The results on the left side are all by our approach.

natural question is whether it is possible to combine multi-level features in a proper way such that stronger and more general feature representations can be constructed for solving all of these tasks.

To solve the above question, rather than simply combining the multi-level features extracted from the trunk of CNNs as done in most existing works [3, 5, 11], we propose to construct a horizontal cascade (Fig. 3a), which connects a sequence of stages together. Each stage is composed of multiple transition nodes, each of which gets input from its former stage, enabling subsequent stages to efficiently select features from the backbone in a dense manner. As the signals from the backbone pass through the stages sequentially, more and more advanced feature representations can be built that can be applied to different tasks. As shown in Fig. 2, our approach is more general compared to existing relevant methods. To evaluate the performance of the proposed architecture, we apply it to three binary tasks—salient object detection, edge detection, and skeleton extraction and show its superiority compared to prior works on multiple widely used benchmarks.

2 Related Works

2.1 Earlier Pioneering Works

Salient Object Detection. Earlier salient object detection methods mostly rely on hand-crafted features, including either local contrast cues [12–14] or global contrast cues [15–18]. Interested readers may refer to some notable review and benchmark papers [19, 20] for detailed descriptions. Apart from the classic methods, a number of deep learning based methods have recently emerged. Among them, some early works [21–24] took as input image patches the bounding boxes of regions with arbitrary shapes and then used CNN features to predict the saliency of each input region (either a bounding box [25] or a superpixel [26, 27]). Later works [4, 5, 9, 28–31], benefiting from the high efficiency of fully convolutional networks [1] (FCNs), utilize the strategy in which spatial information is processed in CNNs, and hence produce remarkable results.

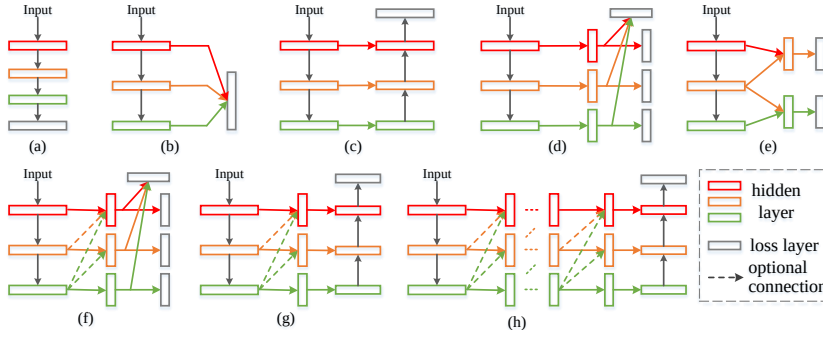

Fig. 2. Architecture comparisons. (b) DCL [4]; (c) MSRNet [9]; (d) HED [3]; (e) COB [10]; (f) SRN [7] and DSS [5]; (g) Our architecture with one stage; (h) A general case of our architecture.

Edge Detection. Early edge detection works [32–34] mostly relied on various gradient operators. Later works, such as [35–37], were driven by manually-designed features and were able to improve the performance compared to gradient-based works. Recently, with the emergence of large scale datasets, learning-based methods [38–41] gradually became the main stream for edge detection. Further, recent CNN-based methods [3, 10, 42–47] have started a new wave in edge detection research. Their capacity in learning multi-level and multi-scale features allowed them to achieve unprecedented performance in this field.

Skeleton Extraction. Recent skeleton detection methods [6, 7] are mainly based on the holistically-nested edge detector (HED). In [6], Shen *et al.* introduced supervision in different blocks by guiding the scale-associated side outputs toward ground-truth skeletons at different scales and then fused multiple scale-associated side outputs in a scale-specific manner to localize skeleton pixels at multiple scales. Ke *et al.* [7] added multiple shortcuts from deeper blocks to shallower ones based on the HED architecture such that high-level semantic information can be effectively transmitted to lower side outputs, yielding stronger features.

2.2 CNN-Based Skip-Layer Architectures

Unlike most classification tasks which adopt the classic bottom-up structures (Fig. 2a), region segmentation and edge detection tasks depend on how homogeneous regions are extracted and how edges are sharpened. Intuitively, considering the fact that lower network layers are capable of capturing local details while higher layers capture high-level contextual details, a good solution to satisfy the above needs might be introducing skip-layer architectures [1]. One of the recent successful CNN-based skip-layer structures is the HED architecture [3], which learns rich hierarchical features by means of adding side supervision to each side output (Fig. 2d). This architecture treats features at different levels equally, and therefore allows enough edge details to be captured through lower side outputs. Afterwards, several follow-up works [6, 10, 46, 47] adopted similar structures (e.g., by introducing deep supervision) to capture richer feature representations by fusing features at different levels. There are also some other work modifying

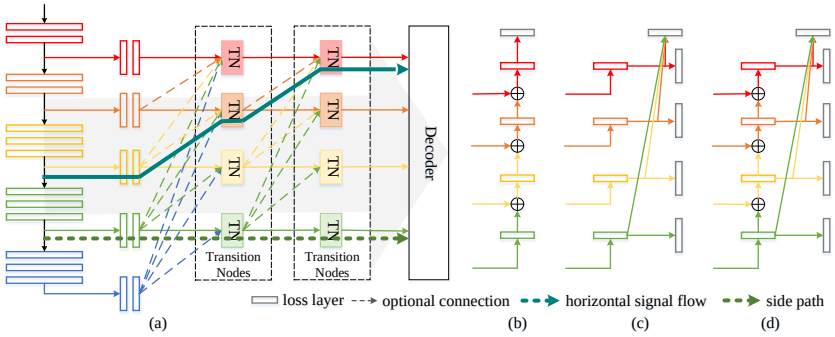

Fig. 3. (a) A typical representation of our proposed unified framework. (b-d) Three different kinds of decoders that are used to merge the extracted multi-level features. Thin solid lines are required while thin dash ones are optional. Thick lines (horizontal signal flow and side path) are used for demonstration purpose only.

this structure by adding short connections [5, 7] from upper layers to lower ones to better leverage multi-level features for salient object detection and skeleton extraction. These approaches, however, only attempt to simply combine multi-level features. There is still a large room for extracting richer feature representations for these pixel-wise binary regression problems.

3 The Proposed Unified Framework

In this section, we elaborate on the similar characteristics shared by salient object detection, skeleton extraction, and edge detection tasks and propose a unified framework (UFNet) that can treat them all.

3.1 Key Observations

Previous works leverage multi-level features by either fusing simple features in a top-down manner (Fig. 2c) or introducing shortcuts followed by side supervision (Figs. 2d and 2f). What they have in common is that each layer in the decoder (the right part of each diagram in Fig. 2) can only receive features from the backbone or its upper layers in the decoder. These types of designs may work well for salient object detection but may fail when applied to edge detection and skeleton extraction (and vice versa). The fundamental reason behind this is the fact that the feature representations formed in the decoders are not powerful enough to deal with all of these tasks. Taking into account the nature of salient object detection, edge detection, and skeleton extraction, a straightforward way to build more advanced feature representations is to add a couple of groups of transition nodes such that multi-level features from the backbone can be sequentially combined multiple times until the representations are strong enough. In the following subsections, we will show how to construct a generalized structure that can include all the features each task favors.

3.2 Overview of Our Proposed Framework

An illustrative diagram of our network is shown in Fig. 3a. Structurally, our architecture can be decoupled into multiple stages $\{S_i\}$ $(0 \leq i \leq N)$, each of which performs different functions. The first stage S_0 corresponds to the backbone of any classification network (VGGNet [48] here). This stage is mainly used to extract the first-tier multi-level and multi-scale features from the input images. The last stage S_N is a decoder and can have different forms depending on the task at hand. The responsibility of the decoder is to fuse multi-level features from its former stage such that stronger representations can be learned and then sent to the loss functions. Each middle stage is composed of multiple transition nodes T, each of which receives input from its last stage and sends responses to the next stage for rebuilding higher-level features. For notational convenience, each block[1] in S_0 is treated as a transition node as well. A sequence of stages forms the so-called *horizontal cascade*, which transmits the multi-level features from the backbone to the decoder horizontally. Our architecture, obviously, can be treated as the generalized case of previous work (See Fig. 2) and hence is quite different from them.

3.3 Transition Nodes

Side Path. To better interpret the architecture of our proposed approach, we introduce the concept of side path in this paragraph, which is similar to the notation of side output in [3, 5]. Each side path, in our case, starts from the end of a block in CNNs and ends before the decoder. A typical example can be found in Fig. 3, which is represented by the dark green dash arrow with an enhanced thickness for highlighting. Obviously, there are totally four standard side paths in Fig. 3 plus a short one which is connected to the last block in S_0. At the beginning of each side path, we can optionally add a stack of consecutive convolutional layers followed by ReLU layers on each according to different tasks. This is inspired by [5], who have shown that adding more convolutional layers improves salient object detection. In what follows, we neglect the specific number of these convolutional layers for the sake of convenience. Detailed settings of each side path can be found in Sec. 3.6.

Transition Node. Formally, for any positive integer k $(k < N)$, let T_m^k denote the mth transition node in S_k. Transition node T_m^k is able to selectively receive signals (features) from transition nodes that are not shallower than itself in its previous stage S_{k-1}. In this way, upper transition nodes can only get inputs from deeper side paths, preserving the original high-level features that are informative for generating homogeneous regions. Additionally, lower transition nodes receive features from multiple levels, allowing these features to be merged efficiently to produce even more advanced representations. Fig. 3a provides an illustration, in which transition nodes are represented by colorful solid rectangles. Notice that, in Fig. 3a, we only show a case where the transition nodes between adjacent stages are densely connected. In fact, the connection patterns can be decided according to different kinds of tasks.

[1] The definition of block here refers to the layers that share the same resolution in a baseline model (e.g., VGGNet [48]).

Internal Structure. The multi-level feature maps extracted from the backbone model usually contains different channel numbers and resolutions. To ensure that features from different levels can be fused together in our architecture, each input of a transition node is first passed through a convolutional layer with the same number of channels. Then, we use deconvolutional layers initialized with fixed parameters for bilinear interpolation such that all the inputs share the same feature map size. The hyper-parameters of deconvolutional layers can be easily inferred from the context of our network, which will be elaborated in the experiments sections. For fusion, all feature maps with the same size are merged together by simple summation. Concatenation operation can also be used here but we empirically found that performance gain is negligible in all tasks. An extra convolutional layer is added to eliminate the aliasing effect caused by the summation operation.

3.4 Horizontal Hierarchy

A number of recent works have leveraged multi-level features by introducing a series of top-down paths based on the backbone of a classification network. In our network, each transition node is able to optionally receive signals from its previous stage. Unlike the fusing strategy in [5] which only combines the score maps from different side paths, our architecture allows more signals to be transmitted to the next stage or to the decoder. In this way, each new transition node is allowed to fuse features at different levels from the backbone, allowing the output features to reach higher levels. By adding a stack of transition node stage, as shown in Fig. 3a, we are able to further advance the feature levels. Therefore, when the number of stages increases, a horizontal hierarchy is formed.

Horizontal Signal Flow. Let S_k be the kth stage of transition nodes. The set of all transition nodes forms a multi-tree, a special directed acyclic graph (DAG) whose vertices and edges are composed of all the transition nodes and the connections between each pair of transition nodes. With these definitions, a horizontal signal flow in the DAG starts from the end of an arbitrary transition node in S_0 and ends before the decoder. Furthermore, the vertices through which it passes should not be lower than its starting transition node. The dark green thick arrow in Fig. 3a depicts a representative horizontal signal flow. When applied to different tasks, the edges can be selectively discarded. In the following experiments sections, we will further elaborate on this and discuss how to better leverage the horizontal signal flows for different types of binary vision tasks.

3.5 Diverse Decoders

The form of the decoder is also very important when facing different tasks. To date, many decoders (Fig. 2) with various structures have been developed. Here, we describe two of them which we found to work very well for the three binary tasks to be solved in this paper, respectively. The first one corresponds to the structure in Fig. 3b, which has been adopted by many segment detection related tasks. This structure gradually fuses the features from different side paths in a top-down manner. Since the feature channels from different side paths may vary, to perform the summation operation, a convolutional

Table 1. Detailed connection information between transition nodes in adjacent stages. Here, $T_c^{\{1,2,3\}}$ means that current transition node gets inputs from $T_{c-1}^1, T_{c-1}^2, T_{c-1}^3$. For edge detection an skeleton extraction, all the settings are the same apart from the channels numbers in each side path.

	Saliency			Edge & Skeleton		
bottom	S_0	S_1	S_2	S_0	S_1	S_2
conv1	T_0^1	$T_1^{\{1,2,3,4,5\}}$	$T_2^{\{1,2,3\}}$	T_0^1	$T_1^{\{1,2,3\}}$	$T_2^{\{1,2,3\}}$
conv2	T_0^2	$T_1^{\{2,3,4,5,6\}}$	$T_2^{\{2,3,4\}}$	T_0^2	$T_1^{\{2,3,4\}}$	$T_2^{\{2,3,4\}}$
conv3	T_0^3	$T_1^{\{3,4,5,6\}}$	$T_2^{\{3,4,5\}}$	T_0^3	$T_1^{\{3,4,5\}}$	$T_2^{\{3,4\}}$
conv4	T_0^4	$T_1^{\{4,5,6\}}$	$T_2^{\{4,5\}}$	T_0^4	$T_1^{\{4,5\}}$	-
conv5	T_0^5	$T_1^{\{5,6\}}$	-	T_0^5	-	-
conv6	T_0^6	-	-	-	-	-

layer with kernel size 3×3 is used, if needed. In spite of only one loss layer, we found that one loss layer performs better than the structure in Fig. 3d which introduces the concept of side supervision [3, 5] (See Table 3). We will provide more details on the behaviors of them in the experiments sections.

For edge detection and skeleton extraction, we employ the same form of decoder as in [3]. Edge detection and skeleton extraction require the ability of sharpening thick lines detected and thereby rely more on low-level features compared to segmentation. Adding side supervision to the end of each side path allows more detailed edge information to be emphasized. On the other hand, the side predictions can also be merged with the weighted-fusion layer as the final output, providing better performance.

3.6 Implementation Details

We use the popular and publicly available Caffe toolbox [49] to implement our framework. As most prior works chose VGGNet [48] as their pre-trained model, for fair comparison, we base our model on this architecture too.

For salient object detection, we replace 3 fully-connected layers with 3 convolutional layers (conv6), the same structure to conv5. We change the stride of pool5 to 1 and the dilatation of conv6 to 2 for large receptive fields, and add 2 convolutional layers at the beginning of each side path following [5]. From side path 1 to 6, the strides are 1, 2, 4, 8, 16, and 16, respectively, and the corresponding channel numbers of convolutional layers are set to 32, 64, 128, 256, 512, and 512, respectively. We adopt the architecture shown in Table 1 and the decoder in Fig. 3b. For edge detection, we change the stride of pool4 layer to 1 and set the dilation rate of convolutional layers in conv5 to 2 as in [47]. The convolutional layers in each transition node are all with 16 channels. The connection patterns can be found in Table 1. We adopt the same decoder as in [3, 47] (Fig. 3c). For skeleton extraction, the network structure is the same as in Table 1 aside from the channel numbers in side paths which correspond to 32, 64, 128, and 256 from side path 1 to 4, respectively. Thus, the strides of side paths 1 to 4 are 1, 2, 4, and 8, respectively. All the convolutional layers mentioned here are with kernel size 3 and stride 1.

Table 2. Quantitative salient object segmentation results over 5 widely used datasets. The best and the second best results in each column are highlighted in red and green, respectively. As can be seen, our approach achieves the best results on all datasets in terms of F-measure.

	Training		MSRA-B [17]		ECSSD [50]		HKU-IS [21]		DUT [51]		SOD [52, 53]	
Model	#Images	Dataset	MaxF	MAE	MaxF	MAE	MaxF	MAE	MaxF	MAE	MaxF	MAE
GC [15]	-	-	0.719	0.159	0.597	0.233	0.588	0.211	0.495	0.218	0.526	0.284
DRFI [15]	2,500	MB	0.845	0.112	0.782	0.170	0.776	0.167	0.664	0.150	0.699	0.223
LEGS [24]	3,340	MB + P	0.870	0.081	0.827	0.118	0.770	0.118	0.669	0.133	0.732	0.195
MC [22]	8,000	MK	0.894	0.054	0.837	0.100	0.798	0.102	0.703	0.088	0.727	0.179
MDF [21]	2,500	MB	0.885	0.066	0.847	0.106	0.861	0.076	0.694	0.092	0.785	0.155
DCL [4]	2,500	MB	0.916	0.047	0.901	0.068	0.904	0.049	0.757	0.080	0.832	0.126
RFCN [30]	10,000	MK	-	-	0.899	0.091	0.896	0.073	0.747	0.095	0.805	0.161
DHSNET [29]	6,000	MK	-	-	0.905	0.061	0.892	0.052	-	-	0.823	0.127
ELD [28]	9,000	MK	-	-	0.865	0.098	0.844	0.071	0.719	0.091	0.760	0.154
DISC [54]	9,000	MK	0.905	0.054	0.809	0.114	0.785	0.103	0.660	0.119	-	-
MSRNet [9]	5,000	MB + H	0.930	0.042	0.913	0.054	0.916	0.039	0.785	0.069	0.847	0.112
DSS [5]	2,500	MB	0.927	0.028	0.915	0.052	0.913	0.039	0.774	0.065	0.842	0.118
UFNet	2,500	MB	0.926	0.039	0.915	0.060	0.910	0.044	0.776	0.069	0.844	0.124
UFNet	5,000	MB + H	0.930	0.039	0.925	0.055	0.934	0.034	0.790	0.068	0.853	0.117
UFNet^{CRF}	5,000	MB + H	0.934	0.031	0.930	0.046	0.939	0.027	0.801	0.058	0.860	0.114

4 Application I: Salient Object Detection

We first apply the proposed approach to salient object detection. This is an important pixel-wise binary problem and has attracted a lot of attention recently.

4.1 Evaluation Measures and Datasets

Here, we use two universally-agreed, standard, and easy-to-understand measures [20] for evaluating the existing deep saliency models. We first report the F-measure score, which simultaneously considers recall and precision, the overlapping area between the subjective ground truth annotation and the resulting prediction maps. The second measure we use is the mean absolute error (MAE) between the estimated saliency map and ground-truth annotation.

We perform evaluations on 5 datasets, including MSRA-B [17], ECSSD [50], HKU-IS [21], SOD [52, 53], and DUT-OMRON [51]. For training, we first use the 2,500 training images from MSRA-B. We also try a larger training set incorporating another 2,500 images from HKU-IS as done in [9]. Notice that all the numbers reported here are from the results the authors have presented or the results we obtained by running their publicly available code.

4.2 Ablation Studies

The hyper-parameters are as follows: weight decay (0.0005), momentum (0.9), and mini-batch size (10). The initial learning rate is set to 5e-3 and is divided by 10 after 8,000 iterations. We run the network for 12,000 iterations and choose our best model according to the performance on the validation set [17]. Further, we also use the fully connected CRF model [55] which is the same as in [5] as a post-processing tool for maintaining spatial coherence.

Table 3. Ablation experiments for analyzing different numbers of stages and decoders. We use the ECSSD dataset as the test set here. The best results are highlighted in **bold**.

#	Methods	#Stages	Decoder	F-measure	MAE
1	DHSNet [29]	0	-	0.907	0.059
2	MSRNet [9]	0	Fig. 3b	0.913	0.054
3	UFNet (Ours)	1	Fig. 3b	0.916	0.055
4	UFNet (Ours)	2	Fig. 3c	0.918	0.053
5	UFNet (Ours)	2	Fig. 3b	**0.923**	**0.051**

The Number of Stages. The number of transition node stages plays an important role in our approach. Some prior works (e.g., [9, 29]) have shown good results with the decoders directly connected to the backbone. However, when we add the first stage of transition nodes, a small improvement is achieved in terms of F-measure score. Quantitative results can be found in Table 3. When we add another stage of transition nodes, further improvements on the F-measure and MAE scores are obtained. We also added additional stages but more stages yielded no improvements. This might be because feature representations after two times fusion have already reached the top level of our architecture.

More Training Data. The amount of training data is essential for CNN-based methods. Besides training on the MSRA-B dataset as done in [4, 5, 18], we also attempt to add more training data as in [9]. In Table 2, we show the results using different training sets. With another 2,500 training images from [21], our performance can be further improved about 1% in terms of F-measure. Therefore, we believe more high-quality training data can help.

The Effect of Horizontal Signal Flows. By default, for salient object detection, we use a dense way to connect each pair of transition nodes from adjacent stages. To show the effect of horizontal signal flows, we attempt to simplify our network by reducing the inputs of each transition node (i.e., the dash arrows in Fig. 3a). When we adopt a similar structure to the edge part in Table 1, the F-measure performance drops slightly (about 0.5 points). When we reduce more horizontal signal flows, the performance decreases further. This phenomenon indicates that more top-down connections between transition nodes helps segmentation type tasks. For more experiments setting, we encourage the readers to refer to our supplementary material.

The Roles of Different Decoders. The structure of the decoder also affects the performance of our approach. We try two different structures (Figs. 3c and 3d) as our decoders. Although the structure in Fig. 3c was helpful in [5], we obtain no performance gain by such a structure but a slight decrease in F-measure (See Table 3). This phenomenon reveals that introducing side supervision as in [5] is not always a good strategy for salient object detection. Different network architectures may favor different types of decoders.

4.3 Comparison with the State-of-the-Art

We exhaustively compare our proposed approach with 12 existing state-of-the-art salient object detection methods including 2 classic methods (GC [15] and DRFI [18]) and 10

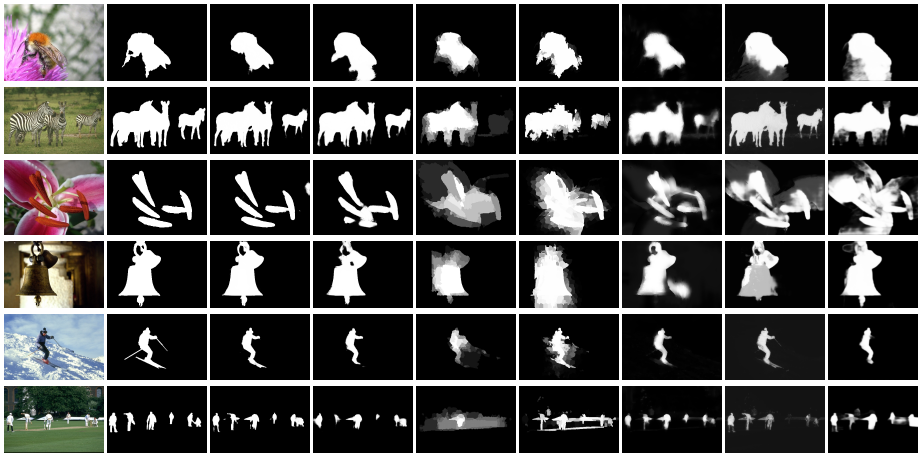

Fig. 4. Visual comparisons of different salient object detection approaches.

recent CNN-based methods (LEGS [24], MC [22], MDF [21], DCL [4], RFCN [30], DHS [29], ELD [28], DISC [54], MSRNet [9], and DSS [5]). Notice that MSRNet [9] and DSS [5] are two of the best models to date. Here, our best results are shown at the bottom of Table 2.

F-measure and MAE Scores. Here, we compare our approach with the aforementioned approaches in terms of F-measure and MAE (See Table 2). As can be seen, our model trained on the MSRA-B dataset already outperforms all of the existing methods in terms of F-measure. With more training data, the results are improved further by a large margin (1% on average). This phenomenon is more pronounced when testing on the HKU-IS dataset. Notice that our approach does better than the best existing model [5, 9] using F-measure (2% improvement on average). Similar patterns can also be observed using the MAE score.

Visual Comparisons. In Fig. 4, we show the visual comparisons with several previous state-of-the-art approaches. In the top row, the source image have salient objects with complex textures. As can be seen, our approach is able to successfully segment all salient objects in images with boundaries being accurately highlighted. Some other methods such as DSS [5] and DCL [4] also produce high quality segmentation maps, although inferior to our results. A similar phenomenon also happens when processing images where contrast between foreground and background is low or when salient objects are tiny and irregular. Compared with existing methods, our approach performs much better in both cases. See for example the case shown at the bottom row of Fig. 4. These results demonstrate that our proposed horizontal hierarchy is capable of capturing rich and robust feature representations when applied to salient object detection. In addition to Fig. 4, we also provide more visual results which can be found in our supplementary materials.

Table 4. Quantitative comparison of our approach with existing edge detection methods. Here, 'MS' means multi-scale test as in [47]. Here, we use three scales {0.5, 1.0, 1.5, 2.0}. 'I' and 'II' correspond to the networks that are with 1 stage and 2 stages, respectively. The best results are highlighted in **bold**.

	Edge	
Method	ODS	OIS
gPb-owt-ucm [37]	0.726	0.757
SE-Var [41]	0.746	0.767
MCG [56]	0.747	0.779
DeepEdge [44]	0.753	0.772
DeepContour [43]	0.756	0.773
HED [3]	0.788	0.808
CEDN [57]	0.788	0.804
RDS [58]	0.792	0.810
COB [10]	0.793	0.820
CED [59]	0.803	0.820
DCNN+sPb [46]	0.813	0.831
RCF-MS [47]	0.811	0.830
UFNet-MS$^{\mathrm{I}}$ (Ours)	0.815	0.834
UFNet-MS$^{\mathrm{II}}$ (Ours)	**0.818**	**0.836**

5 Application II: Edge Detection

We also apply our UFNet to edge detection, one of the popular and basic low-level tasks in computer vision.

5.1 Ablation Analysis

The hyper-parameters used in our experiment include mini-batch size set to 10, momentum set to 0.9, weight decay set to 2e-4, and initial learning rate set to 1e-6 which is divided by 10 after 23,000 iterations. Our network is trained for 30,000 iterations. We evaluate our UFNet on the Berkeley Segmentation Dataset and Benchmark (BSDS 500) [37], which is one of notable benchmarks in the edge detection field. This dataset contains 200 training, 100 validation, and 200 testing images, each with accurately annotated boundaries. Besides, our training set also incorporates the images from the PASCAL Context Dataset [60] and performs data augmentation as in [47] for fair comparisons. Similar to previous works, we use the fixed contour threshold (ODS) and per-image best threshold (OIS) as our measures. Before evaluation, we apply the standard non-maximal suppression algorithm to get thinned edges.

The Number of Stages. The number of stages plays an important role in our approach. We consider the HED network [3] as a special case of UFNet with 0 stages. From Table 4, we observe that adding 1 stage (the bottom row) based on the HED architecture helps us obtain an increase of 0.4% in terms of ODS. When we add 2 stages, the ODS score can be further improved by 0.3 points. We observed no significant improvement when adding more than two stages.

The Number of Channels. As stated in Sec. 3.6, the convolutional layers in each side path are all with 16 channels. We also increased the channel numbers as done in [47]

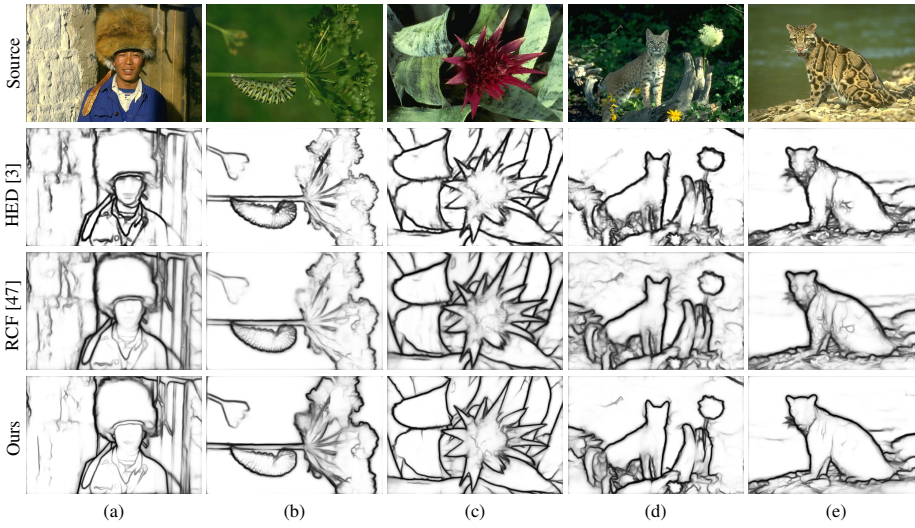

Fig. 5. Visual comparisons with a recently representative approach that leverage pure network features. All the images are from the BSDS 500 dataset [37].

(21 channels). However, the results show that more channels in each side path gives worse performance, leading to a decrease of around 0.2 points in terms of ODS. Similar phenomenon was also encountered when decreasing the number of channels.

The Effect of Horizontal Signal Flows. We also analyze the number of horizontal signal flows. While more horizontal signal flows helps salient object detection, we found that this operation does not boost the performance in edge detection. We also attempt to simplify our network by reducing the inputs of each transition node. According to the experimental results, both options degrade the performance by nearly 0.3 points in ODS. This might be caused by the fact that low-level features are essential to the fusion loss, which bring rich information about the details. More experimental settings and results can be found in our supplementary material.

5.2 Comparison with the State-of-the-Art

We compare our results with results from 12 existing methods, including gPb-owt-ucm [37], SE-Var [41], MCG [56], DeepEdge [44], DeepContour [43], HED [3], CEDN [57], RDS [58], COB [10], CED [59], DCNN+sPb [46], and RCF-MS [47], most of which are CNN-based methods.

Quantitative Analysis. In Table 4, we show the quantitative results of 12 previous works as well as ours. With only one stage, our method achieves ODS of 0.815 and OIS of 0.834, which are already better than the most of the previous works. This indicates that fusing features from different blocks of VGGNet performs better than combining only the feature maps from the same block [58]. On the other hand, more side supervision does help learning rich feature representations [10]. Furthermore, when we add

Table 5. Quantitative comparisons with existing skeleton extraction methods. The best results are highlighted in **bold**.

	Skeleton Datasets (F-measure)	
Method	SK-LARGE	WH-SYMMAX
HED [3]	49.7%	73.2%
FSDS [6]	63.3%	76.9%
LMSDS [11]	64.9%	77.9%
SRN [7]	61.5%	78.0%
UFNet (Ours)	**68.3%**	**80.1%**

another stage of transition nodes as in Table 1, our results can be further enhanced from ODS of 0.815 to ODS of 0.818 $(+0.003)$. For OIS, a similar phenomenon can also be found in Table 4.

Visual Analysis. In Fig. 5, we show some visual comparisons between our approach and a leading representative methods [58]. More visual results can be found in our *supplementary materials*. As can be observed, our approach performs better in detecting the boundaries compared to the other one. In Fig. 5b, it is apparent that the real boundaries of the plants are highlighted well. In addition, thanks to the fusion mechanism in our approach, the features learned by our network are much more powerful compared to [3, 58]. This is because the areas with no edges are rendered much cleaner, especially in Figs. 5a and 5b.

6 Application III: Skeleton Extraction

In this section, we apply our UFNet to skeleton extraction. We will show that our method substantially outperforms priors works by a large margin.

6.1 Ablation Analysis

The hyper-parameters we use are as follows: weight decay set to 0.0002, momentum set to 0.9, mini-batch size set to 10, and initial learning rate of 1e-6 which is divided by 10 after 20,000 iterations. We run the network for 30,000 iterations. A standard non-maximal suppression algorithm is used to obtain thinned skeletons before evaluation.

The Number of Stages. As in salient object detection and edge detection, reducing the number of stages when carrying out skeleton detection degrades the performance. When we remove S_2, the F-measure score drops by more than 2 points. Adding more stages in our approach also leads to no performance gain.

The Number of Channels. In our experiment, we reduce the number of channels in each side path to half of each and observe that the results slightly decrease. When we further reduce the channel numbers to a quarter of each, the performance drops dramatically (by more than 5 points). In addition, we also attempt to increase the number of channels by doubling them but find no performance gain.

The Effect of Horizontal Signal Flows The number of horizontal signal flows also affects the skeleton results. We reduce the number of optional connections between S_1

(a) GT (b) FSDS [6] (c) SRN [7] (d) Ours

Fig. 6. Visual comparisons with two recently representative skeleton extraction approaches. More can be found in our supplementary materials.

and S_2 from 3 to 2 but the F-measure score decreases by around 2 points. Reducing the number of optional connections between S_0 and S_1 leads to a similar phenomenon. This indicates introducing connections between higher and lower layers is essential.

6.2 Comparison with the State-of-the-Arts

We compare our UFNet with 4 recent CNN-based methods (HED [3], FSDS [6], LMSDS [11], and SRN [7]) on 2 popular and challenging datasets including SK-LARGE [11] and WH-SYMMAX [61]. Similar to [6], we use the F-measure score to evaluate the quality of prediction maps. In Table 5, we show quantitative comparisons with existing methods. As can be seen, our method wins dramatically by a large margin (3.4 points) on the SK-LARGE dataset [6]. There is also an improvement of 2.1 points on the WH-SYMMAX dataset [61]. In Fig. 6, we also show some visual illustrations of three approaches (more can be found in our supplementary materials). Owing to the advanced features extracted from our UFNet, our method is able to more accurately locate the exact positions of the skeletons. This point can also be substantiated by the fact that our prediction maps are also much thinner than other works. Both quantitative and visual results unveil that our horizontal hierarchy provides a better way to combine different-level features for skeleton extraction.

7 Discussion and Conclusion

In this paper, we present a novel architecture for binary vision tasks and apply it to three drastically different example tasks including salient object segmentation, edge detection, and skeleton extraction. Notice, however, that our approach is not limited to these tasks and can be potentially applied to a wide variety of binary pixel labeling tasks in computer vision. In order to take more advantage of CNNs, we introduce the concept of transition node, which receives signals from different-level features maps. Exhaustive evaluations and comparisons with recent notable state of the art methods on widely used datasets shows that our framework outperforms all of them in all three tasks, testifying the power of our proposed framework. Further, we structurally analyze our proposed architecture using several ablation experiments in each task and investigate the roles of different design choices in our approach. We hope that our work will encourage subsequent research to design universal architectures for computer vision tasks.

References

1. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015) 3431–3440
2. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR. (2017)
3. Xie, S., Tu, Z.: Holistically-nested edge detection. In: ICCV. (2015) 1395–1403
4. Li, G., Yu, Y.: Deep contrast learning for salient object detection. In: CVPR. (2016)
5. Hou, Q., Cheng, M.M., Hu, X., Borji, A., Tu, Z., Torr, P.: Deeply supervised salient object detection with short connections. In: CVPR. (2017)
6. Shen, W., Zhao, K., Jiang, Y., Wang, Y., Zhang, Z., Bai, X.: Object skeleton extraction in natural images by fusing scale-associated deep side outputs. In: CVPR. (2016) 222–230
7. Ke, W., Chen, J., Jiao, J., Zhao, G., Ye, Q.: Srn: Side-output residual network for object symmetry detection in the wild. arXiv preprint arXiv:1703.02243 (2017)
8. Kokkinos, I.: Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In: CVPR. (2017)
9. Li, G., Xie, Y., Lin, L., Yu, Y.: Instance-level salient object segmentation. In: CVPR. (2017)
10. Maninis, K.K., Pont-Tuset, J., Arbelaez, P., Van Gool, L.: Convolutional oriented boundaries: From image segmentation to high-level tasks. IEEE TPAMI (2017)
11. Shen, W., Zhao, K., Jiang, Y., Wang, Y., Bai, X., Yuille, A.: Deepskeleton: Learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images. IEEE Transactions on Image Processing **26**(11) (2017) 5298–5311
12. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE TPAMI (11) (1998) 1254–1259
13. Klein, D.A., Frintrop, S.: Center-surround divergence of feature statistics for salient object detection. In: ICCV. (2011)
14. Xie, Y., Lu, H., Yang, M.H.: Bayesian saliency via low and mid level cues. IEEE TIP **22**(5) (2013) 1689–1698
15. Cheng, M., Mitra, N.J., Huang, X., Torr, P.H., Hu, S.: Global contrast based salient region detection. IEEE TPAMI (2015)
16. Perazzi, F., Krähenbühl, P., Pritch, Y., Hornung, A.: Saliency filters: Contrast based filtering for salient region detection. In: CVPR. (2012) 733–740
17. Liu, T., Yuan, Z., Sun, J., Wang, J., Zheng, N., Tang, X., Shum, H.Y.: Learning to detect a salient object. IEEE TPAMI **33**(2) (2011) 353–367
18. Jiang, H., Wang, J., Yuan, Z., Wu, Y., Zheng, N., Li, S.: Salient object detection: A discriminative regional feature integration approach. In: CVPR. (2013) 2083–2090
19. Borji, A., Cheng, M.M., Jiang, H., Li, J.: Salient object detection: A survey. arXiv preprint arXiv:1411.5878 (2014)
20. Borji, A., Cheng, M.M., Jiang, H., Li, J.: Salient object detection: A benchmark. IEEE TIP **24**(12) (2015) 5706–5722
21. Li, G., Yu, Y.: Visual saliency based on multiscale deep features. In: CVPR. (2015) 5455–5463
22. Zhao, R., Ouyang, W., Li, H., Wang, X.: Saliency detection by multi-context deep learning. In: CVPR. (2015) 1265–1274
23. Zou, W., Komodakis, N.: Harf: Hierarchy-associated rich features for salient object detection. In: ICCV. (2015) 406–414
24. Wang, L., Lu, H., Ruan, X., Yang, M.H.: Deep networks for saliency detection via local estimation and global search. In: CVPR. (2015) 3183–3192
25. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014) 580–587

26. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE TPAMI **34**(11) (2012) 2274–2282
27. Zhao, J.X., Bo, R., Hou, Q., Cheng, M.M.: Flic: Fast linear iterative clustering with active search. arXiv preprint arXiv:1612.01810 (2016)
28. Gayoung, L., Yu-Wing, T., Junmo, K.: Deep saliency with encoded low level distance map and high level features. In: CVPR. (2016)
29. Liu, N., Han, J.: Dhsnet: Deep hierarchical saliency network for salient object detection. In: CVPR. (2016)
30. Wang, L., Wang, L., Lu, H., Zhang, P., Ruan, X.: Saliency detection with recurrent fully convolutional networks. In: ECCV. (2016)
31. Li, X., Zhao, L., Wei, L., Yang, M.H., Wu, F., Zhuang, Y., Ling, H., Wang, J.: Deepsaliency: Multi-task deep neural network model for salient object detection. IEEE TIP **25**(8) (2016) 3919 – 3930
32. Canny, J.: A computational approach to edge detection. IEEE Transactions on pattern analysis and machine intelligence (6) (1986) 679–698
33. Marr, D., Hildreth, E.: Theory of edge detection. Proceedings of the Royal Society of London B: Biological Sciences **207**(1167) (1980) 187–217
34. Torre, V., Poggio, T.A.: On edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (2) (1986) 147–163
35. Konishi, S., Yuille, A.L., Coughlan, J.M., Zhu, S.C.: Statistical edge detection: Learning and evaluating edge cues. IEEE TPAMI **25**(1) (2003) 57–74
36. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE TPAMI **26**(5) (2004) 530–549
37. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE TPAMI **33**(5) (2011) 898–916
38. Dollar, P., Tu, Z., Belongie, S.: Supervised learning of edges and object boundaries. In: CVPR. Volume 2., IEEE (2006) 1964–1971
39. Ren, X.: Multi-scale improves boundary detection in natural images. ECCV (2008) 533–545
40. Lim, J.J., Zitnick, C.L., Dollár, P.: Sketch tokens: A learned mid-level representation for contour and object detection. In: CVPR. (2013) 3158–3165
41. Dollár, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE TPAMI **37**(8) (2015) 1558–1570
42. Ganin, Y., Lempitsky, V.: Nˆ 4-fields: Neural network nearest neighbor fields for image transforms. In: ACCV, Springer (2014) 536–551
43. Shen, W., Wang, X., Wang, Y., Bai, X., Zhang, Z.: Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In: CVPR. (2015) 3982–3991
44. Bertasius, G., Shi, J., Torresani, L.: Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In: CVPR. (2015) 4380–4389
45. Hwang, J.J., Liu, T.L.: Pixel-wise deep learning for contour detection. In: ICLR. (2015)
46. Kokkinos, I.: Pushing the boundaries of boundary detection using deep learning. arXiv preprint arXiv:1511.07386 (2015)
47. Liu, Y., Cheng, M.M., Hu, X., Wang, K., Bai, X.: Richer convolutional features for edge detection. In: CVPR. (2017)
48. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
49. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACM Multimedia, ACM (2014) 675–678
50. Yan, Q., Xu, L., Shi, J., Jia, J.: Hierarchical saliency detection. In: CVPR. (2013) 1155–1162
51. Yang, C., Zhang, L., Lu, H., Ruan, X., Yang, M.H.: Saliency detection via graph-based manifold ranking. In: CVPR. (2013) 3166–3173

52. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: ICCV. Volume 2. (2001) 416–423
53. Movahedi, V., Elder, J.H.: Design and perceptual validation of performance measures for salient object segmentation. In: CVPR. (2010) 49–56
54. Chen, T., Lin, L., Liu, L., Luo, X., Li, X.: Disc: Deep image saliency computing via progressive representation learning. IEEE TNNLS **27**(6) (2016) 1135–1149
55. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. In: NIPS. (2011)
56. Pont-Tuset, J., Arbelaez, P., Barron, J.T., Marques, F., Malik, J.: Multiscale combinatorial grouping for image segmentation and object proposal generation. IEEE TPAMI **39**(1) (2017) 128–140
57. Yang, J., Price, B., Cohen, S., Lee, H., Yang, M.H.: Object contour detection with a fully convolutional encoder-decoder network. In: CVPR. (2016) 193–202
58. Liu, Y., Lew, M.S.: Learning relaxed deep supervision for better edge detection. In: CVPR. (2016) 231–240
59. Wang, Y., Zhao, X., Huang, K.: Deep crisp boundaries. In: CVPR. (2017) 3892–3900
60. Mottaghi, R., Chen, X., Liu, X., Cho, N.G., Lee, S.W., Fidler, S., Urtasun, R., Yuille, A.: The role of context for object detection and semantic segmentation in the wild. In: CVPR. (2014) 891–898
61. Shen, W., Bai, X., Hu, Z., Zhang, Z.: Multiple instance subspace learning via partial random projection tree for local reflection symmetry in natural images. Pattern Recognition **52** (2016) 306–316