Learning Superpixels with Segmentation-Aware Affinity Loss

Wei-Chih Tu[1] Ming-Yu Liu[2] Varun Jampani[2] Deqing Sun[2] Shao-Yi Chien[1] Ming-Hsuan Yang[2,3] Jan Kautz[2]
[1]National Taiwan University [2]NVIDIA [3]UC Merced

Abstract

Superpixel segmentation has been widely used in many computer vision tasks. Existing superpixel algorithms are mainly based on hand-crafted features, which often fail to preserve weak object boundaries. In this work, we leverage deep neural networks to facilitate extracting superpixels from images. We show a simple integration of deep features with existing superpixel algorithms does not result in better performance as these features do not model segmentation. Instead, we propose a segmentation-aware affinity learning approach for superpixel segmentation. Specifically, we propose a new loss function that takes the segmentation error into account for affinity learning. We also develop the Pixel Affinity Net for affinity prediction. Extensive experimental results show that the proposed algorithm based on the learned segmentation-aware loss performs favorably against the state-of-the-art methods. We also demonstrate the use of the learned superpixels in numerous vision applications with consistent improvements.

1. Introduction

Superpixels are the image regions generated by grouping image pixels. Superpixels provide a more natural representation of image data compared to pixels. In addition, superpixels reduce the number of primitives to operate on, thereby improving the computational efficiency of vision algorithms. The process of extracting superpixels is known as superpixel segmentation or over-segmentation. Superpixels are widely used in both conventional energy-minimization [32, 30] and recent deep learning [23, 13, 9] frameworks with applications to a wide range of problems such as salient object detection [29, 32, 13], and semantic segmentation [23, 9], to name a few.

In light of fundamental importance of superpixels in computer vision, numerous superpixel segmentation algorithms [17, 1, 27, 16, 2] have been proposed in the literature. Despite their differences in problem formulation, existing algorithms mainly rely on hand-crafted features, and thus often fail to separate objects from the backgrounds if no strong boundaries can be identified.

In this work, we leverage deep networks to facilitate extracting superpixels from images. There are several challenges in learning superpixels using deep networks. First, there is no groundtruth for superpixels. Second, the indices of different superpixels are interchangeable. Third, existing superpixel algorithms are not differentiable. To overcome these issues, we propose to learn pixel affinities for graph-based superpixel segmentation. Pixel affinities measure the likelihood of two neighboring pixels belonging to the same object. With better pixel affinities that take object boundaries into account, graph-based algorithms can extract semantically more meaningful superpixels. Empirically, we find that deep features learned from other high-level vision tasks do not perform well for superpixel segmentation. Our experiments show that simply replacing the hand-crafted features with pre-trained deep features for computing pixel affinities does not result in good superpixel segmentation. A closely-related problem to affinity computation is edge detection, where a number of CNN-based methods have been developed. One may expect that edges extracted from the state-of-the-art deep networks [28] provide effective visual cues of pixel affinities for superpixel segmentation. However, our experiments show that this approach results in inferior performance compared to that of using hand-crafted pixel affinities.

We observe that affinities derived from pre-trained deep features or deep edges are not sensitive to segmentation errors. We therefore propose a method for learning segmentation-aware affinities for graph-based superpixel segmentation. Specifically, we propose a deep network, termed Pixel Affinity Net (PAN), for learning pixel affinities and exploiting existing segmentation datasets [3, 6] as supervisory signals where pixel affinities should be low at segmentation boundaries and high elsewhere. To ensure the predicted affinities result in good segmentation quality, we propose a new SEgmentation-Aware Loss (SEAL) function. We first use the predicted affinities to compute superpixels with a graph-based algorithm. Then the SEAL function computes a loss at every pixel based on the computed superpixels and groundtruth segmentation. The loss at each pixel is then back-propagated through the proposed PAN model to adjust its parameters, thereby facilitating the use of deep neural networks for learning superpixels while avoid-

ing back-propagating through the non-differentiable superpixel segmentation algorithm.

Extensive experiments on the BSDS500 [3] and Cityscapes [6] datasets demonstrate that the proposed learning-based approach performs favorably against the state-of-the-art superpixel segmentation methods. In addition, we show that improvements in superpixel accuracy, obtained with our approach, also translates to performance improvements in vision tasks that rely on superpixels, such as semantic segmentation and salient object detection.

We make the following contributions in this work:

- We propose a deep learning based approach to learn pixel affinities for superpixel segmentation. To the best of our knowledge, this is the first deep learning based approach for superpixel segmentation.
- We develop a novel segmentation-aware loss that makes use of segmentation errors to learn affinities for superpixel segmentation.
- We demonstrate that with the learned pixel affinities, the computed superpixels preserve object boundaries better than those with hand-crafted features. Experiments show that our algorithm performs favorably against the state-of-the-arts and helps improve superpixel-based vision applications.

2. Related Work

In this work, we briefly review the superpixel algorithms closely related to this work in proper context.

Graph-based algorithms. Graph-based algorithms consider an image as a planar graph, where pixels are vertices and pixel affinities are computed for connected pixels. These algorithms compute superpixels by partitioning the graph. The Normalized Cuts (NC) [22] method generates superpixels by recursively computing normalized cuts on the graph. The NC method generates compact superpixels but they do not adhere to boundaries well. Felzenszwalb and Huttenlocher (FH) [8] propose a bottom-up method that treats image pixels as disjoint sets and progressively merges the sets based on internal variation [8] of pixel affinities. Although the FH method preserves boundaries well, it often generates both extremely large and small segments. Moreover, it is difficult to control the number of superpixels in the FH method. In [11], Grundmann *et al*. use the χ^2 distance between color histograms of two adjacent sets as a merging criterion for superpixel segmentation. The ERS [17] algorithm merges disjoint sets by maximizing the entropy rate of pixel affinities to extract superpixels. While the above methods differ in graph merging or splitting techniques, they all use hand-crafted features to compute superpixels.

Clustering-based algorithms. Numerous superpixel segmentation methods are developed based on clustering techniques. These algorithms progressively refine an initial clustering of pixels until meeting the specified criteria.

The SEEDS [27] algorithm uses uniform blocks as initial approximation of superpixels and iteratively exchanges neighboring blocks in a coarse-to-fine manner based on an objective function. Yao *et al*. [31] use the block-based initialization and propose a more complex objective function for superpixel segmentation. Typically, these block-based methods do not take number of superpixels as input. Instead, users need to set several parameters manually (e.g., minimum block size and number of scales) based on image resolution and desired number of superpixels.

The SLIC [1] method places initial cluster centers on a uniform grid and performs k-means in the five dimensional CIELab color and position feature space to cluster pixels. The LSC [16] method projects the five dimensional features to a ten dimensional space and performs clustering in the projected space. The Manifold-SLIC [18] method clusters pixels in a reduced two dimensional space. Both LSC and Manifold-SLIC schemes suggest that the segmentation quality can be improved by merely changing the feature representation, which also inspires us to leverage learning techniques for superpixel segmentation. More recently, Achanta *et al*. introduce the SNIC [2] algorithm based on a non-iterative clustering scheme. It only visits most pixels once during clustering while still achieving state-of-the-art performance among the clustering-based methods.

Other approaches. Several other algorithms [26] have been developed based on other techniques such as the Watershed transform [20] and geometric flows [15], also using hand-crafted features.

3. Superpixels Meet Deep Learning

Learning superpixels with deep neural networks is not straightforward due to the following issues. First, there is no groundtruth for superpixel segmentation. As superpixels are over-segmentation of an image, the superpixel boundaries can be arbitrary as long as a superpixel is within an object. Second, the superpixel indices are interchangeable. We can shuffle the indices while keeping the superpixel representation intact. Third, most superpixel algorithms are not differentiable as they often involve discrete operations of clustering or subset selection.

A naïve way to address these issues is to use deep features from pre-trained networks and plug them into existing superpixel algorithms. Since pre-trained deep network features were shown to generalize to different vision tasks [10, 5], one may expect that these features would lead to performance gain in superpixel segmentation. We adopt this strategy using the SNIC [2] and ERS [17] methods.

We first replace the hand-crafted features in the SNIC and ERS algorithms with the deep features extracted from the pre-trained VGG16 model [25]. As illustrated in Fig-

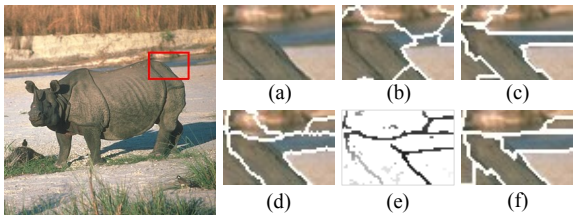

Figure 1: **Segmentation leakage.** (a) shows the typical challenge in natural images that the object has similar colors with background. (b) and (c) show the results of the SNIC and ERS methods using the deep features from VGG16. (d) is the result of the ERS method using the HED-based affinities. (e) is the corresponding affinities derived from HED edges. We also show the result using our learned segmentation-aware affinities in (f).

ure 1 (b) and (c), the extracted superpixels often fail to align with object boundaries. While the deep features provide discriminative information for high-level vision tasks, they are invariant to some spatial information which is of great importance for localizing object boundaries. More details regarding this experiment and quantitative results are presented in Section 5.1 and supplementary materials.

As edge detection is closely related to image segmentation, we also consider utilizing edge information to guide the graph-based superpixel algorithms. By letting pixels with high edge probability have low pixel affinities and pixels with low edge probability have high affinities, the graph-based algorithms should be able to keep object boundaries from merging when computing superpixels. We use the ERS algorithm and derive the pixel affinities from the groundtruth segmentation maps [3], which we refer to as "oracle affinities". We find the performance of the ERS algorithm can be significantly improved with the oracle affinities. We analyze whether similar improvements can be obtained with pixel affinities derived from the state-of-the-art deep edge detectors such as HED [28]. However, no significant improvements can be achieved with the HED-based affinities. Figure 1(d) shows that a few missing boundary pixels in the near-perfect boundary map (Figure 1(e)) lead to merging of foreground and background regions and introduce segmentation errors. More experimental details along with the quantitative results are presented in Section 5.1.

The reason that pre-trained deep features perform poorly for superpixel segmentation can be attributed to the fact that image segmentation is not explicitly considered in the training objectives. For classification networks, the objective is to correctly classify images in terms of feature representations that are invariant to local edge information, thereby making them unsuitable for superpixel segmentation. Deep edge detectors, on the other hand, are trained with the objective of maximizing boundary accuracy. As a result, a few missing boundary pixels would only contribute to a small increase in edge detection loss but can cause a large segmentation error when used as affinities for superpixel segmentation.

4. Learning Segmentation-Aware Affinities

We propose a segmentation-aware pixel affinity learning approach for graph-based superpixel segmentation. It overcomes the challenges discussed in Section 3 via learning pixel affinities by minimizing a novel segmentation-aware loss function. The PAN architecture is designed to predict 4-connected pixel affinities on image graphs. These predicted affinities are then passed to a graph-based superpixel algorithm. Based on the computed superpixels and the groundtruth object segmentation map, the SEAL function is computed and the errors are back-propagated to train the PAN model. An overview of the proposed affinity learning framework is illustrated in Figure 2.

4.1. Segmentation-aware loss

We leverage existing segmentation datasets (e.g., BSDS500 [3]) as supervisory signals to train the PAN model for affinity prediction. We note that the groundtruth segmentation map is for object segmentation rather than for superpixel segmentation. However, as superpixels are over-segmentation of an image, we can still use the object segmentation groundtruth to define a proper loss function for affinity prediction. The SEAL function is for this purpose. To use the segmentation datasets, we first convert the groundtruth segmentation maps into pixel affinities with zeros for the boundary pixels and ones for the remaining pixels. We use horizontal label transitions in the groundtruth segmentation maps to generate horizontal affinities and use vertical transitions to generate vertical affinities. For instance, if there is a groundtruth label transition from pixel (x, y) to pixel $(x + 1, y)$ then we only set the horizontal affinity at (x, y) as zero, but not vertical affinity.

Given the binary groundtruth affinities and the network predictions in both the horizontal and vertical directions, we formulate the affinity learning task as a supervised learning problem and use the binary cross-entropy (BCE) loss L:

$$L(\theta) = -\sum_i \big(t_i \log(a_i) + (1-t_i)\log(1-a_i)\big), \quad (1)$$

where $a_i \in (0, 1)$ denotes predicted affinity at pixel i and $t_i \in \{0, 1\}$ denotes the corresponding target affinity computed from groundtruth segmentation maps. In addition, θ denotes the parameters of the PAN model. Note that training affinities is similar to training edge prediction where the simple BCE loss suffers from imbalance of edge and non-edge samples. To overcome data imbalance, extra weights are introduced in the class-balancing BCE loss [28] to balance two classes when training edge detectors. However, it still does not take segmentation into account. Our experiments (in Section 5.1) also indicate that the affinities trained

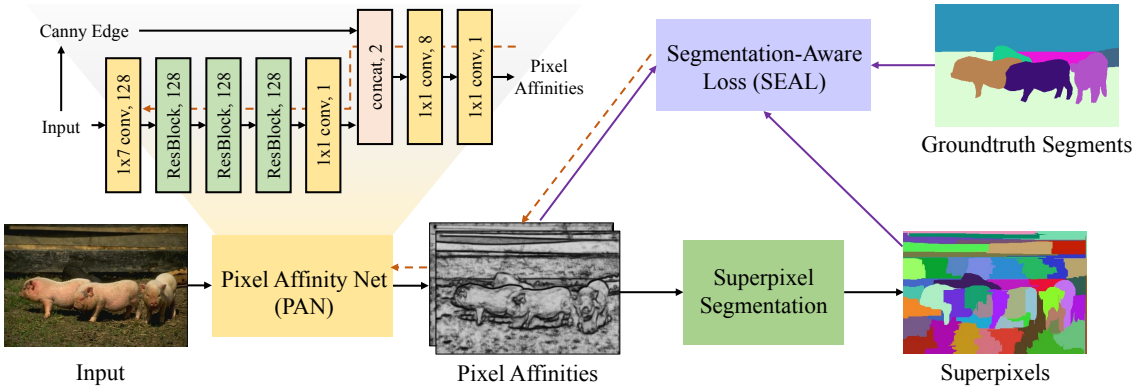

Figure 2: **Illustration of the proposed affinity learning framework.** The PAN model takes the image as input and predicts horizontal and vertical affinities at each pixel. The affinities are then passed into a graph-based superpixel algorithm to compute superpixels. The SEAL function takes the affinities, computed superpixels and the groundtruth segmentation map as input and computes gradients for back-propagation. The dashed lines indicate the flow of gradients.

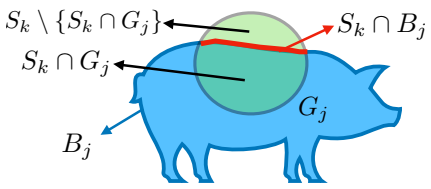

Figure 3: **Segmentation errors.** Illustration of segmentation error of a superpixel S_k (in green) with respect to the groundtruth object boundary B_j (in blue). We weigh the loss on boundary pixels (in red) with the segmentation error $S_k \setminus \{S_k \cap G_j\}$.

with class-balancing BCE loss do not generate satisfactory superpixels as this loss is agnostic to segmentation errors.

We propose the SEAL, which improves the BCE loss by tying it to superpixel segmentation errors. Specifically, we first use the predicted affinities to compute superpixels using a graph-based superpixel segmentation algorithm. The resulting superpixels are then compared with the groundtruth segmentation map to measure the segmentation errors at each superpixel.

As illustrated in Figure 3, we use S_k to refer to a superpixel (shaded green) and G_j to refer to the groundtruth object segment (shaded blue) that overlaps the most with S_k. Because a superpixel must belong to a single object, the over-segmentation error ω_{S_k} is given by the number of pixels in S_k but not in G_j:

$$\omega_{S_k} = |S_k \setminus \{S_k \cap G_j\}| = |S_k| - |S_k \cap G_j|. \quad (2)$$

If the predicted affinities perfectly correlate with object boundaries, ω_{S_k} should be zero. When ω_{S_k} is not zero, it means some predicted affinities are incorrect. The key idea of the SEAL function is to penalize pixel affinity predictions according to the over-segmentation errors.

Specifically, let B_j denote the set of groundtruth boundary pixels (dark blue contour). The pixels that are both in the groundtruth boundaries and in S_k are given by $S_k \cap B_j$ (red pixels in Figure 3). The SEAL function is defined as a weighted BCE loss:

$$L_{\text{SEAL}}(\theta) = -\sum_i (1+\gamma_i)(t_i \log a_i + (1-t_i)\log(1-a_i)), \quad (3)$$

where $\gamma_i = \omega_{S_k}$ if $i \in S_k \cap B_j$ for some superpixel S_k and the corresponding groundtruth segment G_j, otherwise $\gamma_i = 0$. In other words, If the pixels are on the groundtruth boundaries, we weigh the BCE loss with over-segmentation error $(1 + \gamma_i)$; otherwise, we use plain BCE loss. This way a larger over-segmentation error will lead to a larger loss value, which in term will induce stronger gradients during training with the back-propagation algorithm. Furthermore, since we only weigh the groundtruth boundary pixels, we implicitly handle the data imbalance issue.

4.2. Pixel Affinity Net

We propose the PAN model to predict affinities from a given image. The input to the network is a color image and the output is a two-channel affinity map with one representing horizontal affinities and the other for vertical affinities. For each pixel, the network predicts affinities towards right and bottom pixels. For instance, the horizontal affinity at pixel (x, y) indicates the affinity value between pixel (x, y) and $(x + 1, y)$ and the vertical affinity at pixel (x, y) is the affinity value between pixel (x, y) and $(x, y + 1)$. The last column of horizontal affinity and the last row of vertical affinity are discarded during testing and gradients with respect to these locations are set to zero during training.

The local affinity is computed independently of the orientation of an input image. That is, both the horizontal and vertical affinities can be computed using the same network by rotating the input image along the horizontal and vertical axis. We train the PAN model to predict horizontal affinities only and the vertical affinities are computed by passing the

rotated input image. These predicted affinities correspond to edge weights on the image graph, which are then fed into a graph-based superpixel segmentation algorithm.

The PAN model, illustrated in Figure 2, uses 1×7 horizontal convolution kernels in the first layer to capture horizontal changes in the image. This is followed by 3 standard residual blocks (ResBlock) [12] of 128 channels. A single 1×1 convolution layer is then used to convert 128 channels into 1-channel intermediate affinity prediction. This intermediate affinity map is concatenated with the Canny edges [4] and is then passed onto two 1×1 convolution layers for final affinity prediction. We use Canny edges here because they help localize the boundaries. Empirically, they lead to improved performance as will be shown in Section 5. We use sigmoid activations at the layers generating intermediate and final affinity maps to constrain the affinity values between 0 and 1. All other convolutional layers are interleaved with the ReLU function.

5. Experiments

Implementation details. We use the Adam [14] optimization to train the PAN model for 100k iterations. We set the initial learning rate to $1e$-4 and reduce it by a factor of 10 every 30k iterations until the learning rate drops to $1e$-6. Other optimization parameters (β_1, β_2) and weight decay are set to $(0.9, 0.999)$ and $1e$-4 respectively. We use the ERS [17] algorithm to compute superpixels in the main paper. In the supplementary materials, we report additional results using the FH [8] algorithm. During training, the PAN model generates horizontal and vertical affinities that are passed into the 4-connected ERS algorithm to compute superpixels.

For testing, we develop an approach to derive the diagonal affinities from 4-connected affinities and use the 8-connected ERS to compute superpixels. We observe that using the 8-connected ERS with the learned affinities achieves the best performance, which we refer to as the SEAL-ERS. The details for approximating 8-connected affinities from the 4-connected affinities are described in the supplementary materials.

Performance metrics. We use the Achievable Segmentation Accuracy (ASA) and Boundary Recall (BR) metrics for performance evaluation. The ASA score evaluates superpixels by measuring the total effective segmentation area of a superpixel representation with respect to the groundtruth segmentation map. For example, if a superpixel straddles an object boundary, only the larger side is considered correct. The BR score, on the other hand, measures the superpixel boundary adherence with respect to the groundtruth boundary. Higher ASA and BR scores indicate better superpixel segmentation results. More details about these two metrics are presented in the supplementary material. Some existing methods [2, 27] use the Corrected Under-Segmentation Error (CUSE) metric, which is equivalent to $(1 - \text{ASA})$. In this paper, we report all experimental results using the ASA and BR metrics.

5.1. Comparisons with baselines

We compute a number of baseline affinities using several existing techniques and use them with the ERS algorithm [17] for superpixel segmentation. We then evaluate the resulting superpixels with the ASA and BR scores on the BSDS500 [3] test set. These scores for different baseline methods are shown in Figure 4 where the standard ERS algorithm with affinities computed with RGB pixel differences is referred to as "RGB features", and our proposed method is referred to as "Affinities trained with SEAL loss". The plots indicate that the proposed algorithm performs favorably against the standard ERS method in terms of ASA scores with slightly better BR scores.

Affinities with pre-trained deep features. As stated in Section 3, we evaluate affinities computed using deep features from pre-trained networks. Specifically, we extract features from the VGG16 [25] model trained for the ImageNet classification and use them to compute pixel affinities. We experiment with features from different layers and find that conv1_1 layer features result in the best performance for the pre-trained deep feature based superpixels. We refer to this experiment as "VGG16 Features" in Figure 4. The ASA and BR scores indicate that using VGG16 features for affinity measure performs worse than that using basic RGB pixel features. We also observe similar performance drop with the SNIC [2] method using pre-trained deep features, and present the experimental results in the supplementary material.

Edge-based affinities. Edge detection is closely related to segmentation. As discussed in Section 3, we experiment with the affinities computed from the state-of-the-art deep edge detectors. Specifically, we use the HED [28] method and convert the predicted edge map E into affinity map A by $A = 1 - E$. The results with these edge-based affinities are referred to as "HED-based affinities" in Figure 4. Experimental results indicate that the ERS method using HED-based affinities achieves similar ASA score as the original ERS superpixels but with a lower BR. In Figure 1, we show a few missing boundary pixels in the predicted edges can cause large errors in superpixel segmentation.

Class-balancing BCE loss. In addition to existing deep features, we also experiment with another variant of BCE loss function that is commonly used for training edge detectors [28]. The class-balancing BCE loss uses different weights for the boundary and non-boundary pixels in the BCE loss function to account for the data imbalance. Both

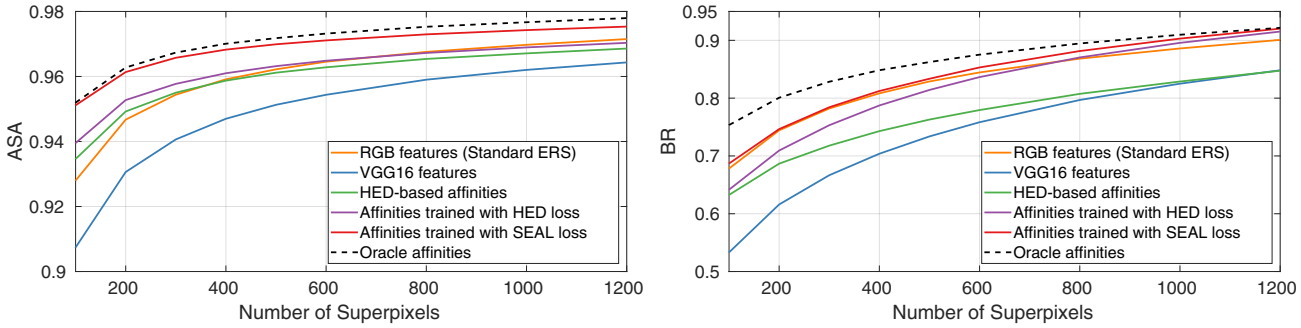

Figure 4: **Comparison to baseline affinities.** We compute ASA and BR scores on the BSDS500 test set for superpixels computed using the ERS [17] algorithm with different baseline affinities and our learned affinities.

this class-balancing loss and the proposed segmentation-aware loss are weighted BCE losses, but the class-balancing loss is still agnostic to superpixel segmentation errors. As a baseline, we train the PAN model with this class-balancing BCE loss and report the resulting superpixel scores in Figure 4 as "Affinities trained with HED loss ". Experimental results indicate that this approach performs worse than that with the affinities trained with the SEAL function.

The oracle. We also carry out experiments using affinities computed from the groundtruth segmentation maps in the BSDS500 dataset. Since, several groundtruth segmentation annotations are provided for each image, we average the affinities computed from each of the annotations and consider that as groundtruth affinities. We refer to the superpixels computed using those groundtruth affinities as "the oracle" and the results in Figure 4 show that the oracle affinities can significantly boost the performance of the ERS algorithm. We note that our approach with learned affinities performs almost as well as that based on the oracle in terms of ASA score.

5.2. Comparisons with the state-of-the-arts

We compare our method with the state-of-the-art superpixel algorithms including the ERS [17], SLIC [1], LSC [16], SEEDS [27] and SNIC [2]. We evaluate on the BSDS500 [3] and the Cityscapes [6] datasets, where accurate segmentation masks are provided.

BSDS500. We train the PAN model using 300 images in the BSDS500 train and validation sets and evaluate on 200 test images. Each image in this dataset is provided with multiple groundtruth annotations. For training, we treat each annotation as an independent sample. With this dataset, we have 1633 train pairs and 1063 test pairs of images and annotations. We apply random image rotations and reflections as data augmentation for training the PAN model. Empirically, we observe that the proposed affinity learning framework is not sensitive to the number of superpixels used in training. We use a fixed number of 300 superpixels during training and apply the trained model to compute superpixels of different numbers.

The experimental results with the ASA and BR scores are shown in Figure 6. We set the boundary tolerance to 1 pixel for computing the BR scores. In addition to separate ASA and BR plots, we show a combined plot in Figure 6 where average ASA and average BR from 100 to 1200 superpixels are used in this plot. Evaluation scores indicate that our proposed SEAL-ERS method performs favorably against the base ERS algorithm and other state-of-the-art superpixel segmentation algorithms. Note that the SEAL-ERS method is able to retain high ASA score even when the number of superpixels is small. Figure 5 shows a couple of visual results where foreground objects have similar color to background. The SEAL-ERS method using the learned affinities demonstrate better boundary-preserving ability for superpixel segmentation.

Cityscapes. We evaluate superpixel segmentation methods on a larger Cityscapes [6] dataset. Each image in this dataset is provided with a pixel-wise segmentation map. We train our model with 2975 Cityscapes train images and evaluate on 500 validation images. As the image resolution is high (1024×2048), we use random 512×512 crops to train our network. Similar to the experimental setup with the BSDS500 dataset, we fix the number of superpixels to 300 during training. For testing, we measure the ASA and BR scores with the number of superpixels ranging from 1000 to 6000. As the images in this dataset contain long and thin objects, we use a larger number of superpixels for all the evaluated methods. We set the boundary tolerance to 3 pixels for computing BR scores, as this dataset images are of high-resolution. Figure 7 shows that significant improvements can be obtained by the SEAL-ERS method for this dataset as well, which demonstrates the generalization capability of the proposed algorithm. Specifically, our approach performs favorably against existing superpixel algorithms in this dataset. Note that the SEEDS [27] algorithm is not included for comparison on this dataset as we are not able

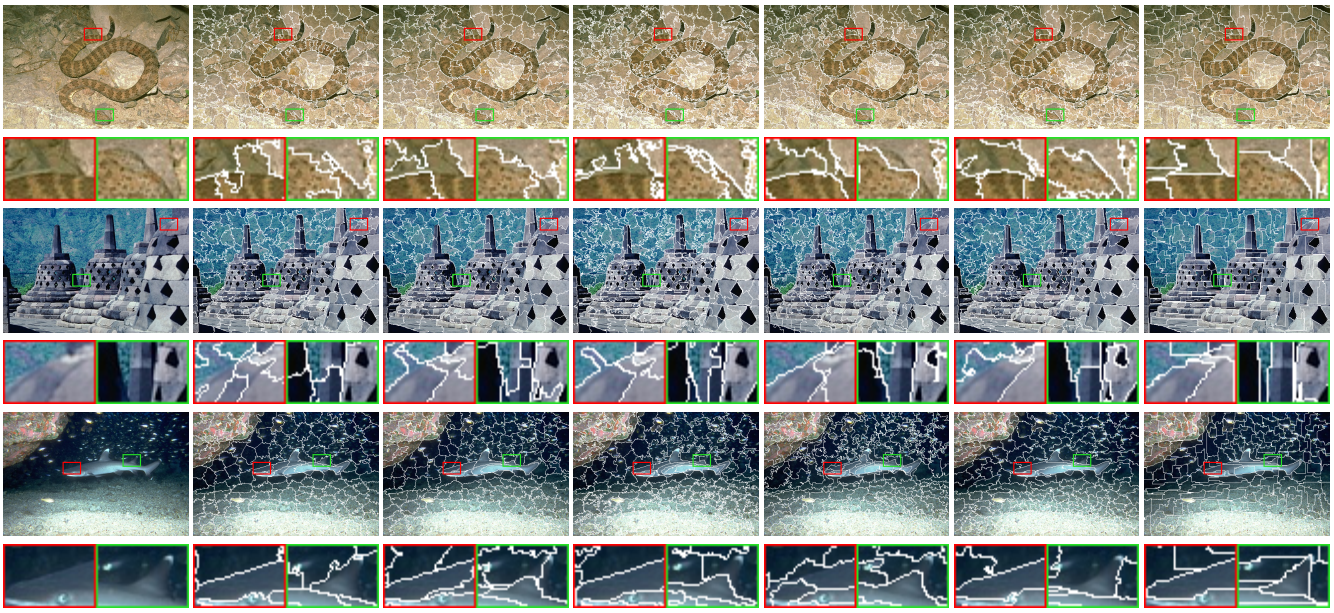

Figure 5: **Visual results.** We show 200 superpixels generated by the existing state-of-the-arts and our SEAL-ERS superpixel algorithm.

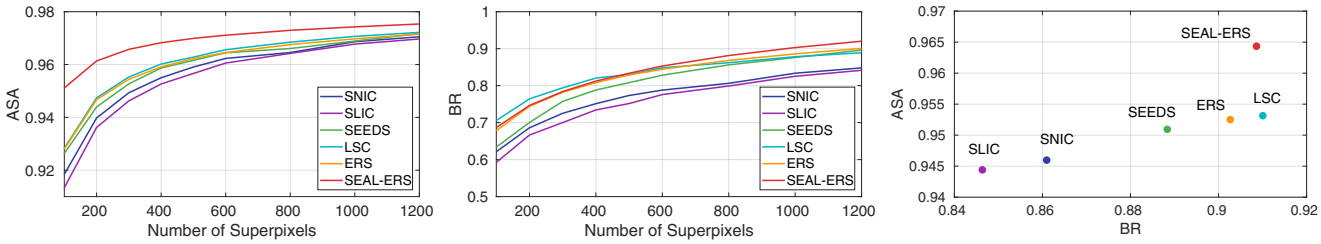

Figure 6: **Results on BSDS500.** We compare our SEAL-ERS method with the state-of-the-art algorithms on the BSDS500 test set.

to determine reasonable parameters (e.g., minimum block size, number of scales) to generate the desired number of superpixels for fair evaluations.

5.3. Ablation study of PAN

We present an ablation study where we evaluate different design choices of the proposed PAN model. There are 3 key features that distinguishes PAN from standard deep architectures: 1). The model predicts only horizontal affinities rather than both horizontal and vertical affinities. We refer to this aspect of our network as "One-way prediction". 2). The model uses horizontal 1D convolution (1×7) filters in the first layer. 3). The model fuses Canny edges with the intermediate affinity prediction. For comparison, we include a baseline model that predicts both horizontal and vertical affinities, uses 7×7 filters instead of 1×7 and without using Canny edge fusion. We evaluate each of these design options of the network. Figure 8 shows that each of the three alternatives in the PAN model perform relatively well. In other words, the proposed PAN model performs robustly with respect to different design choices.

6. Applications

We evaluate whether the improvements in superpixel quality achieved by our learning technique can translate to the improvements in downstream vision tasks that use superpixels. For this study, we choose existing semantic segmentation and salient object detection techniques, and replace the superpixels used in those techniques with our SEAL-ERS superpixels.

Semantic segmentation. CNN models achieve the state-of-the-art performance for semantic segmentation [5, 19]. However, most CNN architectures for semantic segmentation generates lower resolution outputs which are then upsampled using post-processing techniques such as DenseCRF. Recently, Gadde *et al*. [9] propose the Bilateral Inception (BI) networks, where superpixels are used for long-range and edge-aware propagation across CNN units, thereby alleviating the need for post-processing CRF tech-

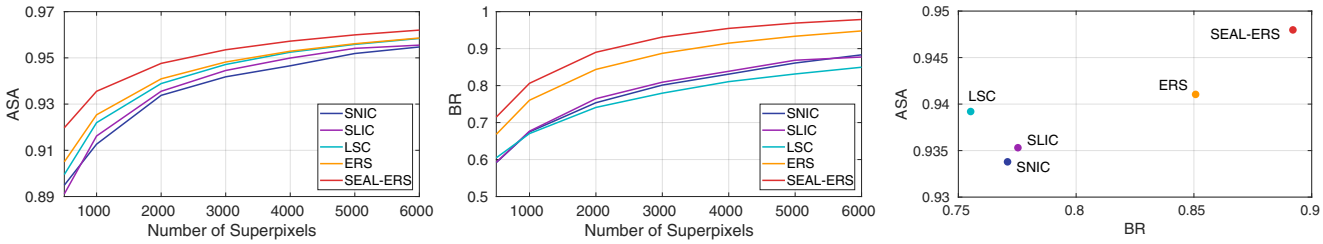

Figure 7: **Results on Cityscapes.** We compare our SEAL-ERS method with the state-of-the-arts on the Cityscapes validation set.

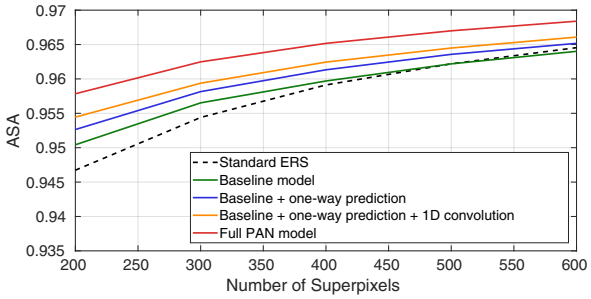

Figure 8: **Ablation study.** We show the effectiveness of each design choice in the PAN model in improving accuracy.

Table 1: **Superpixels for semantic segmentation.** We compute semantic segmentation using the BI network [9] with different types of superpixels and compare the IoU scores on the PascalVOC 2012 test set.

Method	# of Superpixels	IoU
DeepLab [5]	-	68.9
+ CRF [5]	-	72.7
+ BI (SLIC) [9]	1000	74.1
+ BI (ERS)	1000	74.5
+ BI (SEAL-ERS)	600	75.0
+ BI (SEAL-ERS)	1000	**75.4**

niques. In the original BI network [9], 1000 SLIC superpixels are used for segmentation on the PascalVOC dataset. Here, we replace those with the ERS and SEAL-ERS superpixels and evaluate the resulting semantic segmentation on the PascalVOC 2012 test set [7]. We report the standard Intersection over Union (IoU) scores in Table 1. The results indicate that we can also obtain significant IoU improvements when using the learned SEAL-ERS superpixels (75.4) compared to standard ERS superpixels (74.5). In addition, we observe that SEAL-ERS can obtain better IoU (75.0) even when using less superpixels (600).

Salient object detection. Numerous salient object detection algorithms are based on superpixels. We demonstrate the effectiveness of SEAL-ERS superpixels by substituting SLIC superpixels used in existing salient object detection algorithms with our superpixels. We experiment with two salient object detection algorithms: Saliency Optimization (SO) [32] and Graph-based Manifold Ranking (GMR) [29]. We report standard Mean Absolute Error (MAE) and weighted F_β [21] scores on the ECSSD dataset [24]. With the same setting as in [32, 29], we use 200 superpixels for this study. Experimental results in Table 2 show that the use of SEAL-ERS superpixels consistently improves the performance of both the SO and GMR methods in both metrics. These results on semantic segmentation and salient object detection demonstrate the potential of using learned superpixels for downstream vision tasks.

Table 2: **Superpixels for salient object detection.** We run the SO [32] and GMR [29] algorithms with different types of superpixels and evaluate on the ECSSD dataset.

		SLIC [1]	ERS [17]	SEAL-ERS
SO [32]	MAE ↓	0.1719	0.1686	**0.1633**
	Weighted F_β ↑	0.5101	0.5070	**0.5237**
GMR [29]	MAE ↓	0.1892	0.1877	**0.1841**
	Weighted F_β ↑	0.4909	0.4897	**0.4955**

7. Conclusion

In this paper, we present a segmentation-aware affinity learning framework for superpixel segmentation. We propose the PAN model for affinity prediction and develop the SEAL that makes use of superpixel segmentation errors for affinity learning. The resulting SEAL-ERS method generates better boundary-preserving superpixels compared to those using hand-crafted features or deep features trained without taking segmentation into account. Our SEAL-ERS performs favorably against several existing state-of-the-art superpixels. Furthermore, we demonstrate that the SEAL-ERS can be used in numerous vision applications with significant performance gains.

Acknowledgement. M.-H. Yang is supported in part by NSF CAREER (No. 1149783) and gifts from Adobe, Toyota, Panasonic, Samsung, NEC, Verisk, and NVidia.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(11):2274–2282, 2012.

[2] R. Achanta and S. Susstrunk. Superpixels and polygons using simple non-iterative clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(5):898–916, 2011.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 8(6):679–698, 1986.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *International Conference on Learning Representations (ICLR)*, 2015.

[6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, 111(1):98–136, 2015.

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision (IJCV)*, 2004.

[9] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. Gehler. Superpixel convolutional networks using bilateral inceptions. In *European Conference on Computer Vision (ECCV)*, 2016.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[11] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[13] S. He, R. W. Lau, W. Liu, Z. Huang, and Q. Yang. Supercnn: A superpixelwise convolutional neural network for salient object detection. *International Journal of Computer Vision (IJCV)*, 115(3):330–344, 2015.

[14] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv*, abs/1412.6980, 2014.

[15] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(12):2290–2297, 2009.

[16] Z. Li and J. Chen. Superpixel segmentation using linear spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[17] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa. Entropy rate superpixel segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[18] Y.-J. Liu, C.-C. Yu, M.-J. Yu, and Y. He. Manifold slic: A fast method to compute content-sensitive superpixels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[20] V. Machairas, M. Faessel, D. Cárdenas-Peña, T. Chabardes, T. Walter, and E. Decencière. Waterpixels. *IEEE Transactions on Image Processing (TIP)*, 24(11):3707–3716, 2015.

[21] R. Margolin, L. Zelnik-Manor, and A. Tal. How to evaluate foreground maps? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[22] X. Ren and J. Malik. Learning a classification model for segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[23] A. Sharma, O. Tuzel, and M.-Y. Liu. Recursive context propagation network for semantic scene labeling. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[24] J. Shi, Q. Yan, L. Xu, and J. Jia. Hierarchical image saliency detection on extended cssd. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 38(4):717–729, 2016.

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, abs/1409.1556, 2014.

[26] D. Stutz, A. Hermans, and B. Leibe. Superpixels: An evaluation of the state-of-the-art. *arXiv*, abs/1612.01601, 2016.

[27] M. Van den Bergh, X. Boix, G. Roig, and L. Van Gool. Seeds: Superpixels extracted via energy-driven sampling. *International Journal of Computer Vision (IJCV)*, 111(3):298–314, 2015.

[28] S. Xie and Z. Tu. Holistically-nested edge detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[29] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang. Saliency detection via graph-based manifold ranking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[30] F. Yang, H. Lu, and M.-H. Yang. Robust superpixel tracking. *IEEE Transactions on Image Processing (TIP)*, 23(4):1639–1651, 2014.

[31] J. Yao, M. Boben, S. Fidler, and R. Urtasun. Real-time coarse-to-fine topologically preserving segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[32] W. Zhu, S. Liang, Y. Wei, and J. Sun. Saliency optimization from robust background detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.