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

A. Performance Metrics

In this paper, we use the Achievable Segmentation Accuracy (ASA) and Boundary Recall (BR) scores as the performance metrics for superpixel segmentation. We give more details about these two metrics here.

Formally, let N be the number of pixels in a given image and let the image be partitioned into K superpixels $S = \{S_k\}_{k=1}^K$. Both metrics are evaluated using an object segmentation dataset (e.g., BSDS500 [3]). Let $G = \{G_j\}_{j=1}^J$ be the J groundtruth segments in the image. We have $\sum_k |S_k| = \sum_j |G_j| = N$, where $|.|$ returns the number of pixels in a set.

For every superpixel S_k, the ASA metric finds the groundtruth segment G_j that overlaps the most with S_k and find the overlap area A_k between S_k and G_j. The ASA score is computed as the ratio of total overlap area to the total superpixel area:

$$ASA(S,G) = \frac{\sum_{k=1}^K A_k}{\sum_{k=1}^K |S_k|} = \frac{1}{N}\sum_{k=1}^K \max_j |S_k \cap G_j|. \quad (1)$$

The BR score, on the other hand, measures how the superpixel segmentation aligns with groundtruth object boundaries. Let ∂G be the set of all groundtruth boundary pixels. The BR score is computed as:

$$BR(S,G) = \frac{TP(S,G)}{TP(S,G) + FN(S,G)} = \frac{TP(S,G)}{|\partial G|}, \quad (2)$$

where $TP(S,G)$ and $FN(S,G)$ are the number of true positive and false negative boundary pixels in S. Note that $TP(S,G) + FN(S,G)$ is equivalent to the number of groundtruth boundary pixels $|\partial G|$. We compute $TP(S,G)$ by checking every groundtruth boundary pixel. For each groundtruth boundary pixel, if there is any superpixel boundary pixel within a $(2r+1)\times(2r+1)$ neighborhood, $TP(S,G)$ is increased by one. Here, r is set according to image resolution. We use larger r for high resolution images to have larger tolerance to boundary misalignment. In our experiments, we use $r = 1$ for the BSDS500 [3] dataset and $r = 3$ for the Cityscapes [5] dataset.

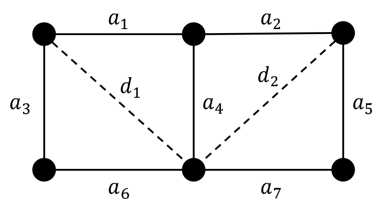


Figure 1: **Diagonal affinities.** We can approximate the affinities on diagonal edges by the affinities on the 4-connected edges.

B. Approximation of Diagonal Affinities

Typically, the graph-based algorithms have better segmentation accuracy in the 8-connected setting, while being slower than that with 4-connected setting. We propose an approach to approximate the diagonal affinities from the horizontal and vertical affinities, so that we can train the model faster with 4-connected setting while using more accurate 8-connected algorithms for testing.

Figure 1 illustrates a 2×3 pixel graph, where we use a to represent the horizontal and vertical affinities and d for the diagonal affinities. To compute d_1, we first compute the average horizontal affinity in the same grid cell

$$\bar{a_h} = \frac{1}{2}(a_1 + a_6), \quad (3)$$

and the average vertical affinity

$$\bar{a_v} = \frac{1}{2}(a_3 + a_4). \quad (4)$$

The diagonal affinity is then approximated by

$$d_1 = \frac{1}{\sqrt{2}}\min(\bar{a_h}, \bar{a_v}) = \frac{1}{2\sqrt{2}}\min(a_1 + a_6, a_3 + a_4). \quad (5)$$

We use $\frac{1}{\sqrt{2}}$ to account for longer spatial distance of diagonal edges. Similarly, we compute d_2 as

$$d_2 = \frac{1}{2\sqrt{2}}\min(a_2 + a_7, a_4 + a_5). \quad (6)$$

We carry out experiments to show the effectiveness of using 8-connected affinities. Figure 2 shows the result of the

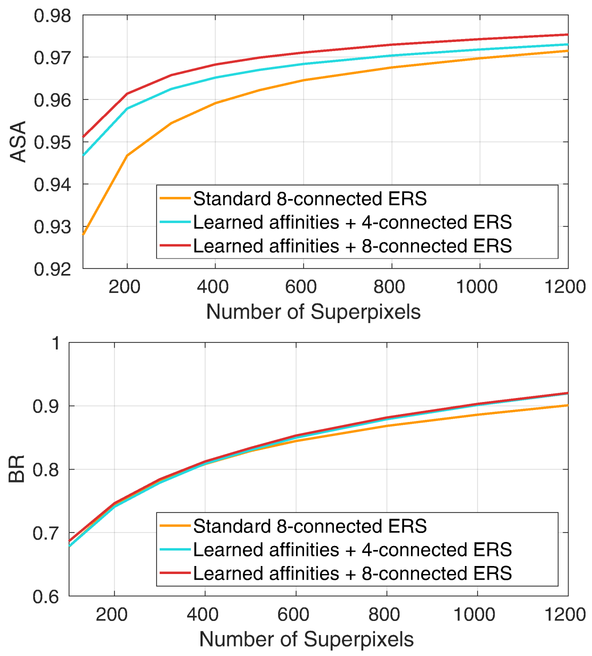


Figure 2: **Effectiveness of diagonal affinities.** We show that the 8-connected ERS algorithm using the approximated 8-connected affinities performs better than that using the learned 4-connected affinities.

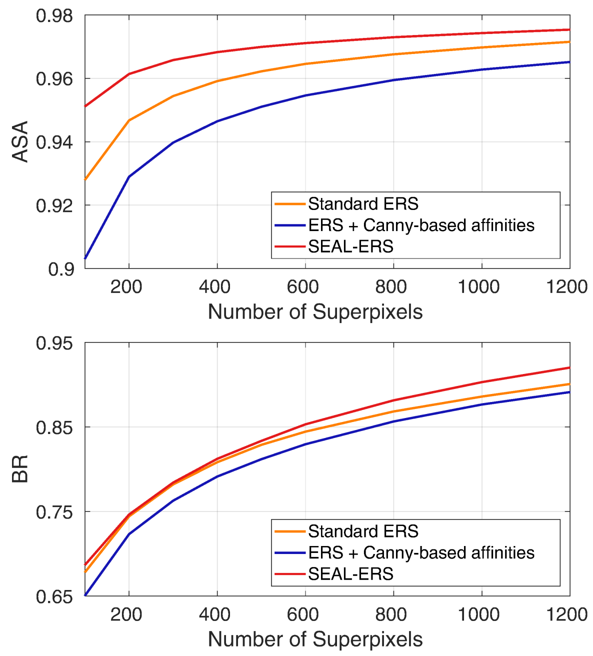


Figure 3: **Experiment on Canny-based affinities.** ASA and BR scores for superpixels computed using ERS with the affinities derived from the Canny edges.

ERS [9] superpixel segmentation algorithm using different input affinities. "Learned affinities" refers to the affinities trained on the 4-connected graph using our segmentation-aware learning framework. As we can see, the segmentation accuracy of the ERS algorithm can be further improved by using the approximated 8-connected affinities while maintaining similar boundary recall. We have also tried to use the approximation and the 8-connected ERS algorithm during training. We find it results in similar testing performance to that of using 4-connected graph for training and 8-connected graph for testing, while being slower.

C. More Baselines

Affinities derived from Canny edges. Similar to the experiment with the HED [13] edges, we also experiment with the affinities derived from the Canny [4] edges. As shown in Figure 3, with Canny-based affinities, the ERS algorithm performs worse than that with standard color difference based affinities. We have similar observation that Canny edges often result in a few missing boundary pixels (i.e., open object contour), attributing to the drop in segmentation accuracy. While the Canny edges alone can not improve the performance, the Canny edges usually can detect local edges, which provides useful auxiliary information for our affinity learning framework. As we show in the main paper, the Canny edge fusion results in better affinity prediction.

Using pre-trained deep features. We use the low-level features extracted from the first four layers of the pretrained VGG16 network [10] to replace hand-crafted features used in the ERS [9] and SNIC [2] algorithms. We discard the pooling layer between conv1_2 and conv2_1 as pooling results in a loss of fine-grained pixel level information. We change the dilation in conv2_1 and conv2_2 accordingly to keep the receptive field the same.

Originally, the ERS algorithm uses RGB color difference to compute affinities of neighboring pixels, and the SNIC method uses CIELab and spatial location as five dimensional features to cluster pixels. We modify the ERS algorithm such that it takes the M-dimensional deep feature $(v_1, v_2, ..., v_M)$ to compute pixel affinity as $\exp(-\frac{d^2}{2\sigma^2})$, in which the feature distance d is computed as

$$d = \sqrt{\frac{1}{M}\sum_{m=1}^{M}(\Delta v_m)^2}. \qquad (7)$$

We modify the SNIC algorithm in a similar way such that it takes d, the distance between deep features, and the pixel location together to cluster pixels. We denote the features extracted from conv1_1 layer as *V1* and the features concatenated by conv1_1 and conv1_2 as *V2*. Similarly, *V4* denotes the features concatenated by all the first four layers.

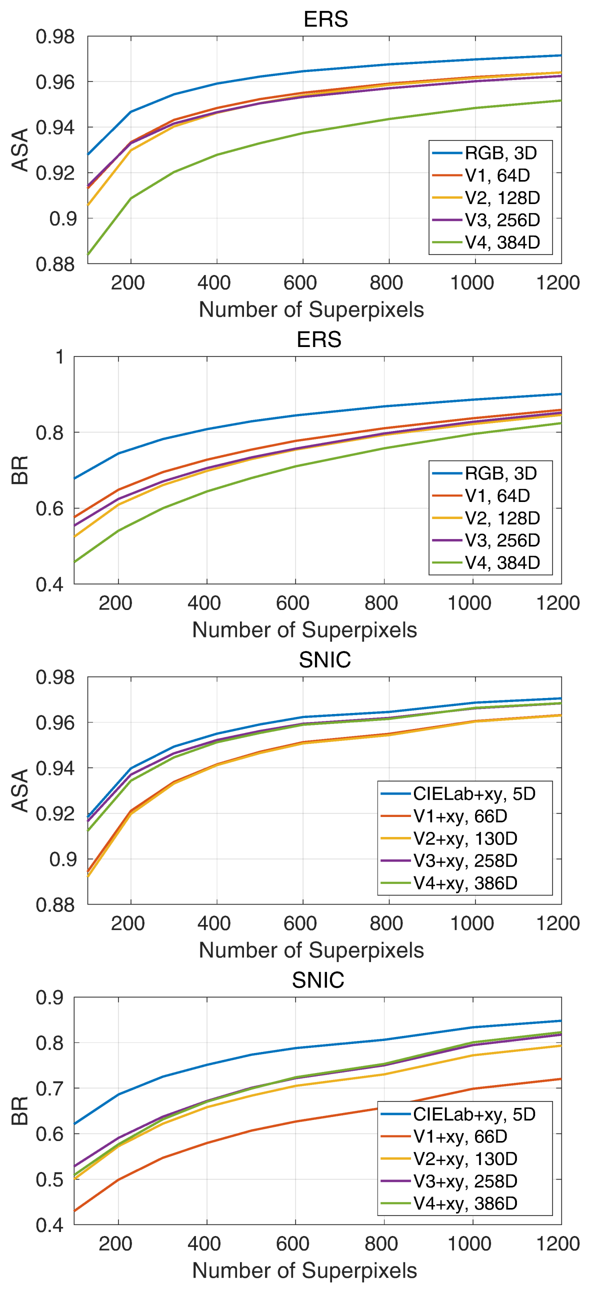


Figure 4: **Using deep features for superpixel segmentation.** We show the results using low-level features extracted from the VGG16 model for superpixel segmentation.

Table 1: **Experimental results on the FH method.** The proposed affinity learning is also effective for the FH [6] method.

Method	ASA	BR
FH	0.929347	0.731893
SEAL-FH	0.948155	0.738872

We evaluate the results using the 200 images in the BSDS500 test set. As we can see in Figure 4, using deep features as a drop-in replacement for the hand-crafted features degrades the performance for both algorithms.

D. Generalization to other Graph-based Algorithms

Our affinity learning framework can be used with any graph-based superpixel algorithm that takes pixel affinities as input. Some other graph-based methods [7, 12], that do not directly use pixel affinities as graph merging criteria but use statistical features (e.g., histograms) computed from sub-graphs, are not applicable to our framework.

In addition to the ERS algorithm, we also experiment with the FH [6] algorithm that uses pixel affinities as input. Similarly, we term the method using the learned affinities and the FH algorithm as SEAL-FH. Table 1 shows the results on the BSDS500 test set. As it is hard to control the number of superpixels for the FH method, we simply average the ASA and BR scores for all the test images. We find that the proposed affinity learning framework is also effective in improving the performance of FH method showcasing the generality of our pixel affinity learning framework.

E. Visual Comparison

In this section, we present more visual comparisons of superpixel segmentation on the BSDS500 [3] and Cityscapes [5] datasets.

BSDS500. Figure 5 shows more visual results on the BSDS500 [3] test set images using 200 superpixels. We also highlight regions where there are weak object boundaries. Results show that our SEAL-ERS method produces superpixels with better boundary-preserving ability. Particularly, our method gives higher priority to object boundaries than texture edges. For example, in the zebra image, the stripes have higher contrast than the boundaries of zebra bodies. Existing methods relying on hand-crafted features tend to form superpixels along the stripes while compromising the object boundaries. On the other hand, our method is able to generate semantically more meaningful superpixels that preserve the boundaries of those zebras well. This would be more useful for high level vision applications.

Cityscapes. We show visual comparisons of results using 1000 superpixels on the Cityscapes [5] dataset in Figure 6 and Figure 7. We also highlight important regions in the scene, such as traffic sign and the boundary between two vehicles. We show the zoom-in comparisons in Figure 8. As we can see, our method can produce semantically more meaningful superpixels.

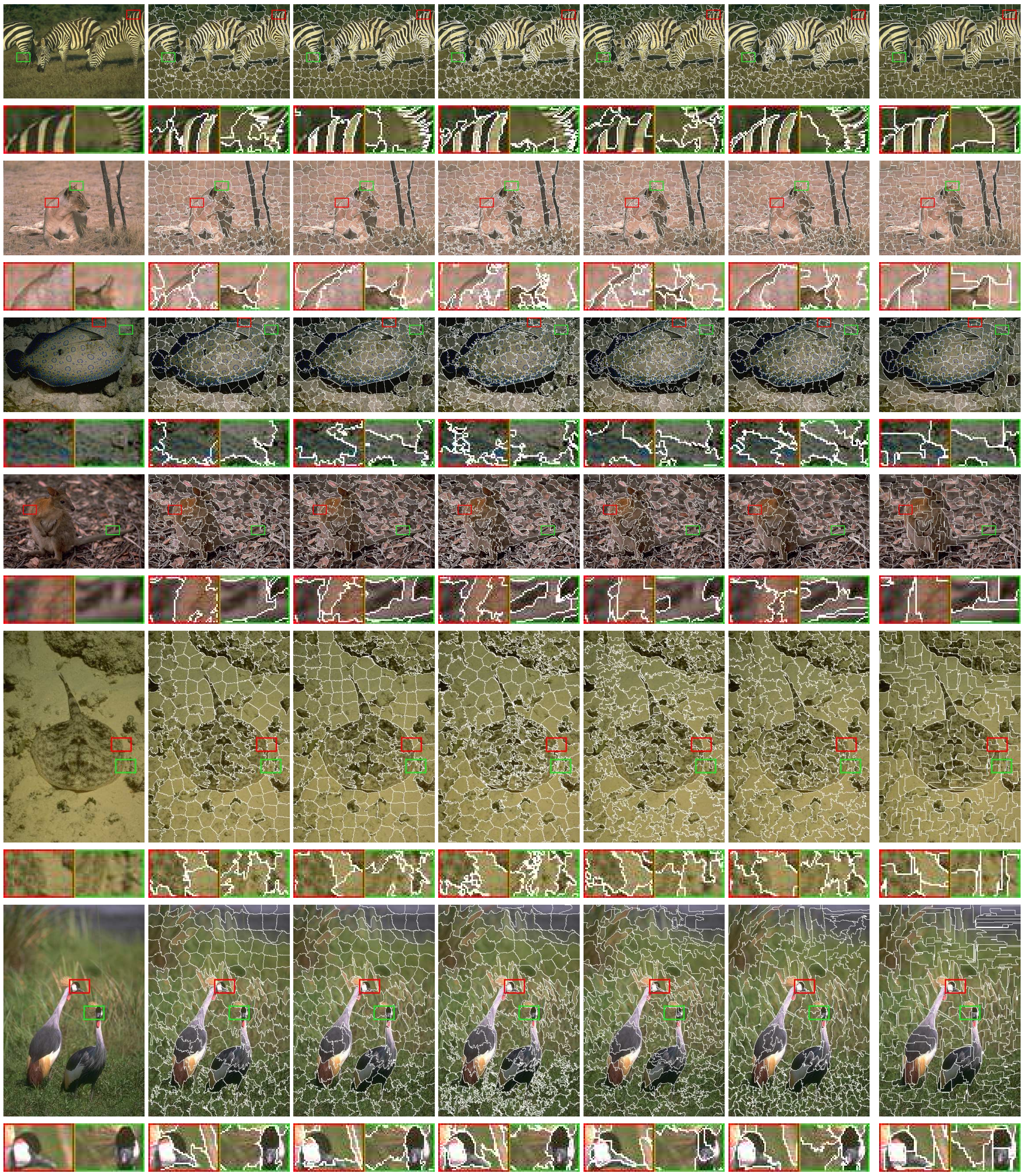
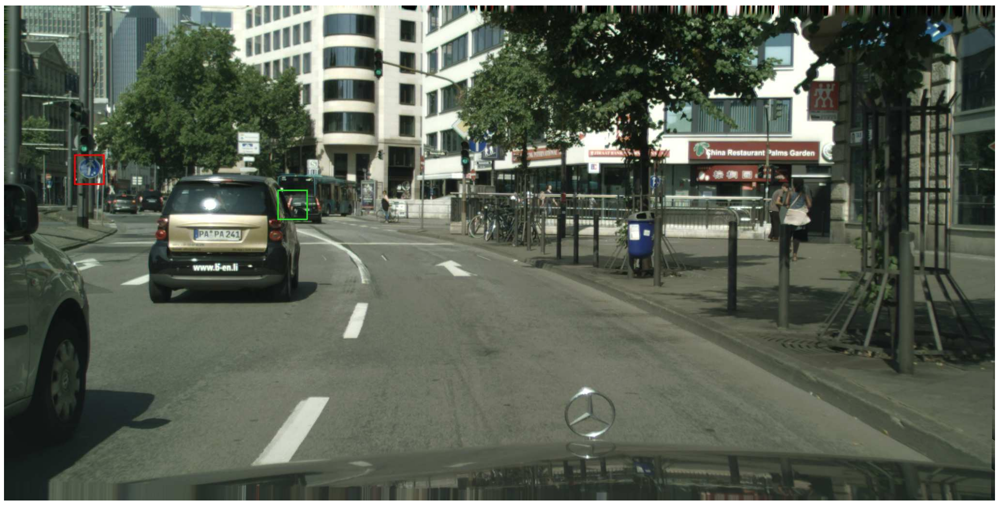
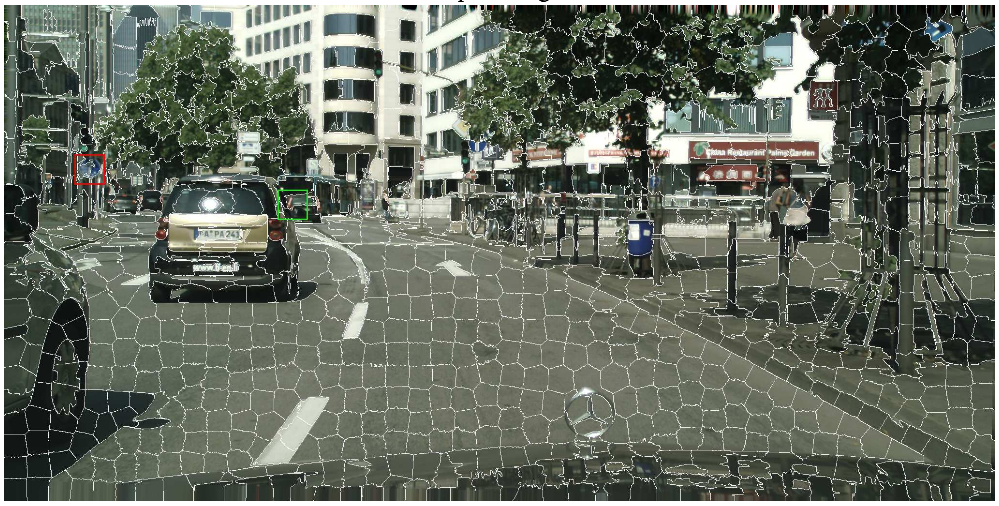


Figure 5: **Visual results on sample BSDS500 [3] test images.** We show the comparison of 200 superpixels generated by the state-of-the-art methods and ours. We highlight the regions that object boundaries are weak. Our method can generate superpixels with better boundary-preserving ability.


Input image


SLIC [1]

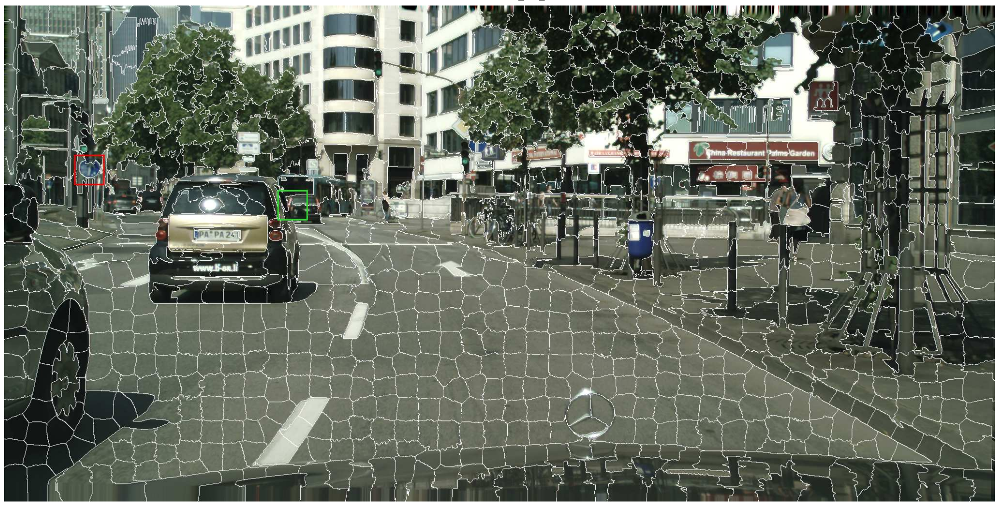
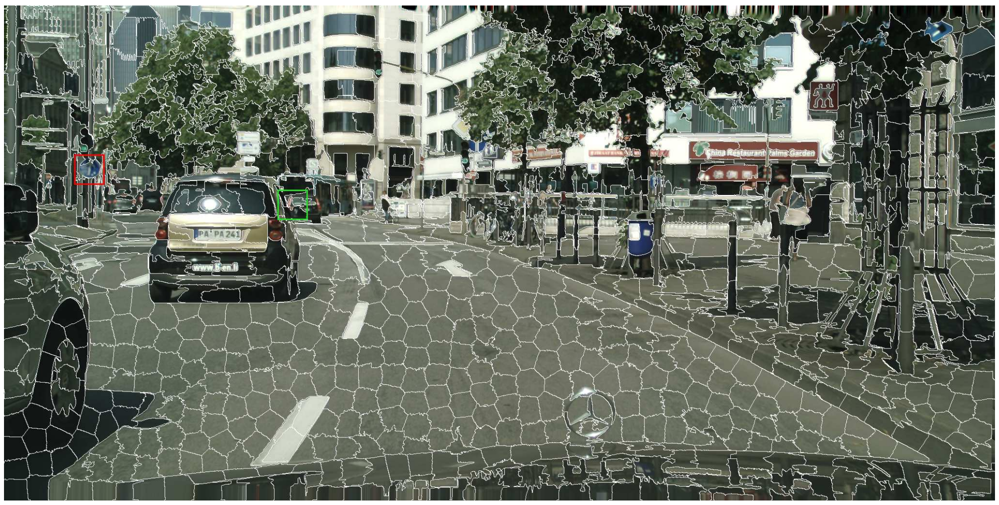
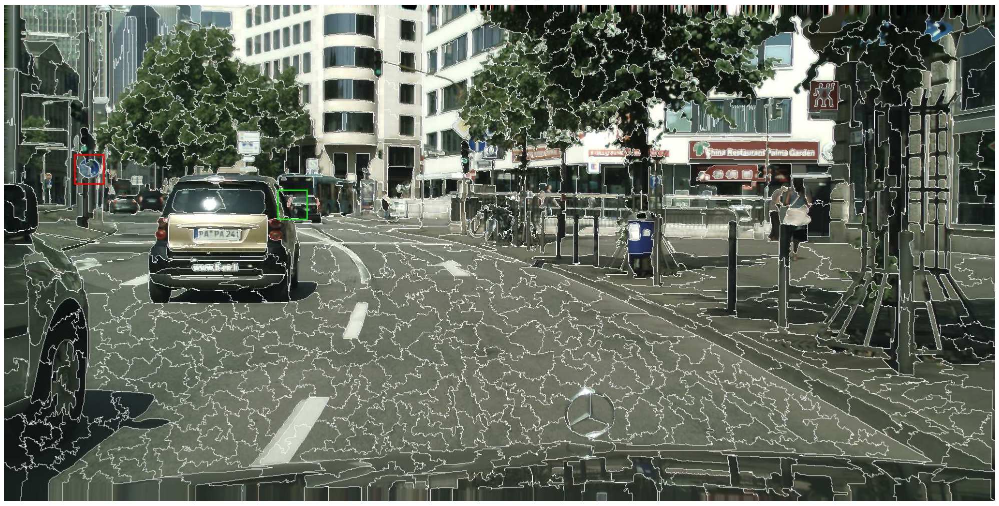

SNIC [2]

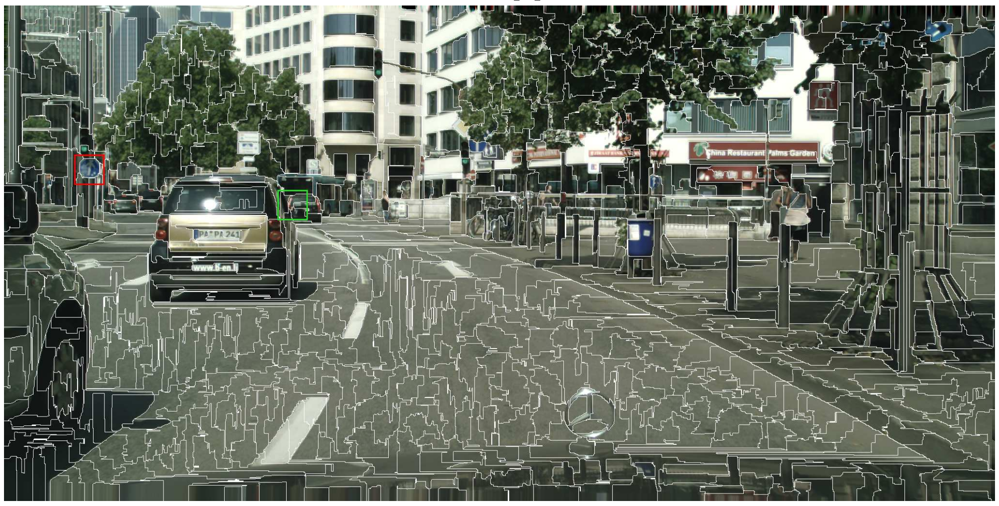
Figure 6: **Sample visual result from the Cityscapes [5] validation set.** We show the comparison of 1000 superpixels generated by the SLIC and SNIC methods.



LSC [8]



ERS [9]



SEAL-ERS (Ours)

Figure 7: **Sample visual result from the Cityscapes [5] validation set.** We show the comparison of 1000 superpixels generated by the LSC, ERS and our SEAL-ERS methods.

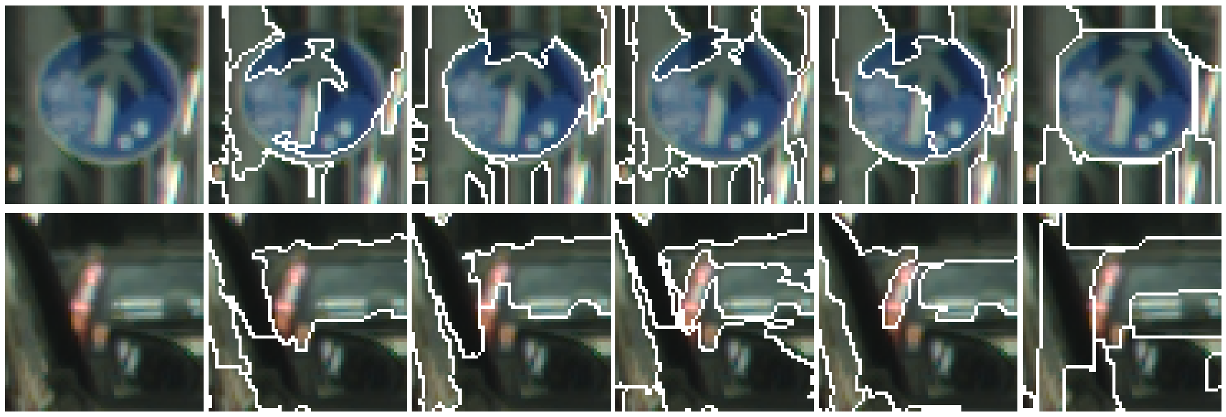


Figure 8: **Semantically meaningful superpixels with learned affinities .** We highlight regions in Figure 6 and Figure 7. Our method is able to produce semantically more meaningful superpixels compared to others.

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] 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.

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

[7] 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.

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

[9] 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.

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

[11] 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.

[12] X. Wei, Q. Yang, Y. Gong, M.-H. Yang, and N. Ahuja. Superpixel hierarchy. *arXiv*, abs/1605.06325, 2016.

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