SUPERPIXEL INTEGRATED GRIDS FOR FAST IMAGE SEGMENTATION

Jack Roberts

Brown University
Department of Computer Science
Providence, RI, USA

Jeova Farias Sales Rocha Neto

Bowdoin College
Department of Computer Science
Brunswick, ME, USA

ABSTRACT

Superpixels have long been used in image simplification to enable more efficient data processing and storage. However, despite their computational potential, their irregular spatial distribution has often forced deep learning approaches to rely on specialized training algorithms and architectures, undermining the original motivation for superpixelations. In this work, we introduce a new superpixel-based data structure, SIGRID (Superpixel-Integrated Grid), as an alternative to full-resolution images in segmentation tasks. By leveraging classical shape descriptors, SIGRID encodes both color and shape information of superpixels while substantially reducing input dimensionality. We evaluate SIGRIDs on four benchmark datasets using two popular convolutional segmentation architectures. Our results show that, despite compressing the original data, SIGRIDs not only match but in some cases surpass the performance of pixel-level representations, all while significantly accelerating model training. This demonstrates that SIGRIDs achieve a favorable balance between accuracy and computational efficiency.

***Index Terms*—** Superpixels, Image Segmentation, CNNs.

1. INTRODUCTION

Image segmentation is a core task in computer vision [1, Chap. 10], enabling applications across diverse fields such as biomedicine [2] and remote sensing [3, 4], where images must be partitioned into compact, meaningful regions. With the rise of deep neural networks, their ability to extract rich information from visual data [1, Chap. 12] naturally extended to segmentation tasks [5]. However, these methods come at the cost of processing and storing vast amounts of data, which in turn requires expensive, high-end hardware.

A classical strategy to address this challenge is to transform pixel-based image representations into superpixel structures [6]. Superpixels group adjacent pixels with similar color or texture into perceptually meaningful regions, simplifying image representation while preserving important boundaries. Owing to their effectiveness for data reduction and abstraction, superpixels soon found applications within deep learning. In some works, they served as multiscale unsupervised feature extractors to enhance CNN-based segmentation algorithms [7, 8]. In others, they were incorporated directly into the model, for instance as image-dependent pooling layers [9], as tokenizers for transformer-based architectures [10], or as the basis for training graph neural networks [3, 11]. Finally, several studies have also proposed deep learning models designed specifically to compute superpixelations [12, 13, 14].

Despite these efforts, most approaches still depend on the design of novel and intricate neural architectures to fully exploit superpixel representations. As summarized in Table 1, this stems in part from the inherently irregular distribution of superpixels, which complicates their integration into deep learning models built on regular grid structures, such as CNNs. In doing so, these methods end up undermining the very motivation behind superpixels: to simplify not only the image representation but also the subsequent processing.

We introduce a new superpixel-based primitive, SIGRID, designed to replace full-resolution images in segmentation tasks. Our approach efficiently converts dense, pixel-based tensors into compact sparse tensors that encode only superpixel appearance and shape information. When used as input to popular segmentation CNNs, these compressed representations achieve performance comparable to, and in some cases exceeding, networks trained on the original images, while requiring only a fraction of the computational resources. The success of SIGRID arises from the fact that superpixels preserve natural image boundaries, offering a strong structural prior that guides the segmentation process. Finally, our work also contributes with a methodology that revives classical shape descriptor theory [1, Chap. 11], once central to computer vision but largely eclipsed by deep learned features, as a means for improving the performances of vision algorithms.

Table 1. Organizational primitives of image data.

Property	**Pixel**	**Superpixel**	**SIGRID**
Number of primitives	Many	Few	Few
Structure	Regular	Irregular	Regular
Compatibility with CNNs	Native	Challenging	Native
Semantic content per primitive	Low	High	High

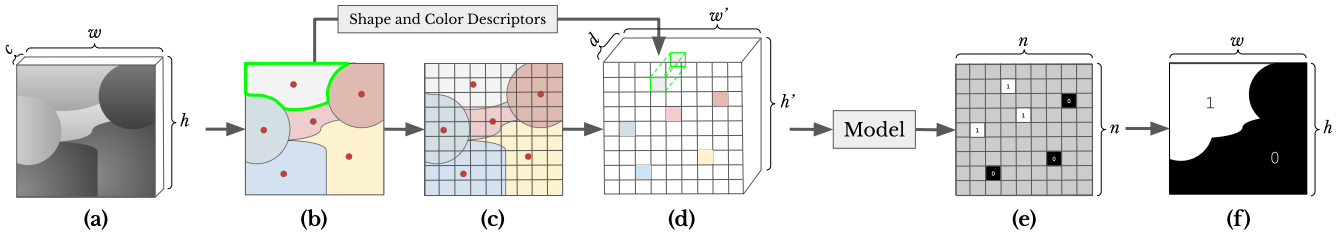

Fig. 1. SIGRID construction and model inference for binary segmentation. (a) Original image I. (b) Superpixelation S ($K = 6$ here) computed from I and superpixel centers (red dots). (c) An $w' \times h'$ grid superimposed on S where superpixels are assigned to grid cells. (d) Assigned grid cells are then populated with superpixel descriptors. (e) The model classifies each assigned grid cell in one of two classes. (f) Cell classes are converted back to pixel-level classification using the original superpixels.

2. METHODOLOGY

2.1. Overview and Motivation

Conventional pixel-level representations store only local color values and redundantly encode similar information across neighboring pixels, resulting in large and often wasteful memory usage. Their regular grid structure, however, is highly compatible with convolution-based operations. In contrast, superpixels compress boundary and appearance cues into a compact set of feature vectors, but their irregular organization makes them difficult to integrate directly into standard architectures such as CNNs.

We introduce SIGRID (Superpixel-Integrated Grid) a structured image representation that leverages superpixels' perceptually coherent regions to construct a compact and meaningful input grid. Each cell in this grid stores descriptors derived from an entire superpixel, including both appearance and shape-based features. This enables downstream models to process more semantically enriched and spatially compressed inputs, reducing computational overhead while preserving critical visual cues. Therefore, SIGRID both retains the compressed semantic prowess and size of superpixels, while enabling their use in convolution-based networks, as Table 1 summarizes.

2.2. SIGRID Construction

Figure 1(a)-(d) illustrates the method to derive a SIGRID $S \in \mathbb{R}^{d \times w' \times h'}$ from an image $I \in \mathbb{R}^{c \times w \times h}$ of c channels and its superpixelation $P \in \{1, \ldots, K\}^{w \times h}$ with K superpixels.

Our algorithm begins by computing the spatial center of each superpixel in P and overlaying an $w' \times h'$ grid on top of P. The grid size n is chosen as the smallest value such that no two centers fall within the same cell (with a small number of collisions allowed in practice; see Section 3). Each superpixel is then assigned to a grid location, and SIGRID S is constructed by placing the corresponding d-dimensional descriptors [15] into their assigned non-empty cells, while leaving the remaining cells zero-filled. These descriptors, discussed in more detail in Section 3, encode both color and shape attributes of each superpixel, thus capturing the essential information in the associated image region. In terms of computational overhead, SIGRIDs can be quickly superpixel generated using fast GPU-friendly algorithms [16] and descriptors computed with scattering techniques [17].

2.3. Network Training and Inference using SIGRIDs

When training an image segmentation model using SIGRIDs, we first convert the training ground-truth segmentations into SIGRID-like objects, i.e. $n \times n$ matrices with segmentation labels on grid locations containing superpixel cells. The remaining grid locations are given "empty" labels, which signal to our optimizer that losses (cross-entropy in the case of segmentation) won't be computed for those tensor coordinates.

We assign a segmentation label to a superpixel based on the majority pixel-level label contained within its boundary. As superpixels closely delineate the natural object borders in the image, pixels within a given superpixel will most likely fall entirely within a single image region, and label ambiguities inside superpixels are rare.

At inference time, the model predicts an $w' \times h'$ grid of binary labels. These predictions are expanded back to pixel space by assigning each superpixel's pixels the predicted label of the SIGRID cell it was assigned to. Figure 1(e)-(f) illustrate this process.

2.4. Comparative Analysis

SIGRID primitives provide a variety of technical and computational advantages to pixels and superpixels:

- They approximately preserve the spatial dispositions of superpixels, while arranging them on a regular grid.
- Because they are regular tensors, akin to images but smaller, they can be readily fed into standard CNNs while preserving their internal architecture.
- They make use of superpixel shape descriptors, which provide immediate structural image information to the network to process. Superxelations also provide natural edge guidance to segmentation networks.
- SIGRIDs are sparse by nature, which leads to memory savings compare to pixels.

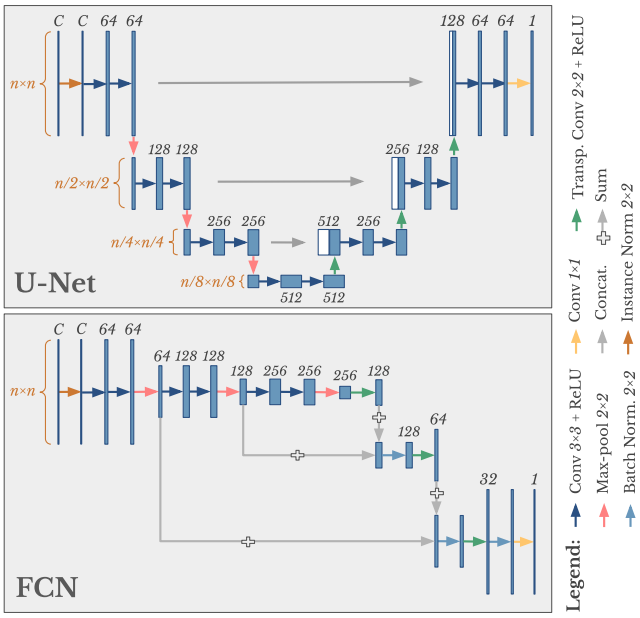

Fig. 2. Architectural details of the CNNs in our experiments.

3. RESULTS AND DISCUSSION

3.1. Implementation Remarks and Experimental Setup

Each image is segmented with a GPU-optimized implementation of the SLIC algorithm [6] [16], using a specified number of segments K and a compactness parameter that balances boundary adherence with region regularity. In practice, when K is large, we allow for a minimal amount of superpixel center collisions, where two or more centroids share the same SIGRID. In this case, we discard the smaller superpixel in area from the pipeline. To address tightly clustered centers when computing SIGRIDs, we also merge superpixels whose centroids are closer than a resolution-dependent threshold $\tau = \max(h, w)/n$, where h and w are the image dimensions and n is the target SIGRID resolution. Pairs with $d_{ij} < \tau$ are merged transitively, after which the segmentation is relabeled and centroids recomputed.

We efficiently compute superpixel descriptors using GPU-backed tensor scattering operations [17], which take values from a source tensor (an image in our case) and accumulate them into a target tensor at specific indices (given by our superpixelation). We implemented and tested the following superpixel descriptors:

- *Appearance-based:* Average RGB color (3 channels),
- *Shape-based:* Superpixel area, width, height, compactness, eccentricity, solidity (1 channel each) [1, Chap. 11], and Hu moments [15] (7 channels).

We evaluate SIGRID on four standard segmentation benchmarks: CUB [18], DUT-OMRON [19], DUTS [20] and ECSSD [21] using their official train/test splits. The images in these datasets have an average size of approximately 310×375 pixels. For both SIGRID and pixel-level training, we augmented the data with random rotation and horizontal/vertical flips. Augmentations are applied prior to any superpixel computation. We set $w' = h' = 80$ for all experiments.

We evaluate SIGRID using two popular convolutional architectures: the U-Net [2] and the Fully Convolutional Network (FCN) [22], whose designs follow the networks used in [23]. Figure 2 shows the implementation details of our tested networks. In both cases, the input of the model is first standardized using an Instance Normalization layer [24], which mitigates scale imbalances arising from its various descriptor values and ensures that they all contribute comparably.

Table 2. Effect of SLIC parameters on segmentation performance. Reported values are Test Pixel IoU, with MaxIoU in parentheses and the percentage of superpixel collisions in square brackets. Experiments use U-Nets trained on CUB with SIGRIDs and average color descriptors only.

	Number of Superpixels (K)					
Compactness	500	700	1000	1500	2000	3000
5	0.781 (0.915) [0%]	0.803 (0.924) [0.05%]	0.810 (0.931) [0.13%]	0.810 (0.936) [0.36%]	0.805 (0.936) [0.73%]	0.790 (0.920) [2.07%]
10	0.784 (0.922) [0%]	0.787 (0.929) [0.03%]	0.802 (0.935) [0.08%]	0.815 (0.939) [0.26%]	0.817 (0.939) [0.59%]	0.803 (0.924) [2.03%]
20	0.786 (0.925) [0%]	0.795 (0.931) [0.05%]	0.807 (0.937) [0.05%]	**0.818** **(0.941)** **[0.16%]**	0.806 (0.942) [0.42%]	0.820 (0.932) [1.92%]
100	0.789 (0.913) [0%]	0.813 (0.923) [0%]	0.816 (0.930) [0.01%]	0.803 (0.938) [0.02%]	0.818 (0.940) [0.12%]	0.819 (0.941) [1.35%]

Table 3. Effect of superpixel descriptors choice. We test Average Color (AC), Area (A), Width (W), Height (H), Compactness (C), Solidity (S), Excentricity (E) and Hu Moments (Hu). ✓ indicates the feature was selected in that configuration. For this superpixel setting, MaxIoU = 0.941.

Descriptors								Performance Metrics			
AC	**A**	**W**	**H**	**C**	**S**	**E**	**Hu**	**Cell Acc.**	**Pixel Acc.**	**Cell IoU**	**Pixel IoU**
✓	✗	✗	✗	✗	✗	✗	✗	95.50	95.31	0.824	0.818
✓	✓	✗	✗	✗	✗	✗	✗	95.61	95.46	0.826	0.820
✓	✗	✓	✓	✗	✗	✗	✗	95.70	95.54	0.830	0.824
✓	✓	✓	✓	✗	✗	✗	✗	95.71	95.54	0.829	0.821
✓	✗	✗	✗	✓	✓	✓	✗	95.71	95.55	0.831	0.824
✗	✗	✗	✗	✗	✗	✗	✓	92.22	92.19	0.710	0.699
✗	✗	✓	✓	✗	✗	✗	✓	92.13	92.09	0.705	0.698
✓	✗	✗	✗	✗	✗	✗	✓	**95.81**	**95.64**	**0.833**	**0.826**
✗	✗	✗	✗	✓	✓	✓	✓	92.93	92.86	0.732	0.722
✓	✓	✓	✓	✓	✓	✓	✓	95.55	95.38	0.826	0.819

Table 4. Comparison of Baseline and SIGRID variants for U-Net and FCN across CUB, DUTS, DUTS-OMRON, and ECSSD datasets. All results are for 100 training epochs. SIGRID models use 80×80 grids with average color and Hu moment features.

	CUB				**DUTS**				**DUTS-OMRON**				**ECSSD**			
	U-Net (Pixel)	U-Net (SIGRID)	FCN (Pixel)	FCN (SIGRID)	U-Net (Pixel)	U-Net (SIGRID)	FCN (Pixel)	FCN (SIGRID)	U-Net (Pixel)	U-Net (SIGRID)	FCN (Pixel)	FCN (SIGRID)	U-Net (Pixel)	U-Net (SIGRID)	FCN (Pixel)	FCN (SIGRID)
Pixel Acc. (%)	96.791	**95.641**	93.847	**93.678**	89.543	**88.281**	83.626	**85.122**	90.326	**91.096**	88.614	**88.908**	83.833	**83.093**	79.876	**80.666**
Pixel IoU	0.871	**0.826**	0.786	**0.768**	0.703	**0.665**	0.613	**0.605**	0.690	**0.682**	0.604	**0.625**	0.631	**0.619**	0.552	**0.589**
Pixel MaxF_β	0.873	**0.797**	0.752	**0.736**	0.626	**0.592**	0.484	**0.488**	0.631	**0.625**	0.585	**0.537**	0.670	**0.629**	0.568	**0.563**
Cell Acc. (%)	–	**95.810**	–	**93.759**	–	**88.520**	–	**85.314**	–	**91.554**	–	**89.347**	–	**83.420**	–	**80.907**
Cell IoU	–	**0.833**	–	**0.765**	–	**0.664**	–	**0.610**	–	**0.698**	–	**0.636**	–	**0.617**	–	**0.589**
Cell MaxF_β	–	**0.807**	–	**0.740**	–	**0.582**	–	**0.490**	–	**0.668**	–	**0.574**	–	**0.635**	–	**0.569**
Time (s/epoch)	89.58	**26.97**	82.61	**18.79**	74.62	**24.41**	63.72	**15.79**	28.92	**9.69**	23.89	**6.29**	5.54	**2.28**	4.60	**1.56**
GFlops	43.12	**7.24**	12.55	**2.14**	43.12	**7.24**	12.55	**2.14**	43.12	**7.24**	12.55	**2.14**	43.12	**7.24**	12.55	**2.14**

We evaluate segmentation quality at both the SIGRID-cell and pixel levels using three standard metrics: Mean Intersection over Union (IoU), accuracy, and the MaxF_β-score (with $\beta = 0.3$). RGB baselines, which process conventional $3 \times w \times h$ images, are evaluated only at the pixel level. For SIGRIDs, we additionally report MaxIoU, defined as the pixel-level IoU that would be obtained if all non-empty SIGRID cells were perfectly classified. This quantity represents the upper bound on pixel-level performance achievable under a given superpixelation. Since superpixels may not align exactly with ground-truth boundaries, we typically have $\mathrm{MaxIoU} < 1$.

All models were trained using binary cross-entropy, AdamW (learning rate 10^{-3}, weight decay 10^{-4}), for 100 epochs with batch size 32, and implemented in Pytorch [25][1] and trained on an NVIDIA A100 GPU with 40 GB of RAM.

3.2. Optimal SLIC Hyperparameters

To select the optimal SLIC hyperparameters (compactness and number of segments K), we tested various superpixel settings for SIGRIDs with just average color descriptors (Table 2). We test U-Nets trained on CUB for these experiments and those of the next section. Increasing segments beyond 1500 yielded marginal IoU improvements but significantly increased discarded superpixels. Thus, we chose $K = 1500$ segments and $\mathrm{compactness} = 20$, balancing minimal superpixel loss (0.16%) and strong segmentation accuracy (IoU: 81.82%). The suitability of this choice across datasets was confirmed with DUTS, DUTS-OMRON, and ECSDD, losing just 0.046%, 0.051% and 0.060% superpixels, respectively.

3.3. Effective SIGRID Channel Configuration

We next identified the most effective superpixel descriptors (Table 3). Using average color alone achieved high performance (pixel IoU: 0.818). Incorporating geometric or shape features offered minimal gains on top, with the addition of Hu moments yielding the highest IoU of 0.826. This result elucidates the importance of aggregating both superpixel appearance and shape data for performance. Our final SIGRID setting, therefore, combines average color and Hu moments.

3.4. SIGRID vs. Pixel Baseline Results

Table 4 compares SIGRID against pixel-based inputs for U-Net and FCN across the four datasets. SIGRID closely matched baseline segmentation performance, with IoU gaps typically under 0.05. Notably, SIGRID-FCN outperformed pixel-FCN on DUTS-OMRON and ECSSD (0.625 vs. 0.604 IoU and 0.589 vs. 0.552, resp.). Beyond accuracy, SIGRID delivers substantial efficiency gains. Training is up to $4\times$ faster, while computational cost (GFLOPs) is reduced by as much as $6\times$, highlighting a favorable trade-off between accuracy and efficiency. Preprocessing overhead remains negligible, with SIGRID construction taking under 0.1s per image across all datasets (specifically, CUB: 0.091s, DUTS: 0.074s, DUT-OMRON: 0.095s, ECSSD: 0.093s).

We attribute SIGRID's performance to two factors:

- Superpixels provide a strong edge guidance to segmentation, improving less complex networks such as FCN.
- Shape descriptors not only offset the detriment caused by the resolution reduction, but also give structural image data for simple models to predict the segmentation.

4. CONCLUSION AND FUTURE WORK

We introduce a novel image data primitive, SIGRID, which combines the semantic richness and compactness of superpixels with the regular structure required by convolutional networks for segmentation. Our results show that SIGRID achieves performance comparable to, and in some cases surpassing, pixel-level baselines, while vastly reducing training and inference time to a fraction of the cost. Looking ahead, we plan to exploit the inherent sparsity of SIGRID through sparse convolution operations to further accelerate CNN training [23] and extend these results to multiregion segmentation.

[1]The code used to generate the results can be found at `www.github.com/JackRobs25/SIGrid`

References

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Pearson, New York, NY, USA, 4th edition, 2018.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*. Springer, 2015, pp. 234–241.

[3] S. Dadsetan, D. Pichler, D. Wilson, N. Hovakimyan, and J. Hobbs, "Superpixels and graph convolutional neural networks for efficient detection of nutrient deficiency stress from aerial imagery," in *IEEE/CVF CVPR*, 2021, pp. 2950–2959.

[4] G. Dang, Z. Mao, T. Zhang, T. Liu, T. Wang, L. Li, Y. Gao, R. Tian, K. Wang, and L. Han, "Joint superpixel and transformer for high resolution remote sensing image classification," *Scientific Reports*, vol. 14, no. 1, pp. 5054, 2024.

[5] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE TPAMI*, vol. 44, no. 7, pp. 3523–3542, 2021.

[6] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

[7] S. He, R. W. H. Lau, W. Liu, Z. Huang, and Q. Yang, "Supercnn: A superpixelwise convolutional neural network for salient object detection," *IJCV*, vol. 115, no. 3, pp. 330–344, 2015.

[8] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *ECCV*. Springer, 2016, pp. 597–613.

[9] S. Kwak, S. Hong, and B. Han, "Weakly supervised semantic segmentation using superpixel pooling network," in *AAAI Conference on Artificial Intelligence*, 2017, vol. 31.

[10] A. Z. Zhu, J. Mei, S. Qiao, H. Yan, Y. Zhu, L.-C. Chen, and H. Kretzschmar, "Superpixel transformers for efficient semantic segmentation," in *IEEE/RSJ IROS*. IEEE, 2023, pp. 7651–7658.

[11] P. H. C. Avelar, A. R. Tavares, T. L. T. da Silveira, C. R. Jung, and L. C. Lamb, "Superpixel image classification with graph attention networks," in *SIBGRAPI*. IEEE, 2020, pp. 203–209.

[12] F. Yang, Q. Sun, H. Jin, and Z. Zhou, "Superpixel segmentation with fully convolutional networks," in *IEEE/CVF CVPR*, 2020, pp. 13964–13973.

[13] V. Jampani, D. Sun, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Superpixel sampling networks," in *ECCV*, 2018, pp. 352–368.

[14] H. Peng, A. I. Aviles-Rivero, and C.-B. Schönlieb, "Hers superpixels: Deep affinity learning for hierarchical entropy rate segmentation," in *IEEE/CVF WACV*, 2022, pp. 217–226.

[15] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE transactions on information theory*, vol. 8, no. 2, pp. 179–187, 1962.

[16] Algy, "fast-slic: High performance slic implementation," 2020, Available at: `https://github.com/Algy/fast-slic`.

[17] M. Fey, "Pytorch scatter," 2023, Version 2.x. Available at: `https://pytorch-scatter.readthedocs.io`.

[18] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," Tech. Rep. CNS-TR-2011-001, California Institute of Technology, Pasadena, CA, 2011.

[19] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang, "Saliency detection via graph-based manifold ranking," in *IEEE/CVF CVPR*, 2013, pp. 3166–3173.

[20] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, and X. Ruan, "Learning to detect salient objects with image-level supervision," in *IEEE CVPR*, 2017.

[21] J. Shi, Q. Yan, L. Xu, and J. Jia, "Hierarchical image saliency detection on extended cssd," *IEEE TPAMI*, vol. 38, no. 4, pp. 717–729, 2015.

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE/CVF CVPR*, 2015, pp. 3431–3440.

[23] B. Graham and L. Van der Maaten, "Submanifold sparse convolutional networks," *arXiv preprint arXiv:1706.01307*, 2017.

[24] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," in *arXiv preprint arXiv:1607.08022*, 2016.

[25] A. Paszke, S. Gross, F. Massa, et al., "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, vol. 32, 2019.