To be determined

Felipe Augusto Lima Reis
Audio-Visual Information Processing Laboratory
Pontifical Catholic University of Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
falreis@sga.pucminas.br

Raquel Almeida
Audio-Visual Information Processing Laboratory
Pontifical Catholic University of Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
raquel.almeida.685026@sga.pucminas.br

Silvio Jamil F. Guimarães
Audio-Visual Information Processing Laboratory
Pontifical Catholic University of Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
sjamil@pucminas.br

Simon Malinowski
Linkmedia, IRISA
Université de Rennes 1
Rennes, France
simon.malinowski@irisa.fr

Zenilton K. G. do Patrocínio Jrs
Audio-Visual Information Processing Laboratory
Pontifical Catholic University of Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
zenilton@pucminas.br

***Abstract*—**

I. Introduction

Image segmentation refers to the partition of an image into a set of regions representing meaningful areas. It is considered a challenging semantic task aiming to determine and group uniform regions for analysis. According to [1], to create an adequate segmented image it is necessary that the output presents some fundamental characteristics, such as: (i) region uniformity and homogeneity in its characteristics, such as gray level, color or texture; (ii) region continuity, without holes; (iii) significant difference to adjacency regions; and (iv) spacial accuracy with smooth boundaries and without raggedness.

Image segmentation is an active topic of research and in a typical approach could be divided in two stages [2]: (i) low-level analysis, which evaluate the pixel characteristics, neighboring relation and it is ideally uncommitted in terms of position, orientation, size and contrast; and (ii) high-level analysis, which maps the low-level characteristics to fulfill the task.

Recently, the deep learning approach drastically changed the computational paradigm for visual tasks. The main advantage of deep learning algorithms is that it does not require an engineered model to operate, meaning that they are capable of learning not only the features to represent the data but also the models to describe it [3]. Facing this new paradigm, researches initially replaced hand-engineered features in the low-level analysis by the features learned in deep models [4]–[6], which mostly achieve the desirable characteristics. More recently, many approaches have been proposed to explore the learned model for the high-level analysis, creating maps from the outputs of different layers in a deep learning network [7]–[10].

The authors are grateful to FAPEMIG (PPM 00006-16), CNPq (Universal 421521/2016-3 and PQ 307062/2016-3), CAPES (MAXIMUM STIC-AmSUD 048/14) and PUC Minas for the financial support to this work.

One challenge on the later strategy is how to combine those maps, considering that they are presented with different sizes and could represent different concepts. In this work it is presented a strategy to combine maps learned in a deep architecture and study the amount of maps necessary to create a viable region proposition for the task of image segmentation.

The remainder of this work is organized as it follows: in Section II it is presented the concepts categorizing the hierarchy of concepts in deep models. In Section III it is presented the related work exploring the deep model information for high-level tasks. In Section IV it is presented the reasoning and strategies proposed in this work. In Section V it is presented the dataset description, the experimental setup and results for the experiments. And, finally, in Section VI the conclusions are laid out.

II. Hierarchies in deep models

Deep learning approaches were initially described as black-box methods, meaning that not much were known about the reasoning and decisions of the created models. Much exertion have been applied to investigate the networks operation, whether by methodical experimentation [11]–[14] or visualization methods [15], [16]. Those efforts provided more clarity of the hierarchical aspects of the deep features, which allowed researches to explore these aspects in their endeavors. The hierarchies learned in deep models are categorized as complex concepts build from simpler ones. When applied for object recognition task for instance, the raw pixel on the input layer is learned as segments and parts until the composition of an object concept at the final layer.

These hierarchies could be particularly observed in convolutional networks, which are a stacked composition of three main layers, namely: (i) convolution; (ii) pooling; and (iii) non-linear activation. In [11] the authors directly assessed the hierarchy of concepts in convolutional networks, analyzing the knowledge representation and the network abstraction at each type of layer. The authors are capable to demonstrate the generic aspect at earlier stages and the specialization at later layers. The findings are conformed with the expected behavior of convolutional networks, but it is possible to observe that most of the learned abstraction is due the convolutional layers and that the pooling and non-linear layers rarely contribute for increasing the abstraction level.

III. Related work

In the earlier years of the deep learning resurgence, the proposal in [4] tackles the task of scene parsing—segmentation task applied for each pixel of the image, aiming to group pixels composing all the identifiable objects in the scene—using hierarchical trees and deep features alongside. Images are used as input for a convolutional network to extract deep features from multiple scales of the images, and in parallel to construct a segmentation tree representing in its nodes dissimilarities of neighboring pixels. The tree nodes are used to pool the correspondent deep features to be processed by a classifier. The classifier scores are used to create histogram of object classes for each node of the segmentation tree, and the final parsing proposal is built using the class entropy distribution for selecting the nodes that cover the entire image. This proposal uses hierarchical trees as auxiliary and external structures for the deep model.

In exploring the hierarchies within the deep model, three main architectures standout in recent years, namely: (i) Holistically-nested Edge Detection (HED) [17]; (ii) Convolutional Oriented Boundaries (COB) [9]; and (iii) Rich Convolutional Features (RCF) [10]. Those networks explicit explore the hierarchies by extracting side outputs of traditional convolutional networks to create boundary maps.

The HED approach applied the boundary maps for the boundary detection task, aiming to identify the limits separating uniform regions. The HED network create a side output layer at each stage of the VGG16 network [5], in which the stages are composed by two or three Convolution+ReLU layers followed by a Max Pooling layer. In HED, each side-output layer is associated with a classifier in a deeply supervised scheme [6], in which the fusion process is also inserted in the network, attributing weights for each side output that will be learned individually and determine its contribution on the final evaluation. The evaluation is performed by a cost-sensitive function to balance the bias towards not-boundary pixels. The HED network significantly improved the performance in multiple datasets and the extended version [7] also applied the network for the segmentation task.

The authors in [8] use the edge maps created by the HED network alongside with other features such as brightness, colors, gradient and variance to describe images. The goal of their proposal was to create an efficient framework to be used as real-time segmentation system, focused on a fusion strategy to update region features.

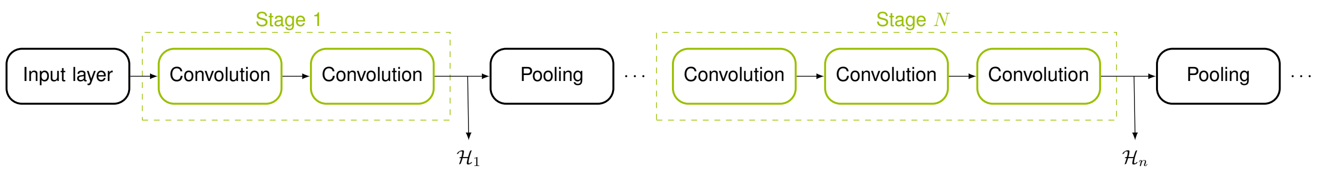
In the COB network, the authors also create edge maps from side activations, differing mainly from HED by the attribution to candidate contours the orientation information and weights representing the contour strength. The contour orientations are estimated by approximation to known polygon segments and are used to create segmentations hierarchies. The segments weights are computed based on the candidate contour neighboring region to measure the confidence that the candidate is a boundary line. The weights are thresholded to determine the granularity of the segment when creating the segmentation hierarchy. The network perform well in multiple tasks such object proposal, object detection, semantic contour and segmentation.

Finally, the RCF network applied in the boundary detection task, which differ from HED by three main modifications. The first modification regards the input layer, in which it is used pyramids to create multiple scales of the images. The scaled images are later interpolated in the output layer, similar to [4]. The second modification regards the number of side output maps, in which there is one at each Convolutial+ReLU layer of the VGG16 network. It is believed that this could create more detailed representations and improve the network accuracy. Following each side output, the merging is performed by a series of operations, namely a 1×1 convolution after each side output, followed by a element-wise sum for stage, an up-sampling and a loss function. The values are then concatenated and pass through a 1×1 convolution for the final evaluation. The last main modification in RCF is the loss function and the ground-truth of the datasets. In the ground-truth images, pixels are weighted based on a vote among multiple human-annotated values. Any pixel that do not achieve a confidence vote value is disregarded by the loss function in the network. The goal is to reduce inconsistencies in the fallible human annotations and mitigate the network confusion in controversial pixels.

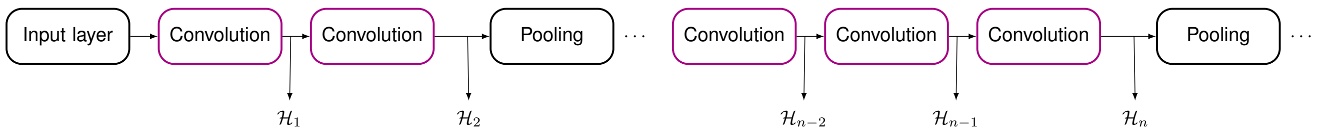
IV. Convolutional side-outputs for image segmentation

Hierarchies are long associated with the image segmentation task [18]–[22], to a degree that it improves a coherent organization of nested regions. The main motivation for using well-defined hierarchies is the idea of different detail level at different hierarchical level [2]. In this work, instead of using a well-defined hand-engineered hierarchical structure, it is proposed to explore the hierarchy of concepts resultant of the convolutional network dynamics.

It is expected that during the network training, the normal flow of the network would keep the desirable low-level features common for this type of network and in parallel, the side outputs at different layers would contain the region propositions presenting different level of details. The idea is to combine the the side output maps into a single proposition to be evaluated in the image segmentation task, driving the learning flow towards adequate regions for the task.



(a) Stage Layer Outputs (SLO)



(b) All Layers Outputs (ALO)

Fig. 1. Illustration for the two side outputs extraction strategies: (a) side outputs extracted at each stage of the network and (b) side outputs extracted at each convolutional layer

The first question on the use of hierarchical maps for the image segmentation task regards which and how many maps are adequate to create coherent proposals. In the ideal scenario, the side outputs would contain enough details to cope with the task, whilst maintaining the stability necessary for a steady training process.

In this work, it is proposed to evaluate two different strategies to extract the side outputs from the VGG architecture [5]. The first one, illustrated in Figure 1a, is inspired by the HED model which create one side output for each VGG stage. In this case, the maps amount to the number of pooling layers on the network. The second one, as illustrated in Figure 1b, is inspired by the RCF model, in which it is created one side output for each convolutional layer of the VGG architecture and theoretically would provide more refined details.

For simplicity, in the remaining of this work, the network using the side outputs extracted at each stage of the VGG will be called Stage Layer Outputs (**SLO**) and All Layers Outputs (**ALO**) for the side outputs extracted at each convolutional layer.

A. *Merging strategies*

The main question in dealing with hierarchies in deep models is how to combine them, considering that they are presented in different scales, could represent different concepts and are not strictly defined but a product of the network dynamics. In this work, the strategy to overcome those challenges is to combine the side outputs by exploring the knowledge of the learning process. To achieve that, it is proposed to explore simple merging functions that would enhance different desirable behavior.

To produce a single proposition to be evaluated, the following operations are proposed to merge the many side outputs:

- *ADD*: Sums all values aiming to balance negative and positive weights on the side outputs;
- *AVG*: Takes the average value aiming to combine the side outputs to create a proposition representing the whole network learning;
- *MAX*: Takes the maximum value aiming represent the most confident value at each region of the side outputs, ignoring low values.

The operations are performed element-wise on each side output after they are re-scaled for the input size, while maintaining the connectivity pattern.

V. Experiments

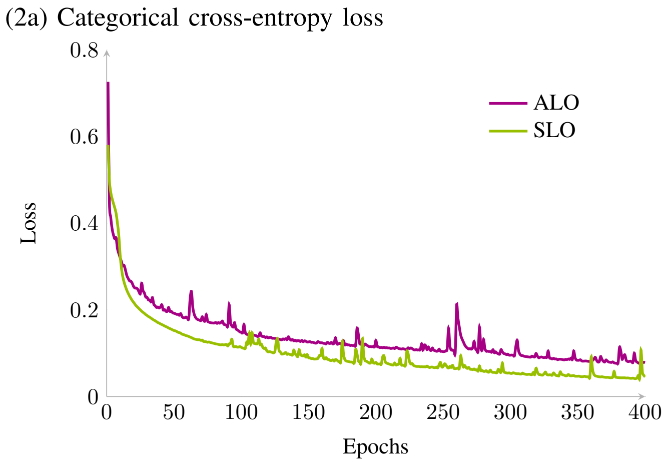
Experiments were conducted in the KITTI Road/Lane Dataset, part of KITTI Vision Benchmarking Suite [23]. The dataset contains images for road and lane estimation for the task of image segmentation. It is consisted of 289 training and 290 test images. The ground-truth is manually annotated for two different road types: (i) road, road area composing all lanes; and (ii) lane, lane the vehicle is currently driving on. It is important to notice that the ground-truth is only available for training set and the test evaluation should be performed using KITTI Server.

In this work, it is used only the road ground-truths and the lane annotations are ignored, due the fact that not all the images contains ground-truths for both categories. Then, we prefer to use the road estimation and build the classifier on a binary problem (road and background). The road type is divided in three different categories of road scenes, namely: (i) uu_road, urban unmarked; (ii) um_road, urban marked; and (ii) umm_road, urban multiple marked lanes.

To increase the number of images in the training set, it is performed some data augmentation procedures. It was added images with pepper/salt noise, horizontal flipping (mirror) and changes in contrast and brightness. Were avoided procedures that would create aberrations, such as the road in the sky and distortions that would change the nature of the objects in the scene, such as cars and pedestrians. These procedures resulted in 1445 images, divided in 1228 samples for training and 217 samples for validation (about 15%).

A. *Experimental setup*

Our network were build using using Keras [24] with Tensorflow [25] and trained for 400 epochs. We used a pre-trained VGG16 model to initialize the weights. Also, we use SGD



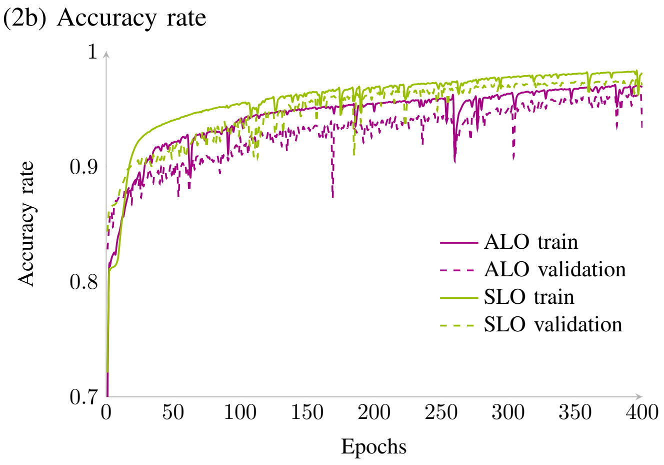


Fig. 2. Learning curves for the compared approaches. Left panel displays the the cross-entropy objective function during the learning step for the validation set. Right panel displays accuracy rate obtained on the training and validations sets during the learning step.

optimization with learning rate set to 1e-4, decay of 1e-6 and momentum of 0.95. The default batch size contains 8 images. Other experiments with different values will be discussed in next sections. All training experiments were performed in GeForce GTX 1080 8GB GPU. HED and RCF projects provided custom functions to balance the number of pixels of edges from the non-edges pixels. Once our problem is not as unbalanced as the edge detection, we decided to use the *categorical cross-entropy* loss function.

For simplicity, in the remaining of this work, the network using the side outputs extracted at each stage of the VGG will be called Stage Layer Outputs (**SLO**) and it is composed by $n = 5$ side outputs. Similarly, for the side outputs extracted at each convolutional layer, it will be called All Layers Outputs (**ALO**) and it is composed by $n = 13$ side outputs.

B. Training results

In Figure 2 it is presented the relevant curves obtained during the learning step for the proposed approaches. As one could see in Figure 2a, both compared approaches presents an expected loss curve and there is no significant difference between both approaches in terms of losses values, although the SLO model appears to be more stable and presents a faster decay than the ALO model.

Regarding the accuracy rate, illustrated in Figure 2b, it is possible to see that the SLO model presents a better performance than the ALO model. The accuracy rate achieved by SLO was 0.974 while ALO it was 0.963 on the validation set (about 1.2% worse). It is also possible to notice that the gap between the accuracy achieved in the training set and the accuracy achieved in validations set is smaller for the SLO model, which indicates that the ALO model is more prone to over-fit the data.

For the performance regarding time the average to process SLO model is 12.2% smaller than the ALO network, and could process 33.60 images per second in training time, while the ALO model process 29.48 images per second.

In summary, for all the metrics in the leaning step, the SLO model presented a slighted superior and more desirable behavior than the ALO model. It is believed that these results are consequence of the considerably larger amount of side outputs in the ALO model, which create more possibilities of interchangeability between confidant values.

In order to improve the results a new set of tests were performed using 2000 training epochs. The best accuracy rate achieve after the new training procedures by the SLO models was **0.980**. As for the ALO model, a more careful design of parameters were tested, particularly, we defined the learning rate as 1e-4, the decay as 1e-6 and used the Nesterov optimization in the process. After 46 epochs the model achieved the best accuracy rate of **0.982**. But the instable behavior persisted, in which in some epochs were close to this top accuracy but in many others the values were close to 0.86 accuracy. Some visual results are presented in Table II.

C. Evaluation results and comparison with the state-of-the-art

After the training procedure, we create a post processing step to reduce possible noises in results proposition. For this, we used the mathematical morphology operation of Opening [26]. This procedure removes small noises created by the foreground (the road)in the background. We defined a set of kernels with the sizes of 5×5, 7×7, 9×9, 11×11 and 13×13 and applied in the images to reduce different sizes of noises. Results using this strategy are under the label **ALO-mm** and **SLO-mm**.

Reminding that the test evaluation could only be performed using KITTI Server, the metrics provided are maximum F1-measure (MaxF), average precision (AP), precision (PRE), recall (REC), false positive rate (FPR) and false negative rate (FNR).

The results achieved on the test set according to each category in the road scenes are presented in Table I. As expected, the SLO model performed better then the ALO

TABLE I
KITTI BENCHMARK EVALUATION RESULTS FOR EACH CATEGORY

um_road

Method	MaxF	AP	PRE	REC	FPR	FNR
SLO	96.92%	87.36%	94.47%	**99.49%**	1.13%	**0.51%**
SLO-mm	**97.01%**	**87.68%**	**94.83%**	99.30%	**1.05%**	0.70%
ALO	96.39%	86.81%	93.87%	99.05%	1.25%	0.95%
ALO-mm	96.65%	87.51%	94.64%	98.74%	1.08%	1.26%
State-of-the-art	**97.05%**	**93.53%**	**97.18%**	**96.92%**	**1.28%**	**3.08%**

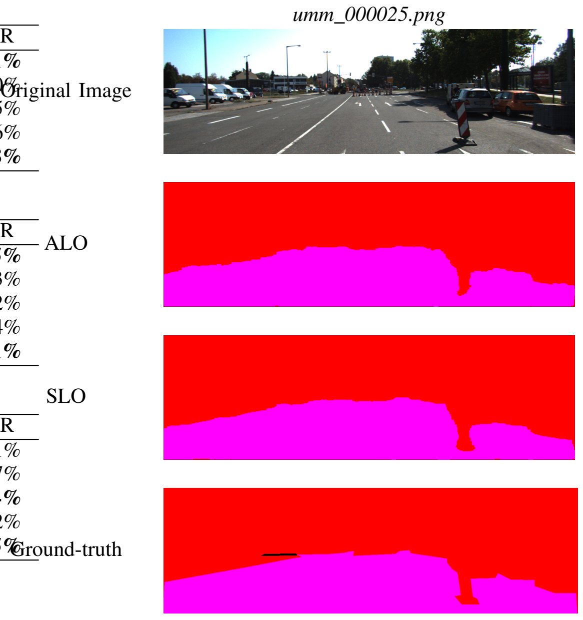
umm_road

Method	MaxF	AP	PRE	REC	FPR	FNR
SLO	97.57%	89.44%	96.05%	**99.15%**	1.24%	**0.85%**
SLO-mm	**97.61%**	**89.67%**	**96.30%**	98.97%	**1.16%**	1.03%
ALO	97.05%	88.83%	95.37%	98.78%	1.46%	1.22%
ALO-mm	97.21%	89.31%	95.90%	98.56%	1.29%	1.44%
State-of-the-art	**97.77%**	**95.64%**	**97.75%**	**97.79%**	**2.48%**	**2.21%**

uu_road

Method	MaxF	AP	PRE	REC	FPR	FNR
SLO	95.16%	**85.73%**	92.94%	97.49%	1.16%	2.51%
SLO-mm	**95.42%**	86.48%	**93.77%**	97.13%	**1.01%**	2.87%
ALO	94.70%	84.87%	92.00%	**97.56%**	1.33%	**2.44%**
ALO-mm	95.20%	86.15%	93.40%	97.08%	1.08%	2.92%
State-of-the-art	**95.95%**	**95.25%**	**95.25%**	**95.65%**	**1.21%**	**4.35%**

TABLE II
VISUAL ILLUSTRATION OF THE OBTAINED RESULTS



model in all almost all of the cases. Particularly when using the post processing procedure with mathematical morphology. It is also possible to notice that although the post-processing slightly improved the overall performance, it also increased the number of false negatives. This could be an indications that perhaps the applied kernel sizes are not adequate and are removing more of the foreground than the desired.

If compared with the state-of-the-art (anonymous submission on the KITTI Server platform), the proposed method is comparable and sometimes superior, regarding the maximum F1-measure and the recall metrics. This is due the fact that although the reported state-of-the-art on the dataset presents a superior average precision, it also almost always presents a higher rate of false positives an negatives. This indicates that the proposed methods are more precise in delineating the regions to be segmented.

VI. CONCLUSION

This work addressed the problem of merging side outputs extracted from the convolutional layer model VGG to create region propositions for the task of image segmentation. It was proposed to use a $max()$ function to enhance confident values during training to be evaluated using a cross-entropy loss function. It was also studied the impact that the number of side outputs have on the proposed strategy and if a simple mathematical morphology operation could enhance the performance on the task.

Experiments demonstrated that the $max()$ function is viable for merging maps with different sizes and connotations, and could place the proposed strategy among the state-of-the-art approaches for the task on the Kitti dataset. It was also demonstrated that a large amount of side outputs increases the network confusion during the training step, but could also create jumps that could lead to better performance, in terms of accuracy. The post-processing strategy slightly improved the performance, but requires further studies.

This research opens novel opportunities for study such as: (i) exploring different merging functions, less susceptible a values fluctuations; (ii) explore regularization techniques to sustain larger amounts of side outputs consistent; and (iii) insert the mathematical morphology kernels on the learning process to search for the best kernel size.

The code and a file containing all dependencies to reproduce the experiments is public available online in https://github.com/falreis/segmentation-eval.

REFERENCES

[1] D. Domnguez and R. R. Morales, *Image Segmentation: Advances*, 2016, vol. 1, no. 1.

[2] L. Guigues, J. P. Cocquerez, and H. Le Men, "Scale-sets image analysis," *International Journal of Computer Vision*, vol. 68, no. 3, pp. 289–317, 2006.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, vol. 1, 2016.

[4] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning Hierarchical Features for Scene Labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, Aug 2013.

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

[6] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-Supervised Nets," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Lebanon and S. V. N. Vishwanathan, Eds., vol. 38. San Diego, California, USA: PMLR, 09–12 May 2015, pp. 562–570.

[7] S. Xie and Z. Tu, "Holistically-nested edge detection," *International Journal of Computer Vision*, vol. 125, no. 1, pp. 3–18, Dec 2017.

[8] M.-M. Cheng, Y. Liu, Q. Hou, J. Bian, P. Torr, S.-M. Hu, and Z. Tu, "HFS: Hierarchical Feature Selection forEfficient Image Segmentation," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 867–882.

[9] K.-K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. Van Gool, "Convolutional oriented boundaries," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 580–596.

[10] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer convolutional features for edge detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 00, Jul 2017, pp. 5872–5881.

[11] R. Ilin, T. Watson, and R. Kozma, "Abstraction hierarchy in deep learning neural networks," in *International Joint Conference on Neural Networks, 30.* Anchorage, Alaska: IEEE Computer Society, 2017, pp. 768–774.

[12] C.-C. J. Kuo, "Understanding convolutional neural networks with a mathematical model," *Journal of Visual Communication and Image Representation*, vol. 41, pp. 406–413, 2016.

[13] D. Eigen, J. Rolfe, R. Fergus, and Y. Lecun, "Understanding deep architectures using a recursive convolutional network," in *International Conference on Learning Representations*. Banff, Canada: Computational and Biological Learning Society, 2014.

[14] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *International Conference on Learning Representations*. Toulon, France: Computational and Biological Learning Society, 2017.

[15] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *CoRR*, 2013.

[16] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision*, vol. 8689. ZURICH, Switzerland: Springer International Publishing, 2014, pp. 818–833.

[17] S. Xie and Z. Tu, "Holistically-nested edge detection," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[18] R. Jones, "Component trees for image filtering and segmentation," in *IEEE Workshop on Nonlinear Signal and Image Processing, E. Coyle, Ed., Mackinac Island*, vol. 9, 1997.

[19] J. Cardelino, G. Randall, M. Bertalmio, and V. Caselles, "Region based segmentation using the tree of shapes," in *2006 International Conference on Image Processing*, Oct 2006, pp. 2421–2424.

[20] P. Soille and L. Najman, "On morphological hierarchical representations for image processing and spatial data clustering," in *Applications of Discrete Geometry and Mathematical Morphology*, U. Köthe, A. Montanvert, and P. Soille, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 43–67.

[21] Y. Xu, T. Graud, and L. Najman, "Hierarchical image simplification and segmentation based on mumfordshah-salient level line selection," *Pattern Recognition Letters*, vol. 83, pp. 278 – 286, 2016.

[22] J. Cousty, L. Najman, Y. Kenmochi, and S. Guimarães, "Hierarchical segmentations with graphs: Quasi-flat zones, minimum spanning trees, and saliency maps," *Journal of Mathematical Imaging and Vision*, vol. 60, no. 4, pp. 479–502, May 2018.

[23] J. Fritsch, T. Kuehnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.

[24] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[25] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[26] L. Najman and H. Talbot, *Mathematical morphology: from theory to applications*, 2013.