CM3: Convolutional-Max and Mathematical Morphology to Image Segmentation

Felipe Augusto Lima Reis, Raquel Almeida, Silvio Jamil F. Guimarães, Zenilton K. G. do Patrocínio Jr
Audio-Visual Information Processing Laboratory (VIPLAB)
Pontifical Catholic University of Minas Gerais (PUC Minas)
Belo Horizonte, Minas Gerais, Brazil
{falreis, raquel.almeida.685026}@sga.pucminas.br, {sjamil, zenilton}@pucminas.br

***Abstract*—**
***Index Terms*—component, formatting, style, styling, insert**

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 relations 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]–[7], 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 [8]–[11].

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 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.

The remainder of this work is organized as it follows: Section II contains the related works; Section III describes the proposed method; Section IV shows the dataset description, the experimental setup and results for the experiments; and, finally, Section V concludes this paper.

II. Related work

Acho q a idia de multiescala no est diretamente relacionada a pegar partes da rede neural. Tanto q o RCF pega sadas laterais e ainda usa multiescala do lado de fora da rede para completar a tarefa. Se for falar de redes neurais multiescala, necessrio falar do MCG.

In the earlier years of the deep learning resurgence, an strategy in [4] (extended version in [5]), 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, to represent 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.

The proposal in [4] with an auxiliary hierarchical structure was one of the first strategies to extend the use of deep features to a complex task. It is important to bear in mind that 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 [12]–[15] or visualization methods [16]–[18]. Those efforts provided more clarity of the hierarchical aspects of the deep features, which allowed researches to explore these aspects in their endeavors.

In exploring the hierarchies of deep features, three main architectures stand out in recent years, namely: (i) Holistically-nested Edge Detection (HED); (ii) Convolutional Oriented Boundaries (COB); and (iii) Rich Convolutional Features (RCF). Those networks explicit explore the hierarchies

by extracting side outputs of traditional convolutional networks to create boundary maps which are also learned in the network.

To the best of our knowledge, the first network exploring this strategy was HED (extended version in [8]), which applied the boundary maps for the boundary detection task, aiming to identify the limits separating uniform regions. The HED network create an side-output layer at each stage of the VGG16 network [6], in which the stages are composed by two 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 [7]. The layers create edge maps, which are scaled and fused at the end, to be evaluate by a cost-sensitive function to balance the bias towards not-boundary pixels. The HED network significantly improved the performance in multiple datasets. The extended version also applied the network for the segmentation task. The authors in [9] 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 maily 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 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. RCF creates a side output at each Convolutial+ReLU layer of the VGG16 network, which is believed to create more detailed representations and improve the network accuracy. The last modification is in 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 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.

In [19] the authors pursued a similar direction of the afore mentioned networks. The proposal consists of joint strategies for the recognition task in large scale, specifically: NAO ENTENDI.

III. 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 [12]–[15] or visualization methods [16]–[18]. 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 [12] 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.

A. Merging strategy

Hierarchies are long associated with the image segmentation task, to a degree that it improves a coherent organization of nested regions. In this work, instead of hand-engineering the hierarchical structures of a typical approach, it is proposed a strategy to merge hierarchical maps created from the outputs at different layers in a convolutional network.

To merge the side outputs it is taken the result of the most confident side-output by using a $max()$ function on the side output maps. This is equivalent to trust only the most confident value, ignoring low values. This operation does not imply that all network is learning a task, but means that at least one has learned. Formally, the max operation for side outputs can be defined as follows: Let a set of side outputs $\mathcal{H} = \{\mathcal{H}_1, \mathcal{H}_2, ..., \mathcal{H}_n\}$. The fused map m could be defined as the Equation 1

$$m = \max_{1 \leq j \leq n} (\mathcal{H}_j) \tag{1}$$

The convolutional network model used in this work is the VGG network [6], proposed in 2014 as one of the first attempts to create deeper models for the task of object recognition. The architecture is a composition of multiple stacked convolutional layers. Following each two or three layers of convolution is placed a max-pooling layer. Also, all hidden layers are supplied with a ReLU non-linear rectification. Both HED network [8] and, later, RCF [11] were based in VGG for the task of edge detection by removing the final output of

the network and create side outputs combined in a new fused output.

Inspired by the HED model, we created side outputs for each VGG stage, as illustrated in Figure 1a, which amounts to the number of pooling layers on the network. Also, inspired by the RCF model, it is proposed to create one side output for each convolutional layer of the network, as illustrated in Figure 1b. The RCF also adopted a convolution of 1×1 in every stage of the network. In this work, otherwise, it is used the side output without any other combination, applying the $max()$ operation with the raw data from each layer.

IV. Experiments

A. Experimental setup

Our network were build using using Keras [20] with Tensorflow [21]. We used a pre-trained VGG16 model to initialize the weights. Also, we use SGD 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.

To increase the number of images in the training set, we made some data augmentation procedures. The techniques we used was pepper noise, horizontal flipping (mirror), changes in contrast and brightness. These procedures resulted in 1445 images, splited in 1228 samples for training and 217 validate samples (about 15%).

B. The Kitti Road/Lane dataset

KITTI Road/Lane Dataset, part of KITTI Vision Benchmarking Suite, contains images for road and lane estimation. Consists of 289 training and 290 test images . Ground truth is manually annotated for two different road types: road - road area (the composition of all lanes), and lane (the ego-lane; lane the vehicle is currently driving on) [22].

In this paper, we use only the road ground-truths and ignore lane annotations. Some images contains two different ground-truths, one for lane and other for road. Then, we prefer to use road estimation and build only one classifier. Also, it is important to say that ground truth is only available for training set. The test evaluation should be performed using KITTI Server [22].

The results in this paper were performed using KITTI Road Evaluation Benchmark, provided with KITTI Road Dataset.

C. Results

To compare the results of Stage Layer Outputs and All Layers Outputs, we trained the network with the ground-truths. This section, for comparison purposes, will call Stage Layer Outputs as SLO, and All Layers Outputs, ALO. This section contains data available for training for 400 epochs with parameters describer in Section IV-A.

Figure 1 shows the accuracy behaviour in the validation set for both networks. It is possible to see that SLO network has better performance than ALO network. The accuracy for SLO network was 0.97433 while ALO achieved 0.96278 (about 1.2% worse).

Figure 2 shows the loss behaviour for both networks. Once SLO had the better accuracy than ALO, the loss of was expected smaller, as shown in Figure 2.

For performance comparison, we provide Figure 3 and 4. Figure 3 shows that average time to process SLO network is 12.2% smaller than ALO network. Figure 4 shows that SLO can process 33.60 images per second in training time, while ALO process 29.48 images per second.

Once the experiments described above does not reached the best value, we kept training the network for more epochs, in a

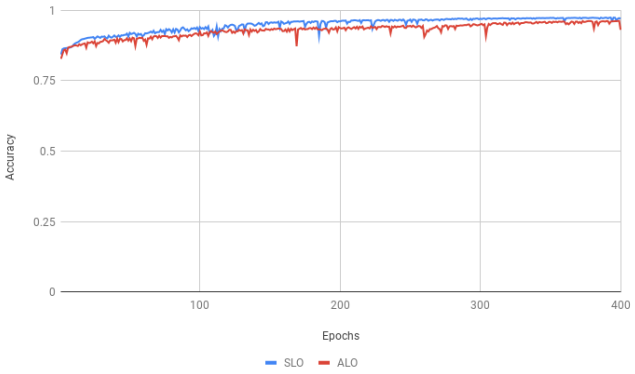


Fig. 1: Validation accuracy

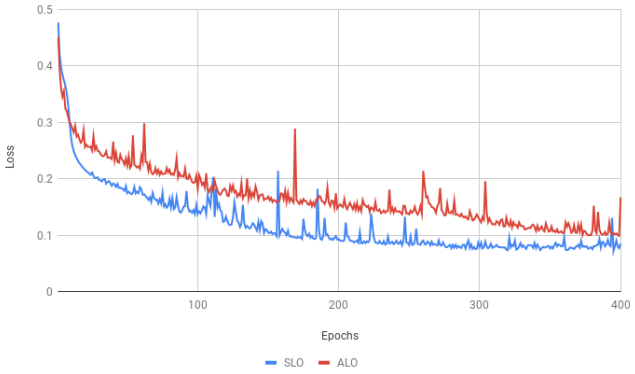


Fig. 2: Validation loss

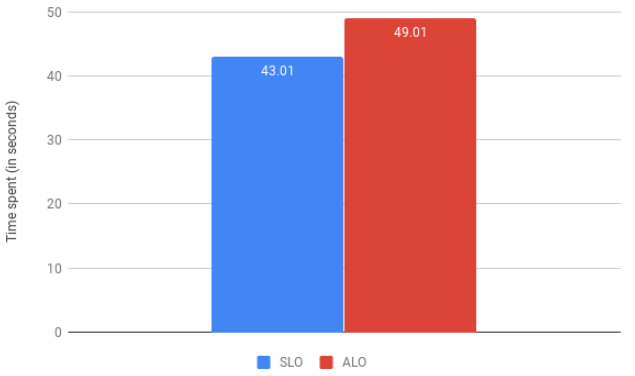


Fig. 3: Average training time

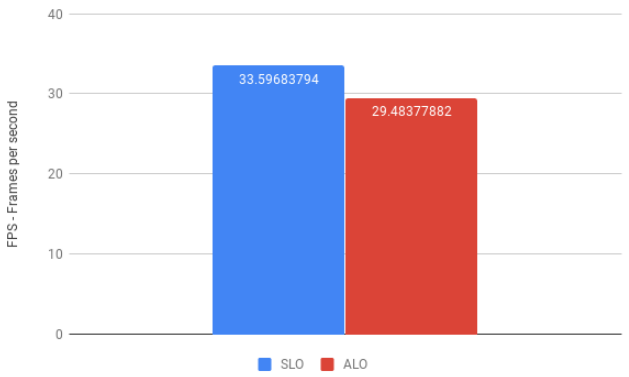


Fig. 4: Frames per second

new test. For SLO network the training, we trained for 2000 epochs using the parameters defined in Section IV-A. The best accuracy achieve after the training procedures was 0.98080.

ALO network was trained using diferent parameters. Once this network requires less carefully parameters, we defined learning rate as 1e-4, decay as 1e-6 and we used Nesterov optimization. We trained the network util overfitting and get the best value for the validation set. It was necessary 46 epochs to achieve the best accuracy value of 0.98225. Some epochs were close to this values but many ones were smaller, with values closes to 0.86 accuracy.

After the training procedure, we create a post processing step to reduce possible noises in results proposition. For this, we used the mathematical morfology operation of Opening. This procedure removes small noises that could exists in the images, affecting the quality of the results. We defined a set of kernels with the sizes of 5×5, 7×7, 9×9, 11×11 and 13×13 applied in the images to reduce different sizes of noises.

The results achieved by both networks, without and with Matematical Morfology post processing were evaluated with KITTI Road Benchmark. The results are available in Table I. It is possible to see that Opening operation resulted in a small gain for both networks in all classes evaluated.

The visual results are available in Table II.

Stage Layer Outputs - Without Matematical Morfology

Category	MaxF	AP	PRE	REC	FPR	FNR
um_road	96.92%	87.36%	94.47%	99.49%	1.13%	0.51%
umm_road	97.57%	89.44%	96.05%	99.15%	1.24%	0.85%
uu_road	95.16%	85.73%	92.94%	97.49%	1.16%	2.51%

Stage Layer Outputs - With Matematical Morfology

Category	MaxF	AP	PRE	REC	FPR	FNR
um_road	97.01%	87.68%	94.83%	99.30%	1.05%	0.70%
umm_road	97.61%	89.67%	96.30%	98.97%	1.16%	1.03%
uu_road	95.42%	86.48%	93.77%	97.13%	1.01%	2.87%

All Layers Outputs - Without Matematical Morfology

Category	MaxF	AP	PRE	REC	FPR	FNR
um_road	96.39%	86.81%	93.87%	99.05%	1.25%	0.95%
umm_road	97.05%	88.83%	95.37%	98.78%	1.46%	1.22%
uu_road	94.70%	84.87%	92.00%	97.56%	1.33%	2.44%

All Layers Outputs - With Matematical Morfology

Category	MaxF	AP	PRE	REC	FPR	FNR
um_road	96.65%	87.51%	94.64%	98.74%	1.08%	1.26%
umm_road	97.21%	89.31%	95.90%	98.56%	1.29%	1.44%
uu_road	95.20%	86.15%	93.40%	97.08%	1.08%	2.92%

TABLE I: KITTI benchmark evaluation results for in each category [1]

V. Conclusion

The code is public available online in https://github.com/falreis/segmentation-eval. The Anaconda environment file, containing all dependencies to reproduce the experiments is also available in the repository.

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, "Scene Parsing with Multiscale Feature Learning, Purity Trees, and Optimal Covers," in *29th International Conference on Machine Learning (ICML 2012)*, A. McCallum, Ed., Edinburgh, United Kingdom, Jun 2012, pp. 1–8.

[5] 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.

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

[7] 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.

[8] S. Xie and Z. Tu, "Holistically-nested edge detection," *Int. J. Comput. Vision*, vol. 125, no. 1-3, pp. 3–18, 12 2017. [Online]. Available: https://doi.org/10.1007/s11263-017-1004-z

[9] 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.

[10] K. Maninis, J. Pont-Tuset, P. Arbelez, and L. V. Gool, "Convolutional oriented boundaries: From image segmentation to high-level tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 819–833, April 2018.

[11] Y. Liu, 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)*, July 2017, pp. 5872–5881.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[1] Table abbreviations: *MaxF*: Maximum F1-measure, *AP*: Average precision, *PRE*: Precision, *REC*: Recall, *FPR*: False Positive Rate, *FNR*: False Negative Rate

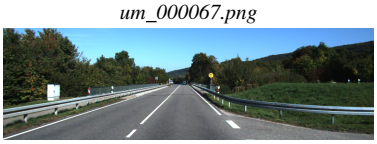
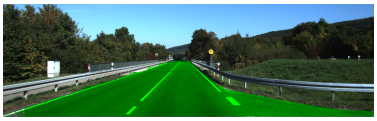
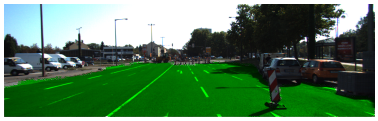
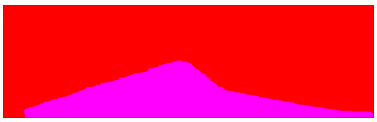
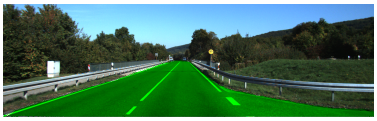
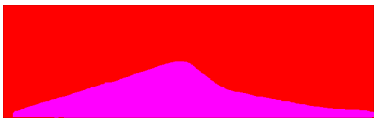
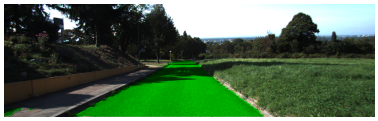
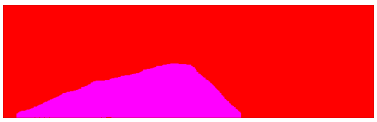
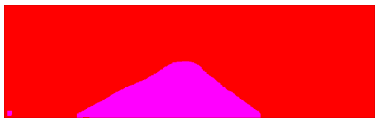
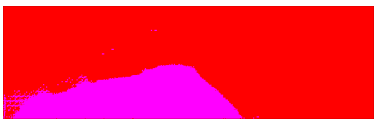
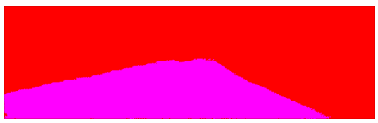
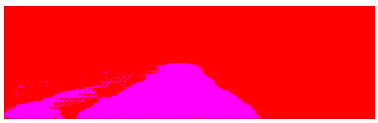
Train set			
	um_000067.png	*umm_000025.png*	*uu_000009.png*
Original Images			
ALO Marked Road			
ALO Ground-truth			
SLO Marked Road			
SLO Ground-truth			
Test set			
	um_000022.png	*umm_000078.png*	*uu_000041.png*
Original Images			
ALO Marked Road			
ALO Groud-truth			
SLO Marked Road			
SLO Groud-truth			

TABLE II: Visual results for image segmentation

[16] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *CoRR*, 2013.
[17] 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.
[18] B. Alsallakh, A. Jourabloo, M. Ye, X. Liu, and L. Ren, "Do convolutional neural networks learn class hierarchy?" *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 152–165, 2018.
[19] J. Fan, T. Zhao, Z. Kuang, Y. Zheng, J. Zhang, J. Yu, and J. Peng, "HD-MTL: Hierarchical Deep Multi-Task Learning for Large-Scale Visual Recognition," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1923–1938, Apr 2017.
[20] F. Chollet *et al.*, "Keras," https://keras.io, 2015.
[21] 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/
[22] 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.