arXiv:submit/2751858 [cs.CV] 1 Jul 2019

DeepTEGINN: Deep Learning Based Tools to Extract Graphs from Images of Neural Networks

Gustavo Borges Moreno e Mello
Dept. of Mech., Elec. and Chem. Engineering
Oslo Metropolitan University
Oslo, Norway
gustavo.mello@oslomet.no

Vibeke Devold Valderhaug
Dept. of Neuromedicine and Movement Science
Norwegian University of Science and Technology
Trondheim, Norway
vibeke.d.valderhaug@ntnu.no

Sidney Pontes-Filho
Dept. of Information Technology
Oslo Metropolitan University
Oslo, Norway
sidneyp@oslomet.no

Evi Zouganeli
Dept. of Mech., Elec. and Chem. Engineering
Oslo Metropolitan University
Oslo, Norway
evi.zouganeli@oslomet.no

Ioanna Sandvig
Dept. of Neuromedicine and Movement Science
Norwegian University of Science and Technology
Trondheim, Norway
ioanna.sandvig@ntnu.no

Stefano Nichele
Dept. of Information Technology
Oslo Metropolitan University
Oslo, Norway
stefano.nichele@oslomet.no

***Abstract*—In the brain, the structure of a network of neurons defines how these neurons implement the computations that underlie the mind and the behavior of animals and humans. Provided that we can describe the network of neurons as a graph, we can employ methods from graph theory to investigate its structure or use cellular automata to mathematically assess its function. Although, software for the analysis of graphs and cellular automata are widely available. Graph extraction from the image of networks of brain cells remains difficult. Nervous tissue is heterogeneous, and differences in anatomy may reflect relevant differences in function. Here we introduce a deep learning based toolbox to extracts graphs from images of brain tissue. This toolbox provides an easy-to-use framework allowing system neuroscientists to generate graphs based on images of brain tissue by combining methods from image processing, deep learning, and graph theory. The goals are to simplify the training and usage of deep learning methods for computer vision and facilitate its integration into graph extraction pipelines. In this way, the toolbox provides an alternative to the required laborious manual process of tracing, sorting and classifying. We expect to democratize the machine learning methods to a wider community of users beyond the computer vision experts and improve the time-efficiency of graph extraction from large brain image datasets, which may lead to further understanding of the human mind.**

***Index Terms*—neural network, deep learning, graph, segmentation, cellular automata, in-painting**

I. Introduction

One of the goals of systems neuroscience is to obtain a mechanistic model that describes and explains how network of neurons in the brain implements perception, thought, and behavior. Because structure implements function in biology, an unavoidable step towards these mechanistic models is to describe the structural connectivity of the network of neurons in the brain. Once a connectivity map (i.e., the description of the network) from a network of neurons is obtained, the it can be described as a graph or a cellular automata. This description may then be used as constraint to leverage methods from graph theory [1] to further analyze the network structure, or its functional complexity [2], which could ultimately contribute in three ways. First, by moving further the understanding on how to use biological substrates for computing [3], [4]. Second, by improving computing methods in AI by supplying biologically derived network structures that can be used as reservoir networks [5]. Finally by offering an mathematical abstraction of the biological system that can be used to compare cell cultures with genetic diseases with the healthy ones [6], providing ground for medical advancements.

Nonetheless, obtaining these matrices of structural connections between the nodes of the nervous system is a challenging and cumbersome endeavour. Especially from microscopy images (see Figure 1. Although some methods have been recently developed to automatize the process, they still rely on basic image processing steps that demand substantial time to find suitable parameters and curate the results [7]. Furthermore, these automatic methods are not very robust, and as an effect the gold standard approach still is to trace these connections manually.

Additionally, and unlike other biological substrates from which networks can be extracted, the brain is constituted of billions of neurons, with diversified morphology, and a few orders of magnitude higher number of connections (i.e., synapses) [8]. This additional complexity implies that different nodes in a graph network should have different properties and represent different cell types or structures in the brain. Because this structural information is critical to understand the brain, it is of utmost importance that automatic tools take them into account, which is not currently available to the best of our knowledge.

Furthermore, experimental constraints (e.g., multi-electrode arrays, patching pipettes) frequently obstruct the view of part of the image (black lines in Figure 1), consequently preventing the connection of nodes that otherwise would be connected. Although these gaps can be easily handled through human intervention (i.e., by estimating the edges that connect two nodes), no simple image processing method can properly handle this problem as the data in the obstructed area is missing. To cope with this problem, one must be able to reconstruct the missing data by inferring how it would look like based on the surrounding area and what is typically known about the morphology.

Modern machine learning techniques that leverage the power of convolutional neural networks (ConvNets) may be used to automatize many of the steps mentioned before. In-painting algorithms can be used to estimate missing data caused by image obstruction, object detection algorithms can employed to locate and classify diverse structures in the brain, unsupervised segmentation algorithms can be leveraged to extract skeletonized versions of the image, just to name a few. This would allow for a comprehensive graph extraction from image in neuroscientific settings. The challenge regarding employing ConvNets is that due to its novelty and complexity, deep learning methods are not widely available to non-specialists in computer vision. Additionally, most of these methods require training, which by itself is generally poorly documented, preventing non-expert in computer vision from experimenting and benefiting from ConvNets in their neuroscientific research.

Motivated by the foregoing shortcomings, we present a deep learning based toolbox to extract graphs from images of brain tissue. This toolbox is a framework constituted by an extensible library of methods that can be integrated into a computer vision pipeline. The library is based on a combination of standard image processing algorithms available in OpenCV [9] and SKimage [10], and deep learning based methods for object detection, image/line segmentation, in-painting and style transfer implemented in Pytorch [11]. Additionally, the toolbox has a graphical user interface (GUI) that simplifies the steps of assembling the graph extraction pipeline, which includes the training of the supervised machine learning algorithms.

The main contribution of this paper is to democratize deep learning based methods for computer vision to the neuroscientific community through a reusable, flexible and scalable tool. Through this toolbox, we hope to make deep learning methods more widely accessible to neuroscientists.

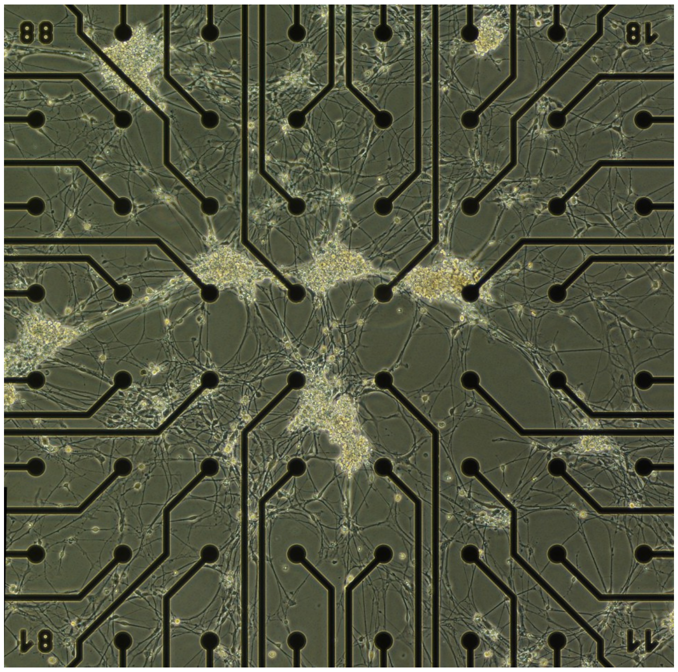

Fig. 1. Raw image example as acquired from the microscope. The black lines ending in circle are ”blind-spots” created by the multi-electrode array.

II. Image acquisition

The images were prepared as follows: Human cortical neural networks were differentiated and matured from iPSC-derived NSCs (ax0019, Axol bioscience), and fluorescently labelled using a two-color LIVE/DEAD viability/cytotoxicity kit (MP03224, Invitrogen). 0.8ul Ethidium homodimer-1 (2mM in DMSO/H2O 1:4) and 0,4ul Calcein AM (4mM in anhydrous DMSO) was diluted in 2ml PBS and applied to the neural networks for 15 minutes in 37°C. The former produces an intense red fluorescence in dead or dying cells, while the latter produces an green fluorescence in live cells. The fluorescently labelled neural networks were imaged with a 10X objective using a automated EVOS 2 fluorescence microscope.

Each multi-electrode array (MEA) cell culture chamber was briefly sterilized using ethanol, washed with water, UV-treated over-night, and hydrophilized by application of foetal bovine serum for 30-60 minutes at room temperature. The surface was subsequently double-coated using poly-L-ornithine (0,01%) and laminin. The appropriate neuronal cell culture media were heated to 37°C and used to create a single-cell suspension, from which 100.000 cells were seeded directly onto the electrode area of each MEA in a dropwise manner. For some cultures, a feeder-layer of astrocytes (5000 per MEA) was first established, upon which 50 000 neuronal cells were seeded onto. The MEA neuronal cultures were kept in a standard humidified air incubator (5% CO2, 20%O2, 37°C), and 50% of the media were changed every 2-3 days. Phase contrast images were acquired at various stages of neuronal differentiation and maturation on the MEAs using the laboratory light microscope Carl Zeiss Axiovert 25 with 5 and 10X objectives.

III. The pipeline

The kernel of the toolbox is the graph extraction pipeline. It enables visualization, correction and analysis of the structures depicted in the input image. This pipeline, is constituted by an ordered sequence of methods, which will output a graph representation of the network from the input image. Additionally, some of the steps of the pipeline sequence require preparation (i.e. training). The obtained graph provides weights, edge lengths and node type, which should reflect anatomical structure.

The default pipeline combines the following steps: pre-processing, structure detection, segmentation, thinning, graph extraction, graph pruning, and training. What follows is a high-level description of the steps.

A. Pre-processing

Pre-processing involves doing image transformations that allows the subsequent algorithms to perform more robustly. There is a set of image processing steps that may be employed interchangeably. Most of them are standard image transformations like color space change and filtering (including sharpening and blurring), widely available through OpenCV and SKimage libraries. Additionally, deep learning based algorithms were included, namely: style transfer [12] and in-painting [13]. These two methods rely on VGG16 networks pre-trained on ImageNet dataset [14] and must be further fine-tuned to properly work with the dataset from the experiments (see training). The addition of in-painting(Figure 2 D) and style transfer (Figure 2 E) enables coping with data lost caused by obstruction of the field of view during experiments, and differences in imaging settings respectively (See black marks in Figure 1).

B. Structure detection

In order to detect nodes that belong to different cell types, the object detection algorithm, yolov3 (You Only Look Once, version 3.0) [15], was included in the pipeline. This method operates by detecting combinations of spatial features in the image, locating their position and area, and classifying them under a predefined category (namely: astrocytes, neurons and clusters of neurons; see Figure 2 F) with an explicit probability. This algorithm requires training, and substantial amount of data to be trained (see training). The center of these detected areas is used by later steps in the pipeline to discriminate synaptic nodes from cell body nodes.

C. Segmentation

We provide two interchangeable avenues for segmentation. Guided watershed [16] and W-Net [17]. We noticed that depending on the characteristics of input image these algorithms perform the best. The main goal of this step is to separate the structures that compose the network from everything else and compose a mask.

D. Thinning

The next step is to skeletonize the mask so no pixel in the mask has two or more neighbor pixels that belongs to the mask and are neighbor to each other (see Figure 2 H, yellow lines). To do that, we implemented the improved Zhang-Suen Thinning algorithm [18]. This method was chosen because it produced less artifacts in the intersection of lines (blobs and missing pixels).

E. Graph extraction

Once obtained the skeletonized image we then detect the positions of nodes and the edges that connect them so we can create a graph. The graph is generated through the NetworkX [19] library. To detect the nodes, we filter the image with series of 3x3 filters. Each filter represent one possible scenario for a node where the center pixel belongs to the filter and 1 or at least 3 other neighbours also belong to it and are not neighbours to each other. This guarantees that intersection nodes and end-of-the-line nodes are contemplated, but that points that belongs to lines are ignored.

Edges between nodes are detected by the following steps. Firstly, the skeleton is segmented in edges by removing the node pixels from the skeletonized mask, each one of these edges has its own label automatically defined as 1 to the number of available edges. Secondly, these segmented edges are dilated. Thirdly, the edges that overlap with two nodes are added to the graph as a bidirectional edge. Finally, the overlapping segmented edge is removed from the set of possible edges. The process repeats until no edges are left (see Figure 2 H blue lines).

If the Structure Detection step has been executed successfully, the nodes which are closest to the center of the regions of interest generated by the object detection algorithm will acquire the category of the identified object (eg., neuron, cluster).

This type of representation of the network in graph is analogous to a connectivity estimation, which is highly relevant in neuroscience to infer functionality. Hence the graph extraction method can contribute to connectivity analysis as performed by Maccione et al. [2] and Ullo et al. [20], but without the cumbersome and time-demanding step of extracting the connectivity map manually.

F. Graph pruning

As the last step, it is given to the user the opportunity to edit the graph extracted by removing or adding new edges, tracing new edges between them and assigning properties to each node.

G. Training

In order to use the methods that depend on supervised learning, the user has to provide first a set of good examples so the algorithm can be properly trained. This process is usually poorly documented and the data format that the algorithm should receive is usually obscure. We make this step explicit, by declaring exactly what should be the data format, and

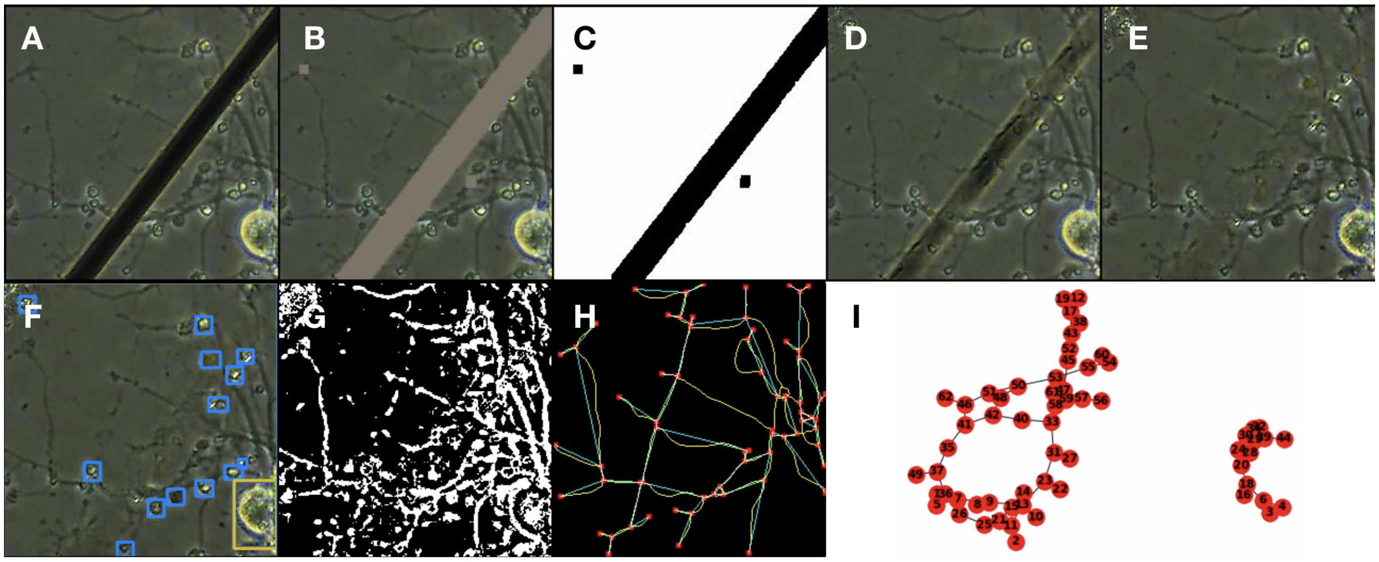

Fig. 2. Graph extraction example. A- 256 x 256 Crop of a raw image as acquired from the microscope. B- Segmentation of the electrode area in grey. C- Mask of the area to be in-painted. D- Intermediate step in the inpainting process. E- End result of the in-painting algorithm. F- Cells identified by type with yolov3, blue boxes are neurons, yellow box is a cluster of cells, astrocytes are omitted. G- Intermediate step of the structure segmentation. H- Graph extraction example, the red represents the nodes, the blue are the edges connecting the nodes and the yellow is the thinned (skeletonized) version of the network identified. I- Graph output describing the network in H.

providing a simple interface that the users can use to generate the training data themselves and train the model.

H. Training in-painting

To train the in-painting [13] network, we need a set of ground truth images, and a set of masks that would match in shape and size the typical artifact obstructing the original image. To generate this data, we segmented the dark areas of original raw image using a simple threshold. We dilated the segmented areas with a 5x5 circular kernel to include the edges of the artifact areas. We randomly cropped the mask image in images of 256x256 pixels. The images in which 1/4th of the area was occupied by the electrode mask were selected for the mask pool. Each mask was then copied 35 times and rotated cumulatively by 10 degrees until we had 36 versions of the same mask in all orientations. To extract the ground-truth images, we cropped patches of 256x256 pixels from the original image where no pixel in the patch overlapped with the cordinates of a pixel belonging to a mask. To expand the dataset, the selected patches were flipped and rotated 90, 180 and 270 degrees.

I. Training object detection

To train the Yolov3, we defined regions of interests (ROIs) by drawing a bounding boxes from a subset of patches (100 images, randomly picked). This ROIs are constrained by the vertical and horizontal coordinates of its centroid and its height and with as ratios of the original image. Each ROI is labeled as an instance of a class of objects.

In our dataset, we defined three labeled structures: neurons, astrocytes and cluster of neurons. Only neurons and cluster of neurons were relevant for the graph extraction, thus only these two labels are displayed. The labeling of astrocytes was required to prevent falsely detecting astrocytes as neurons. Once training data is available, training follows by pointing the location of the data in the storage unit and running the training function.

Because the amount of data was limited. The network was pre-trained with the COCO dataset [21] to learn and then fine-tuned and cross-validated using the labeled images. Our implementation of Yolov3 operates by predicting 3 boxes in 2 different scales. Thus, the tensor is N x N x[2*(4+1+3)], where is for the 4 bounding box offsets, 1 for objectness prediction, and 3 is for the class predictions. Furthermore we chose 6 clusters in the k-means algorithm to establish our binding box priors. In our dataset the 6 clusters were (7x9), (15x16), (22x19), (31x32), (55 x 49), (89x91).

The training progress was displayed on every set of epochs, which could defined by the user, and it could be interrupted at any time. The set of weights with the smaller error was highlighted to facilitate the use of the pipeline.

IV. Discussion

Many solutions exist to extract network graphs from images, including some generic flexible tools. But these tools assume that the network to be extracted is homogeneous (i.e., all the nodes are equal). This is a major problem in neuro-science because biological neurons form highly heterogeneous networks. In particular, the tools available cannot account for the difference between neurons and synapses as nodes. Additionally, they cannot account for differences between cell types (i.e., neurons vs. glia). This is a major source of error for describing a network. The abundance of false positives can lead to a description that is much bigger and dense, hence increasing the level of complexity which by itself increases the challenge of analysis. Our toolbox circumvent this problem by integrating machine learning methods into an easy to use pipeline to extract graphs from network of neurons. Because the algorithm detects objects by category, further developments may be implemented to extract sub-populations of neurons and include more cell types. One possible avenue is to develop a specific dataset for brain cell-type detection, which is currently unavailable in the best of our knowledge.

A second major challenge in the path of automatizing graph extraction from images of cultured cells is that these images often come with major artifacts (e.g., electrodes, pipettes, objects that obstruct the view). We eliminate these artifacts by combining in-painting techniques and style transfer through deep learning methodologies. Although not perfect, we demonstrate that both techniques can provide qualitatively satisfactory results. Allowing to reconstruct a plausible network, despite the artifact. One further point of development could be to apply techniques to improve the resolution, as it may increase the performance of in-painting and style transfer techniques.

We anticipate that this toolbox will enable neuroscientists to extract graphs from network of neurons in a more time-efficient way and consequently contribute in the pursue of the understanding of perception, intelligence and behavior.

Acknowledgment

This work was supported by Norwegian Research Council SOCRATES project (grant number 270961) and received internal support as a lighthouse project in Computer Vision from the Faculty of Technology, Art and Design (TKD) at Oslo Metropolitan University, Norway.

Repository

Code and example data can be found in the following repository: https://github.com/gmorenomello/deepteginn

References

[1] M. M. Hassan and G. L. Hogg, "A review of graph theory application to the facilities layout problem," *Omega*, vol. 15, no. 4, pp. 291–300, 1987.

[2] A. Maccione, M. Garofalo, T. Nieus, M. Tedesco, L. Berdondini, and S. Martinoia, "Multiscale functional connectivity estimation on low-density neuronal cultures recorded by high-density cmos micro electrode arrays," *Journal of neuroscience methods*, vol. 207, no. 2, pp. 161–171, 2012.

[3] H. Broersma, J. F. Miller, and S. Nichele, "Computational matter: Evolving computational functions in nanoscale materials," in *Advances in Unconventional Computing*, pp. 397–428, Springer, 2017.

[4] P. Aaser, M. Knudsen, O. H. Ramstad, R. van de Wijdeven, S. Nichele, I. Sandvig, G. Tufte, U. Stefan Bauer, Ø. Halaas, S. Hendseth, *et al.*, "Towards making a cyborg: A closed-loop reservoir-neuro system," in *Artificial Life Conference Proceedings 14*, pp. 430–437, MIT Press, 2017.

[5] S. Nichele and A. Molund, "Deep reservoir computing using cellular automata," *arXiv preprint arXiv:1703.02806*, 2017.

[6] I. Sandvig, I. L. Augestad, A. K. Håberg, and A. Sandvig, "Neuroplasticity in stroke recovery. the role of microglia in engaging and modifying synapses and networks," *European Journal of Neuroscience*, vol. 47, no. 12, pp. 1414–1428, 2018.

[7] M. Dirnberger, T. Kehl, and A. Neumann, "Nefi: Network extraction from images," *Scientific reports*, vol. 5, p. 15669, 2015.

[8] E. R. Kandel, J. H. Schwartz, T. M. Jessell, D. of Biochemistry, M. B. T. Jessell, S. Siegelbaum, and A. Hudspeth, *Principles of neural science*, vol. 4. McGraw-hill New York, 2000.

[9] B. Gary, "Opencv is an open-source, computer-vision library for extracting and processing meaningful data from images.," *Dr. Dobbs*, no. 25, pp. 120–126, 2000.

[10] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "scikit-image: image processing in python," *PeerJ*, vol. 2, p. e453, 2014.

[11] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," *Open review preprint https://openreview.net/pdf?id=BJJsrmfCZ*, 2017.

[12] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[13] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 85–100, 2018.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[15] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[16] V. Osma-Ruiz, J. I. Godino-Llorente, N. Sáenz-Lechón, and P. Gómez-Vilda, "An improved watershed algorithm based on efficient computation of shortest paths," *Pattern Recognition*, vol. 40, no. 3, pp. 1078–1090, 2007.

[17] X. Xia and B. Kulis, "W-net: A deep model for fully unsupervised image segmentation," *arXiv preprint arXiv:1711.08506*, 2017.

[18] W. Chen, L. Sui, Z. Xu, and Y. Lang, "Improved zhang-suen thinning algorithm in binary line drawing applications," in *2012 International Conference on Systems and Informatics (ICSAI2012)*, pp. 1947–1950, IEEE, 2012.

[19] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[20] S. Ullo, T. R. Nieus, D. Sona, A. Maccione, L. Berdondini, and V. Murino, "Functional connectivity estimation over large networks at cellular resolution based on electrophysiological recordings and structural prior," *Frontiers in Neuroanatomy*, vol. 8, p. 137, 2014.

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.