doi:10.1017/S143192761800315X
Microsc. Microanal. 24 (Suppl 1), 2018

Dragonfly as a Platform for Easy Image-based Deep Learning Applications

Ruslana Makovetsky[1], Nicolas Piche[1], Mike Marsh[2]

[1.] Object Research Systems. Montreal, Canada.
[2.] Object Research Systems. Denver, USA.

Deep learning (DL) refers to a class of model-based computational techniques for processing data and solving problems, often with results that deliver higher accuracy compared to human experts executing the same tasks. Adoption of DL solutions has accelerated in recent years in part because of success in domains that have been otherwise intractable with traditional algorithms and in part because of the rapidly growing speed of high-performance DL engines. Here, we apply the popular DL engine TensorFlow [1] to image processing tasks and make it user-friendly for microscopists and other image scientists to apply convolutional neural networks (CNNs) in our Dragonfly software platform. We discuss applications in image filtering and image segmentation.

DL is well-suited for the class of problems that human experts solve intuitively, but for which computer scientists have difficulty encoding in to standard algorithms. DL works by taking sample outputs that can be used as training data for a CNN to "learn" how to transform the associated input data. The resulting trained network can then be applied to unseen inputs in order to generate valuable outputs. For example, CNNs can be trained to take grayscale images as input and produce an image segmentation.

For this work, we have adopted the TensorFlow DL engine [1] along with the Keras Python framework [2], a modular and extensible toolkit for TensorFlow applications. The DL workflow typically obeys the following schema. A CNN expert designs the network topology and encodes it as a Keras CNN. After connecting input data and output data to the CNN, the user iterates through a workflow where the network is trained, results are inspected, and then the network parameters (and possibly network topology) are adjusted before retraining. After iterations of CNN optimization, the result is a model that can reproduce outputs similar to the outputs used for training. That CNN can then be applied to previously unseen data. Using the CNN to convert inputs into outputs is called inference, and inference of a CNN is a high-performance task that executes quickly on CPUs and even faster on GPUs.

Dragonfly takes Keras-format Python-encoded CNNs and gives users an interface for connecting inputs and outputs, tuning parameters, iterating training, and inferring the trained CNN on new inputs. Because Dragonfly has rich tools for image segmentation and image visualization, the integrated platform makes it much simpler for CNN experts to create training data and execute their CNN development workflows. Furthermore, the software platform makes it easy for non CNN experts to share CNNs, retrain CNNs, and apply them; applying CNNs in Dragonfly is no different than applying any other image filter in the software's Image Processing Toolbox.

One of the simplest DL applications to understand is the Autoencoder, which is a CNN that takes an input image, passes it through layers that encode the essence of the image in a reduced dimensionality encoding, and then pass it through layers that decode in order to restore the original image. By encoding and then decoding the image with a CNN, only the salient features are restored. Because the noise is not reconstructed, the Autoencoder is an image compression technique that behaves as a noise mitigation filter. We report the performance of Autoencoders in Dragonfly here to be comparable to mainstream denoising filters.

When users have large datasets that are too tedious to segment manually and intractable by automated segmentation routines, a CNN can be used as an image filter to enhance the contrast of the features that must be segmented. For example, by training a CNN with some images curated manually, we easily developed a filter which greatly enhances the image features required for segmentation. We show both cortical pore emphasis in bone microCT and plasma membrane emphasis in serial section TEM of Drosophila neurons [3].

Despite the rapid advancement of DL tools in many domains, progress has been constrained by the absence of a feature-rich platform for CNN developers to validate behavior in image applications. Because Dragonfly lets users not only iterate development, but also deploy the inference engine, users can now share their CNNs easily with collaborators. This is a big advance in taking arcane DL development tools and putting them directly into the hands of microscopists for easy adoption [4].

References:

[1] M Adabi *et al*, 12th USENIX Symposium on OSDI (2015), pp265-283.
[2] F Chollet, GitHub (2015), https://github.com/fchollet/keras
[3] S Gerhard *et al*, (2013), Figshare, http://dx.doi.org/10.6084/m9.figshare.856713
[4] The authors thank Dr. Tim Ryan (Pennsylvania State University) for contributing experimental data.

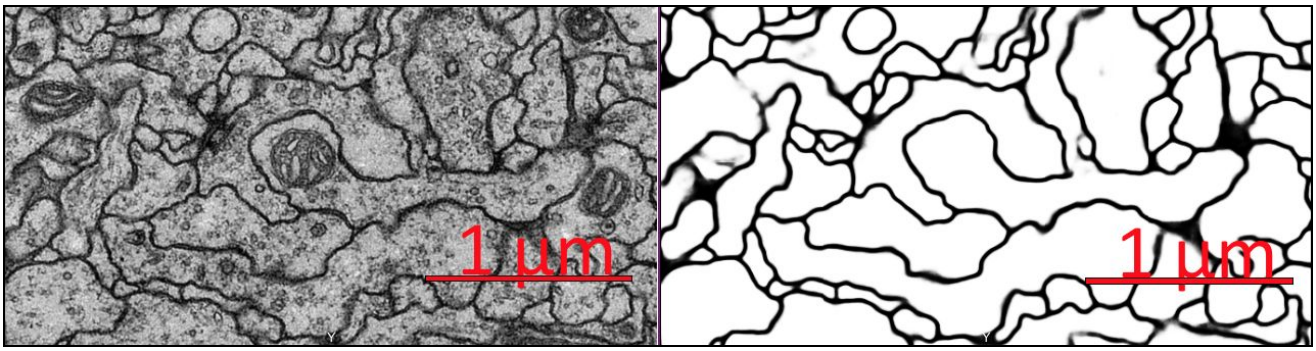

Figure 1. (Left) Input TEM image. (Right) Output of CNN, tuned for plasma membrane contrast enhancement.

https://doi.org/10.1017/S143192761800315X Published online by Cambridge University Press