Bioinformatics, 37(1), 2021, 97–106
doi: 10.1093/bioinformatics/btaa1094
Advance Access Publication Date: 8 January 2021
Original Paper

OXFORD

Bioimage informatics

EM-stellar: benchmarking deep learning for electron microscopy image segmentation

Afshin Khadangi [1], **Thomas Boudier** [2] **and Vijay Rajagopal** [1,*]

[1]Department of Biomedical Engineering, University of Melbourne, Victoria, 3000, Australia and [2]Institute of Molecular Biology, Academia Sinica, Taipei, Taiwan

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on July 29, 2020; revised on October 6, 2020; editorial decision on December 6, 2020; accepted on December 22, 2020

Abstract

Motivation: The inherent low contrast of electron microscopy (EM) datasets presents a significant challenge for rapid segmentation of cellular ultrastructures from EM data. This challenge is particularly prominent when working with high-resolution big-datasets that are now acquired using electron tomography and serial block-face imaging techniques. Deep learning (DL) methods offer an exciting opportunity to automate the segmentation process by learning from manual annotations of a small sample of EM data. While many DL methods are being rapidly adopted to segment EM data no benchmark analysis has been conducted on these methods to date.

Results: We present EM-stellar, a platform that is hosted on Google Colab that can be used to benchmark the performance of a range of state-of-the-art DL methods on user-provided datasets. Using EM-stellar we show that the performance of any DL method is dependent on the properties of the images being segmented. It also follows that no single DL method performs consistently across all performance evaluation metrics.

Availability and implementation: EM-stellar (code and data) is written in Python and is freely available under MIT license on GitHub (https://github.com/cellsmb/em-stellar).

Contact: vijay.rajagopal@unimelb.edu.au

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Electron microscopy (EM) is a fundamentally important modality for basic biomedical science research. In recent years, we have seen significant advances in EM technologies with the advent of, first, electron tomography and, more recently, focused ion-beam and serial block-face scanning EM (Hussain *et al.*, 2018). These technologies generate giga- and tera-bytes of high-resolution images of subcellular architecture which must be segmented manually or using image segmentation algorithms. The inherent low contrast in EM has motivated the use of crowd-sourcing platforms, and image segmentation challenges such as to reduce the image postprocessing time. Deep learning (DL) is a powerful approach to image segmentation that is being widely explored as a way to segment high-throughput biological datasets, including EM images (Xing *et al.*, 2018). In recent years, there have been several efforts to streamline the usage of such technologies for the community (Haberl *et al.*, 2018; Khadangi *et al.*, 2020; Schmidt *et al.*, 2018; Von Chamier *et al.*, 2020). One crucial question that arises is whether we can use such platforms to segment all types of EM data and whether they have inherent limitations in segmenting particular types of ultrastructures. Typically, these platforms utilize one unique or a limited number of available deep neural network architectures. No study has investigated the relationship between the choice of the DL method and the segmentation performance. Moreover, segmentation performance evaluation remains under investigated as current studies use a limited number of the evaluation metrics, and the impact of the choice of evaluation metric has not been investigated.

One of the other challenges that the community faces when using such methods is the lack of sophisticated DL utilities and functionalities as the current platforms often resort to demo-based DL applications. For example, such platforms opt for one single optimization method or use a limited set of evaluation criteria as the inbuilt evaluation metrics of the popular application programming interfaces (API) are often limited. Or the segmentation objective function or loss function is constrained to a limited number of inbuilt functions that such API provide. However, we have witnessed in recent years that successful DL applications in computer vision problems rely on a strategic blend of data processing, network architecture, optimization method, loss function, validation method, validation criteria and hyperparameter tuning. We have also seen how the choice of network architecture, loss function or optimization method can affect the DL performance (Janocha and Czarnecki, 2017; Khadangi *et al.*, 2018, 2019; Khadangi and Zarandi, 2016; Lin *et al.*, 2017; Liu *et al.*, 2019; Tan and Le, 2019). In addition to the above, the

cost-effectiveness or the computational efficiency of DL applications have not been investigated before.

We present EM-stellar (the official implementation is provided as a Colab Notebook on GitHub (https://github.com/cellsmb/em-stellar). We also provide a python implementation for the users who might face issues with storage limitations on Google Drive or have data security and privacy concerns), a comprehensive interface between the user and the DL application that is dedicated to EM image segmentation. Figure 1 represents the workflow of EM-stellar and the analysis that we have investigated in this study. EM-stellar provides a wide range of DL network architectures, evaluation metrics and easy to use utilities. Such utilities involve a wide range of custom loss functions, validation criteria, state-of-the-art optimization methods that minimize the hyperarameter tuning and K-fold cross-validation. Moreover, it enables the user to benchmark the performances of a diverse set of DL algorithms and use the desired methods as the ensemble of models for the final inference stage. EM-stellar is provided as the self-explanatory Jupyter Notebook for Google Colab which simplifies the use of such sophisticated technologies and utilities to a set of simple user clicks. This approach will save lots of time as the user will not face problems with software dependencies installment, and they do not need to learn the workflow of applications as EM-stellar is ready to use interface with guidelines of the usage for the users.

2 Materials and methods

2.1 Deep convolutional neural networks

Here, we briefly outline the deep neural networks that we have included in this study. The methods have been ordered chronologically by year and month. Software packages used in this study had open-source licenses. We use modified versions of the mentioned networks to test their performance on experimental datasets, and the corresponding architectures were correctly implemented to the best of our knowledge as validated with the literature.

1. VGG (Simonyan and Zisserman, 2014): VGG was proposed in 2014 as one of the first deep convolutional neural networks used for ImageNet challenge (Deng *et al.*, 2009) where the authors had proposed to push the depth to 16 or even 19 layers. The network uses tiny 3×3 convolution filters and rectified linear units ((ReLU) as the main activation function. We have used VGG-16 in this study as the encoder, then we have

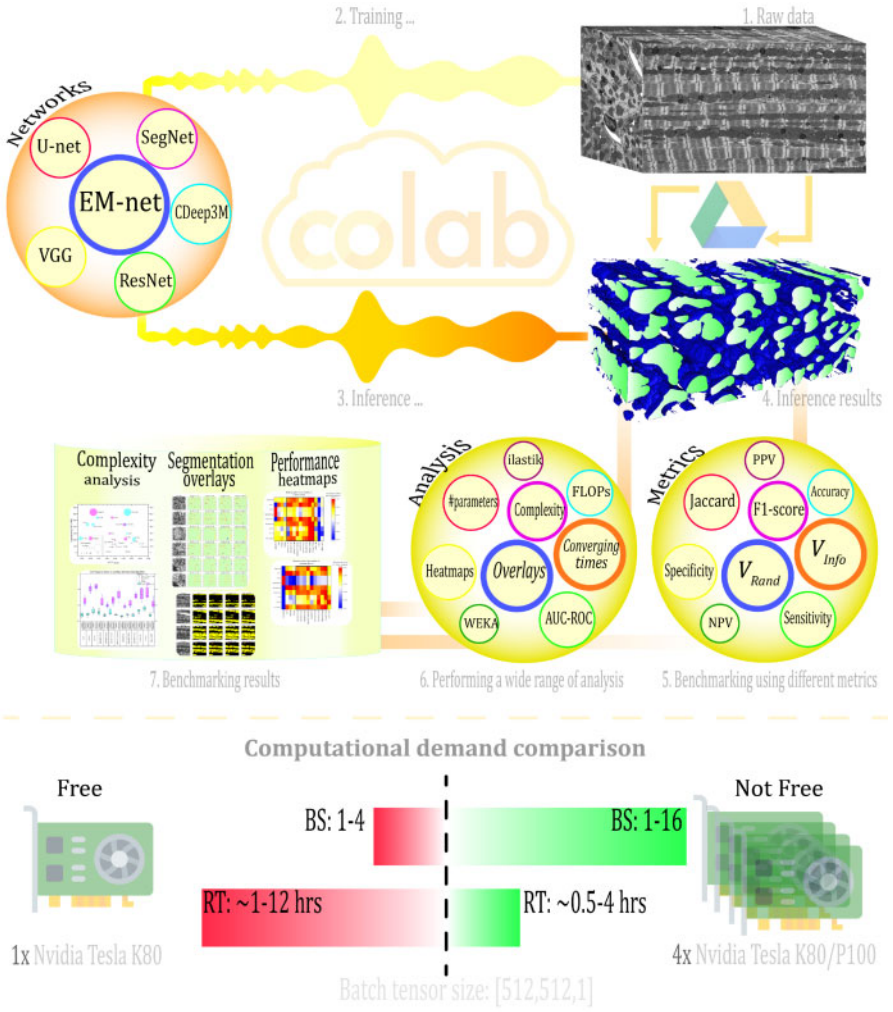

Fig. 1. Overview of the EM-stellar. Top of the figure shows the workflow of the EM-stellar. The user uploads the raw data to Google Drive and uses networks to segment the raw data. A wide range of metrics can be used to monitor the validation or to assess the inference performance. Moreover, in this study, we have performed a wide range of analysis including complexity analysis, convergence times. The bottom (computational demand comparison, BS and RT represent batch size and running time, respectively) shows effect of the batch size on the segmentation performance and the computational demand. As shown, increasing the batch size will require more GPU resources and, hence incurring more charges. We also show that increasing the batch size during the training expedites the convergence and often leads to better inference results. Moreover, we have also compared the DL performance with the previously developed machine-learning software packages including ilastik and Weka (we provide detailed analysis on our experiments with Weka and ilastik in Supplementary Information)

employed two-dimensional (2D) upsampling with skip connections to retrieve the feature channels back to the original resolution. Batch normalization (Ioffe and Szegedy, 2015) has been used in the network to stabilize the training. The kernels were initialized using He-normal initialization method (He *et al.*, 2015).

2. PReLU-net (He *et al.*, 2015): Parametric rectified linear unit (PReLU) was proposed in 2015 to generalize ReLU. The authors had also proposed a novel initialization method (He-normal) for kernels which improved the classification performance on ImageNet challenge. We adapted PReLU-net to use in this study where batch normalization has been used to stabilize the training.
3. U-net (Ronneberger *et al.*, 2015): U-net was proposed in 2015 for medical and biological image segmentation. The network uses two symmetric paths, namely called contractive and expansive paths to enhance capturing context and localization, respectively. In this study, we have implemented U-net in two versions: in the first version, we have used batch normalization across all the layers; however, the second version lacks batch normalization layers. ResNet (He *et al.*, 2016): Deep residual learning was proposed in 2016 to ease the training deep neural networks by reducing their complexities. The authors had evaluated a 152-layer residual network which had eight times more depth than VGG, but representing lower complexity as compared to VGG. In this study, we have used ResNet-50 as the encoder, and a U-net like decoder with skip connections have been utilized to retrieve the feature channels back to the original input resolution.
4. SegNet (Badrinarayanan *et al.*, 2017): SegNet was proposed in 2017 as a deep, fully convolutional neural network for semantic pixel-wise segmentation. The network consists of an encoder network followed by a decoder network and subsequently pixel-wise classification layer. We have used SegNet in this study to segment the EM images; however, we have modified the output layer to suit the binary classification task.
5. CDeep3M (Haberl *et al.*, 2018): CDeep3M was proposed in 2018 to facilitate access to complex computational environments and high-performance computational resources for the community. The authors had implemented InceptionResnetV2 (Szegedy *et al.*, 2017) using Caffe on Amazon Web Services (AWS) EC2 instance. Access to these facilities requires the user to pay for the resources on an hourly rate basis for both training and inference. CDeep3M offers 2D and three-dimensional (3D) segmentation pipelines.
6. EM-net (Khadangi *et al.*, 2020): EM-net was proposed in 2020 for 2D segmentation of EM images. The authors have proposed trainable linear units which generalize PReLU and ReLU and have evaluated the proposed network and the base classifiers on a FIB-SEM cardiac dataset and ISBI challenge for neuronal stacks segmentation. EM-net represents lower computational complexity in terms of the number of trainable parameters and floating point operations per second (FLOPs).

2.2 Data

We used two publicly available datasets in this study. The first dataset includes left ventricular myocyte FIB-SEM image datasets collected from mice, as described previously (Glancy *et al.*, 2015). We extracted 24 random patches from this dataset each having 512 × 512 pixels for training, testing and validation. After we manually annotated mitochondria on this sample, we split data randomly into training, validation and testing by 16/24, 4/24 and 4/24, respectively. The second dataset involves mice lateral habenula serial block-face scanning electron microscopy (SBEM) 1024 × 1024 × 80 voxels, as described previously (Haberl *et al.*, 2018). We extracted 320 random patches from this dataset and the corresponding binary mitochondria masks that had already been annotated for training, validation and testing. We split data randomly into training, validation and testing by 80%, 10% and 10%, respectively. All the random data splits were performed using K-fold cross-validation, and the inference performance is reported based on the best fold model.

2.3 Training and testing

All the experiments in this study except CDeep3M were implemented using TensorFlow GPU 1.8.0 CUDA 9.0 (Abadi *et al.*, 2016) and KERAS 2.2.4 (others, 2015). These experiments were performed on a GPU cluster, HPC Spartan (Lafayette *et al.*, 2016) as described in(Khadangi *et al.* (2020). A stack of 100GB GPU instance was launched on AWS p3.2xlarge in the US West (Oregon) region to train CDeep3M. CDeep3M is implemented in Caffe, and we utilized the default settings for training as described in Haberl *et al.* (2018) and Khadangi *et al.* (2020). More details about the training and testing are provided in Supplementary Information.

3 Results

3.1 Overview of DL methods

We performed an extensive survey of the literature to identify state-of-the-art deep neural networks that have been utilized for EM image segmentation. We chose CDeep3M (Haberl *et al.*, 2018), EM-net (Khadangi *et al.*, 2020), PReLU-net (He *et al.*, 2015), ResNet-50 (He *et al.*, 2016), SegNet (Badrinarayanan *et al.*, 2017), U-net (Ronneberger *et al.*, 2015) and VGG-16 (Simonyan and Zisserman, 2014) for our experiments. Among these methods, we have experimented EM-net with all of its seven base classifiers bringing the total number of networks and methods to a maximum of thirteen. We used two publicly available focused ion-beam scanning electron microscopy (FIB-SEM) datasets for our experiments and evaluation purposes. We utilized a wide range of segmentation evaluation metrics to compare the results including F1-score, Foreground-restricted Rand Scoring after border thinning [$V^{Rand}{}_{(thinned)}$] Foreground-restricted Information-Theoretic Scoring after border thinning [$V^{Info}{}_{(thinned)}$] (Arganda-Carreras *et al.*, 2015). More details about datasets and evaluation metrics are highlighted in methods section.

3.2 EM-net variants demonstrate reliable learning capacity on both small and large datasets

We trained and evaluated chosen networks with two FIB-SEM datasets, including one small cardiac dataset comprising 24 serial sections each of pixel size 512 × 512, and another large neuronal dataset consisting of 320 serial sections with the same image size as the cardiac dataset. Mitochondria were manually annotated on both datasets. Figure 2 illustrates the results of evaluating networks on the test datasets that were held out randomly and not used for training. The result values have been normalized using min–max normalization per metric category for comparison.

As shown, despite the difference in size between these two datasets, EM-net variants (grouped within a box on both heatmaps) demonstrate competitive evaluation metric values when compared to other methods. The ensemble of top EM-net base classifiers outperforms other methods majority of the metrics on the cardiac dataset; however, the segmentation performance metric values were not as high performing on the neuronal dataset based on average voting.

3.3 No one network can fit them all

Figure 2 shows how the underlying texture and intensity distribution of different datasets, and the target ultrastructures can affect the performance of a deep neural network in segmenting a dataset. One network cannot achieve high performance for all datasets—one network cannot fit them all. Considering U-net BN and EM-net V2 4X, both methods demonstrate only above-average performance on

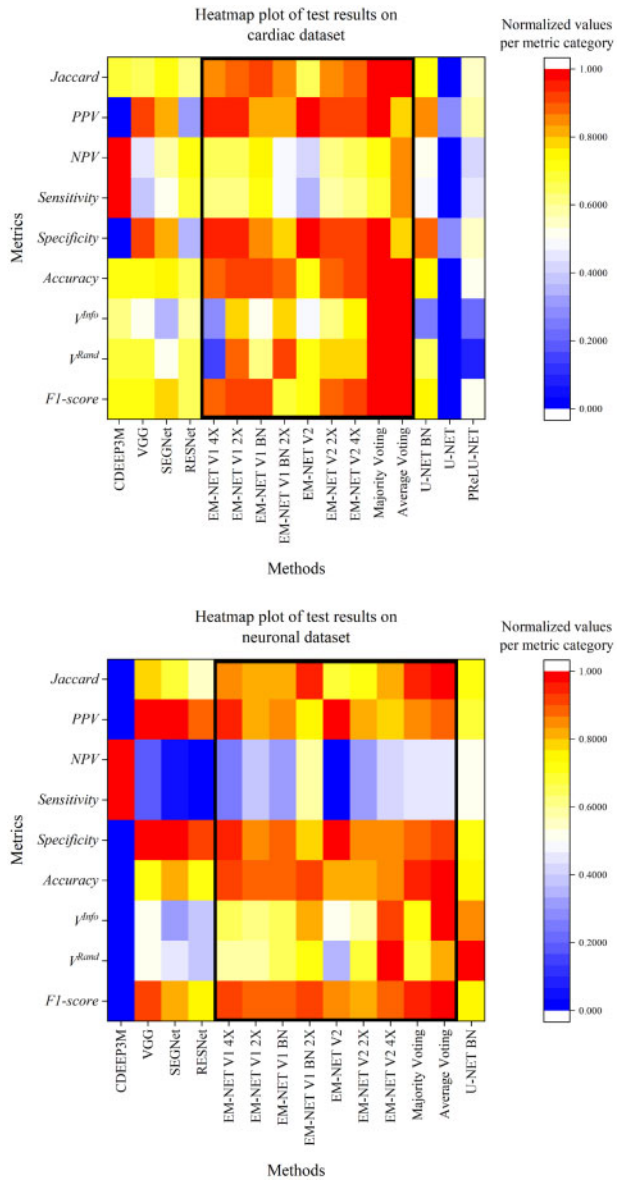

Fig. 2. Heatmap of evaluation metrics for different methods based on the test datasets. The values are normalized using min–max normalization per metric category. The black boxes correspond to EM-net base classifiers, including the ensemble methods. Top: cardiac, bottom: neuronal data

segmenting the cardiac dataset in terms of the $V^{Rand}{}_{(thinned)}$ score; however, they achieve top performance based on the same metric for the neuronal data. Additionally, almost all of the methods represent relatively similar performance based on sensitivity metric for the cardiac dataset, whereas they have dropped below 50% on the neuronal dataset. CDeep3M demonstrates above-average performance for the cardiac dataset, whereas it provides inferior performance on the neuronal dataset. However, Figure 2 implies that an ensemble of top classifiers may lead to reliable performance across different datasets.

3.4 Evaluation metrics remain subjective and probably unique to the deep neural network

Choosing the right evaluation metric for segmenting EM data is a critical and still challenging step as depending on the objective of the segmentation, the user might prefer a specific set of segmentation metrics (Arganda-Carreras *et al.*, 2015). In other words, there is no one universal evaluation metric for such tasks. The choice of such metrics might even depend on the segmentation task; e.g. 2D or 3D segmentation may require different evaluation metrics. One previous study (Taha and Hanbury, 2015) has investigated benchmarking segmentation metrics for biomedical images in the 3D setting. Still, most of the studies have opted for F1-score and Jaccard index as the segmentation metric of choice. In one other research (Caicedo *et al.*, 2019), the same metrics have been utilized as the main evaluation metrics for nucleus segmentation.

We extended our analysis to monitor the response of the neural networks to different evaluation metrics. We followed this aim as the evaluation metrics reported for EM image segmentation remains sparse in the literature, and no study has investigated such a broad range of analysis on evaluation metrics. Our analysis shows that performances of these networks are subject to change depending on the evaluation criteria. Take the result of U-net BN on neuronal test dataset as an example shown in Figure 2. This network achieved top-performing $V^{Rand}{}_{(thinned)}$ score; however, it demonstrated average performance when using other metrics, including accuracy, e.g. Moreover, our analysis shows that the Jaccard similarity index and F1-score are mostly correlated for those instances that have achieved top Jaccard index scores.

In addition to the above, we found that some methods demonstrate unique behavior when applied to different datasets. As shown in Figure 2, CDeep3M demonstrates the same performance for specificity and PPV, meaning that this network produces minimal false-negative segmentation instances. However, the performance of VGG implies that this network delivers low sensitivity and high specificity on both neuronal and cardiac datasets. These findings suggest that the architecture of the deep neuronal networks and the underlying layers can affect the performance of the networks when evaluated with different metrics. In summary, the users might prefer one network over another depending on the desirable evaluation metrics, and they should not expect that one method will be the top performer for all the metrics.

3.5 Convergence times vary depending on the size of the dataset or the underlying data structures

Figure 3 shows the convergence times of the networks on both cardiac and neuronal datasets according to the validation metrics that we have chosen during the training. The convergence times imply that the large ground-truth datasets (in this case, neuronal dataset), and potentially diverse structural variations in the data will impact the convergence time of the network. However, this does not necessarily mean that the convergence times are positively correlated with the data size only.

Take EM-net V1 BN and V1 BN 2X as an example shown in Figure 3. Based on a comparison between the convergence times of these two networks for the cardiac dataset, one user might expect that V1 BN 2X will demonstrate lower mean and median values for the convergence times relative to the V1 BN on the neuronal dataset as well. However, Figure 3 shows precisely the opposite. On the other hand, U-net and U-net BN demonstrate relatively similar convergence times based on their mean and median values for the neuronal dataset; however, U-net BN has converged faster than U-net on the cardiac dataset. Our analysis shows that convergence times does not only depend on the ground-truth data size but is also affected by underlying data structures and the feature bank of the network used for the training. In general, EM-net V1 2X and EM-net V2 show less sensitivity to the data size or data structures, as shown in Figure 3.

3.6 Complex networks might not perform well and might also exhaust resources

Figure 4 illustrates the ball chart reporting the complexity of the networks in terms of Giga FLOPs, the associated number of parameters and top $V^{Rand}{}_{(thinned)}$ score (thresholded to above 0.90) on the corresponding test datasets. The operations are reported for one iteration based on an input tensor with the shape of (1, 512, 512, 1) representing a single batch of monochromic image. As shown, ResNet and CDeep3M required the lowest and highest FLOPs, respectively.

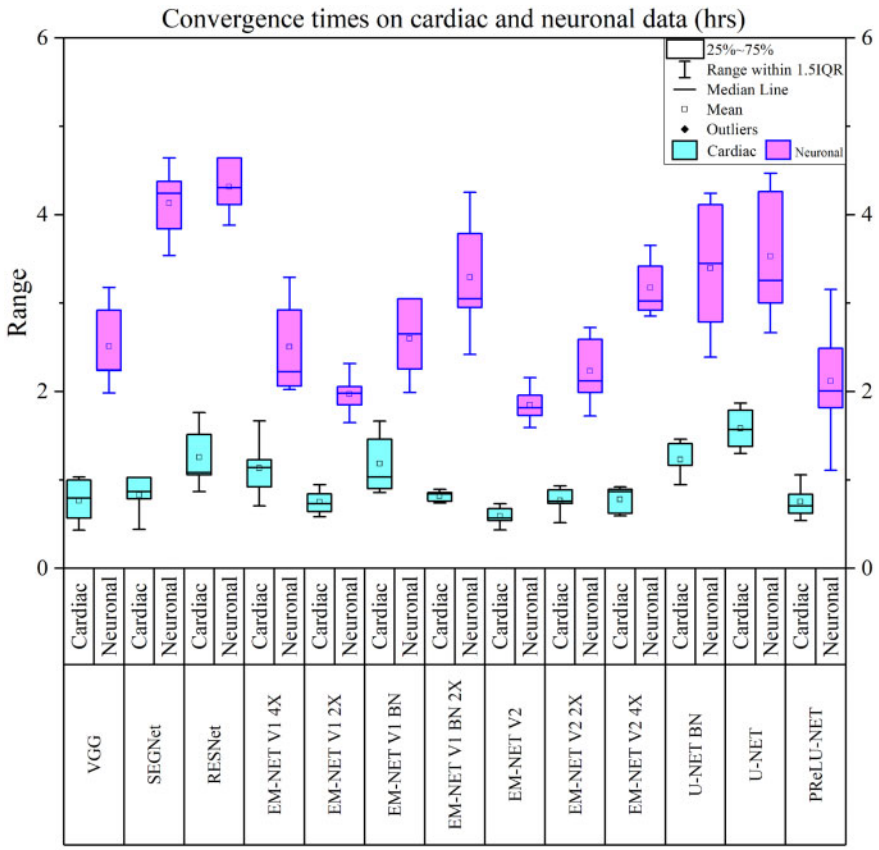

Fig. 3. Convergence times (hours) of different networks for cardiac and neuronal data based on the different evaluation metrics. These times have been reported based on our runnings on four parallelly pooled Nvidia Tesla P100 GPUs

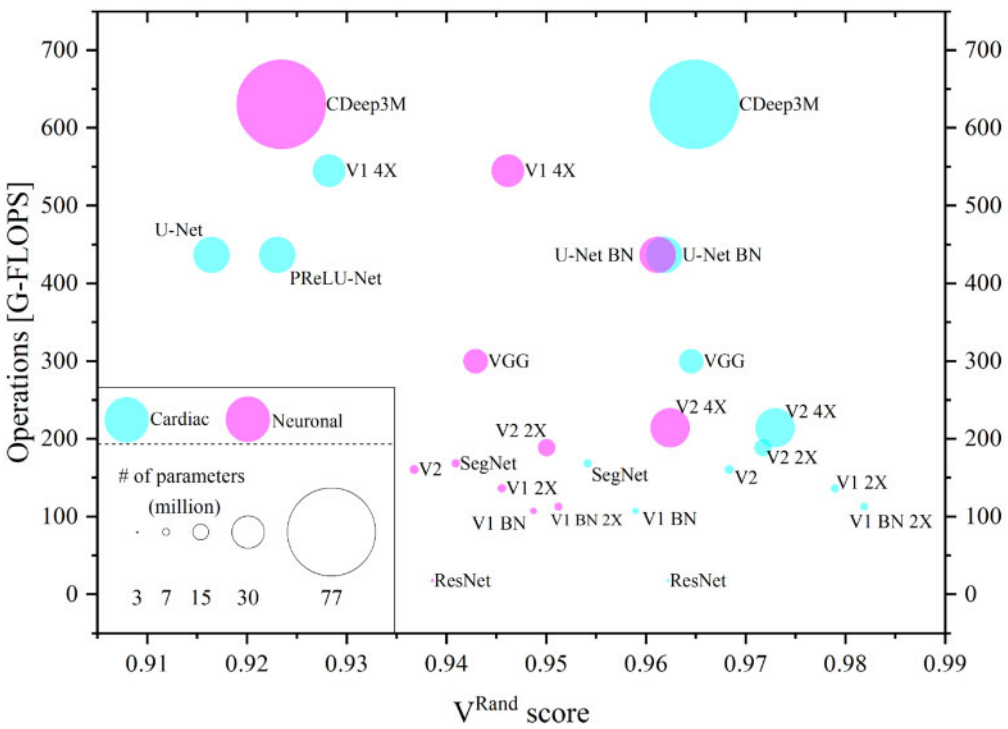

Fig. 4. Ball chart reporting complexity of networks based on Giga FLOPs and the corresponding performances in terms of the $V^{Rand}{}_{(thinned)}$ score. This figure also illustrates the number of trainable parameters for individual networks (millions)

First, the number of parameters does not directly reflect the complexity. Considering EM-net V2 4X and U-net BN, both these networks have a relatively similar number of parameters; however, EM-net V2 4X required less computational resources to perform the same job as compared to U-net BN. Second, this figure shows that the high number of parameters or complexity of the networks do not necessarily yield top test performances. This figure shows how EM-net V1 2X and V1 BN 2X have achieved top $V^{Rand}{}_{(thinned)}$ score on the cardiac dataset despite their very low complexity and the number of parameters. However, we can observe that their performances have been not as good when tested on the neuronal dataset but they are still competitive when compared to the VGG and CDeep3M results. Finally, we can observe that U-net BN demonstrates similar performance in terms of $V^{Rand}{}_{(thinned)}$ score for both cardiac and neuronal datasets.

3.7 Visualization of the intermediate layers reveals the redundancy of the feature channels

We visualized over 140 000 intermediate feature channels for U-net BN, VGG and EM-net V2 4X on the test datasets. We analyzed the 2D correlation between each of the individual channels leading to over than 9 billion feature correlation maps. Figure 5 illustrates the distributions of the correlation maps between each block of the networks. Each of these blocks shares the same characteristic as the underlying feature channels, which have the same resolutions. For example, B1 represents the distributions of the feature correlation maps for these three networks within their corresponding block one, and they all have the same feature resolutions in this case (512, 512) in x and y as we have used for training datasets. We have also visualized the interblock feature correlation maps by which we can analyze the relationships of the feature maps within each of the blocks of these networks. For example, take B4 and B3 in y and x axes, respectively. This location corresponds to 2D contours of the feature correlation distributions between these blocks. It suggests that U-net represents similar feature correlation distributions in blocks three and four; however, VGG and EM-net show much spread distributions which means they extract less redundant feature maps. We have also visualized the scatter plots of these feature correlation maps distributions in the upper-diagonal plots.

Our analysis shows that in general, VGG and EM-net demonstrate fewer feature correlations within each block and even between the individual blocks as compared to U-net. The

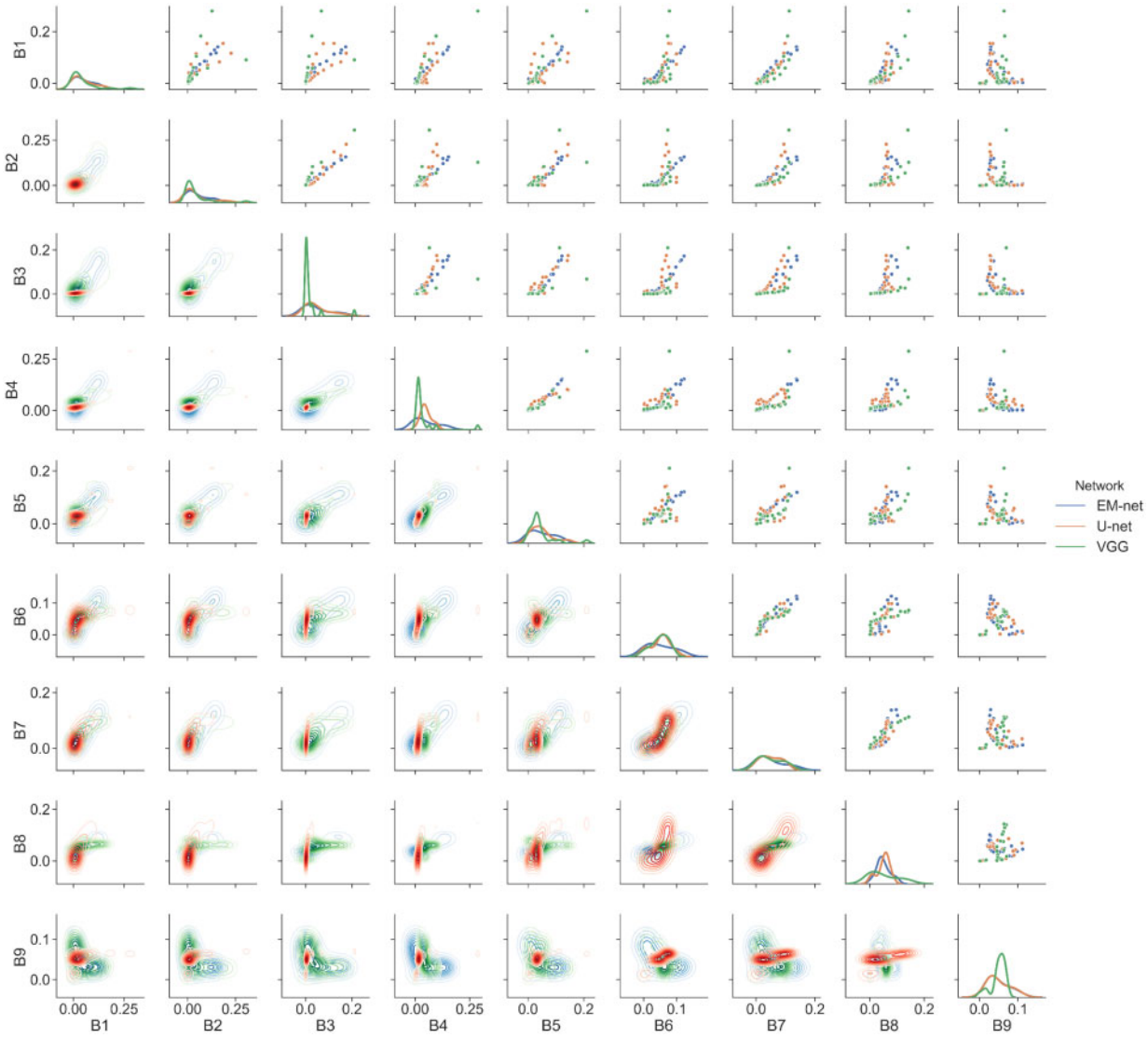

Fig. 5. Block-wise feature correlation map distributions for EM-net, U-net and VGG. B1 corresponds to the first block (maximum feature resolution), B5 represents the bottleneck of the networks (minimum resolution), and B9 represents the block corresponding to the output node. The upper-diagonal plots show the scatter plots of the feature correlation maps, the bottom-diagonal plots represent the 2D contours of the block-wise distributions and diagonal plots correspond to univariate feature correlation distributions of individual blocks

distributions of these feature channel correlation maps reach their maximum between blocks three and four in VGG, implying that features are less correlated within these two blocks. However, EM-net demonstrates less correlation between the feature channels within the block five called 'the bottleneck' (where almost 30–50% of the features are concentrated here) as compared to the two others.

From the interblock feature correlation map perspective, we can observe that U-net demonstrates a high correlation between the correlation map distributions of the different blocks as these distributions are centered or peaked. This implies that feature maps extracted by the U-net could potentially lead to redundant feature maps, especially in the bottleneck as almost all the blocks represent the same level of feature correlation map distributions. One study (Iglovikov and Shvets, 2018) has investigated this phenomenon by tweaking the U-net architecture where they have substituted the encoder of the U-net to the encoder of the VGG-11, and the authors have obtained better performance in terms of Jaccard similarity index.

3.8 Visualization of the segmentation masks reveals that CDeep3M is less prone to false-negative

We have visualized the overlays of the segmentation results on the sample cardiac and neuronal test datasets. Figures 6 and 7 illustrate the overlay of binary masks on the sample cardiac and neuronal test datasets, respectively. These illustrations have been obtained based on the results of EM-net (average and majority voting), CDeep3M and U-net BN. As shown, CDeep3M provides minimal false-negative or missing mitochondria on these images; however, the number of false-positives is higher than EM-net and U-net. U-net and EM-net offer a higher number of false-negative instances in comparison with CDeep3M as they are more prone to missing mitochondria.

We have used a threshold value of 0.5 to obtain binary masks of the segmentation probability maps resulted from the networks in these visualizations, as illustrated in Figures 6 and 7. However, metrics like $V^{Rand}{}_{(thinned)}$ and $V^{Info}{}_{(thinned)}$ handle such a limitation by thinning the border or using a threshold step value of 0.1. As a result, we can monitor the desired performance metric and finally determine the best performing threshold value.

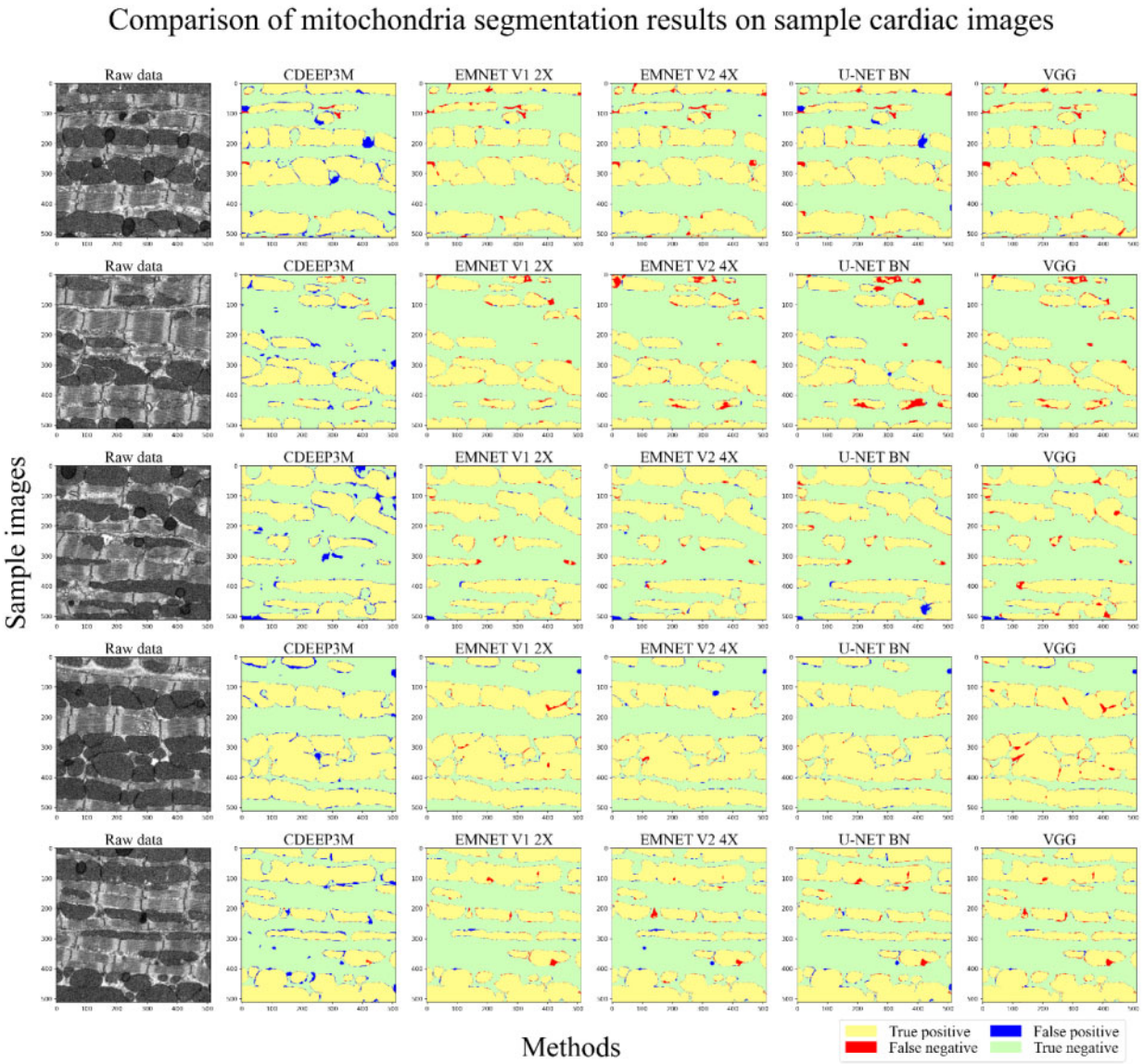

Fig. 6. Comparison of mitochondria segmentation results on sample cardiac test dataset. Yellow, green, red and blue correspond to true-positive, true-negative, false-negative (missing mitochondria) and false-positive. EM-net, U-net and VGG are less prone to false-positives; however; CDeep3M demonstrates minimum false-negative segmentation errors. The left column represents the sample test FIB-SEM images of cardiomyocytes. Other columns correspond to the overlay of result masks for CDeep3M, EM-net, U-net BN and VGG

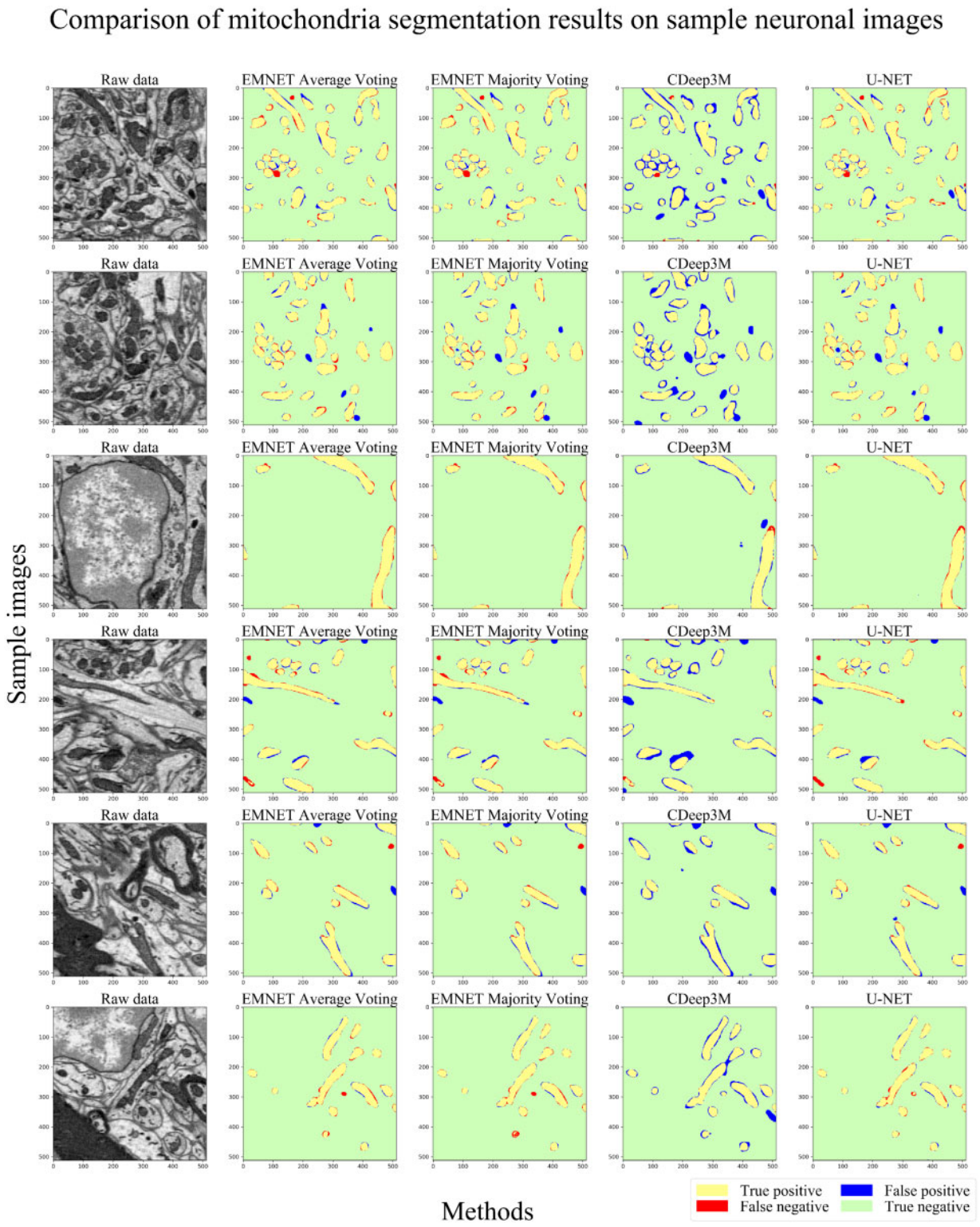

Fig. 7. Comparison of mitochondria segmentation results on sample neuronal test dataset. Yellow, green, red and blue correspond to true-positive, true-negative, false-negative (missing mitochondria) and false-positive. EM-net ensembles and U-net are less prone to false-positives; however; CDeep3M demonstrates minimum false-negative segmentation errors. The left column represents the sample test SBEM images of mice brain cells. Other columns correspond to the overlay of result masks for EM-net ensembles based on average and majority voting, CDeep3M and U-net

4 Discussion

We have presented EM-stellar, a framework for benchmarking DL methods for EM image segmentation that is hosted on Google Colab. Although a couple of reviews of using DL methods for microscopy image analysis and segmentation have been reported (Carneiro *et al.*, 2017; Xing *et al.*, 2018), a comprehensive evaluation of the segmentation methods has not been conducted to date.

In this paper, we have compared seven different deep convolutional neural networks for EM image segmentation. Most of the studies reported in the literature are limited to one tissue type such as neuronal microscopy datasets; however, we report our analysis not only using a neuronal dataset but also using cardiac EM data. We also have extended our study to analyze the performance of these methods using a wide range of segmentation metrics.

Moreover, we report the computational complexity of these algorithms and their associated computational demand. This is the first study in the literature that reports such analysis in the context of EM image segmentation, which is implemented in the cloud for persistent reusability by biologists. Our Colab notebook enables the users to benefit from state-of-the-art software and hardware resources in the context of DL to achieve the maximum segmentation performance.

We found considerable variation in the segmentation performance metrics across individual algorithms. Our study shows that different deep neural networks perform differently when using a single segmentation metric. Among many validation performance monitoring criteria, high validation F1-score and Jaccard similarity index are associated with high test Jaccard, F1-sore and $V^{Rand}{}_{(thinned)}$ scores. In terms of the objective function, we have found that using binary cross-entropy for highly imbalanced binary segmentation tasks will not necessarily lead to best inference results and using focal loss (Lin *et al.*, 2017) is highly recommended in such cases. In terms of the optimization methods, using warm-up strategy (Goyal *et al.*, 2017) has led to best inference performance in ISBI challenge and mitochondria segmentation in both cardiac and neuronal datasets. For small and limited training datasets, complex networks tend to overfit more often; however, they show reliable performance as exposure to an abundant training dataset. Convergence times and computational resource expense depend on both variations of structures in image data and changes across serial sections. Moreover, our experiments suggest that training these networks on GPUs in parallel mode with increased batch size boost the segmentation performance and minimize the convergence times.

We also report the segmentation performance using ilastik and Weka (see Supplementary Information). Our experiments suggest that DL methods perform better than ilastik and Weka in terms of accuracy. However, ilastik and Weka offer a significant advantage over DL methods as they can save user's time by segmenting the data with limited and sparse ground-truth labels in the expense of accuracy and require less training time and computational resources.

Finally, we highlight the importance of ensemble learning in EM image segmentation. Our experiments show that using only one type of classifier or deep neural network, or even one randomly chosen validation dataset will not lead to maximum test segmentation performance. Hence, we have equipped EM-stellar with ensemble learning which enables the user to select the inference model based on majority or average voting. Moreover, EM-stellar allows the user to benefit from K-fold cross-validation, which maximizes the chance of obtaining maximum inference performance.

To summarize, EM-stellar is a cloud-based platform hosted on Google Colab which gives free access to GPU and TPU resources and enables the user to use state-of-the-art DL methods across a wide range of segmentation performance metrics. It is equipped with several machine-learning strategies including K-fold cross-validation, different loss functions and optimization methods. It enables the user to choose top-performing models for ensemble learning based on report insights that are provided at the end of the training. We recommend the users to use U-net BN and EM-net V2 4X for segmentation when a large training dataset is available; otherwise, they may use EM-net V1 2X. However, the users might opt for arbitrary networks if they aim at using an ensemble of different models. We plan to utilize TPUs in the future as part of EM-stellar release versions and integrate other state-of-the-art networks such as EfficientNet (Tan and Le, 2019) to deliver maximum performance and efficiency. Table 1 illustrates a summary of our findings on the network selection strategy for the prospective users.

Table 1. Summary of network/method and evaluation criteria strategy selection based on the data size

Data size	Evaluation criteria	Network/method	Error prevalence
Small	V^{Rand}, V^{Info} *or* F1 − score	EM-net V1 2X	False-negative
		Average voting	False-positive
		Majority voting	False-negative
Large	V^{Rand}, V^{Info}	CDeep3M	False-positive
		EM-net V1 4X, U-net Average voting	False-negative
	Jaccard similarity index	Majority voting	

This summary is acquired based on our experiments and may not be generalizable under other data or methodology settings.

Acknowledgements

This research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Facility was established with the assistance of [LIEF Grant LE170100200]. We also thank Dr. Brian Glancy at NIH/NHLBI for access to the FIB-SEM data.

Financial Support: none declared.

Conflict of Interest: none declared.

References

Abadi,M. *et al.* (2016) Tensorflow: a system for large-scale machine learning. In: *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283.

Arganda-Carreras,I. *et al.* (2015) Crowdsourcing the creation of image segmentation algorithms for connectomics. *Front. Neuroanat.*, **9**, 142.

Badrinarayanan,V. *et al.* (2017) Segnet: a deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, **39**, 2481–2495.

Caicedo,J.C. *et al.* (2019) Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. *Cytometry Part A*, **95**, 952–965.

Carneiro,G. *et al.* (2017) Review of deep learning methods in mammography, cardiovascular, and microscopy image analysis. In: *Deep Learning and Convolutional Neural Networks for Medical Image Computing*. Springer, pp. 11–32.

Deng,J. *et al.* (2009) Imagenet: a large-scale hierarchical image database. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 248–255.

Glancy,B. *et al.* (2015) Mitochondrial reticulum for cellular energy distribution in muscle. *Nature*, **523**, 617–620.

Goyal,P. *et al.* (2017) Accurate, large minibatch SGD: training imagenet in 1 hour. arXiv:170602677.

Haberl,M.G. *et al.* (2018) CDeep3M—plug-and-play cloud-based deep learning for image segmentation. *Nat. Methods*, **15**, 677–680.

He,K. *et al.* (2015) Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034.

He,K. *et al.* (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

Hussain,A. *et al.* (2018) An automated workflow for segmenting single adult cardiac cells from large-volume serial block-face scanning electron microscopy data. *J. Struct. Biol.*, **202**, 275–285.

Iglovikov,V. and Shvets,A. (2018) Ternausnet: U-net with vgg11 encoder pre-trained on imagenet for image segmentation. arXiv:180105746.

Ioffe,S. and Szegedy,C. (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:150203167.

Janocha,K. and Czarnecki,W.M. (2017) On loss functions for deep neural networks in classification. *arXiv:170205659*, **1**/2016.

Khadangi,A. *et al.* (2018) Automated framework to reconstruct 3D model of cardiac Z-disk: an image processing approach. In: *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, pp. 877–884.

Khadangi,A. *et al.* (2019) Automated segmentation of cardiomyocyte Z-disks from high-throughput scanning electron microscopy data. *BMC Med. Inform. Decis. Making*, **19**, 1–14.

Khadangi,A. *et al.* (2020) EM-net: deep learning for electron microscopy image segmentation. *bioRxiv*.

Khadangi,A. and Zarandi,M.F. (2016) From type-2 fuzzy rate-based neural networks to social networks' behaviors. In: *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, pp. 1970–1975.

Lafayette,L. *et al.* (2016) Spartan performance and flexibility: an HPC-cloud chimera.

Lin,T.-Y. *et al.* (2017) Focal loss for dense object detection. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988.

Liu,L. *et al.* (2019) On the variance of the adaptive learning rate and beyond. arXiv:190803265.

others FCa (2015) Keras. https://keras.io.

Ronneberger,O. *et al.* (2015) U-net: convolutional networks for biomedical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 234–241.

Schmidt,U. *et al.* (2018) Cell detection with star-convex polygons. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 265–273.

Simonyan,K. and Zisserman,A. (2014) Very deep convolutional networks for large-scale image recognition. *arXiv:14091556*.

Szegedy,C. *et al.* (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. In: *Thirty-First AAAI Conference on Artificial Intelligence*.

Taha,A.A. and Hanbury,A. (2015) Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Med. Imaging*, **15**, 29.

Tan,M. and Le,Q.V. (2019) EfficientNet: rethinking model scaling for convolutional neural networks. *arXiv:190511946*.

Von Chamier,L. *et al.* (2020) ZeroCostDL4Mic: an open platform to simplify access and use of deep-learning in microscopy. *BioRxiv*.

Xing,F. *et al.* (2018) Deep learning in microscopy image analysis: a survey. *IEEE Trans. Neural Netw. Learn. Syst.*, **29**, 4550–4568.