Deep Learning for Single Molecule Localization Microscopy

Micheal Brandon Reed

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Single Molecule Localization Microscopy (SMLM) enables researchers to interrogate nanoscale spatial details in a range of systems. Biological investigations benefit greatly from SMLM due to its ability to quantitatively investigate details of great importance such as protein distribution in the cell membrane or protein-protein interactions(Baddeley & Bewersdorf, 2018). However, SMLM set-ups can be quite expensive, imaging times can be lengthy and data analysis can require expert knowledge. Recently, Deep Learning (DL) algorithms have been developed to reduce imaging time and automate data analysis. Naturally each subsequent model aims to address a different shortcoming of the prior work and so there is a family of model architectures. However, there is not a singular location for researchers to have access to these models dedicated to SMLM. We developed the Deep Learning for Single Molecule Localization Microscopy (DL4SMLM) in Python using the PyTorch framework to democratize access to these models and lower the barrier of entry to SMLM. DL4SMLM is a Python package that uses the Pytorch DL framework to implement Convolutional Neural Nets (CNN) dedicated to SMLM. We created it for researchers and engineers who wish to use DL for SMLM in whichever field they practice granted they have sufficient data.

Statement of Need

DL4SMLM is designed to automate the training, validation and testing of different machine learning models dedicated toward the single molecule localization task in a single software suite. Currently, such models are located in their respective code reposit ries and/or enmeshed into software plug-ins. While this makes their models publicly available it hinders rapid re-training, prototyping and comparison of different models on the same dataset. By providing this software environment, investigators can generat e their training data (simulated or experimentrally collected), train various models, and then decide which model is best suited for their task. Because our software is built atop of the Pytorch framework, our functions allow the user to designate which device, CPU or GPU, the inference process will occur on. We allow this flexibility on the inference process because we want invesigators to be empowered to assess how their models will perform on the CPU of their choice especially where they wish to push their models onto compute limited devices such as mobile phones.

Features

Models

Currently, there are six models:Super Resolution Convolutional Neural Network (SRCNN)(Dong et al., 2015), Deep-Stochastic Optical Reconstruction Microscopy (Deep-STORM)(Nehme et al., 2018), Skip-STORM, Unet for STORM (U-STORM)(W. Yao et al., 2018), Deep Residual STORM (DRL-STORM)(B. Yao et al., 2020), and Fast Dense Image Reconstruction based Deep Learning STORM (FID-STORM)(Zhou et al., 2023). SRCNN and Skip-STORM are

The Journal of Open Source Software

novel architectures to the SMLM field. While SRCNN has been applied to perform the super resolution task in more traditional image processing tasks, its utility in SMLM is unknown. Skip-STORM's architecture is similar to Deep-STORM but a skip connection between the initial image and final layer is introduced to enable it to have a spatial context during its reconstruction phase which could assist it in emitter dense images. U-STORM is inspired by U-Net and adopts a similar biphasic architecture where in the first half the initial image is max pooled while simultaneously increasing channel number and the second half involves the up sampling and reduction in channel. It also contains skip connections that connect the features of the first half to the those of second half during image reconstruction. These models are implemented using the object-oriented programming paradigm enabling researchers to instantiate multiple models for a single data set to allow downstream comparison of inference performance.

Loss Functions

We have implemented three loss functions: L1L2, weighted mean square error, and weighted mean absolute error. Each loss function accepts a lambda parameter which controls the sparsity of the predicted output.The lower the lambda value the sparser the output while a higher lambda value preserves more of the signal from the original diffraction limited image. The L1L2 loss metric was first introduced by Nehme et al. (2018) and involves a gaussian convolution of the predicted spikes in the ground truth image. We chose to implement the weighted mean square error and weighted mean absolute error in instances where the gaussian convolution is unneeded in the ground truth image but the user still desires control over the sparsity of the predicted image.

Helper Functions

A custom ImageDataset class havs been implemented using PyTorch's Dataset functionality to automate the loading and normalization of the noisy diffraction limited images and their super resolved counterparts. This dataset can then be loaded into the DataLoader class in PyTorch.To assist researchers in visualizing the emitter localization performance of their model, we have implemented a function that visualizes the diffraction limited image of the emitters, its super-resolved version, and the predicted super-resolved image according to the trained model.This empowers the researcher to troubleshoot the performance of the model and determine if the model should be retrained with different parameters or to proceed with the experiment. To faciliate the devolpment of powerful yet accurate models we have implemented two functions that enable knowledge distillation between a teacher model and a student model: Hint Learning and Knowledge Transfer. Hint training automates the learning process of a student ne work to have its intermediate representation mirror that of a teacher network. Knowledge Transfer uses the attentive imitation loss function from Saputra et al. (2019), the teacher model, and the ground truth data set to optimize the student network.

Acknowledgements

This work is supported by NIH award 1R16GM145671 and NSF award MCB 2027738. This work was performed at the Joint School of Nanoscience and Nanoengineering (JSNN), a member of the National Nanotechnology Coordinated Infrastructure (NNCI), which is supported by the National Science Foundation (Grant ECCS-2025462).

References

Baddeley, D., & Bewersdorf, J. (2018). Biological insight from super-resolution microscopy: What we can learn from localization-based images. *Annual Review of Biochemistry*, *87*(1), 965–989.

Dong, C., Loy, C. C., He, K., & Tang, X. (2015). Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(2), 295–307.

Nehme, E., Weiss, L. E., Michaeli, T., & Shechtman, Y. (2018). Deep-STORM: Super-resolution single-molecule microscopy by deep learning. *Optica*, *5*(4), 458–464. https://doi.org/10.1364/OPTICA.5.000458

Saputra, M. R. U., De Gusmao, P. P., Almalioglu, Y., Markham, A., & Trigoni, N. (2019). Distilling knowledge from a deep pose regressor network. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 263–272.

Yao, B., Li, W., Pan, W., Yang, Z., Chen, D., Li, J., & Qu, J. (2020). Image reconstruction with a deep convolutional neural network in high-density super-resolution microscopy. *Opt. Express*, *28*(10), 15432–15446. https://doi.org/10.1364/OE.392358

Yao, W., Zeng, Z., Lian, C., & Tang, H. (2018). Pixel-wise regression using u-net and its application on pansharpening. *Neurocomputing*, *312*, 364–371. https://doi.org/https://doi.org/10.1016/j.neucom.2018.05.103

Zhou, Z., Wu, J., Wang, Z., & Huang, Z.-L. (2023). Deep learning using a residual deconvolutional network enables real-time high-density single-molecule localization microscopy. *Biomed. Opt. Express*, *14*(4), 1833–1847. https://doi.org/10.1364/BOE.484540