**deeptracknm.py**

* deeptrack.py(https://github.com/softmatterlab/DeepTrack) is a python based deep learning algorithm developed by Helgadottir et al., 2019. deeptracknm.py incorporates a few minor modifications to the original deeptrack.py code as required for analysis of images generated from neuromorphic data. The following pipelines and codes have been created using functions defined by DeepTrack. The following link and paper have a more detailed overview of DeepTrack:
* https://github.com/softmatterlab/DeepTrack
* Helgadottir, S., Argun, A. & Volpe, G. Digital video microscopy enhanced by deep learning. *Optica***6**, 506-513, doi:10.1364/OPTICA.6.000506 (2019).

**beadtracksim.py**

* beadtracksim.py is an algorithm that uses DeepTrack's *image\_generator* function to simulate images of varying image parameters, create and train a neural network of user defined sizes, and to test the trained neural network on simulated particles. Each of these processes are carried out in separate cells as follows:
* The first cell is used to generate simulation images of particles. The image parameters for generating simulated particles are defined by DeepTrack's function *get\_image\_parameters* that has been modified to suite the required simulations. The algorithm generates simulated particle images using the Bessel function from user defined distributions for image parameters like particle radii, particle intensity, Bessel function order, background level, signal to noise ratio, ellipticity of the particle, and gradients, all in terms of pixel values. The algorithm also allows for multiple particles to be simulated in a single image, and define the size of the image generated( in pixels).
* The second cell is used to define the architecture of the neural network as follows:
* *input\_shape*: Enter the shape of the input data to the neural network.
* *conv\_layers\_dimensions*: Enter the number and size of the convolutional layers required inthe network.
* *dense\_layers\_dimensions:* Enter the number and size of the dense layers needed in the network.
* *create\_deep\_learning\_network* is a DeepTrack function that is used to create a convoluted neural network of given dimensions and architecture.
* The third cell is used to train the neural network that was created by user defined parameters. The inputs in this cell are:
* *sample\_sizes:* Enter the size of the batch of images to be generated.
* *iteration\_numbers:* Enter the number of iterations each batch has to go through.
* *train\_deep\_\_learning\_network* is a DeepTrack function that uses *image\_generator* and training parameters to train the neural network created.
* The fourth cell uses the *image\_generator* function to generate simulated function to test the trained neural network for its accuracy.
* The final cell is used to save the trained neural network.

**snrseries.py**

* snrseries.py is an algorithm that uses DeepTrack's *image\_generator* function to simulate images of varying image parameters. The parameters are defined by DeepTrack's function *get\_image\_parameters* that has been modified to suite the required simulations. The algorithm generates simulated particle images using the Bessel function from user defined distributions for image parameters like particle radii, particle intensity, Bessel function order, background level, signal to noise ratio, ellipticity of the particle, and gradients, all in terms of pixel values. The algorithm also allows for multiple particles to be simulated in a single image, and define the size of the image. A trained neural network can be loaded and used to predict the centroids of simulated particles to test the prediction respect to changing signal to noise ratio. This code can also be edited to train the neural network using the simulated particle images.

**beadtrack.py**

* beadtrack.py is a pipeline using functions from DeepTrack to load experimental images/videos of particles, and have a trained neural network predict the centroids of all the particles in the input data. The pipeline has been divided into cells defining different steps.
* The first cell is used to preprocess and upload the experimental data either in form of image or video, and define the scanning box parameters using the variables:
* *video\_file\_name*: Enter the name of the image or video file with the extension.
* *box\_half\_size*: Enter the size of the scanning box window that scans each frame.
* *box\_scanning\_size*: Enter the step size for the scanning window.
* *frame\_enhance and frame\_normalise*: Enter values to optimise and adjust pixel values for preprocessing.
* The second cell is used to load a pre-trained convoluted neural network:
* *saved\_network\_file\_name*: Enter the name of the trained neural network.
* *network*: Used to load the neural networkfor localisation.
* The third cell uses DeepTrack's *track\_video* function to track particles by scanning each frame of the video by the predefined scanning box or window.
* The fourth cell defines filter parameters to average multiple localisations of the same particle(*particle\_maximum\_interdistance*), and radial threshold to localise a particle for a scanning box(*particle\_radial\_distance\_threshold*).
* The fifth cell uses DeepTrack's *plot\_scanning\_boxes* function to display particular scanning boxes and the localisation in each case. The scanning boxes to be displayed are defined by the variables *frames\_to\_be\_shown*, *rows\_to\_be\_shown*, and *columns\_to\_be\_shown*.
* The sixth cell uses DeepTrack's *show\_tracked\_frames* function to display entire frames and all the localisation in each case. The frames to be displayed are defined by the variables *number\_frames\_to\_be\_shown,* and *frame.*

**References**

* Helgadottir, S., Argun, A. & Volpe, G. Digital video microscopy enhanced by deep learning. *Optica***6**, 506-513, doi:10.1364/OPTICA.6.000506 (2019).