Seg_Cell: a newly developed Python tool for a fast and automatic counting of cells in a .lif file

ISTITUTO ITALIANO DI TECNOLOGIA
ELECTRON SPECTROSCOPY AND NANOSCOPY

M. Lizzano[1] [2], A. Griesi[1], G. Divitini[1]

Email: Mattia.lizzano@iit.it
Github Page: https://github.com/IIT-ElSpy/Seg_cell

[1] Istituto Italiano di Tecnologia, Via morego 30, 16163, GE, Italy
[2] Università degli Studi di Genova, Via dodecaneso 33, 16146, GE, Italy

Abstract

The analysis of optical microscopy images to extract statistical information on tumoral cell morphology is crucial to a lot of biological research projects. However, currently available tools often lack the capability to iteratively analyse large datasets of images contained in a .lif file with high accuracy, while maintaining relatively low processing times per image. To remove this bottleneck, we developed a Python script that overcomes these limitations, enabling faster and more precise analysis of large image datasets containing hundred of cells per image. This new tool has proven to be highly accurate in counting the number of cells, while maintaining a low processing time per image.

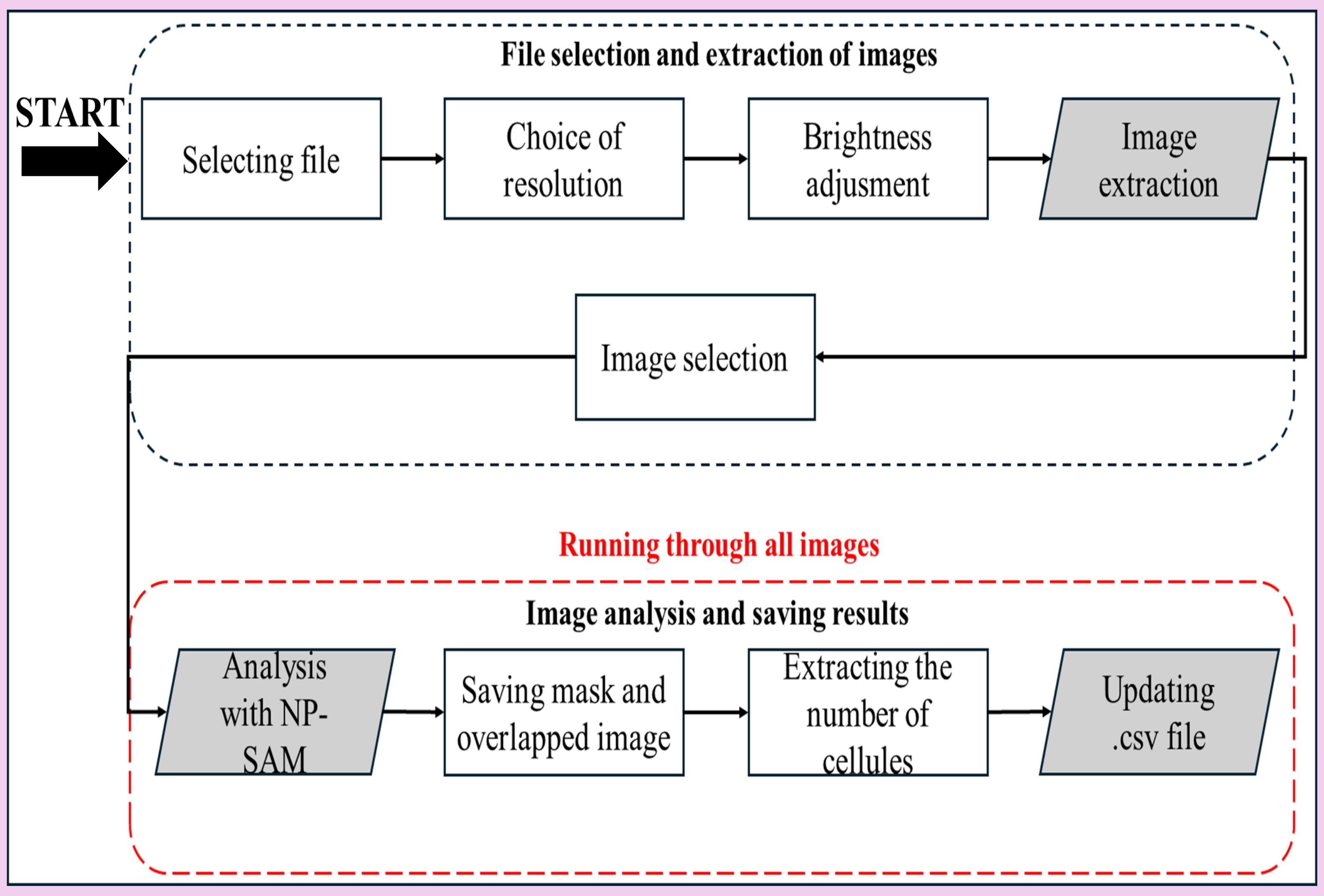
Introduction

The analysis of biological samples, such as tumour cells, often requires the calculation of statistical parameters, such as the number of tumour foci or cells. These values are obtained by averaging the results from each image within a dataset, and the accuracy of these measurements improves as the size of the dataset increases. Estimating these parameters can be done using several open-source tools, such as *FIJI* or *scikit-learn*, which allow for manual (or automatic) counting of cells in an image. However, the method for counting has both advantages and disadvantages: manual analysis ensures greater accuracy but requires more time compared to automatic counting, albeit with the risk of operator bias. The latter is less precise, as automatic detection of cell numbers can be inaccurate due to overlaps that are difficult for the software to identify. In both cases, the available tools allow for the analysis of one image at a time, making the analysis of large datasets time-consuming and potentially imprecise. To overcome this limitation, our group developed a Python script, Seg_cell, that processes .lif files (Leica Image File), extracts the images, and analyses them iteratively using segmentation software designed for electron microscopy image analysis, NP-SAM [1]. The final result consists of images with a mask showing the segmented cells and an .csv file containing the cell count for each image.

Workflow

As described in the figure, the user selects a .lif file and whether to reduce the image size, either using the original (2048×2048 being the common format for .lif files) or a lower resolution obtained by rebinning (512×512). This choice affects the time required for the analysis. After a guided manual brightness adjustment, the script extracts all the images from the .lif file (which contains a stack of 3-channel images) and places them into a specific folder. The user can then choose whether to analyse all images or only a subset. The NP-SAM algorithm is then invoked to perform the segmentation and return the number of cells, an image with a mask highlighting the segmented cells, and an overlaid image combining the mask and the original. This process is repeated for every image in the folder, providing an accurate and automated tool for counting cells in each image of the dataset.

An analysis conducted using both 2048×2048 and 512×512-pixel datasets showed that NP-SAM achieves 95–98% accuracy in counting cells in both cases, while maintaining a processing time of ~5 minutes per image (for 2048×2048) or ~3 minutes (512×512-pixel images). These results were obtained using an AMD Ryzen Threadripper PRO 32-core processor, and the processing time may vary depending on the hardware used, with potential for significant improvements using a GPU. Future updates will focus on integrating additional parameters, as well as extending the analysis to the other channels within the .lif file.


File selection and extraction of images
START
Selecting file
Choice of resolution
Brightness adjusment
Image extraction
Image selection
Running through all images
Image analysis and saving results
Analysis with NP-SAM
Saving mask and overlapped image
Extracting the number of cellules
Updating .csv file

Conclusion

We developed a tool capable of automatically analysing all the images contained within a .lif file, significantly optimising the workflow for processing large datasets. Future developments aim to integrate the analysis of additional morphological components, and the multiple channels present within the file, with the goal of creating a unified pipeline that ensures efficiency and precision in the extraction of statistical parameters, enhancing the robustness and reliability of the analysis.

1. Larsen R, Villadsen TL, Mathiesen JK, Jensen KMØ, Boejesen ED. NP-SAM: Implementing the Segment Anything Model for Easy Nanoparticle Segmentation in Electron Microscopy Images. ChemRxiv. 2023; doi:10.26434/chemrxiv-2023-k73qz-v2