Check for updates

OpSeF: Open Source Python Framework for Collaborative Instance Segmentation of Bioimages

Tobias M. Rasse[1]*, Réka Hollandi[2] and Peter Horvath[2,3]

[1] Scientific Service Group Microscopy, Max Planck Institute for Heart and Lung Research, Bad Nauheim, Germany,
[2] Synthetic and Systems Biology Unit, Biological Research Center (BRC), Szeged, Hungary, [3] Institute for Molecular Medicine Finland (FIMM), University of Helsinki, Helsinki, Finland

Various pre-trained deep learning models for the segmentation of bioimages have been made available as developer-to-end-user solutions. They are optimized for ease of use and usually require neither knowledge of machine learning nor coding skills. However, individually testing these tools is tedious and success is uncertain. Here, we present the Open Segmentation Framework (OpSeF), a Python framework for deep learning-based instance segmentation. OpSeF aims at facilitating the collaboration of biomedical users with experienced image analysts. It builds on the analysts' knowledge in Python, machine learning, and workflow design to solve complex analysis tasks at any scale in a reproducible, well-documented way. OpSeF defines standard inputs and outputs, thereby facilitating modular workflow design and interoperability with other software. Users play an important role in problem definition, quality control, and manual refinement of results. OpSeF semi-automates preprocessing, convolutional neural network (CNN)-based segmentation in 2D or 3D, and postprocessing. It facilitates benchmarking of multiple models in parallel. OpSeF streamlines the optimization of parameters for pre- and postprocessing such, that an available model may frequently be used without retraining. Even if sufficiently good results are not achievable with this approach, intermediate results can inform the analysts in the selection of the most promising CNN-architecture in which the biomedical user might invest the effort of manually labeling training data. We provide Jupyter notebooks that document sample workflows based on various image collections. Analysts may find these notebooks useful to illustrate common segmentation challenges, as they prepare the advanced user for gradually taking over some of their tasks and completing their projects independently. The notebooks may also be used to explore the analysis options available within OpSeF in an interactive way and to document and share final workflows. Currently, three mechanistically distinct CNN-based segmentation methods, the U-Net implementation used in Cellprofiler 3.0, StarDist, and Cellpose have been integrated within OpSeF. The addition of new networks requires little; the addition of new models requires no coding skills. Thus, OpSeF might soon become both an interactive model repository, in which pre-trained models might be shared, evaluated, and reused with ease.

Keywords: deep learning, biomedical image analysis, segmentation, convolutional neural network, U-net, cellpose, StarDist, python

INTRODUCTION

Phenomics, the assessment of the set of physical and biochemical properties that completely characterize an organism, has long been recognized as one of the most significant challenges in modern biology (Houle et al., 2010). Microscopy is a crucial technology to study phenotypic characteristics. Advances in high-throughput microscopy (Lang et al., 2006; Neumann et al., 2010; Chessel and Carazo Salas, 2019), slide scanner technology (Webster and Dunstan, 2014; Wang et al., 2019), light-sheet microscopy (Swoger et al., 2014; Ueda et al., 2020), semi-automated (Bykov et al., 2019; Schorb et al., 2019) and volume electron microscopy (Titze and Genoud, 2016; Vidavsky et al., 2016), as well as correlative light- and electron microscopy (Hoffman et al., 2020) have revolutionized the imaging of organisms, tissues, organoids, cells, and subcellular structures. Due to the massive amount of data produced by these approaches, the traditional biomedical image analysis tool of "visual inspection" is no longer feasible, and classical, non-machine learning-based image analysis is often not robust enough to extract phenotypic characteristics reliably in a non-supervised manner.

Thus, the advances mentioned above were enabled by breakthroughs in the application of machine learning methods to biological images. Traditional machine learning techniques, based on random-forest classifiers and support vector machines, were made accessible to biologists with little to no knowledge in machine learning, using stand-alone tools such as ilastik (Haubold et al., 2016; Berg et al., 2019; Kreshuk and Zhang, 2019) or QuPath (Bankhead et al., 2017). Alternatively, they were integrated into several image analysis platforms such as Cellprofiler (Lamprecht et al., 2007), Cellprofiler Analyst (Jones et al., 2009), Icy (de Chaumont et al., 2012), ImageJ (Schneider et al., 2012; Arganda-Carreras et al., 2017) or KNIME (Sieb et al., 2007).

More recently, deep learning methods, initially developed for computer vision challenges, such as face recognition or autonomous cars, have been applied to biomedical image analysis (Cireşan et al., 2012, 2013). The U-Net is the most commonly used deep convolutional neural network specifically designed for semantic segmentation of biomedical images (Ronneberger et al., 2015). In the following years, neural networks were broadly applied to biomedical images (Zhang et al., 2015; Akram et al., 2016; Albarqouni et al., 2016; Milletari et al., 2016; Moeskops et al., 2016; Van Valen et al., 2016; Çiçek et al., 2016; Rajchl et al., 2017). Segmentation challenges like the 2018 Data Science Bowl (DSB) further promoted the adaptation of computer vision algorithms like Mask R-CNN (He et al., 2017) to biological analysis challenges (Caicedo et al., 2019). The DSB included various classes of nuclei. Schmidt et al. use the same dataset to demonstrate that star-convex polygons are better suited to represent densely packed cells (Schmidt et al., 2018) than axis-aligned bounding boxes used in Mask R-CNN (Hollandi et al., 2020b). Training of deep learning models typically involves tedious annotation to create ground truth labels. Approaches that address this limitation include optimizing the annotation workflow by starting with reasonably good predictions (Hollandi et al., 2020a), applying specific preprocessing steps such that an existing model can be used (Whitehead, 2020), and the use of generalist algorithms trained on highly variable images (Stringer et al., 2020). Following the latter approach, Stringer et al. trained a neural network to predict vector flows generated by the reversible transformation of a highly diverse image collection. Their model includes a function to auto-estimate the scale. It works well for specialized and generalized data (Stringer et al., 2020).

Recently, various such pre-trained deep learning segmentation models have been published that are intended for non-machine learning experts in the field of biomedical image processing (Schmidt et al., 2018; Hollandi et al., 2020b; Stringer et al., 2020). Testing such models on new data sets can be time-consuming and might not always give good results. Pre-trained models might fail because the test images do not resemble the data network was trained on sufficiently well. Alternatively, the underlying network architecture and specification, or the way data is internally represented and processed might not be suited for the presented task. Biomedical users with no background in computer science are often unable to distinguish these possibilities. They might erroneously conclude that their problem is in principle not suited for deep learning-based segmentation. Thus, they might hesitate to create annotations to re-train the most appropriate architecture. Here, we present the **Op**en **Se**gmentation **F**ramework OpSeF, a Python framework for deep-learning-based instance segmentation of cells and nuclei. OpSeF has primarily been developed for staff image analysts with solid knowledge in image analysis, thorough understating of the principles of machine learning, and basic skills in Python. It wraps scikit-image, a collection of Python algorithms for image processing (van der Walt et al., 2014), the U-Net implementation used in Cellprofiler 3.0 (McQuin et al., 2018), StarDist (Schmidt et al., 2018; Weigert et al., 2019, 2020), and Cellpose (Stringer et al., 2020) in a single framework. OpSeF defines the standard in- and outputs, facilitates modular workflow design, and interoperability with other software (Weigert et al., 2020). Moreover, it streamlines and semi-automates preprocessing, CNN-based segmentation, postprocessing as well as evaluation of results. Jupyter notebooks (Kluyver et al., 2016) serve as a minimal graphical user interface. Most computations are performed head-less and can be executed on local workstations as well as on GPU clusters. Segmentation results can be easily imported and refined in ImageJ using AnnotatorJ (Hollandi et al., 2020a).

MATERIALS AND METHODS

Data Description

Cobblestones

Images of cobblestones were taken with a Samsung Galaxy S6 Active Smartphone.

Leaves

Noise was added to the demo data from "YAPiC - Yet Another Pixel Classifier" available at https://github.com/

yapic/yapic/tree/master/docs/example_data using the *Add Noise* function in ImageJ.

Small Fluorescent Nuclei

Images of Hek293 human embryonic kidney stained with a nuclear dye from the image set BBBC038v1 (Caicedo et al., 2019) available from the Broad Bioimage Benchmark Collection (BBBC) were used. Metadata is not available for this image set to confirm staining conditions. Images were rescaled from 360×360 pixels to 512×512 pixels.

3D Colon Tissue

We used the low signal-to-noise variant of the image set BBBC027 (Svoboda et al., 2011) from the BBBC showing 3D colon tissue images.

Epithelial Cells

Images of cervical cells from the image set BBBC038v1 (Caicedo et al., 2019) available from the BBBC display cells stained with a dye that labels membranes weakly and nuclei strongly. The staining pattern is reminiscent of images of methylene blue-stained cells. However, metadata is not available for this image set to confirm staining conditions.

Skeletal Muscle

A methylene blue-stained skeletal muscle section was recorded on a Nikon Eclipse Ni-E microscope equipped with a Märzhäuser SlideExpress2 system for automated handling of slides. The pixel size is 0.37×0.37 μm. Thirteen large patches of 2048×2048 pixels size were manually extracted from the original 44712×55444 pixels large image. Color images were converted to grayscale.

Kidney

HE stained kidney paraffin sections were recorded on a Nikon Eclipse Ni-E microscope equipped with a Märzhäuser SlideExpress2 system for automated handling of slides. The pixel size is 180×180 nm. The original, stitched, 34816×51200 pixels large image was split into two large patches (18432×6144 and 22528×5120 pixel). Next, the Eosin staining was extracted using the *Color Deconvolution* ImageJ plugin. This plugin implements the method described by Ruifrok and Johnston (2001).

Arabidopsis Flowers

H2B:mRuby2 was used for the visualization of somatic nuclei of *Arabidopsis thaliana* flower. The flower was scanned from eight views differing by 45° increments in a Zeiss Z1 light-sheet microscope (Valuchova et al., 2020). We used a single view to mimic a challenging 3D segmentation problem. Image files are available in the Image Data Resource (Williams et al., 2017) under the accession code: idr0077.

Mouse Blastocysts

The DAPI signal from densely packed E3.5 mouse blastocysts nuclei was recorded on a Leica SP8 confocal microscope using a $40\times$ 1.30 NA oil objective (Blin et al., 2019). Image files are available in the Image Data Resource (Williams et al., 2017) under the accession code: idr0062.

Neural Monolayer

The DAPI signal of a neural monolayer was recorded on a Leica SpE confocal microscope using a $63\times$ 1.30 NA oil objective (Blin et al., 2019). Image files are available in the Image Data Resource (Williams et al., 2017) under the accession code: idr0062.

Algorithm

Ideally, OpSeF is used as part of collaborative image analysis projects, to which both the user and the image analyst contribute their unique expertise (**Figure 1A**). All analyst tasks are optimized for deployment on Linux workstations or GPU clusters, all user tasks may be performed on any laptop in ImageJ. If challenges arise, the image analyst (**Figure 1A**) might consult other OpSeF users or the developer of tools used within OpSeF. The analyst will – to the benefit of future users – become more skilled using CNN-based segmentation in analysis workflows. The user, who knows the sample best, plays an important role in validating results and discovering artifacts (**Figure 1A**). Exemplary workflows and new models might be shared to the benefit of other OpSeF users (**Figure 1B**).
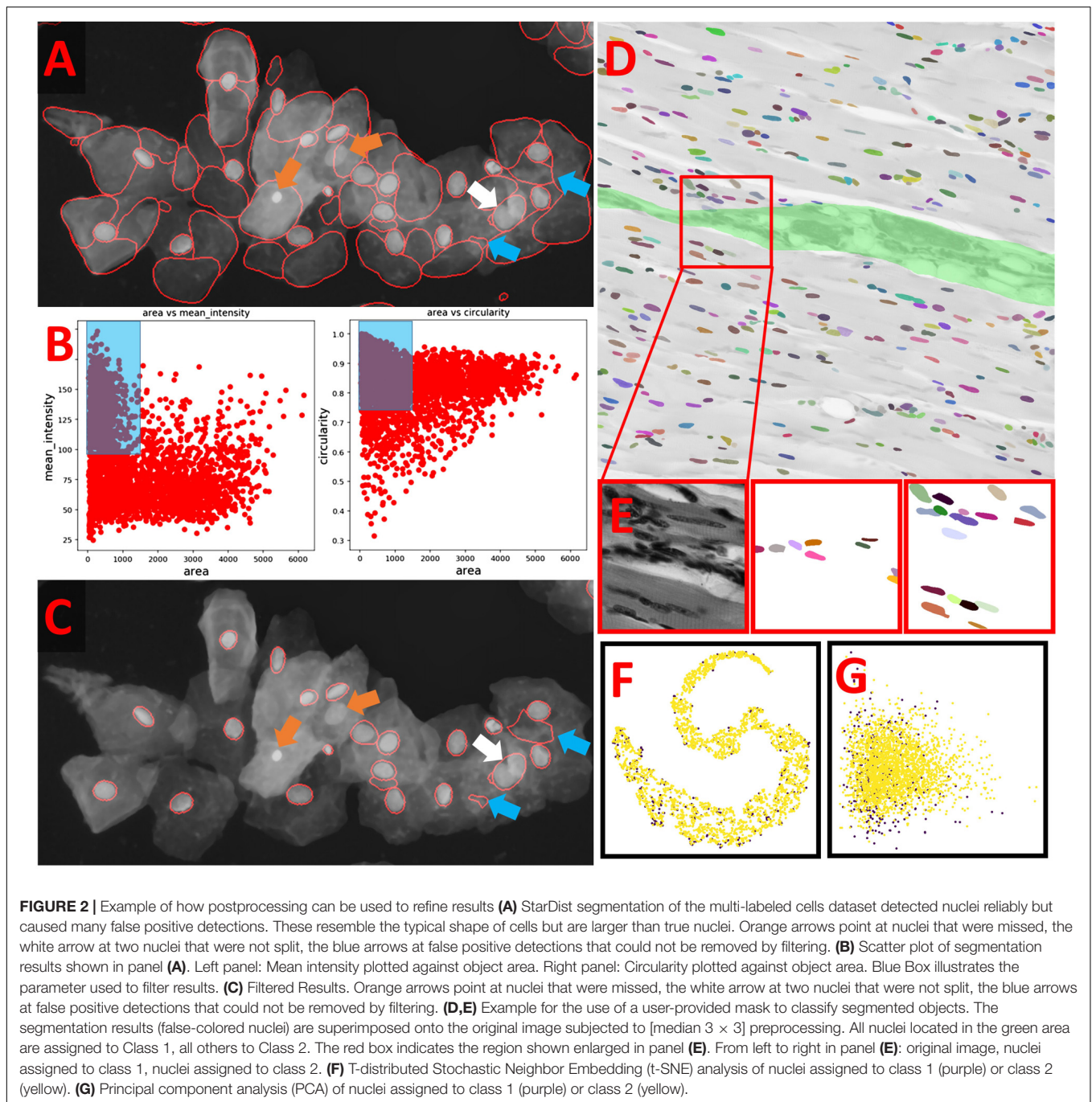
OpSeF's analysis pipeline consists of four principal sets of functions to *import and reshape* the data, to *preprocess* it, to *segment* objects, and to *analyze and classify* results (**Figure 1C**). Currently, OpSeF can process individual tiff files and the proprietary Leica '.lif' container file format. During *import and reshape*, the following options are available for tiff-input: *tile* in 2D and 3D, *scale*, and make *sub-stacks*. For lif-files, only the make *sub-stacks* option is supported. Preprocessing is mainly based on scikit-image (van der Walt et al., 2014). It consists of a linear workflow in which 2D images are filtered, the background is removed, and stacks are projected. Next, the following optional preprocessing operations might be performed: histogram adjustment (Zuiderveld, 1994), edge enhancement, and inversion of images. Available segmentation options include the pre-trained U-Net used in Cellprofiler 3.0 (McQuin et al., 2018), the so-called "2D_paper_dsb2018" StarDist model (Schmidt et al., 2018) and Cellpose (Stringer et al., 2020). The 2D StarDist model (Schmidt et al., 2018) was trained on a subset of fluorescent images from the 2018 Data Science Bowl (DSB) (Caicedo et al., 2019). Although good performance on non-fluorescent images cannot be taken for granted, the StarDist versatile model, which was trained on the same data, generalizes well and can be used to segment cells in diaminobenzidene and hematoxylin stained tissue sections (Whitehead, 2020). We thus used the 2D_paper_dsb2018 StarDist model for all 2D examples. Available options for preprocessing in 3D are limited (**Figure 1B**, lower panel). Segmentation in 3D is computationally more demanding. Thus, we recommend a two-stage strategy for Cellpose 3D. Preprocessing parameters are first explored on representative planes in 2D. Next, further optimization in 3D is performed. Either way, preprocessing and selection of the ideal model for segmentation are one functional unit. **Figure 1D** illustrates this concept with a processing pipeline, in which three different models are applied to four different preprocessing pipelines each. The resulting images are classified
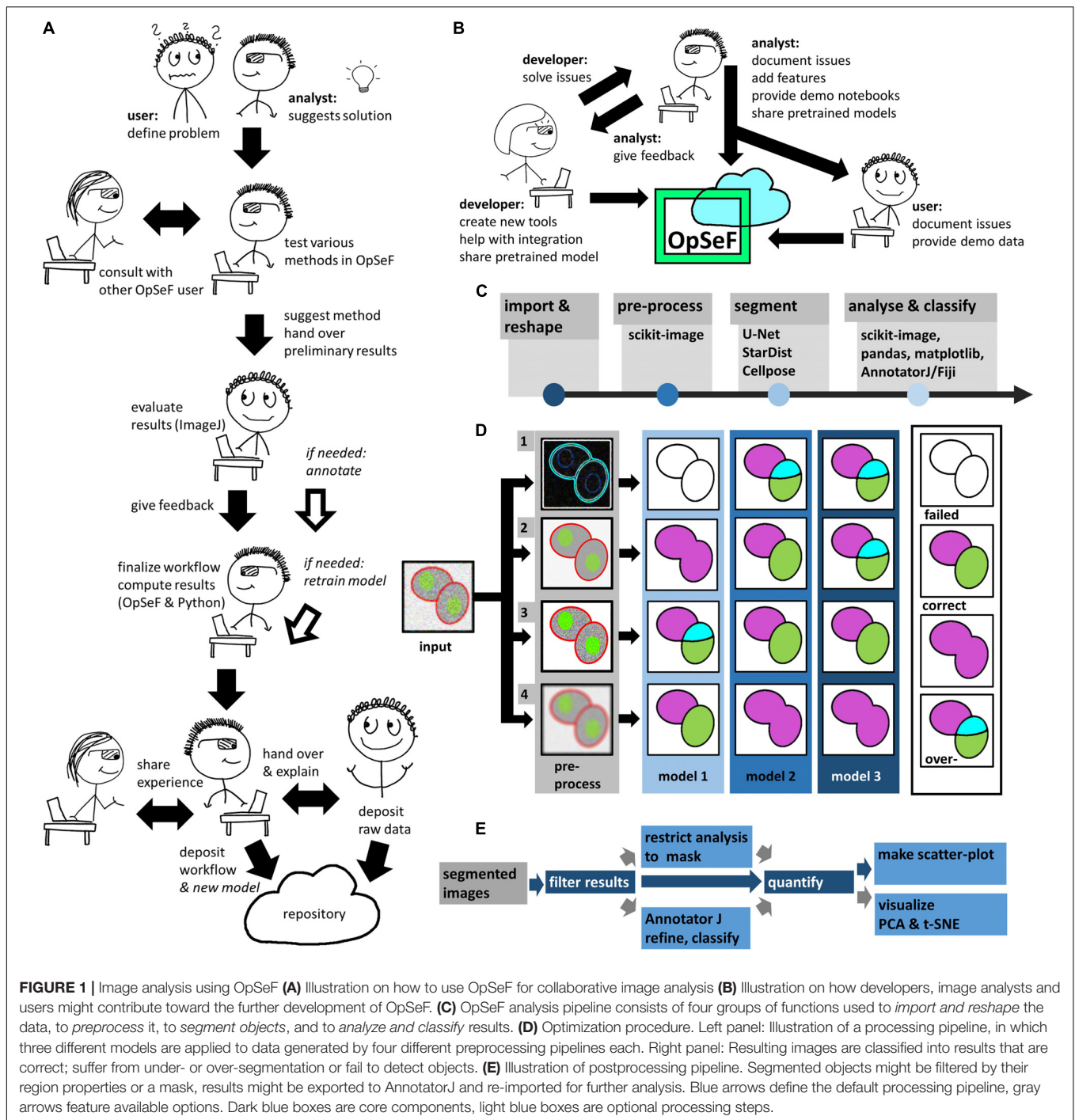
FIGURE 1 | Image analysis using OpSeF **(A)** Illustration on how to use OpSeF for collaborative image analysis **(B)** Illustration on how developers, image analysts and users might contribute toward the further development of OpSeF. **(C)** OpSeF analysis pipeline consists of four groups of functions used to *import and reshape* the data, to *preprocess* it, to *segment objects*, and to *analyze and classify* results. **(D)** Optimization procedure. Left panel: Illustration of a processing pipeline, in which three different models are applied to data generated by four different preprocessing pipelines each. Right panel: Resulting images are classified into results that are correct; suffer from under- or over-segmentation or fail to detect objects. **(E)** Illustration of postprocessing pipeline. Segmented objects might be filtered by their region properties or a mask, results might be exported to AnnotatorJ and re-imported for further analysis. Blue arrows define the default processing pipeline, gray arrows feature available options. Dark blue boxes are core components, light blue boxes are optional processing steps.

into results that are mostly correct, suffer from under- or over-segmentation, or largely fail to detect objects. In the given example, the combination of preprocessing pipeline three and model two gives overall the best result. We recommend an iterative optimization which starts with a large number of models, and relatively few, but conceptually different preprocessing pipelines. For most datasets, some models outperform others. In this case, we recommend fine-tuning the most promising preprocessing pipelines in combination with the most promising

model. OpSeF uses matplotlib (Virtanen et al., 2020) to visualize results in Jupyter notebooks and to export exemplary results that may be used as figures in publications. All data is managed in pandas (Virtanen et al., 2020) and might be exported as csv file. Scikit-image (van der Walt et al., 2014), and scikit-learn (Pedregosa et al., 2011; **Figures 1E**, **2A–C**) are used for pre- and postprocessing of segmentation results, which might e.g., be filtered based on their size, shape or other object properties (**Figure 2B**). Segmentation objects may further be

FIGURE 2 | Example of how postprocessing can be used to refine results **(A)** StarDist segmentation of the multi-labeled cells dataset detected nuclei reliably but caused many false positive detections. These resemble the typical shape of cells but are larger than true nuclei. Orange arrows point at nuclei that were missed, the white arrow at two nuclei that were not split, the blue arrows at false positive detections that could not be removed by filtering. **(B)** Scatter plot of segmentation results shown in panel **(A)**. Left panel: Mean intensity plotted against object area. Right panel: Circularity plotted against object area. Blue Box illustrates the parameter used to filter results. **(C)** Filtered Results. Orange arrows point at nuclei that were missed, the white arrow at two nuclei that were not split, the blue arrows at false positive detections that could not be removed by filtering. **(D,E)** Example for the use of a user-provided mask to classify segmented objects. The segmentation results (false-colored nuclei) are superimposed onto the original image subjected to [median 3×3] preprocessing. All nuclei located in the green area are assigned to Class 1, all others to Class 2. The red box indicates the region shown enlarged in panel **(E)**. From left to right in panel **(E)**: original image, nuclei assigned to class 1, nuclei assigned to class 2. **(F)** T-distributed Stochastic Neighbor Embedding (t-SNE) analysis of nuclei assigned to class 1 (purple) or class 2 (yellow). **(G)** Principal component analysis (PCA) of nuclei assigned to class 1 (purple) or class 2 (yellow).

refined by a user-provided (**Figures 2D,E**) or an autogenerated mask. Results might be exported to AnnotatorJ (Hollandi et al., 2020a) for editing or classification in ImageJ. AnnotatorJ is an ImageJ plugin that helps hand-labeling data with deep learning-supported semi-automatic annotation and further convenient functions to create and edit object contours easily. It has been extended with a classification mode and import/export fitting the data structure used in OpSeF. After refinement, results can be re-imported and further analyzed in OpSeF. Analysis options include scatter plots of region properties

(**Figure 2B**), T-distributed Stochastic Neighbor Embedding (t-SNE) analysis (**Figure 2F**), and principal component analysis (PCA) (**Figure 2G**).

RESULTS

We provide demonstration notebooks to illustrate how OpSeF might be used to elucidate efficiently whether a given segmentation task is solvable with state of the art deep

convolutional neural networks (CNNs). In the first step, preprocessing parameters are optimized. Next, we test whether the chosen model performs well without re-training. Finally, we assess how well it generalizes on heterogeneous datasets.

Preprocessing can be used to make the input image more closely resemble the visual appearance of the data on which the models for the CNNs bundled with OpSeF were trained on, e.g., by filtering and resizing. Additionally, preprocessing steps can be used to normalize data and reduce heterogeneity. Generally, there is not a single, universally best preprocessing pipeline. Instead, well-performing combinations of preprocessing pipelines and matching CNN-models can be identified. Even the definition of a "good" result depends on the biological question posed and may vary from project to project. For cell tracking, very reproducible cell identification will be of utmost importance; for other applications, the accuracy of the outline might be more crucial. To harness the full power of CNN-based segmentation models and to build trust in their more widespread use, it is essential to understand under which conditions they are prone to fail.

We use various demo datasets to challenge the CNN-based segmentation pipelines. Jupyter notebooks document how OpSeF was used to obtain reliable results. These notebooks are provided as a starting point for the iterative optimization of user projects and as a tool for interactive user training.

The first two datasets – cobblestones and leaves – are generic, non-microscopic image collections, designed to illustrate common analysis challenges. Further datasets exemplify the segmentation of a monolayer of fluorescent cells, fluorescent tissues, cells in which various compartments have been stained with the same dye, as well as histological sections stained with one or two dyes. The latter dataset exemplifies additionally how OpSeF can be used to process large 2D images.

Nuclei and cells used to train CNN-based segmentation are most commonly round or ellipsoid shaped. Objects in the cobblestone dataset are approximately square-shaped. Thus, the notebook may be used as an example to explore the segmentation of non-round cells (e.g., many plant cells, neurons). Heterogeneous intensities within objects and in the border region, as well as a five-fold variation of object size, challenge segmentation pipelines further. In the first round of optimization, minor smoothing [median filter with 3×3 kernel (median 3×3)] and background subtraction were applied. Next, the effect of additional histogram equalization, edge enhancement, and image inversion was tested. The resulting four preprocessed images were segmented with all models [Cellpose nuclei, Cellpose Cyto, StarDist, and U-Net]. The Cellpose scale-factor range [0.2, 0.4, 0.6] was explored. Among the 32 resulting segmentation pipelines, the combination of image inversion and the Cellpose Cyto 0.4 model produced the best results in both training images (**Figures 3A–C**) without further optimization. The segmentation generalized well to the entire dataset. Only in one image, three objects were missed, and one object was over-segmented. Borders around these stones are very hard to distinguish for a human observer, and even further training might not resolve the presented segmentation tasks (**Figures 3D–F**). Cellpose has been trained on a large variety of images and had been reported to perform well on objects of similar shape [compare **Figure 4**,
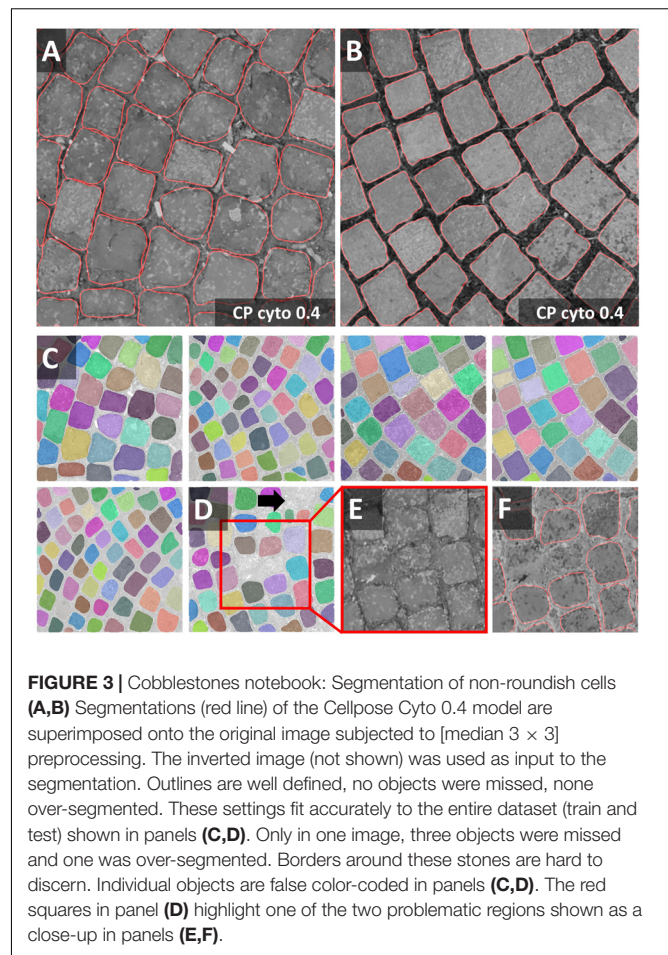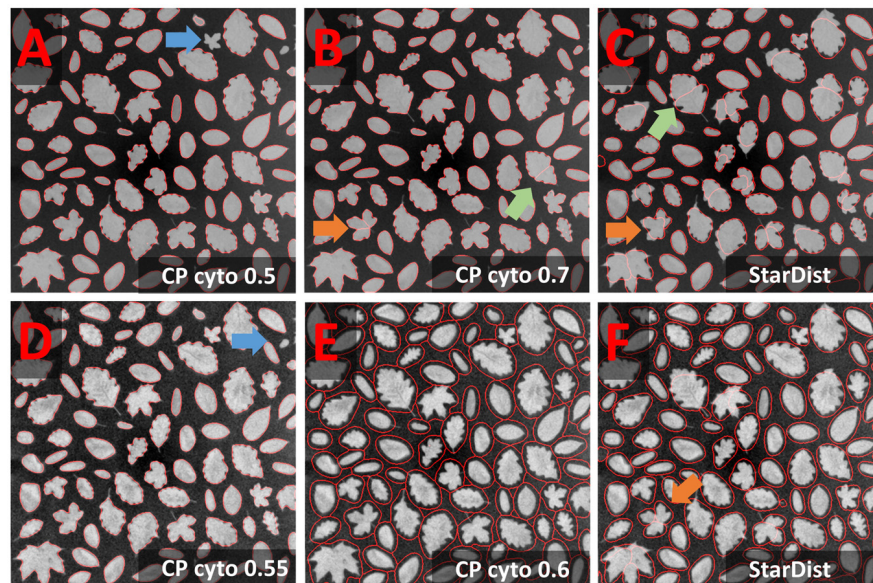

FIGURE 3 | Cobblestones notebook: Segmentation of non-roundish cells **(A,B)** Segmentations (red line) of the Cellpose Cyto 0.4 model are superimposed onto the original image subjected to [median 3×3] preprocessing. The inverted image (not shown) was used as input to the segmentation. Outlines are well defined, no objects were missed, none over-segmented. These settings fit accurately to the entire dataset (train and test) shown in panels **(C,D)**. Only in one image, three objects were missed and one was over-segmented. Borders around these stones are hard to discern. Individual objects are false color-coded in panels **(C,D)**. The red squares in panel **(D)** highlight one of the two problematic regions shown as a close-up in panels **(E,F)**.

Images 21, 22, 27 in Stringer et al. (2020)]. Thus, it is no surprise that Cellpose outperformed the StarDist 2D model (Schmidt et al., 2018), which had been trained only on fluorescent images.

Segmentation of the leaves dataset seems trivial and could easily be solved by any threshold-based approach. Nevertheless, it challenges CNN-based segmentation due to the presence of concave and convex shapes. Moreover, objects contain dark lines, vary 20-fold in area, and are presented on a heterogeneous background. Preprocessing was performed as described for the cobblestone dataset. The most promising result was obtained with the Cellpose Cyto 0.5 model in combination with [median 3×3 & image inversion] preprocessing (**Figures 4A,B**) and the StarDist model with [median 3×3 & histogram equalization] preprocessing (**Figure 4C**). Outlines were well defined, few objects were missed (blue arrow in **Figure 4A**), few over-segmented (green and orange arrow in **Figures 4B,C**). The Cellpose Cyto 0.7 model gave similar results.

Maple leaves (orange arrows in **Figures 4B,C**) were most frequently over-segmented. Their shape resembles a cluster of touching cells. Thus, the observed over-segmentation might be caused by the attempt of the CNN to reconcile their shapes with structures it has been trained on. Oak leaves were the second most frequently over-segmented leaf type. These leaves contain dark leaf veins that might be interpreted as

FIGURE 4 | Leaves notebook: Segmentation of rare concave cells **(A–C)** Segmentation results (red line) of the Cellpose Cyto 0.5 and 0.7 and StarDist model are superimposed onto the original image subjected to [median 3 × 3] preprocessing. The inverted image (not shown) was used as input to the segmentation. Outlines are well defined, few objects were missed (**A:** blue arrow), some over-segmented (**B,C:** green and orange arrow). Green arrow points at an oak leave with prominent leaf veins, orange arrow at maple leaves with less prominent leave veins. **(D–F)** Results of further optimization. Further smoothing **(E)** reduced the rate of false negatives (blue arrow) and over-segmentation in the Cellpose Cyto model. However, object outlines were less precise **(E)**.
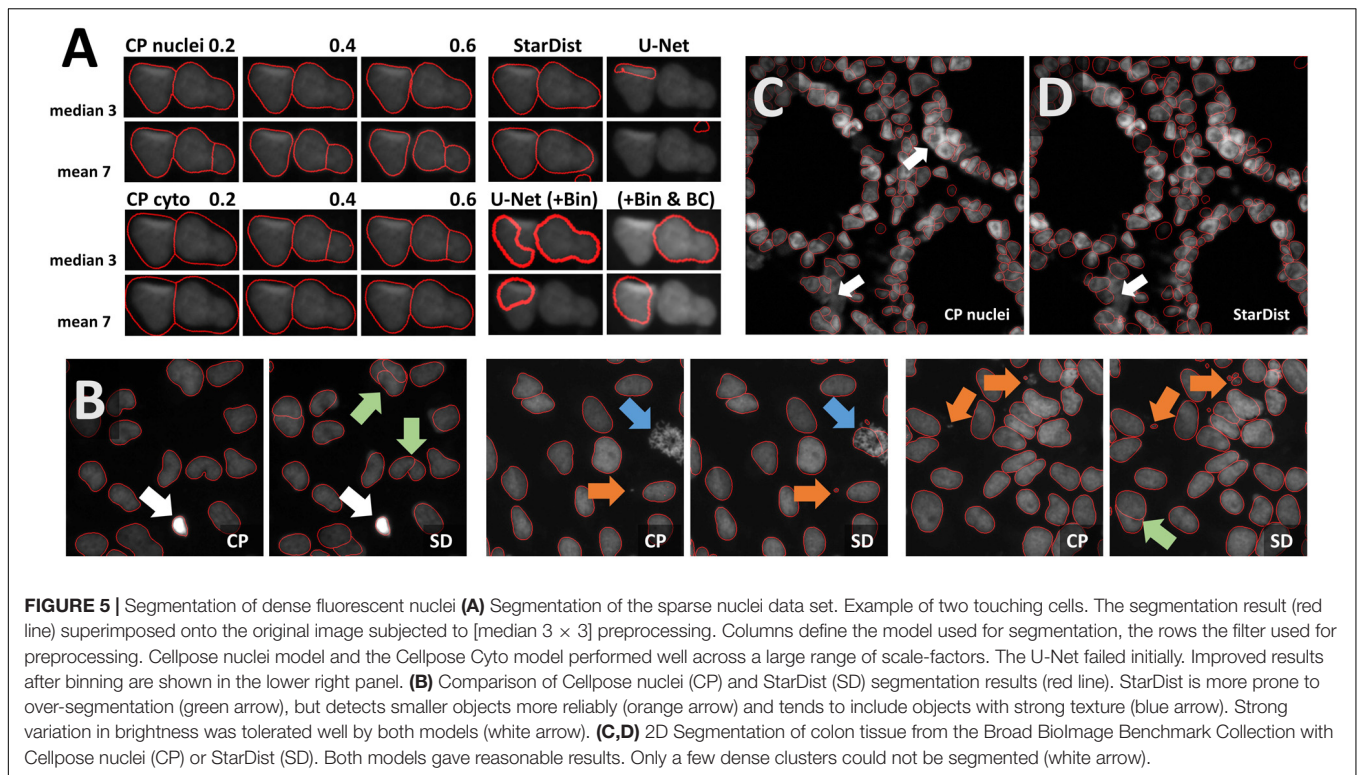
cell borders. However, erroneous segmentation mostly does not follow these veins (green arrow in **Figure 4B**). Next, the effect of stronger smoothing [mean 7 × 7] was explored. The Cellpose nuclei model (**Figure 4E**) reduced the rate of false-negative detections (**Figure 4D** blue arrow) and over-segmentation (**Figure 4F** orange arrow) at the expense of loss in precision of object outlines. Parameter combinations tested in **Figures 4D,E** generalize well in the entire dataset.

Next, we used OpSeF to segment nuclei in a monolayer of cells. Most nuclei are well separated. We focused our analysis on the few touching nuclei. Both the Cellpose nuclei model and the Cellpose Cyto model performed well across a broad range of scale-factors. Interestingly, strong smoothing made the Cellpose nuclei but not the Cellpose Cyto model more prone to over-segmentation (**Figure 5A**). The StarDist model performed well, while the U-Net failed surprisingly, given the seemingly simple task. Pixel intensities have a small dynamic range, and nuclei are dim and rather large. To elucidate whether any of these issues led to this poor performance, we binned the input 2 × 2 (U-Net+BIN panel in **Figure 5A**) and adjusted brightness and contrast. Adjusting brightness and contrast alone had no beneficial effect (data not shown). The U-Net performed much better on the binned input. Subsequently, we batch-processed the entire dataset. StarDist was more prone to over-segmentation (green arrow in **Figure 5B**), but detected smaller objects more faithfully (orange arrow in **Figure 5B**). This might indicate that the size of test objects was larger than the size of train objects. StarDist was more likely to include atypical objects, e.g., nuclei during cell division that display a strong texture (blue arrow in **Figure 5B**). Substantial variation in brightness was

well tolerated by both models (white arrow in **Figure 5B**). Both models complement each other well.

We also tested a more complex dataset: 3D colon tissue from the Broad Bioimage Benchmark Collection. This synthetic dataset is ideally suited to assess segmenting clustered nuclei in tissues. We chose the low signal-to-noise variant, which allowed us to test denoising strategies. Sum, maximum, and median projection of three Z-planes was tested in combination with the preprocessing variants previously described for the monolayer of cells dataset. Twelve different preprocessing pipelines were combined with all models [Cellpose nuclei, Cellpose Cyto, StarDist, and U-Net]. The Cellpose scale-factor range [0.15, 0.25, 0.4, 0.6] was explored. Many segmentation decisions in the 3D colon tissue dataset are hard to perform even for human experts. Within this limitation, [median projection & histogram equalization] preprocessing produced reasonable results without any further optimization in combination with either Cellpose nuclei 0.4 or the StarDist model (**Figures 5C,D**). Only a few cell clusters were not segmented (**Figures 5C,D** white arrow). Both models performed equally well on the entire data set.

We subsequently tried to segment a single layer of irregular-shaped epithelial cells, in which the nucleus and cell membranes had been stained with the same dye. In the first run, minor [median 3 × 3] or strong [mean 7 × 7] smoothing was applied. Next, the effect of additional histogram equalization, edge enhancement, and image inversion was tested. The resulting eight preprocessed images were segmented with all models [Cellpose nuclei, Cellpose Cyto, StarDist, and U-Net]. The Cellpose scale-factor range [0.6, 0.8, 1.0, 1.4, 1.8] was explored. The size of nuclei varied more than five-fold. We thus focused

FIGURE 5 | Segmentation of dense fluorescent nuclei **(A)** Segmentation of the sparse nuclei data set. Example of two touching cells. The segmentation result (red line) superimposed onto the original image subjected to [median 3 × 3] preprocessing. Columns define the model used for segmentation, the rows the filter used for preprocessing. Cellpose nuclei model and the Cellpose Cyto model performed well across a large range of scale-factors. The U-Net failed initially. Improved results after binning are shown in the lower right panel. **(B)** Comparison of Cellpose nuclei (CP) and StarDist (SD) segmentation results (red line). StarDist is more prone to over-segmentation (green arrow), but detects smaller objects more reliably (orange arrow) and tends to include objects with strong texture (blue arrow). Strong variation in brightness was tolerated well by both models (white arrow). **(C,D)** 2D Segmentation of colon tissue from the Broad BioImage Benchmark Collection with Cellpose nuclei (CP) or StarDist (SD). Both models gave reasonable results. Only a few dense clusters could not be segmented (white arrow).

our analysis on a cluster of particularly large nuclei and a cluster of small nuclei. The Cellpose nuclei 1.4 and StarDist model detected both small and large nuclei similarly well (**Figure 6A**). StarDist segmentation results included many cell-shaped false positive detections. Given the model was trained on different data, retraining would be the best way to improve performance. Alternatively, false-positive detections, which were much larger than true nuclei, could be filtered out during postprocessing. While the U-Net did not perform well on the same input [median 3 × 3] (**Figure 6A**), it returned better results (**Figure 6A**) upon [mean 7 × 7 & histogram equalization] preprocessing. As weak smoothing was beneficial for the Cellpose and StarDist pipelines and stronger smoothing for the U-Net pipelines, we explored the effect of intermediate smoothing [median 5 × 5] for Cellpose and StarDist and even stronger smoothing [mean 9 × 9] for the U-Net pipelines. A slight improvement was observed. Thus, we used [median 5 × 5] preprocessing in combination with Cellpose nuclei 1.5 or StarDist model to process the entire dataset. Cellpose frequently failed to detect bright, round nuclei (**Figure 6B**, arrows) and StarDist (**Figure 6C**) had many false detections. Thus, re-training or postprocessing is required.

In the DSB, most algorithms performed better on images classified as small or large fluorescent, compared to images classified as "purple tissue" or "pink and purple" tissue. We used methylene blue-stained skeletal muscle sections as a sample dataset for tissue stained with a single dye and Hematoxylin and eosin (HE) stained kidney paraffin sections as an example for multi-dye stained tissue. Analysis of tissue sections might be compromised by heterogenous image quality cause e.g., by

artifacts created at the junctions of tiles. To account for these artifacts all workflows used the fused image as input to the analysis pipeline.

While most nuclei in the skeletal muscle dataset are well separated, some form dense clusters, others are out of focus (**Figure 7A**). The size of nuclei varies ten-fold; their shape ranges from elongated to round. The same preprocessing and model as described for the epithelial cells dataset were used; the Cellpose scale-factor range [0.2, 0.4, 0.6] was explored. [Median 3 × 3 & invert image] preprocessing combined with the Cellpose nuclei 0.6 model produced satisfactory results without further optimization (**Figure 7B**). Outlines were well defined, some objects were missed, few over-segmented. Neither StarDist nor the U-Net performed similarly well. We could not overcome this limitation by adaptation of preprocessing or binning. The performance of other – most likely more appropriate – StarDist model (2D_versatile_fluo, 2D_versatile_he) was not tested. Processing of the entire dataset identified inadequate segmentation of dense clusters (**Figure 7C**, white arrow) and occasional over-segmentation of large, elongated nuclei (**Figure 7C**, orange arrow) as the main limitations. Nuclei that are out-of-focus were frequently missed (**Figure 7C**, blue arrow). Limiting the analysis to in-focus nuclei is feasible.

Cell density is very heterogeneous in the kidney dataset. The Eosin signal from a HE stained kidney paraffin section (**Figures 8A,B**) was obtained by color deconvolution. Nuclei are densely packed within glomeruli and rather sparse in the proximal and distal tubules. Two stitched images were split using OpSeF's "to tiles" function. Initial optimization was performed on a batch of 16 image tiles, the entire dataset contains 864 tiles.
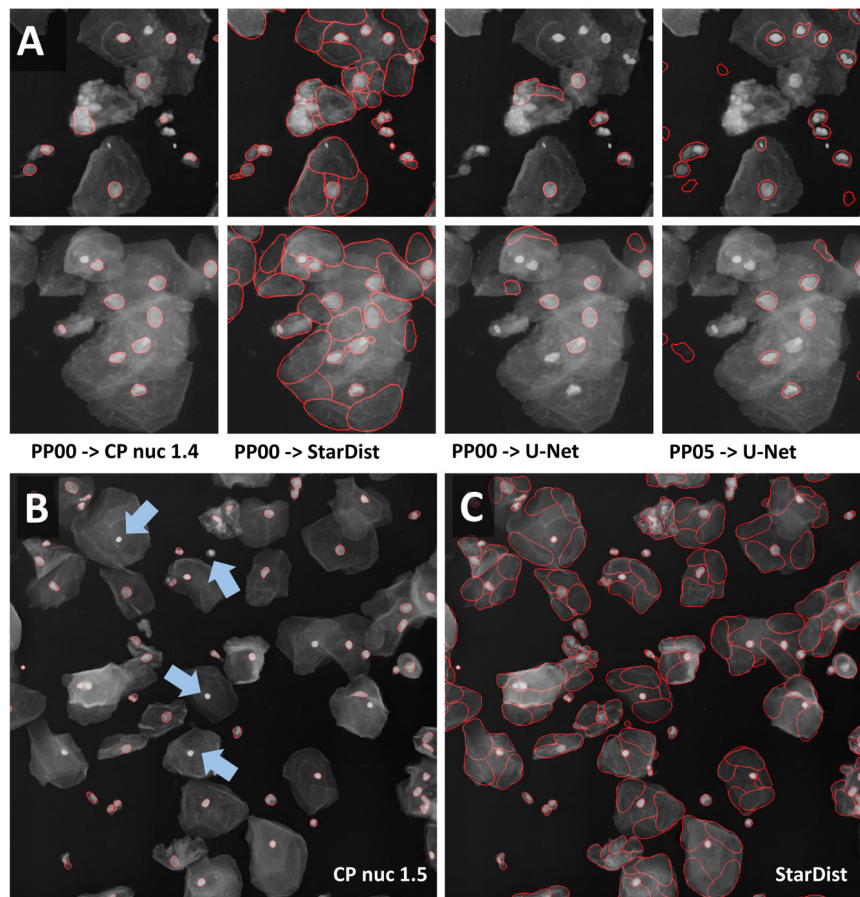
FIGURE 6 | Segmentation of multi-labeled cells **(A)** Images of large epithelial cells from the 2018 Data Science Bowl collection were used to test the segmentation of a single layer of cells. These cells vary in size. The Cellpose nuclei 1.4 model and [median 3 × 3] preprocessing gave a reasonable segmentation for large and small nuclei. StarDist segmentation based on the same input detected nuclei more reliably. However, many false-positive detections were present. Interestingly, the shape of false detections resembles the typical shape of cells well. The U-Net did not perform well with the same preprocessing, but with [mean 7 × 7 & histogram equalization] preprocessing. **(B,C)** [Median 5 × 5] preprocessing in combination with the Cellpose 1.5 nuclei or the StarDist model was applied to the entire data set. **(B)** The Cellpose model missed reproducibly round, very bright nuclei (blue arrow). **(C)** StarDist predicted many false-positive cells.

The same preprocessing and model were used as described for the skeletal muscle dataset, the Cellpose scale-factor range [0.6, 0.8, 1.0, 1.4, 1.8] was explored. [Median 3 × 3 & histogram equalization] preprocessing in combination with the Cellpose nuclei 0.6 model produced fine results (**Figure 8C**). [Mean 7 × 7 & histogram equalization] preprocessing in combination with StarDist performed similarly well (**Figure 8C**). The latter pipeline resulted in more false-positive detections (**Figure 8C**, purple arrows). The U-Net performed worse, and more nuclei were missed (**Figure 8C**, blue arrow). All models failed for dense cell clusters (**Figures 8C,D**, white arrow).

Next, we sought to expand the capability of OpSeF to volume segmentation. To this aim, we trained a StarDist 3D model using the annotation of *Arabidopsis thaliana* lateral root nuclei dataset provided by Wolny et al. (2020). Images were obtained on a Luxendo MuVi SPIM light-sheet microscope (Wolny et al., 2020). We first tested the model with a similar, publically available dataset. Valuchova et al. (2020) studied differentiation within *Arabidopsis* flowers. To this aim, the authors obtained eight views

of H2B:mRuby2 labeled somatic nuclei on a Zeiss Z1 light-sheet microscope. We used a single view to mimic a challenging 3D segmentation (**Figures 9A–C**). Changes in image quality along the optical axis, in particular, deteriorating image quality deeper in the tissue (**Figure 9C**) are a major challenge for any segmentation algorithm. While the segmentation quality of the interactive H-Watershed ImageJ plugin (Vincent and Soille, 1991; Najman and Schmitt, 1996; Lotufo and Falcao, 2000; Schindelin et al., 2015), a state of the art traditional image processing method, is still acceptable in planes with good contrast (**Figure 9B**, xy Slice), results deeper in the tissue are inferior to CNN-based segmentation (data not shown). The H-Watershed plugin consequently fails to segment precisely in 3D (**Figure 9C**, zy-slices). The StarDist 3D model, which was trained on a similar dataset, performs slightly better than the Cellpose nuclei model. To evaluate the performance of these models further, we used the DISCEPTS dataset (Blin et al., 2019). DICEPTS stands for "DifferentiatingStemCells & Embryos are a PainToSegment" and contains various datasets of densely packed
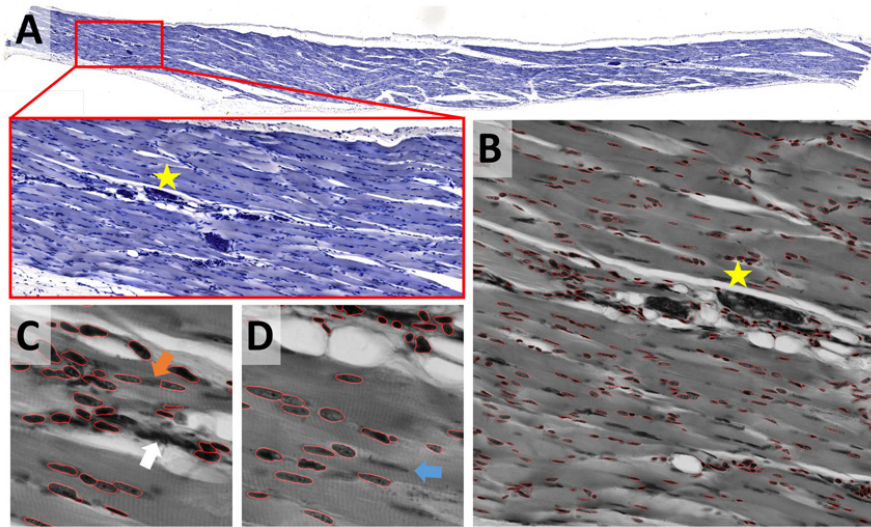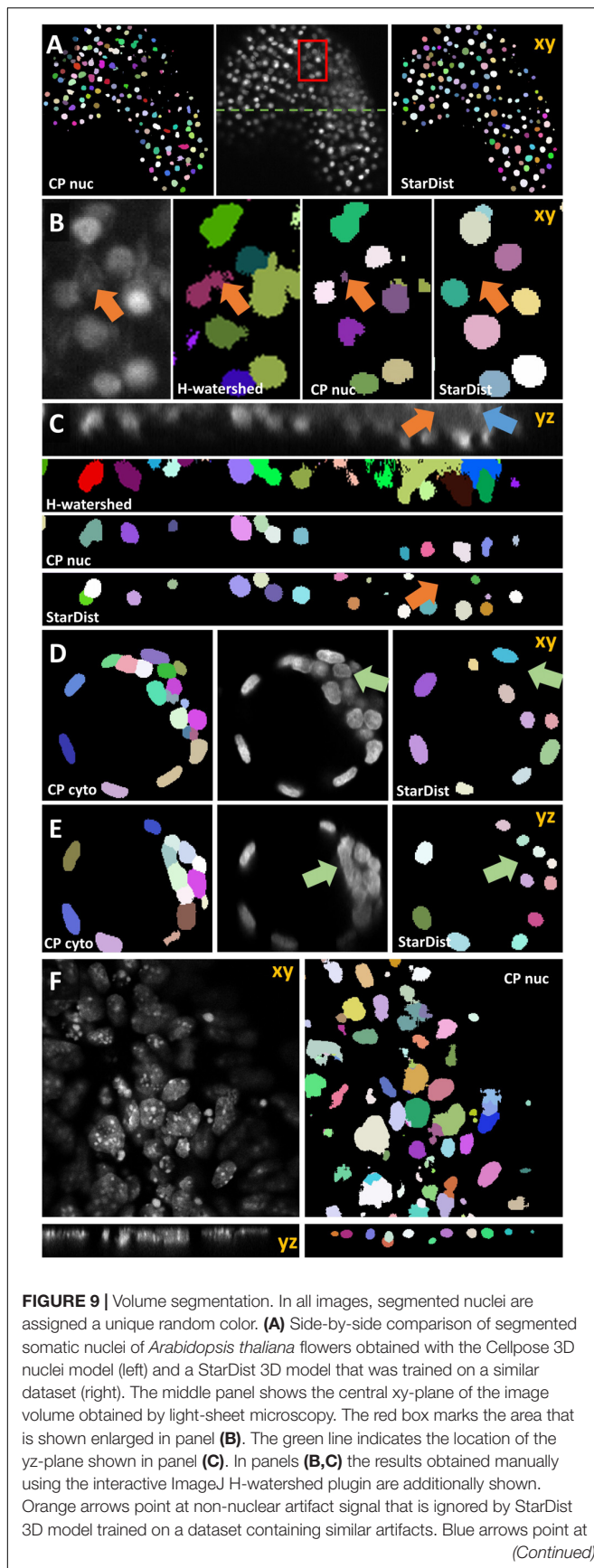
FIGURE 7 | Skeletal muscle notebook: Segmenting irregular nuclei in tissue **(A)** A methylene blue-stained skeletal muscle section was used to test the segmentation of tissue that has been stained with one dye. **(B)** Segmentation was tested on 2048 × 2048 pixels large image patches. The star shown in panels **(A,B)** is located at the same position within the image displayed at different zoom factors. The segmentation result (red line) of the Cellpose nuclei 0.6 model is super-imposed onto the original image subjected to [median 3 × 3] preprocessing. **(C,D)** Close-up on regions that were difficult to segment. Segmentation of dense clusters (white arrow) often failed, and occasional over-segmentation of large, elongated nuclei (orange arrow) was observed. Nuclei that are out-of-focus (blue arrow) were frequently missed (blue arrow).
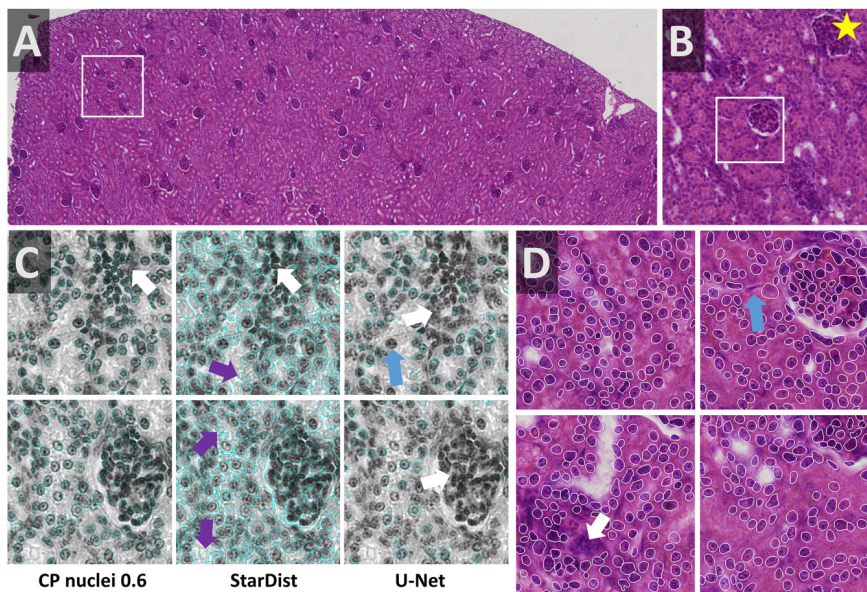
FIGURE 8 | Kidney notebook: Segmenting cell cluster in tissues **(A,B)** Part of a HE stained kidney paraffin section used to test segmentation of tissue stained with two dyes. The white box in panel **(A)** highlights the region shown enlarged in panel **(B)**. Star in panels **(A,B,D)** marks the same glomerulus. **(C)** Eosin signal was extracted by color deconvolution. The segmentation result (blue line) is superimposed onto the original image subjected to [median 3 × 3] preprocessing. Cellpose nuclei 0.6 model with scale-factor 0.6 in combination with [median 3 × 3 & histogram equalization] preprocessing and the StarDist model with [mean 7 × 7 & histogram equalization] performed similarly well **(C,D)**. StarDist resulted in more false-positive detections (purple arrows). The U-Net performed worse, more nuclei were missed (blue arrow). All models failed in very dense areas (white arrow).

nuclei that are heterogeneous in shape, size, or texture. Blin et al. elegantly solved the segmentation challenge by labeling the nuclear envelope. We thought to assess the performance of models contained in OpSeF on the more challenging DAPI signal. While the StarDist model trained on *Arabidopsis thaliana* lateral root nuclei shows satisfactory performance on E3.5 mouse

FIGURE 9 | Volume segmentation. In all images, segmented nuclei are assigned a unique random color. **(A)** Side-by-side comparison of segmented somatic nuclei of *Arabidopsis thaliana* flowers obtained with the Cellpose 3D nuclei model (left) and a StarDist 3D model that was trained on a similar dataset (right). The middle panel shows the central xy-plane of the image volume obtained by light-sheet microscopy. The red box marks the area that is shown enlarged in panel **(B)**. The green line indicates the location of the yz-plane shown in panel **(C)**. In panels **(B,C)** the results obtained manually using the interactive ImageJ H-watershed plugin are additionally shown. Orange arrows point at non-nuclear artifact signal that is ignored by StarDist 3D model trained on a dataset containing similar artifacts. Blue arrows point at

(Continued)

blastocysts, notably, the size of nuclei is underestimated, and cells in dense clusters are sometimes missed. Fine-tuning of the non-maximum suppression and the detection threshold might suffice to obtain more precise segmentation results.

Interestingly, the more versatile Cellpose cyto, rather than the Cellpose nuclei model, is ideally suited for segmenting the E3.5 mouse blastocysts nuclei (**Figures 9D,E**). Next, we used the neural "monolayer" dataset (**Figure 9F**). In this dataset flat cells form tight 3D clusters. It proved to be challenging to segment (Blin et al., 2019). Our pre-trained StarDist model failed to give satisfactory segmentation results (data not shown). The presented cells contain – in contrast to *Arabidopsis thaliana* lateral root nuclei – strong texture in their nuclei. The more versatile Cellpose nuclei model displayed promising initial results with little fine-tuning (**Figure 9F**) that might be further improved by re-training the model on the appropriate ground truth.

DISCUSSION

Intended Use and Future Developments

Examining the relationship between biochemical changes and morphological alterations in diseased tissues is crucial to understand and treat complex diseases. Traditionally, microscopic images are inspected visually. This approach limits the possibilities for the characterization of phenotypes to more obvious changes that occur later in disease progression. The manual investigation of subtle alterations at the single-cell level, which often requires quantitative assays, is hampered by the data volume. A whole slide tissue image might contain over one million cells. Despite the improvement in machine learning technology, completely unsupervised analysis pipelines have not been widely accepted. Thus, one of the major challenges for the coming years will be the development of efficient strategies to keep the human-expert in the loop. Many biomedical users still perceive deep learning models as black boxes. The mathematical foundation of how CNNs make decisions is improving. OpSeF facilitates understanding the strength of pre-trained models and network architecture on the descriptive, operational level. Thereby, awareness of intrinsic limitations such as the inability of StarDist to segment non-star-convex shapes well, or issues relating to the limited field-of-view of neural networks can be reached. It further allows us to quickly assess how robust models are against artifacts such as shadows present in light-sheet microscopy or how well they are in

predicting cell shapes accurately that are neither round nor ellipsoid shaped (e.g., neurons, amoebas). Collectively, increased awareness of limitations and better interpretability of results will be pivotal to increase the acceptance of machine learning methods. It will improve the quality control of results and allow efficient integration of expert knowledge in analysis pipelines (Holzinger et al., 2019a,b).

As illustrated in the provided Jupyter notebooks, the U-Net often performed worst. Why is that the case? As previously reported the learning capacity of a single, regular U-Net is limited (Caicedo et al., 2019). Alternatively, the similarity of train and test images might be insufficient. Either way, the provision of a set of U-Nets trained on diverse data, might be a promising approach to address this limitation. [ods.ai] topcoders, the winning team of the 2018 Data Science Bowl (Caicedo et al., 2019), combined simple U-Nets with dedicated pre- and postprocessing pipelines. In doing so, they outperformed teams using more complex models like Mask R-CNN (Caicedo et al., 2019). OpSeF allows for the straightforward integration of a large number of pre-trained CNNs. We plan to include the possibility of saving pixel probabilities in future releases of OpSeF. This option will grant users more flexibility in designing custom postprocessing pipelines that ensemble results from a set of useful intermediate predictions.

OpSeF allows semi-automated exploration of a large number of possible combinations of preprocessing pipelines and segmentation models. Even if satisfactory results are not achievable with pre-trained models, OpSeF results may be used as a guide for which CNN architecture, re-training on manually created labels might be promising. The generation of training data is greatly facilitated by a seamless integration in ImageJ using the AnnotatorJ plugin. We hope that many OpSeF users will contribute their training data to open repositories and will make new models available for integration in OpSeF. Thus, OpSeF might soon become, an interactive model repository, in which an appropriate model might be identified with reasonable effort. Community provided Jupyter notebooks might be used to teach students in courses how to optimize CNN based analysis pipelines. This could educate them and make them less dependent on turn-key solutions that often trade performance for simplicity and offer little insight into the reasons why the CNN-based segmentation works or fails. The better users understand the model they use, the more they will trust them and, the better they will be able to quality control them. We hope that OpSeF will be widely accepted as a framework through which novel models might be made available to other image analysts in an efficient way.

Integrating Various Segmentation Strategies and Quality Control of Results

Multiple strategies for instance segmentation have been pursued. The U-Net belongs to the "pixel to object" class of methods: each pixel is first assigned to a semantic class (e.g., cell or background), then pixels are grouped into objects (Ronneberger et al., 2015). Mask R-CNNs belong to the "object to pixel" class

of methods (He et al., 2017): the initial prediction of bounding boxes for each object is followed by a semantic segmentation. Following an intermediate approach, Schmidt et al. first predict star-convex polygons that approximate the shape of cells and use non-maximum suppression to prune redundant predictions (Schmidt et al., 2018; Weigert et al., 2019, 2020). Stringer et al. use stimulated diffusion originating from the center of a cell to convert segmentation masks into flow fields. The neural network is then trained to predict flow fields, which can be converted back into segmentation masks (Stringer et al., 2020). Each of these methods has specific strengths and weaknesses. The use of flow fields as auxiliary representation proved to be a great advantage for predicting cell shapes that are not roundish. At the same time, Cellpose is the most computationally demanding model used. In our hands, Cellpose tended to result in more obviously erroneously missed objects, in particular, if objects displayed a distinct appearance compared to their neighbors (blue arrows in **Figures 5B**, **6B**, **7D**). StarDist is much less computationally demanding, and star-convex polygons are well suited to approximate elliptical cell shapes. The pre-trained StarDist model implemented in OpSeF might be less precise in predicting novel shapes it has not been trained on, e.g., maple leaves (**Figure 4C**). This limitation can be overcome by retraining, given the object is star-convex, which includes certain concave shapes such as maple leaves. However, some cell-types (e.g., neurons, amoeba) are typically non star-convex, and StarDist – due to "limitation by design" – cannot be expected to segment these objects precisely. Segmentation errors by the StarDist model were generally plausible. It tended to predict cell-like shapes, even if they are not present (**Figure 6B**). Although the tendency of StarDist to fail gracefully might be advantageous in most instances, this feature requires particularly careful quality control to detect and fix errors. The "pixel-to-object" class of methods is less suited for segmentation of dense cell clusters. The misclassification of just a few pixels might lead to the fusion of neighboring cells.

OpSeF integrates three mechanistically distinct methods for CNN-based segmentation in a single framework. This allows comparing these methods easily. Nonetheless, we decided against integrating an automated evaluation, e.g., by determining F1 score, false positive and false negative rates, and accuracy. Firstly, for most projects no ground-truth is available. Secondly, we want to encourage the user to visually inspect segmentation results. Reviewing 100 different segmentation results opened in ImageJ as stack takes only a few minutes and gives valuable insight into when and how segmentations fail. This knowledge is easily missed when just looking at the output scores of commonly used metrics but might have a significant impact on the biological conclusion. Even segmentation results from a model with 95% precision and 95% recall for the overall cell population might be not suited to determine the abundance of a rare cell type if these cells are systematically missed, detected less accurately in the mutant situation, or preferentially localized to areas in the tissue that are not segmented well. Although it is difficult to capture such issues with standard metrics, they are readily observed by a human expert. Learning more about the circumstances in which

certain types of CNN-based segmentation fail helps to decide when human experts are essential for quality control of results. Moreover, it is pivotal for the design of postprocessing pipelines. These might select among multiple segmentation hypotheses – on an object by object basis – the one which gives the most consistent results for reconstructing complex cells-shapes in large 3D volumes or for cell-tracking.

Optimizing Results and Computational Cost

Image analysis pipelines are generally a compromise between ease-of-use and performance as well as between computational cost and accuracy. Until now, rather simple, standard U-Nets are most frequently used models in the major image analysis tools. In contrast, the winning model of the 2018 Data Science Bowl by the [ods.ai] topcoders team used sophisticated data postprocessing to combine the output of 32 different neural networks (Caicedo et al., 2019). The high computational cost currently limits the widespread use of this or similar approaches. OpSeF is an ideal platform to find the computationally most efficient solution to a segmentation task. The [ods.ai] topcoders algorithm was designed to segment five different classes of nuclei: "small" and "large fluorescent," "grayscale," "purple tissue" and "pink and purple tissue" (Caicedo et al., 2019). Stringer et al. used an even broader collection of images that included cells of unusual appearance and natural images of regular cell-like shapes such as shells, garlic, pearls, and stones (Stringer et al., 2020).

The availability of such versatile models is precious, in particular, for users, who are unable to train custom models or lack resources to search for the most efficient pre-trained model. For most biological applications, however, no one-fits-all solution is required. Instead, potentially appropriate models might be pre-selected, optimized, and tested using OpSeF. Ideally, an image analyst and a biomedical researcher will jointly fine-tune the analysis pipeline and quality control results. This way, resulting analysis workflows will have the best chances of being both robust and accurate, and an ideal balance between manual effort, computational cost, and accuracy might be reached.

Comparison of the models available within OpSeF revealed that the same task of segmenting 100 images using StarDist took 1.5-fold, Cellpose with fixed scale-factor 3.5-fold, and Cellpose with flexible scale-factor 5-fold longer compared to segmentation with the U-Net.

The systematic search of the optimal parameter and ideal performance might be dispensable if only a few images are to be processed that can be easily manually curated, but highly valuable if massive datasets produced by slide-scanner, light-sheet microscopes or volume EM techniques are to be processed.

Deployment Strategies

We decided against providing OpSeF as an interactive cloud solution. A local solution uses existing resources best, avoids limitations generated by the down- and upload of large datasets, and addresses concerns regarding the security of clinical datasets. Although the provision of plugins is perceived as crucial to speed up the adoption of new methods, image analysts increasingly use the Jupyter notebooks that allow them to document workflows step-by-step. This is a significant advantage compared to interactive solutions, in which parameters used for analysis are not automatically logged. Biologists might hesitate to use Jupyter notebooks for analysis due to an initial steep learning curve. Once technical challenges such as the establishment of the conda environment are overcome, notebooks allow them to integrate data analysis and documentation with ease. Notebooks might be deposited in repositories along with the raw data. This builds more trust in published results by improving transparency and reproducibility.

DATA AVAILABILITY STATEMENT

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found below:
Test data for the Jupyter notebooks is available at:
https://owncloud.gwdg.de/index.php/s/nSUqVXkkfUDPG5b.
Test data for AnnotatorJ is available at:
https://owncloud.gwdg.de/index.php/s/dUMM6JRXsuhTncS.
The source code is available at:
https://github.com/trasse/OpSeF-IV.

AUTHOR CONTRIBUTIONS

TR designed and developed the OpSeF, and analyzed the data. RH and PH designed and developed the AnnotatorJ integration. TR wrote the manuscript with contributions from all authors.

FUNDING

ACKNOWLEDGMENTS

REFERENCES

Akram, S. U., Kannala, J., Eklund, L., and Heikkilä, J. (2016). "Cell segmentation proposal network for microscopy image analysis," in *Deep Learning and Data Labeling for Medical Applications*, eds G. Carneiro, D. Mateus, L. Peter, A. Bradley, J. M. R. S. Tavares, V. Belagiannis, et al. (New York, NY: Springer International Publishing), 21–29. doi: 10.1007/978-3-319-46976-8_3

Albarqouni, S., Baur, C., Achilles, F., Belagiannis, V., Demirci, S., and Navab, N. (2016). AggNet: deep learning from crowds for mitosis detection in breast cancer histology images. *IEEE Trans. Med. Imaging* 35, 1313–1321. doi: 10.1109/TMI.2016.2528120

Arganda-Carreras, I., Kaynig, V., Rueden, C., Eliceiri, K. W., Schindelin, J., Cardona, A., et al. (2017). Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification. *Bioinformatics* 33, 2424–2426. doi: 10.1093/bioinformatics/btx180

Bankhead, P., Loughrey, M. B., Fernandez, J. A., Dombrowski, Y., McArt, D. G., Dunne, P. D., et al. (2017). QuPath: open source software for digital pathology image analysis. *Sci. Rep.* 7:16878. doi: 10.1038/s41598-017-17204-5

Berg, S., Kutra, D., Kroeger, T., Straehle, C. N., Kausler, B. X., Haubold, C., et al. (2019). ilastik: interactive machine learning for (bio)image analysis. *Nat. Methods* 16, 1226–1232. doi: 10.1038/s41592-019-0582-9

Blin, G., Sadurska, D., Portero Migueles, R., Chen, N., Watson, J. A., and Lowell, S. (2019). Nessys: a new set of tools for the automated detection of nuclei within intact tissues and dense 3D cultures. *PLoS Biol.* 17:e3000388. doi: 10.1371/journal.pbio.3000388

Bykov, Y. S., Cohen, N., Gabrielli, N., Manenschijn, H., Welsch, S., Chlanda, P., et al. (2019). High-throughput ultrastructure screening using electron microscopy and fluorescent barcoding. *J. Cell Biol.* 218, 2797–2811. doi: 10.1083/jcb.201812081

Caicedo, J. C., Goodman, A., Karhohs, K. W., Cimini, B. A., Ackerman, J., Haghighi, M., et al. (2019). Nucleus segmentation across imaging experiments: the 2018 data science bowl. *Nat. Methods* 16, 1247–1253. doi: 10.1038/s41592-019-0612-7

Chessel, A., and Carazo Salas, R. E. (2019). From observing to predicting single-cell structure and function with high-throughput/high-content microscopy. *Essays Biochem.* 63, 197–208. doi: 10.1042/EBC20180044

Çiçek, Ö, Abdulkadir, A., Lienkamp, S. S., Brox, T., and Ronneberger, O. (2016). "3D U-net: learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, eds S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells (New York, NY: Springer International Publishing), 424–432. doi: 10.1007/978-3-319-46723-8_49

Cireşan, D. C., Giusti, A., Gambardella, L. M., and Schmidhuber, J. (2012). "Deep neural networks segment neuronal membranes in electron microscopy images," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, (Lake Tahoe, Nev: Curran Associates Inc.).

Cireşan, D. C., Giusti, A., Gambardella, L. M., and Schmidhuber, J. (2013). "Mitosis detection in breast cancer histology images with deep neural networks," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, eds K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab (Berlin: Springer), 411–418. doi: 10.1007/978-3-642-40763-5_51

de Chaumont, F., Dallongeville, S., Chenouard, N., Herve, N., Pop, S., Provoost, T., et al. (2012). Icy: an open bioimage informatics platform for extended reproducible research. *Nat. Methods* 9, 690–696. doi: 10.1038/nmeth.2075

Haubold, C., Schiegg, M., Kreshuk, A., Berg, S., Koethe, U., and Hamprecht, F. A. (2016). Segmenting and tracking multiple dividing targets using ilastik. *Adv. Anat. Embryol. Cell Biol.* 219, 199–229. doi: 10.1007/978-3-319-28549-8_8

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). "Mask R-CNN," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, (Venice: IEEE).

Hoffman, D. P., Shtengel, G., Xu, C. S., Campbell, K. R., Freeman, M., Wang, L., et al. (2020). Correlative three-dimensional super-resolution and block-face electron microscopy of whole vitreously frozen cells. *Science* 367:eaaz5357. doi: 10.1126/science.aaz5357

Hollandi, R., Diósdi, Á, Hollandi, G., Moshkov, N., and Horváth, P. (2020a). AnnotatorJ: an ImageJ plugin to ease hand-annotation of cellular compartments. *Mol. Biol. Cell* 31, 2179–2186. doi: 10.1091/mbc.E20-02-0156

Hollandi, R., Szkalisity, A., Toth, T., Tasnadi, E., Molnar, C., Mathe, B., et al. (2020b). nucleAIzer: a parameter-free deep learning framework for nucleus segmentation using image style transfer. *Cell Syst.* 10, 453–458.e6. doi: 10.1016/j.cels.2020.04.003

Holzinger, A., Haibe-Kains, B., and Jurisica, I. (2019a). Why imaging data alone is not enough: AI-based integration of imaging, omics, and clinical data. *Eur. J. Nucl. Med. Mol. Imaging* 46, 2722–2730. doi: 10.1007/s00259-019-04382-9

Holzinger, A., Langs, G., Denk, H., Zatloukal, K., and Muller, H. (2019b). Causability and explainability of artificial intelligence in medicine. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* 9:e1312. doi: 10.1002/widm.1312

Houle, D., Govindaraju, D. R., and Omholt, S. (2010). Phenomics: the next challenge. *Nat. Rev. Genet.* 11, 855–866. doi: 10.1038/nrg2897

Jones, T. R., Carpenter, A. E., Lamprecht, M. R., Moffat, J., Silver, S. J., Grenier, J. K., et al. (2009). Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning. *Proc. Natl. Acad. Sci. U.S.A.* 106, 1826–1831. doi: 10.1073/pnas.0808843106

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B. E., Bussonnier, M., Frederic, J., et al. (2016). "Jupyter Notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, eds F. Loizides and B. Schmidt (Amsterdam: IOS Press).

Kreshuk, A., and Zhang, C. (2019). Machine learning: advanced image segmentation using ilastik. *Methods Mol. Biol.* 2040, 449–463. doi: 10.1007/978-1-4939-9686-5_21

Lamprecht, M. R., Sabatini, D. M., and Carpenter, A. E. (2007). CellProfiler: free, versatile software for automated biological image analysis. *Biotechniques* 42, 71–75. doi: 10.2144/000112257

Lang, P., Yeow, K., Nichols, A., and Scheer, A. (2006). Cellular imaging in drug discovery. *Nat. Rev. Drug Discov.* 5, 343–356. doi: 10.1038/nrd2008

Lotufo, R., and Falcao, A. (2000). "The ordered queue and the optimality of the watershed approaches," in *Mathematical Morphology and its Applications to Image and Signal Processing*, eds J. Goutsias, L. Vincent, and D. S. Bloomberg (Boston, MA: Springer), 341–350. doi: 10.1007/0-306-47025-x_37

McQuin, C., Goodman, A., Chernyshev, V., Kamentsky, L., Cimini, B. A., Karhohs, K. W., et al. (2018). CellProfiler 3.0: next-generation image processing for biology. *PLoS Biol.* 16:e2005970. doi: 10.1371/journal.pbio.2005970

Milletari, F., Navab, N., and Ahmadi, S. (2016). "V-Net: fully convolutional neural networks for volumetric medical image segmentation," in *Proceedings of the 2016 Fourth International Conference on 3D Vision (3DV)*, (Stanford, CA: IEEE), 565–571.

Moeskops, P., Viergever, M. A., Mendrik, A. M., Vries, L. S. D., Benders, M. J. N. L., and Išgum, I. (2016). Automatic segmentation of MR brain images with a convolutional neural network. *IEEE Trans. Med. Imaging* 35, 1252–1261. doi: 10.1109/TMI.2016.2548501

Najman, L., and Schmitt, M. (1996). Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Trans. Pattern Anal. Machine Intell.* 18, 1163–1173. doi: 10.1109/34.546254

Neumann, B., Walter, T., Heriche, J. K., Bulkescher, J., Erfle, H., Conrad, C., et al. (2010). Phenotypic profiling of the human genome by time-lapse microscopy reveals cell division genes. *Nature* 464, 721–727. doi: 10.1038/nature08869

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: machine learning in python. *J. Machine Learn. Res.* 12, 2825–2830.

Rajchl, M., Lee, M. C. H., Oktay, O., Kamnitsas, K., Passerat-Palmbach, J., Bai, W., et al. (2017). DeepCut: object segmentation from bounding box annotations using convolutional neural networks. *IEEE Trans. Med. Imaging* 36, 674–683. doi: 10.1109/TMI.2016.2621185

Rasse, T. M., Hollandi, R., and Horváth, P. (2020). OpSeF IV: open source python framework for segmentation of biomedical images. *bioRxiv*[Preprint] doi: 10.1101/2020.04.29.068023

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: convolutional networks for biomedical image segmentation. *arXiv*[Preprint] Available online at: https://arxiv.org/abs/1505.04597 (Accessed March 15, 2020),

Ruifrok, A. C., and Johnston, D. A. (2001). Quantification of histochemical staining by color deconvolution. *Anal. Quant Cytol. Histol.* 23, 291–299.

Schindelin, J., Rueden, C. T., Hiner, M. C., and Eliceiri, K. W. (2015). The imageJ ecosystem: an open platform for biomedical image analysis. *Mol. Reprod. Dev.* 82, 518–529. doi: 10.1002/mrd.22489

Schmidt, U., Weigert, M., Broaddus, C., and Myers, G. (2018). "Cell detection with star-convex polygons," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, eds A. F. Frangi, J. A. Schnabel, C. Davatzikos, C.

Alberola-López, and G. Fichtinger (Cham: Springer International Publishing), 265–273. doi: 10.1007/978-3-030-00934-2_30

Schneider, C. A., Rasband, W. S., and Eliceiri, K. W. (2012). NIH Image to ImageJ: 25 years of image analysis. *Nat. Methods* 9, 671–675. doi: 10.1038/nmeth.2089

Schorb, M., Haberbosch, I., Hagen, W. J. H., Schwab, Y., and Mastronarde, D. N. (2019). Software tools for automated transmission electron microscopy. *Nat. Methods* 16, 471–477. doi: 10.1038/s41592-019-0396-9

Sieb, C., Meinl, T., and Berthold, M. R. (2007). Parallel and distributed data pipelining with KNIME. *Mediterr. J. Comput. Netw.* 3, 43–51.

Stringer, C., Michaelos, M., and Pachitariu, M. (2020). Cellpose: a generalist algorithm for cellular segmentation. *bioRxiv*[Preprint] doi: 10.1101/2020.02.02. 931238

Svoboda, D., Homola, O., and Stejskal, S. (2011). *Generation of 3D Digital Phantoms of Colon Tissue*. Berlin: Springer, 31–39.

Swoger, J., Pampaloni, F., and Stelzer, E. H. (2014). Light-sheet-based fluorescence microscopy for three-dimensional imaging of biological samples. *Cold Spring Harb. Protoc.* 2014, 1–8. doi: 10.1101/pdb.top080168

Titze, B., and Genoud, C. (2016). Volume scanning electron microscopy for imaging biological ultrastructure. *Biol. Cell* 108, 307–323. doi: 10.1111/boc. 201600024

Ueda, H. R., Erturk, A., Chung, K., Gradinaru, V., Chedotal, A., Tomancak, P., et al. (2020). Tissue clearing and its applications in neuroscience. *Nat. Rev. Neurosci.* 21, 61–79. doi: 10.1038/s41583-019-0250-1

Valuchova, S., Mikulkova, P., Pecinkova, J., Klimova, J., Krumnikl, M., Bainar, P., et al. (2020). Imaging plant germline differentiation within Arabidopsis flowers by light sheet microscopy. *eLife* 9:e52546. doi: 10.7554/eLife.52546

van der Walt, S., Schonberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., et al. (2014). scikit-image: image processing in Python. *PeerJ* 2:e453. doi: 10.7717/peerj.453

Van Valen, D. A., Kudo, T., Lane, K. M., Macklin, D. N., Quach, N. T., DeFelice, M. M., et al. (2016). Deep learning automates the quantitative analysis of individual cells in live-cell imaging experiments. *PLoS Comput. Biol.* 12:e1005177. doi: 10.1371/journal.pcbi.1005177

Vidavsky, N., Akiva, A., Kaplan-Ashiri, I., Rechav, K., Addadi, L., Weiner, S., et al. (2016). Cryo-FIB-SEM serial milling and block face imaging: large volume structural analysis of biological tissues preserved close to their native state. *J. Struct. Biol.* 196, 487–495. doi: 10.1016/j.jsb.2016.09.016

Vincent, L., and Soille, P. (1991). Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Anal. Machine Intell.* 13, 583–598. doi: 10.1109/34.87344

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* 17, 261–272. doi: 10.1038/s41592-019-0686-2

Wang, S., Yang, D. M., Rong, R., Zhan, X., and Xiao, G. (2019). Pathology image analysis using segmentation deep learning algorithms. *Am. J. Pathol.* 189, 1686–1698. doi: 10.1016/j.ajpath.2019.05.007

Webster, J. D., and Dunstan, R. W. (2014). Whole-slide imaging and automated image analysis: considerations and opportunities in the practice of pathology. *Vet. Pathol.* 51, 211–223. doi: 10.1177/0300985813503570

Weigert, M., Schmidt, U., Haase, R., Sugawara, K., and Myers, G. (2019). Star-convex polyhedra for 3D object detection and segmentation in microscopy. *arXiv*[Preprint] doi: 10.1109/WACV45572.2020.9093435

Weigert, M., Schmidt, U., Haase, R., Sugawara, K., and Myers, G. (2020). "Star-convex Polyhedra for 3D object detection and segmentation in microscopy," in *Proceedings of The IEEE Winter Conference on Applications of Computer Vision (WACV)*, (Piscataway, NJ: IEEE), 3655–3662.

Whitehead, L. (2020). *Tweet: Correct. It's the Pre-Trained "Versatile" Model. (on use of StarDist)*. Available online at: https://twitter.com/DrLachie/status/ 1224207976350633985 (accessed February 5, 2020).

Williams, E., Moore, J., Li, S. W., Rustici, G., Tarkowska, A., Chessel, A., et al. (2017). The image data resource: a bioimage data integration and publication platform. *Nat. Methods* 14, 775–781. doi: 10.1038/nmeth.4326

Wolny, A., Cerrone, L., Vijayan, A., Tofanelli, R., Barro, A. V., Louveaux, M., et al. (2020). Accurate and versatile 3D segmentation of plant tissues at cellular resolution. *eLife* 9:e57613. doi: 10.7554/eLife.57613

Zhang, W., Li, R., Deng, H., Wang, L., Lin, W., Ji, S., et al. (2015). Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. *Neuroimage* 108, 214–224. doi: 10.1016/j.neuroimage.2014. 12.061

Zuiderveld, K. (1994). "Contrast limited adaptive histogram equalization," in *Graphics gems IV*, ed. P. S. Heckbert (Cambridge, MA: Academic Press Professional, Inc), 474–485. doi: 10.1016/b978-0-12-336156-1.50061-6