Apr 30, 2021

iPSC Image Analysis: From raw pics to .csv file

Jesse Cohn[1]

[1]Washington University, Saint Louis

1 Works for me This protocol is published without a DOI.

Jesse Cohn

ABSTRACT

This protocol describes the pipeline I've been using to analyze images from the InCell of the Kampmann i3N-CRISPRi cell lines. The broad overall steps are:

1. Making StarDist image labels of all the nuclei pictures
2. Create a CellProfiler batch file for the label images
3. Run the CellProfiler analysis on the cluster
4. Collapse all of the CellProfiler output csv's to a single file for analysis

To follow this pipeline you'll need to be using the same settings on the InCell that I used for imaging (the Jesse_96Well_iPSCs.xaqp protocol on the InCell).

If you change any of the settings for these things, there are places in the pipeline that will need tweaking accordingly.

All of the scripts and pipelines needed at various steps are attached to that step or available at: https://github.com/jc6213/CRANIUM . There are also example analysis Jupyter Notebooks there for ease.

PROTOCOL CITATION

Jesse Cohn 2021. iPSC Image Analysis: From raw pics to .csv file. **protocols.io**
https://protocols.io/view/ipsc-image-analysis-from-raw-pics-to-csv-file-buainsce

LICENSE

CREATED

Apr 16, 2021

LAST MODIFIED

Apr 30, 2021

PROTOCOL INTEGER ID

49194

Installing needed software

1 This protocol utilizes a Python package called "StarDist" to identify nuclei and make "label images", which are then subsequently analyzed in CellProfiler.

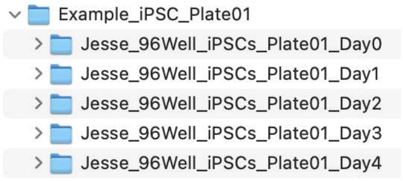
- To install a local version of CellProfiler to the cluster, see the first step of the Neuron Image Analysis protocol.
- For StarDist, see this website for instructions on how to install the package (and for information about the package in general): https://github.com/stardist/stardist

Prepping the files/folders

2 Image your plates with the "**Jesse_96Well_iPSCs.xaqp**" acquisition protocol on the InCell. This only images the green channel and should output images in the format:
B - 02(fld 1).tif
where "B - 02" is the well, and "fld 01" is which field the image is (there should be 9 fields per well). It's important the files are in this format, as the downstream analysis pipelines parse this name for metadata.

3 I typically wait to run the analysis until all of my imaging is done, because you can use the "StarDistOnParentFolder.py" script to make label images for all of the images in subfolders of a parent folder, so if you have all of your images, you can make the labels all in one go. Alternatively, you can use "StarDistOnIndivFolder.py" to do a single folder of images.

Set up a folder like this:



```
Example_iPSC_Plate01
    Jesse_96Well_iPSCs_Plate01_Day0
    Jesse_96Well_iPSCs_Plate01_Day1
    Jesse_96Well_iPSCs_Plate01_Day2
    Jesse_96Well_iPSCs_Plate01_Day3
    Jesse_96Well_iPSCs_Plate01_Day4
```

where each of the subfolders in the parent folder is your folder full of images straight from the scan.

Be sure that each subfolder is labeled with "Plate{2 numbers}_Day{number}" at some point, as CellProfiler gleans this info from the folder name via a regular expression.

Making StarDist label images

4 Get the "StarDistOnParentFolder.py" script: **StarDistOnParentFolder.py**

The only argument to include at the command line when running the script is the path to the parent folder (so the path to Example_iPSC_Plate01 in the above example). So at the command line we'd run:

```
python3 StarDistOnParentFolder.py /Volumes/SeagateExpansionDrive/Example_iPSC_Plate01
```

5 This script can take a while. This should create a folder called "labelimages" in the folder supplied, and should make sub-folders matching the original sub-folders, with "_labels" appended onto the end. Each sub-folder should be populated with label images of the original green images in that folder (with "_labels" appended onto the file name).

5.1 Alternatively, you can do just a single folder of images instead, if needed: **StarDistOnIndivFolder.py**

Run it like above, the only argument given is the path to the folder. This should create a "labelimages" folder in the parent folder of the folder supplied (or add to it if it's already there), and create a folder of the label images in there.

Making the CellProfiler batch file

6 In the parent folder (Example_iPSC_Plate01), make a new folder called 'output'. So you should now have your original folders with images, the "labelimages" folder and an "output" folder.

7 Get the "**iPSC_Nuclei_Analysis.cppipe**" pipeline for CellProfiler: **iPSC_Nuclei_Analysis.cppipe**

8 Load CellProfiler and Import the pipeline (File -> Import -> Pipeline From File...)

9 Select the "Images" module at the very top of the lefthand window, then drag the "labelimages" folder into the area that says "Drop files and folders here".

10 Go to the "NamesAndTypes" module, click the button that says "Update", and at the very bottom of the CellProfiler window it should say "Found __ image sets". Make note of how many image sets there are.

11 At the bottom of the lefthand side, click the button that says "**Output Settings**"

12 Change the "**Default Input Folder**" to the path pointing to your "**labelimages**" folder

13 Change the "**Default Output Folder**" to the path pointing to the "**output**" folder you made in step 5.

14 Click on the "**CreateBatchFiles**" module in the lefthand window.

15 Change the "**Local Root Path**" to be the path to the directory the parent folder is in. For example, the full path to my parent folder is:

```
/Volumes/SeagateExpansionDrive/Example_iPSC_Plate01
```

so I would change the Local Root Path to be:

```
/Volumes/SeagateExpansionDrive/
```

16 Change the "**Cluster Root Path**" to be the path to where you plan to do the analysis on the cluster. I typically do mine in my main scratch folder, so I would change "Cluster Root Path" to:

```
/scratch/rmlab/1/Jesse/
```

17 At the bottom of the lefthand side, click "Analyze Images". It may pop up a warning about there being too many imaging groups, just say OK, I haven't had that actually cause an issue yet.

18 A file called "Batch_data.h5" should appear in the "output" folder you made in step 5.

Preppings things on the cluster

19 Open a terminal window and log into the cluster

20 In the "Cluster Root Path" directory you specified in step 14, make a folder with the same name as your parent folder. For example, I would run:

```
mkdir /scratch/rmlab/1/Jesse/Example_Plate_iPSC_Plate01
```

21 Now open another terminal window and use rsync with the -r flag to move the 'labelimages' folder and the 'output' folder into the directory you just made on the cluster:

```
rsync -r /Volumes/SeagateExpansionDrive/Example_iPSC_Plate01/labelimages
/scratch/rmlab/1/Jesse/Example_Plate_iPSC_Plate01

rsync -r /Volumes/SeagateExpansionDrive/Example_iPSC_Plate01/output
/scratch/rmlab/1/Jesse/Example_Plate_iPSC_Plate01
```

Running the pipeline on the cluster

22 Make a 'lookup' file that will tell each instance of CellProfiler which image sets to analyze. Each line of the lookup file will tell each instance of CellProfiler which image sets to analyze. You provide it with the 'start' image set and the 'stop' image step. We need each instance to analyze 10 images, so if I had 45 image sets, my lookup file would be a text file with the body:

```
1 10
11 20
21 30
31 40
41 45
```

The total number of image sets will be the number you noted back in step 10. If there are a lot of image sets, it may be easier to make it real quick in Python:

22.1 Start the python interactive mode:

```
python3
```

22.2 Change the following code to make the file in the right directory and with a useful name, and change the "total" variable to reflect the total number of image sets you have:

```
with open('/scratch/rmlab/1/Jesse/iPSCPlate01_lookup.txt', 'w') as file:
    total = 45
    starts = list(range(1, (total + 1), 10))
    stops = list(range(10, total, 10)) + [total]
    for start, stop in zip(starts, stops):
        _ = file.write('{} {}\n'.format(start, stop))
```

22.3 Exit the python interactive mode:

```
exit()
```

You should now have the lookup file in the place you specified in the above code.

23 For the sbatch submission script, copy this code and save as a .sh file:

```
#!/bin/bash

#SBATCH --cpus-per-task=1
#SBATCH --mem=25000
#SBATCH --array=1-20%10

module load python/3.8.8
module load py-virtualenv/16.4.1-python-3.8.8
module load jdk

read startimage stopimage < <(sed -n ${SLURM_ARRAY_TASK_ID}p iPSCPlate01_lookup.txt )

. /home/jessecohn/env/bin/activate
cellprofiler -p /scratch/rmlab/1/Jesse/Example_Plate_iPSC_Plate01/output/Batch_data.h5 -c -
r -f ${startimage} -l ${stopimage}
```

Several parts will need editing:

```
#SBATCH --array=1-20%8
```

Change this line to have however many lines are in your lookup file (which you can get with wc -l on your lookup file). Like if you had **30** lines in your lookup file, it would become **#SBATCH --array=1-30%10** (the %10 just specifies to only run 10 instances of CellProfiler at a time).

```
read startimage stopimage < <(sed -n ${SLURM_ARRAY_TASK_ID}p ExamplePlate001_lookup.txt )
```

Change "iPSCPlate01_lookup.txt" to point to your lookup file.

```
. /home/jessecohn/env/bin/activate
cellprofiler -p /scratch/rmlab/1/Jesse/Example_Plate_iPSC_Plate01/output/Batch_data.h5 -c -
r -f ${startimage} -l ${stopimage}
```

The first line of this section will need to be edited to point to the env where you have CellProfiler. If you installed it using the code from the very first step of the protocol, this will probably just be a matter of replacing 'jessecohn' with your cluster user name.

Then in the next line, replace '/scratch/rmlab/1/Jesse/ExamplePlate001/output/Batch_data.h5' with the path to your Batch_data.h5 file.

24 Submit the sbatch file.

```
sbatch CellProfilerCall.sh
```

25 This will take a while to run, usually a few hours. The output folder should start to fill up with .csv files in the format of "**[plate]_[well]_[field]_[day].csv**", one for every image in the data set. Once all instances of CellProfiler are done running and all of the .csv files have accumulated, you can concatenate them all into one final file per plate by running this command in the output directory:

```
awk 'FNR==1 && NR!=1{next;}{print}' *.csv > ExamplePlate001_Final.csv
```

That's it! The data should be ready to analyze.