Manuscript

Xitong Zhang, Bolun Lin

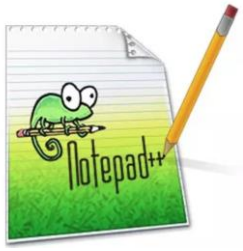
Tools

- Data Sets Establishment: Fiji

- Remote Connection (Windows): Putty

- File transfer client: WinSCP (Windows), Cyberduck (macOS & Windows)

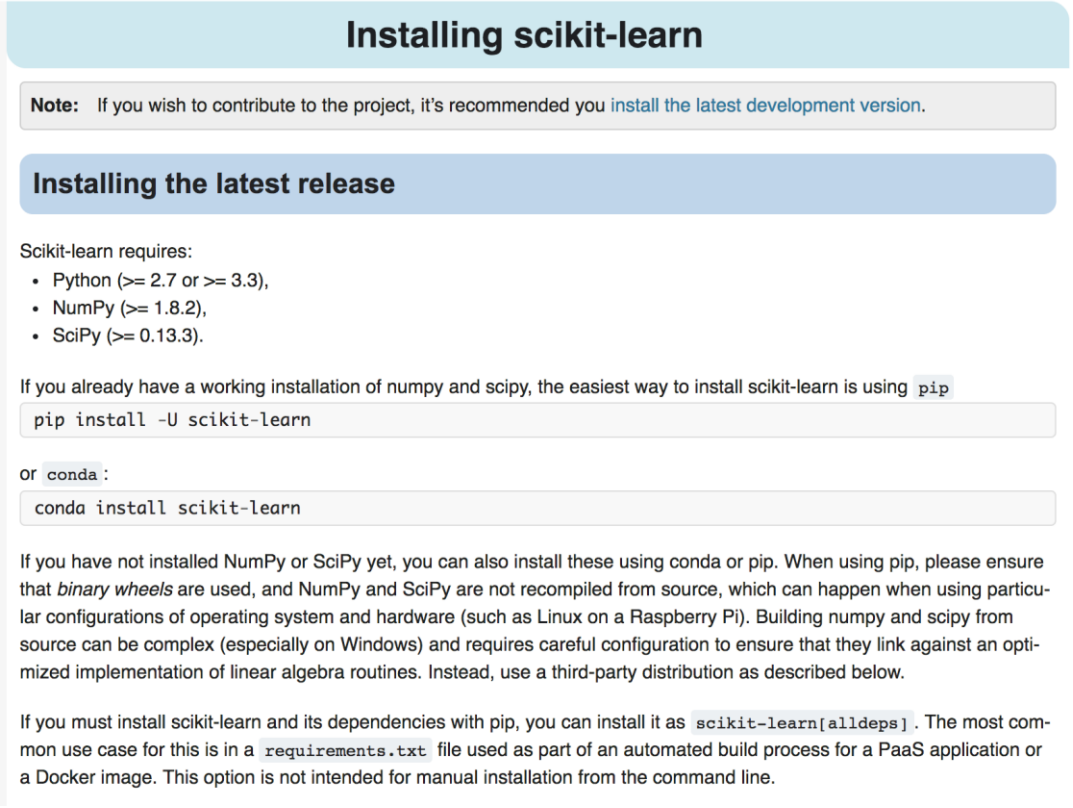
- Text Editor: Sublime text, Notepad++, Jupyter Notebook



Python Environment

- Version: Python 2.7
- Packages for environment establishment: Pip, Anaconda
- Packages:
  1. Opencv
  2. Scikit-learn
  3. Keras with Tensorflow backend
  4. Numpy, OS, GC …

Package Installation



Installing scikit-learn

Note: If you wish to contribute to the project, it's recommended you install the latest development version.

Installing the latest release

Scikit-learn requires:

- Python (>= 2.7 or >= 3.3),
- NumPy (>= 1.8.2),
- SciPy (>= 0.13.3).

If you already have a working installation of numpy and scipy, the easiest way to install scikit-learn is using `pip`

```
pip install -U scikit-learn
```

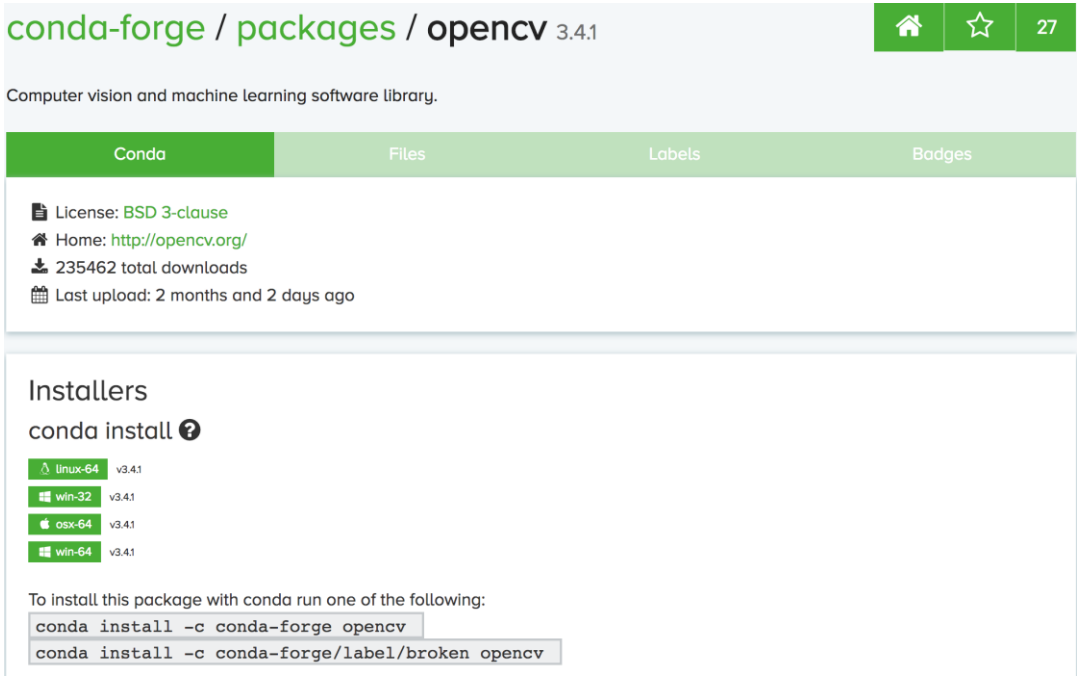
or `conda`:

```
conda install scikit-learn
```

If you have not installed NumPy or SciPy yet, you can also install these using conda or pip. When using pip, please ensure that *binary wheels* are used, and NumPy and SciPy are not recompiled from source, which can happen when using particular configurations of operating system and hardware (such as Linux on a Raspberry Pi). Building numpy and scipy from source can be complex (especially on Windows) and requires careful configuration to ensure that they link against an optimized implementation of linear algebra routines. Instead, use a third-party distribution as described below.

If you must install scikit-learn and its dependencies with pip, you can install it as `scikit-learn[alldeps]`. The most common use case for this is in a `requirements.txt` file used as part of an automated build process for a PaaS application or a Docker image. This option is not intended for manual installation from the command line.

http://scikit-learn.org/stable/install.html



conda-forge / packages / opencv 3.4.1

27

Computer vision and machine learning software library.

Conda | Files | Labels | Badges

License: BSD 3-clause
Home: http://opencv.org/
235462 total downloads
Last upload: 2 months and 2 days ago

Installers

conda install

linux-64 v3.4.1
win-32 v3.4.1
osx-64 v3.4.1
win-64 v3.4.1

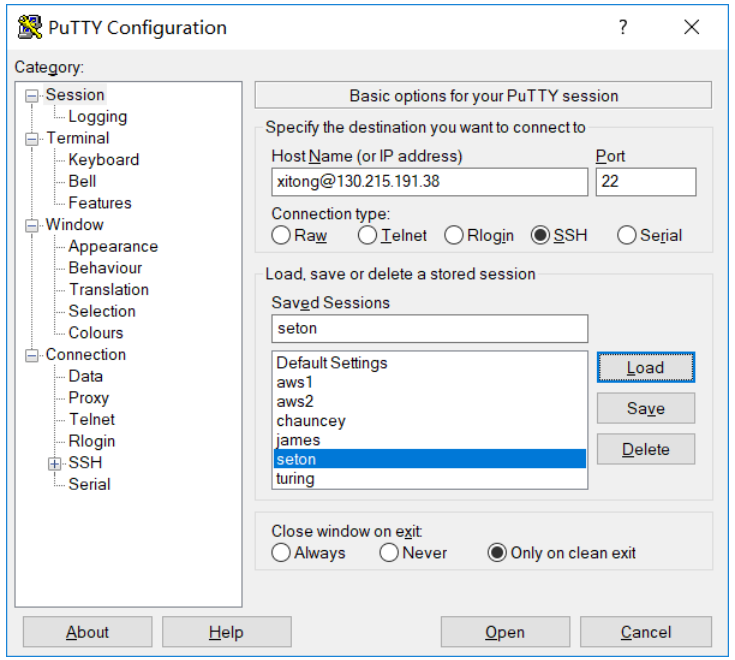
To install this package with conda run one of the following:

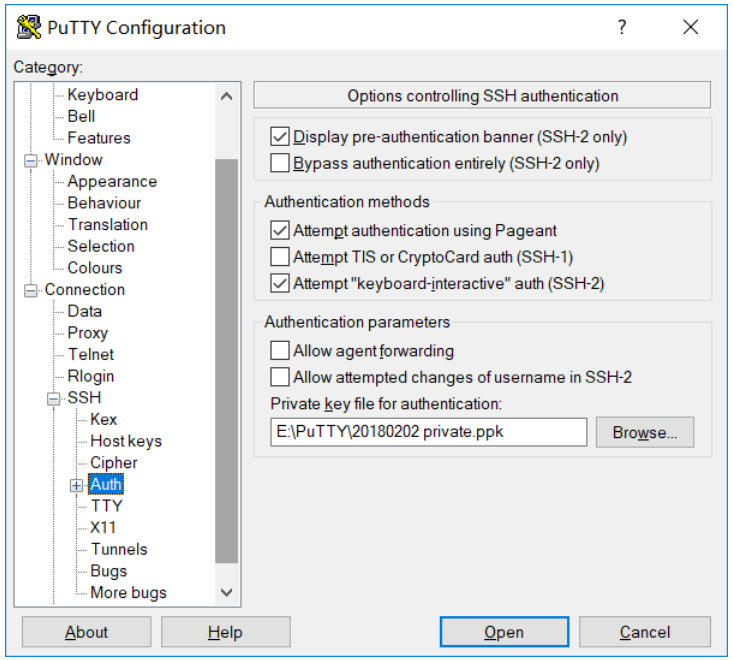
```
conda install -c conda-forge opencv
conda install -c conda-forge/label/broken opencv
```

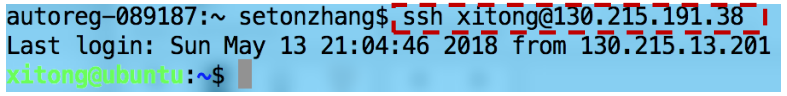
https://anaconda.org/conda-forge/opencv

Remote Connection

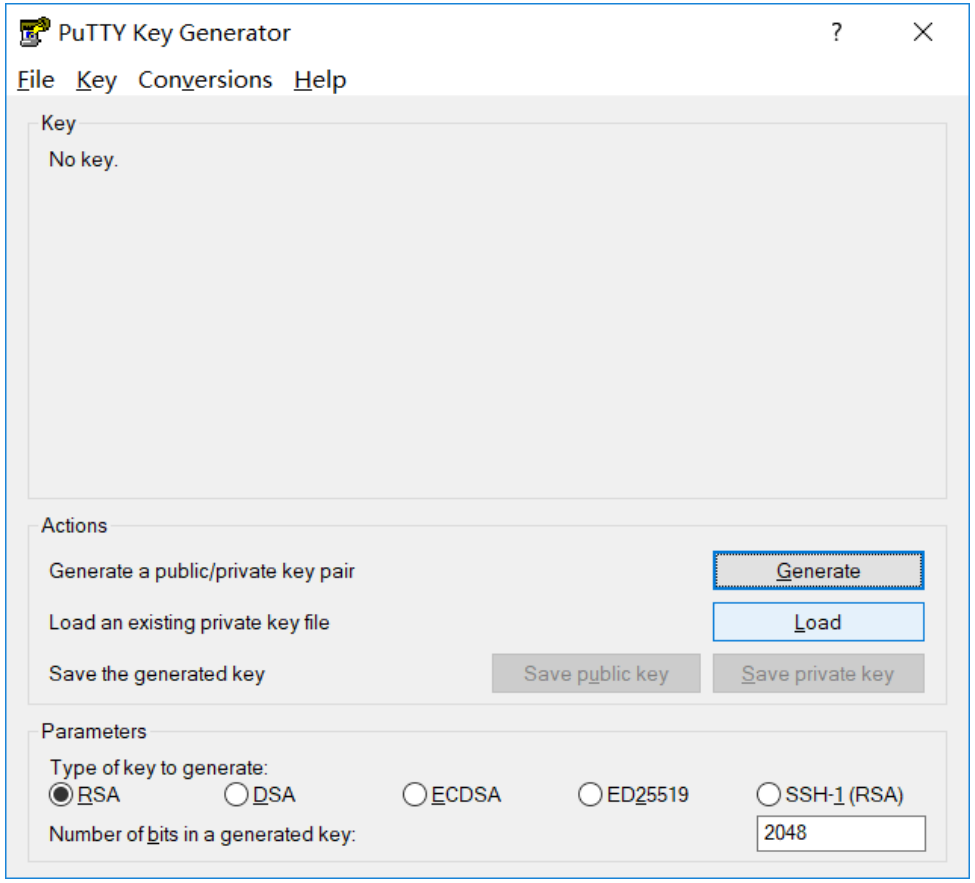
		Windows	Mac
Login	Installation	Install Putty	Use Terminal (no installation required)
	Login command	Create a session on Putty. Host: xitong@130.215.191.38; Port : 22; Connection type: SSH.	Type command in Terminal: ssh xitong@130.215.191.38;

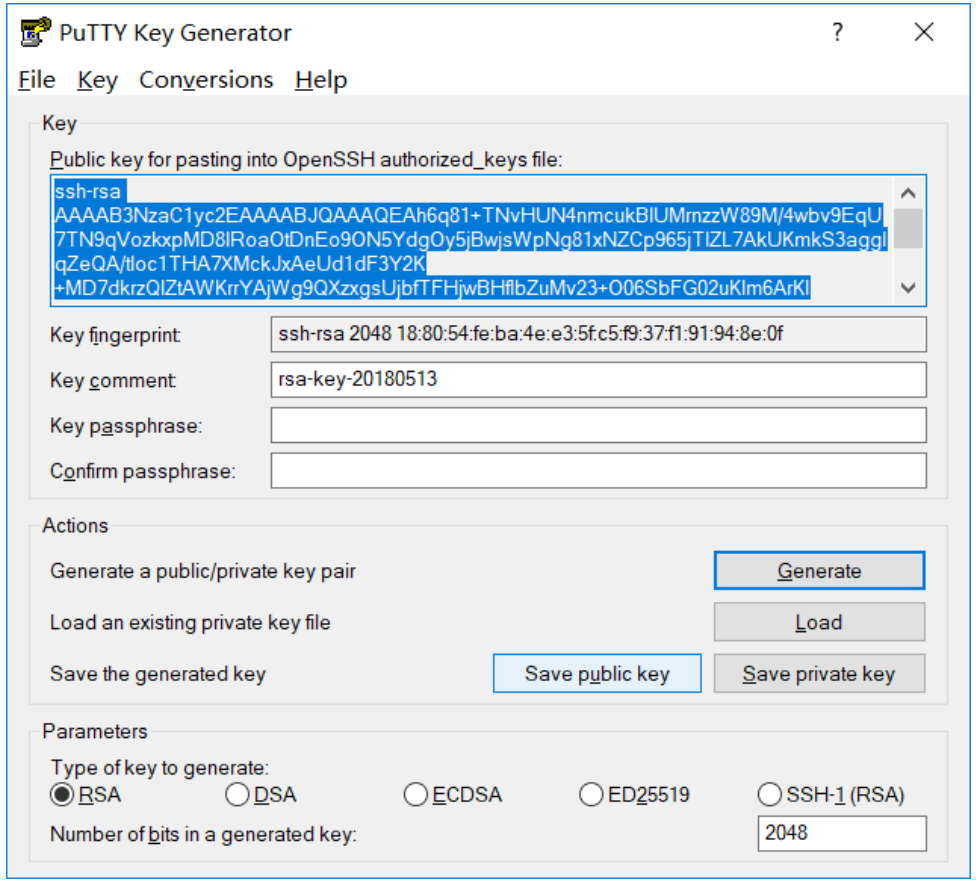






SSH Key Generation (Putty)





SSH Key Generation (macOS)

Linux Generate RSA SSH Keys

in Linux last updated May 27, 2010

How do I generate ssh RSA keys under Linux operating systems?

You need to use the ssh-keygen command as follows to generate RSA keys (open terminal and type the following command):

```
ssh-keygen -t rsa
```

OR

```
ssh-keygen
```

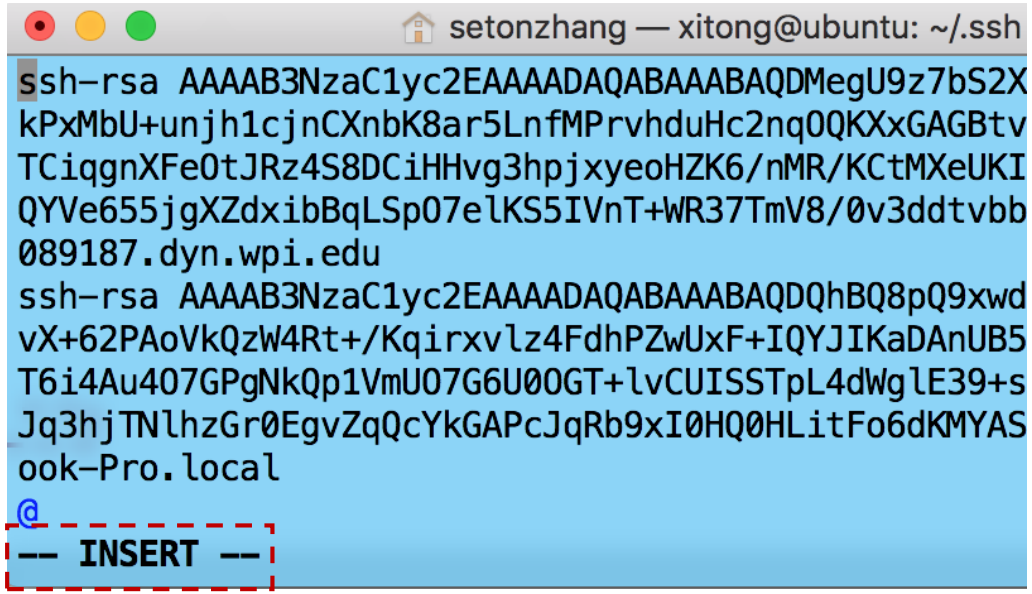
```
Enter file in which to save the key (/home/vivek/.ssh/id_rsa):
Created directory '/home/vivek/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vivek/.ssh/id_rsa.
Your public key has been saved in /home/vivek/.ssh/id_rsa.pub.
The key fingerprint is:
58:3a:80:a5:df:17:b0:af:4f:90:07:c5:3c:01:50:c2 vivek@debian
```

Share SSH Public Key

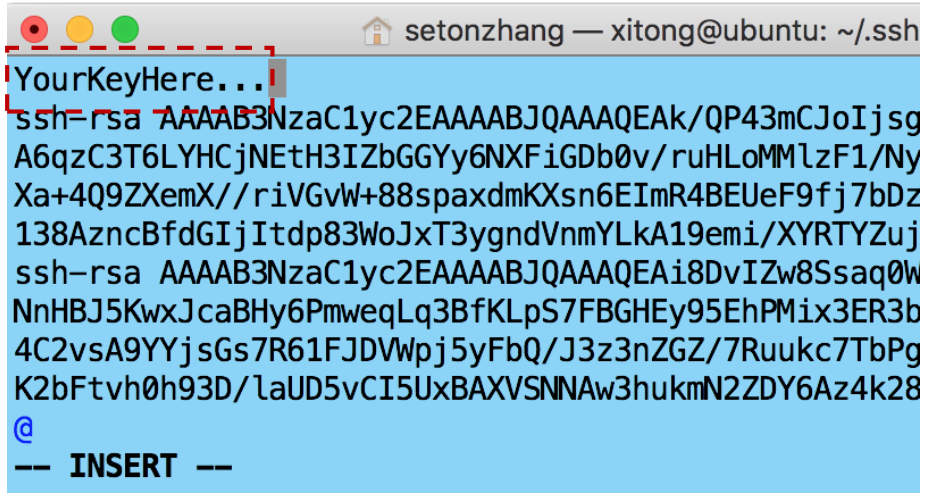
(1). Open “authorized_keys”, Password: zhang1994.

```
autoreg-089187:~ setonzhang$ ssh xitong@130.215.191.38
Last login: Sun May 13 21:05:44 2018 from 130.215.126.177
xitong@ubuntu:~$ cd ~/.ssh/
xitong@ubuntu:~/.ssh$ ls
authorized_keys  known_hosts
xitong@ubuntu:~/.ssh$ sudo vim authorized_keys
[sudo] password for xitong:
```

(2)Type “i” to the insert mode.

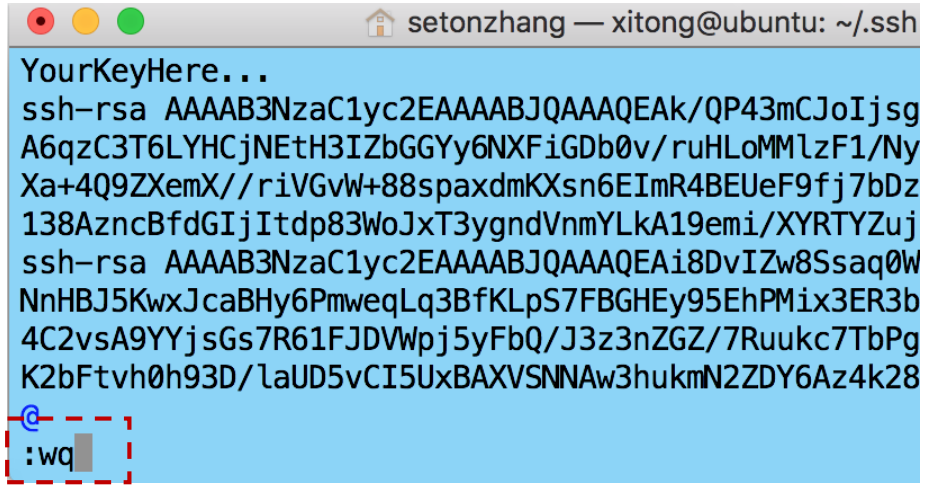


(3) Insert the generated keys.



(4) Type “esc” to exit the insert mode.

(5) Type “:wq” to overwrite the file and exit.



(6) If there is a false operation and difficult to fix, type “esc” to exit the current mode and type “:q!” to quit forcefully.

More commands, see vim

Linux Command

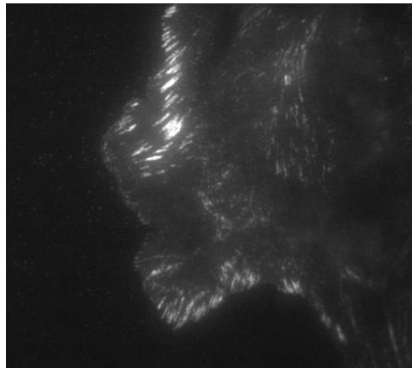
Command	Description
cd	Linux Commands
ls	Linux Commands
cp	Linux Commands
mkdir	Linux Commands
rm	Linux Commands
mv	Linux Commands
nvidia-smi	Check the status of GPUs
python	Run the process: python XXX.py
screen	[-S]+"screen_name": open a screen [-ls]: list all existing screens [-r]+"screen_name" : resume a screen by screen name "ctrl+a+d": detach from a screen
kill	Linux Commands
"tab"	Filename completion

Mount Remote Disk

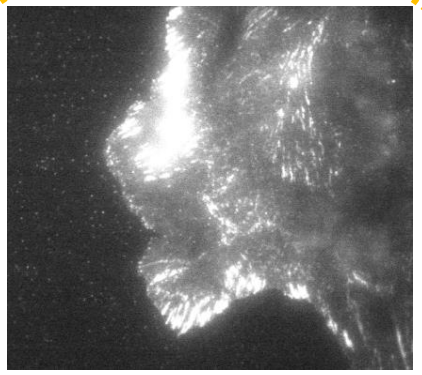
```
Connection to 130.215.191.38 closed.
autoreg-089187:~ setonzhang$ ssh xitong@130.215.191.38
Last login: Sun May 13 22:56:03 2018 from 130.215.13.201
xitong@ubuntu:~$ vim .smbcredentials
```

```
username=
password=
```

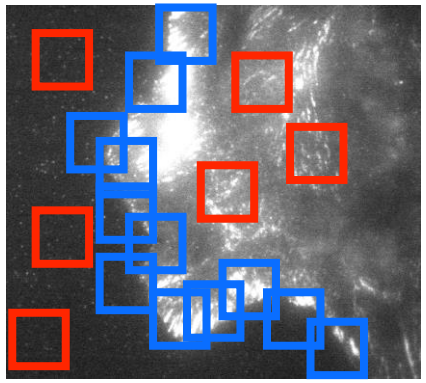
Training Process



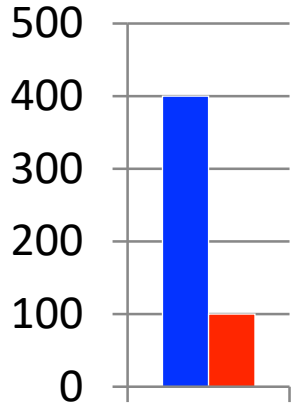
A. Raw



B. Pre-processing

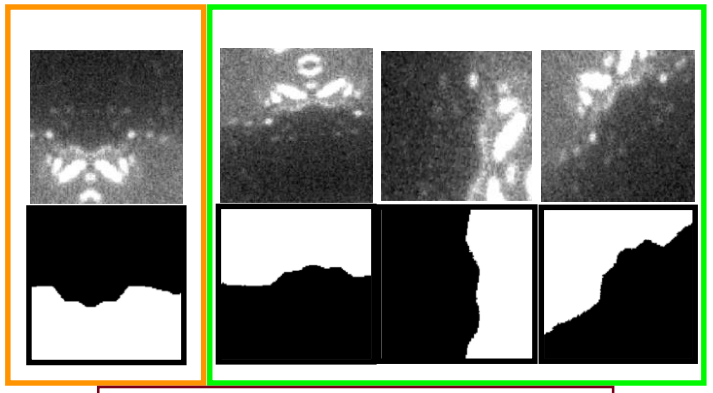


C. Non-uniformly sampling



Before

After

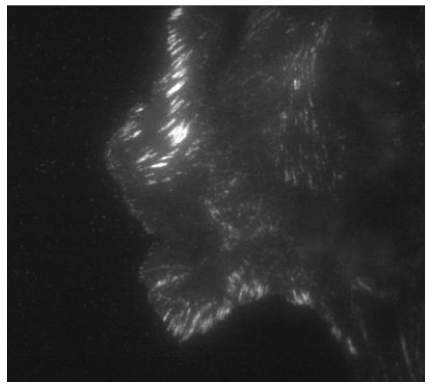


- Shear
- Rotate
- Rescale
- Flip

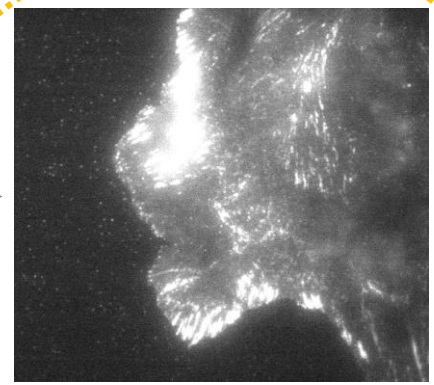
D. Data Augmentation

Deep Convolutional Network

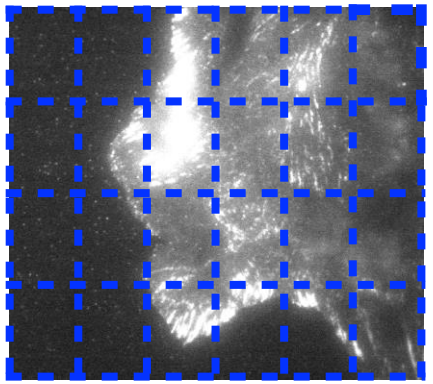
Testing Process



A. Raw

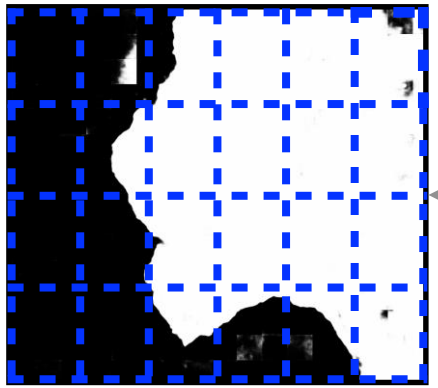


B. Pre-processing



C. Uniformly separation

Deep Convolutional Network



E. Output concatenation

F. Post-processing

Steps

1. Go to the folder which includes the code of the project.
2. Type “python train.py”.
3. After the training finished, type “python predict.py”.
4. The prediction files are saved in the ”average_hist/result”.
5. Quantitative analysis

Code

1. Edit the dataset name in the “train.py” and the “predict.py” in the code directory. (i.e. from “test3” to your dataset name)

2. Edit the parameter “n_train” in the “train.py” and the “predict.py” to meet the number of training images.

3. Edit the parameter “n_total” in the “predict.py” to meet the total number of images.