

# Manuscript

Xitong Zhang, Bolun Lin

# Tools

- Data Sets Establishment: Fiji



- Remote Connection (Windows): Putty

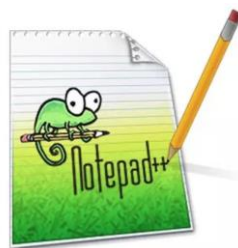


**PuTTY**

- 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 ( $\geq 2.7$  or  $\geq 3.3$ ),
- NumPy ( $\geq 1.8.2$ ),
- SciPy ( $\geq 0.13.3$ ).

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:

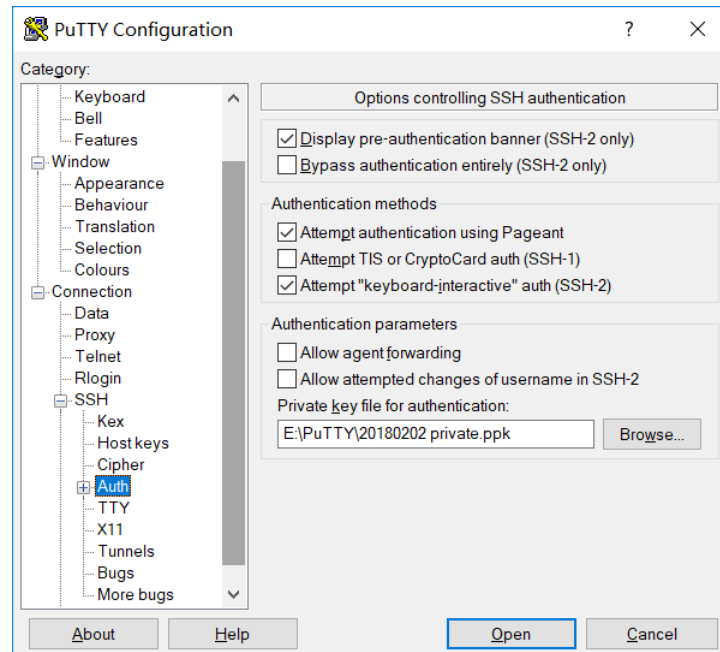
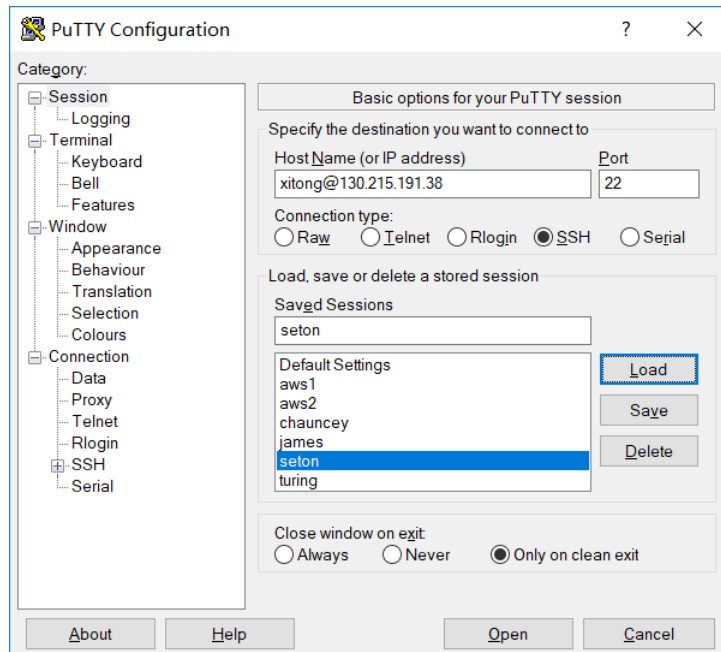
```
conda install -c conda-forge opencv
```

```
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;



```
autoreg-089187:~ setonzhang$ ssh xitong@130.215.191.38
Last login: Sun May 13 21:04:46 2018 from 130.215.13.201
xitong@ubuntu:~$
```

# SSH Key Generation (Putty)

**PuTTY Key Generator** ? X

File Key Conversions Help

Key

No key.

Actions

Generate a public/private key pair **Generate**

Load an existing private key file **Load**

Save the generated key **Save public key** **Save private key**

Parameters

Type of key to generate:  
☒ RSA ☐ DSA ☐ ECDSA ☐ ED25519 ☐ SSH-1 (RSA)

Number of bits in a generated key:

**PuTTY Key Generator** ? X

File Key Conversions Help

Key

Public key for pasting into OpenSSH authorized\_keys file:

```
ssh-rsa  
AAAAB3NzaC1yc2EAAAABJQAAQEAh6q81+TNvHUN4nmcukBILUMrzzW89M/4wbv9EqU  
7TN9qVozkxpMD8lRoaOtDnEo9ON5YdgOy5jBwjsWpNg81xNZCp965jTIZL7AkUKmkS3aggl  
qZeQA/tloc1THA7XMckJxAeUd1dF3Y2K  
+MD7dkrzQIZtAWKrrYAjWg9QXzxsUjbfTFHjwBHflbZuMv23+006SbFG02uKIm6ArKI
```

Key fingerprint:

Key comment:

Key passphrase:

Confirm passphrase:

Actions

Generate a public/private key pair **Generate**

Load an existing private key file **Load**

Save the generated key **Save public key** **Save private key**

Parameters

Type of key to generate:  
☒ RSA ☐ DSA ☐ ECDSA ☐ ED25519 ☐ SSH-1 (RSA)

Number of bits in a generated key:

# 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.

```
setonzhang — xitong@ubuntu: ~/.ssh
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDMegU9z7bS2X
kPxMbU+unjh1cJnCXnbK8ar5LnfMPrvhduHc2nq0QKXxGAGBtv
TCiqgnXFe0tJRz4S8DCiHHvg3hpjxyeoHZK6/nMR/KCtMXeUKI
QYVe655jgXZdxibBqLSp07elKS5IVnT+WR37TmV8/0v3ddtvbb
089187.dyn.wpi.edu
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDQhBQ8pQ9xwd
vX+62PAoVkJzW4Rt+/Kqirxvlz4FdhPZwUxF+IQYJIKaDAnUB5
T6i4Au407GPgNkQp1VmU07G6U00GT+lvCUISSpL4dWglE39+s
Jq3hjTNlhZGr0EgvZqQcYkGAPcJqRb9xI0HQ0HLitFo6dKMYAS
ook-Pro.local
@
-- INSERT --
```

(3) Insert the generated keys.

```
setonzhang — xitong@ubuntu: ~/.ssh
YourKeyHere...
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEak/QP43mCJoIjsg
A6qzC3T6LYHCjNEtH3IZbGGYy6NXFiGDb0v/ruHLoMMLzF1/Ny
Xa+4Q9ZXemX//riVGvW+88spaxdmKXsn6EImR4BEUeF9fj7bDz
138AzncBfdGIjItdp83WoJxT3ygnDVnmYLkA19emi/XYRTYZuj
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEai8DvIZw8Ssaq0W
NnHBJ5KwxJcaBHy6PmweqLq3BfKLpS7FBGHEy95EhPMix3ER3b
4C2vsA9YYjsGs7R61FJDVWpj5yFbQ/J3z3nZGZ/7Ruukc7TbPg
K2bFtvh0h93D/laUD5vCI5UxBAXVSNNAw3hukmN2ZDY6Az4k28
@
-- INSERT --
```

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

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

```
setonzhang — xitong@ubuntu: ~/.ssh
YourKeyHere...
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEak/QP43mCJoIjsg
A6qzC3T6LYHCjNEtH3IZbGGYy6NXFiGDb0v/ruHLoMMLzF1/Ny
Xa+4Q9ZXemX//riVGvW+88spaxdmKXsn6EImR4BEUeF9fj7bDz
138AzncBfdGIjItdp83WoJxT3ygnDVnmYLkA19emi/XYRTYZuj
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEai8DvIZw8Ssaq0W
NnHBJ5KwxJcaBHy6PmweqLq3BfKLpS7FBGHEy95EhPMix3ER3b
4C2vsA9YYjsGs7R61FJDVWpj5yFbQ/J3z3nZGZ/7Ruukc7TbPg
K2bFtvh0h93D/laUD5vCI5UxBAXVSNNAw3hukmN2ZDY6Az4k28
@
:wq
```

(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

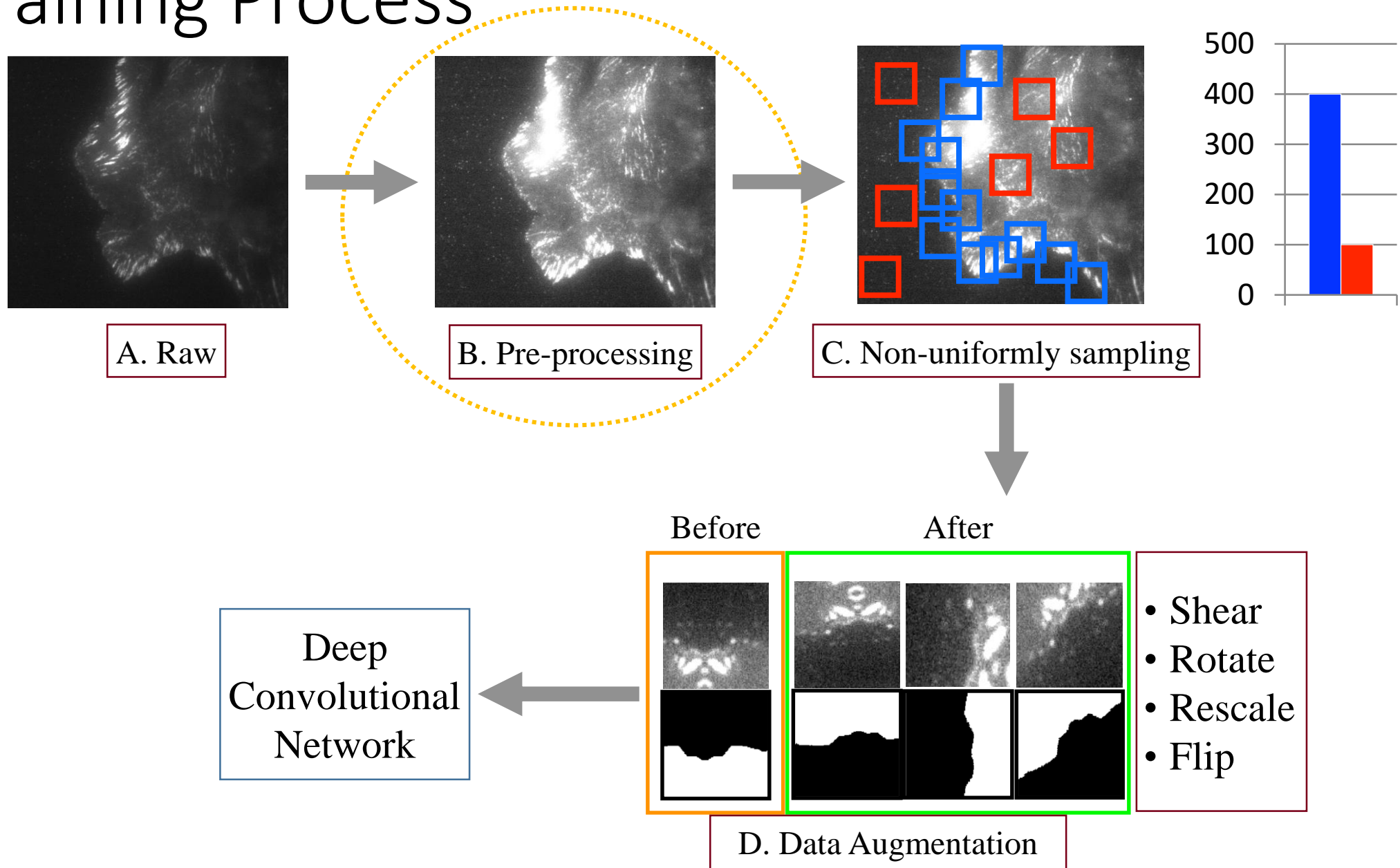
Command	Description
cd	<a href="#">Linux Commands</a>
ls	<a href="#">Linux Commands</a>
cp	<a href="#">Linux Commands</a>
mkdir	<a href="#">Linux Commands</a>
rm	<a href="#">Linux Commands</a>
mv	<a href="#">Linux Commands</a>
nvidia-smi	Check the status of GPUs
python	Run the process: python XXX.py
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	<a href="#">Linux Commands</a>
"tab"	Filename completion

# Mount Remote Disk

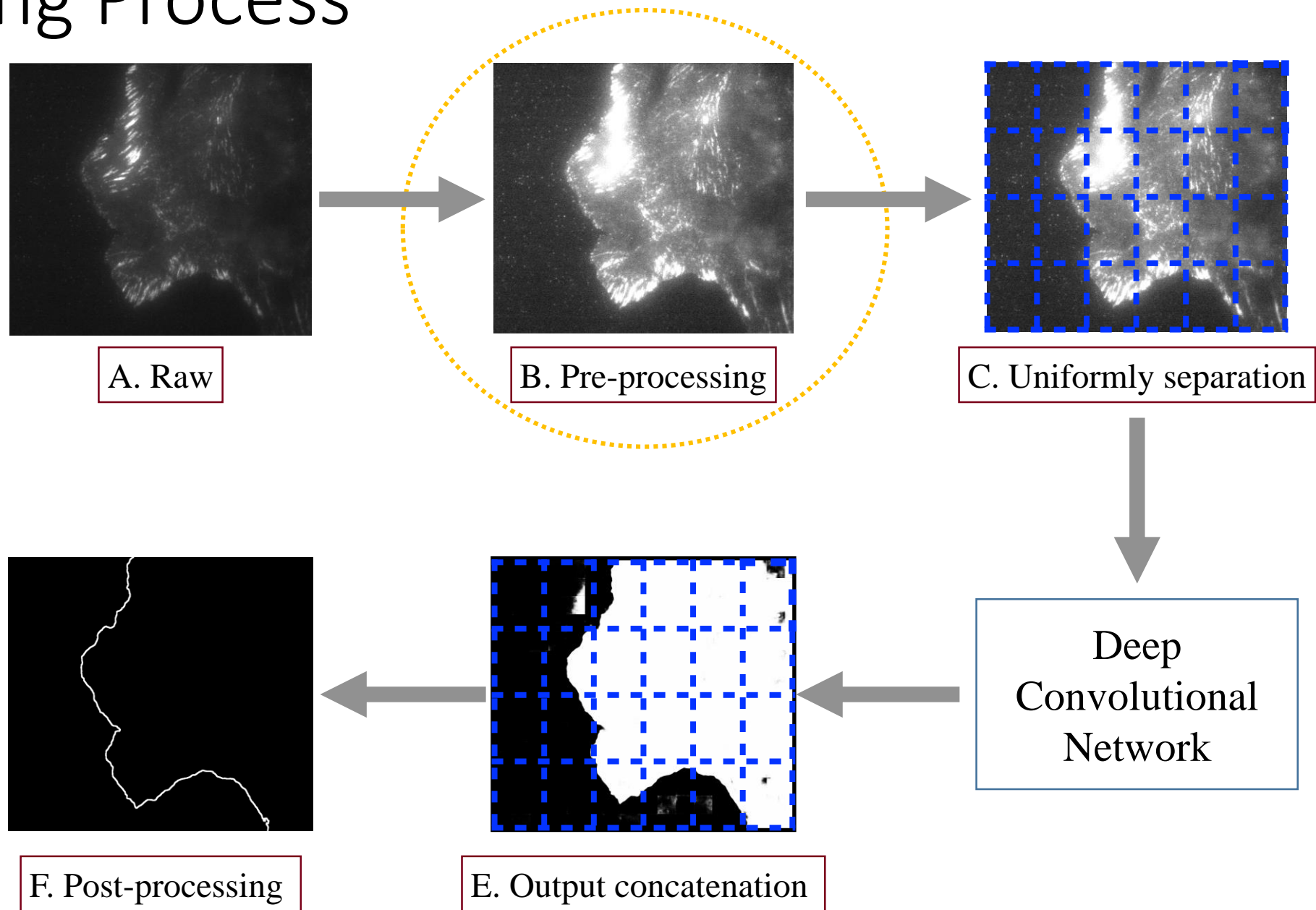
```
Connection to 130.215.191.38 closed.  
autoreg-089187:~ setonzhang$ ssh xitong@130.215.191.38  
Last login: Sun May 13 22:56:03 2018 from 130.215.13.201  
xitong@ubuntu:~$ vim .smbcredentials
```

```
username=  
password=
```

# Training Process



# Testing Process



# 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.