

# Manual of VirionFinder Release Version 1.0

## Table of Contents

### Part I. Physical host version

1. Operating system.....	2
2. Requirements.....	2
3. Preparation.....	2
4. Usage.....	3
5. Output.....	4

### Part II. Virtual machine version

1. Install the virtual machine and run VirionFinder in virtual machine.....	5
2. Exchange file between physical host and virtual machine.....	9

Any question, please do not hesitate to contact me: [fangzc@smu.edu.cn](mailto:fangzc@smu.edu.cn)

VirionFinder can run either in the physical host or virtual machine. The physical host version can speed up with GPU. For non-computer professionals, we recommend using the virtual machine version.

## Part I . Physical host version

### 1. Operating system

Linux (VirionFinder has been tested on Ubuntu 16.04)

### 2. Requirements

- Python 2.7.12 (<https://www.python.org/>)
- Python packages:
  - numpy 1.13.1(<http://www.numpy.org/>)
  - h5py 2.6.0(<http://www.h5py.org/>)
- TensorFlow 1.4.1 (<https://www.tensorflow.org/>)
- Keras 2.0.8 (<https://keras.io/>)
- MATLAB Component Runtime (MCR) R2018a  
(<https://www.mathworks.com/products/compiler/matlab-runtime.html>) or  
MATLAB R2018a (<https://www.mathworks.com/products/matlab.html>)

#### Note:

1. For compatibility, we recommend installing the tools with the similar version as described above.
2. If GPU is available in your machine, we recommend installing a GPU version of the TensorFlow to speed up the program.
3. VirionFinder can be run with either an executable file or a MATLAB script. If you run VirionFinder through the executable file, you need to install the MCR while MATLAB is not necessary. If you run VirionFinder through the MATLAB script, MATLAB is required.

### 3. Preparation

Please install numpy, h5py, TensorFlow, Keras, MCR (or MATLAB) according to their manuals.

The numpy, h5py, TensorFlow, and Keras are python packages, which can be installed with “pip”. If “pip” is not already installed in your machine, use the command “sudo apt-get install python-pip python-dev” to install “pip”. Here are example commands of installing the above python packages using “pip”.

**pip install numpy**

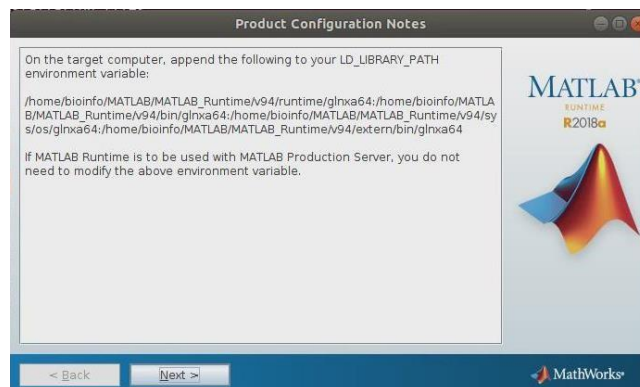
```
pip install h5py
pip install tensorflow==1.4.1 #CPU version
pip install tensorflow-gpu==1.4.1 #GPU version
pip install keras==2.0.8
```

If you are going to install a GPU version of the TensorFlow, specified NVIDIA software should be installed. See <https://www.tensorflow.org/install/gpu> to know whether your machine can install TensorFlow with GPU support.

When running VirionFinder through the executable file, MCR should be installed. See <https://www.mathworks.com/help/compiler/install-the-matlab-runtime.html> to install MCR. On the target computer, please append the following to your LD\_LIBRARY\_PATH environment variable according to the tips of MCR:

```
<MCR_installation_folder>/v94/runtime/glnxa64
<MCR_installation_folder>/v94/bin/glnxa64
<MCR_installation_folder>/v94/sys/os/glnxa64
<MCR_installation_folder>/v94/extern/bin/glnxa64
```

A screenshot of the tips when installing MCR is shown below:



When running VirionFinder through the MATLAB script, please see <https://www.mathworks.com/support/> to install the MATLAB.

## 4. Usage

To run VirionFinder, please download VirionFinder package from GitHub website <https://github.com/zhenchengfang/VirionFinder>.

### 4.1. Run by executable file (in command line)

In this form, please simply executes the command:

```
./VirionFinder <input_file_folder>/input_file.faa <output_file_folder>/output_file.csv
```

The input file must be in fasta format containing the sequences to be identified. The users can use the file “example.faa” which contains 1074 sequences in the program folder to test the VirionFinder by simply executing the command:

## **./VirionFinder example.faa result.csv**

### **4.2. Run by MATLAB script (in MATLAB GUI)**

In this form, please execute the following command directly in the MATLAB command window.

**VirionFinder(<input\_file\_folder>/input\_file.faa', <output\_file\_folder>/output\_file.csv')**

For example, if you want to identify the sequences in example.fna, please execute:

**VirionFinder('example.faa', 'result.csv')**

Remember to set the working path of MATLAB to the program folder before running the program.

### **4.3. Run VirionFinder over a large file (-b option)**

If the RAM of your machine is small, or your file is very large, you can you -b option to let the program read the file in block to reduce the memory requirements and speed up the program. For example, if you want to let the program to predict 1000 sequences at a time, please execute:

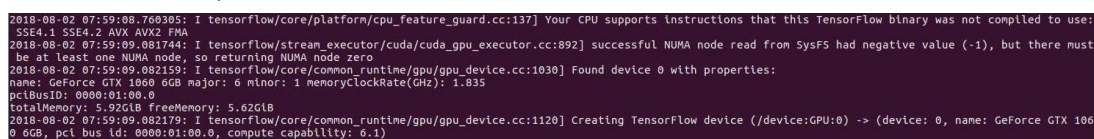
**./VirionFinder example.faa result.csv -b 1000** (Run by executable file)

*or*

**VirionFinder('example.faa', 'result.csv', '-b', '1000')** (Run by MATLAB script)

The default value of -b is 10000.

**Note:** When running VirionFinder, you can ignore the warning about the information of the CPU/GPU, as shown in the screenshot below.



```
2018-08-02 07:59:08.760305: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2018-08-02 07:59:09.081744: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:892] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2018-08-02 07:59:09.082159: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1030] Found device 0 with properties:
  name: GeForce GTX 1060 6GB major: 6 minor: 1 memoryClockRate(GHz): 1.835
  pciBusID: 0000:01:00.0
totalMemory: 5.92GiB freeMemory: 5.62GiB
2018-08-02 07:59:09.082179: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1120] Creating TensorFlow device (/device:GPU:0) -> (device: 0, name: GeForce GTX 1060 6GB, pci bus id: 0000:01:00.0, compute capability: 6.1)
```

## **5. Output**

The output of VirionFinder consists of 4 columns, representing “sequence header” (the same with the corresponding header in the fasta file), “sequence length”, “the probability score that the sequence belongs to the PVP”, and “prediction”, respectively. Here is a screenshot of the output file:

	A	B	C	D
1	Header	Length	Score	Prediction
2	p1	367	0.911312	PVP
3	p2	489	0.739012	PVP
4	p3	792	0.882658	PVP
5	p4	367	0.911312	PVP
6	p5	745	0.973464	PVP

**Note:**

The current version of VirionFinder uses “comma-separated values (CSV)” as the format of the output file. Please use “.csv” as the extension of the output file. VirionFinder will automatically add the “.csv” extension to the file name if the output file does not take “.csv” as its extension”

## Part II. Virtual machine version

Running VirionFinder in the virtual machine is much easier for user who is not familiar with the command line, and the virtual machine can be installed in any PC. Note that the running time of the virtual machine version may be longer because the virtual machine could not speed up with GPU. We also modified the code to separate the input sequences into more batches to reduce memory requirements, which may increase the running time.

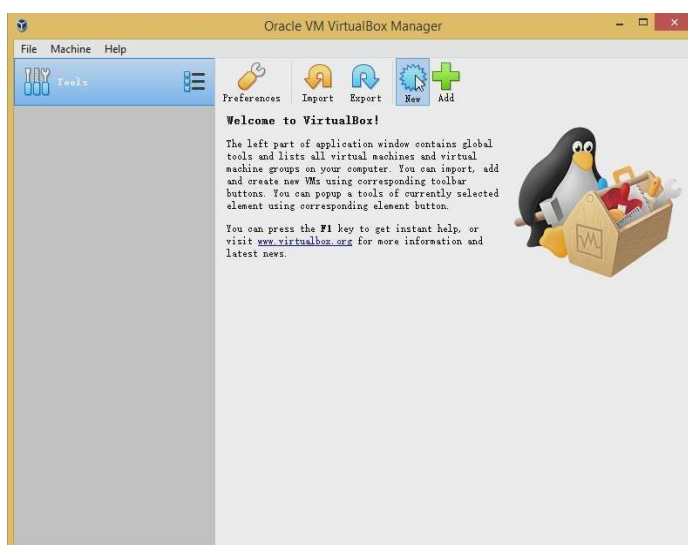
The following is the step by step guide to run VirionFinder in virtual machine.

### 1. Install the virtual machine and run VirionFinder in virtual machine.

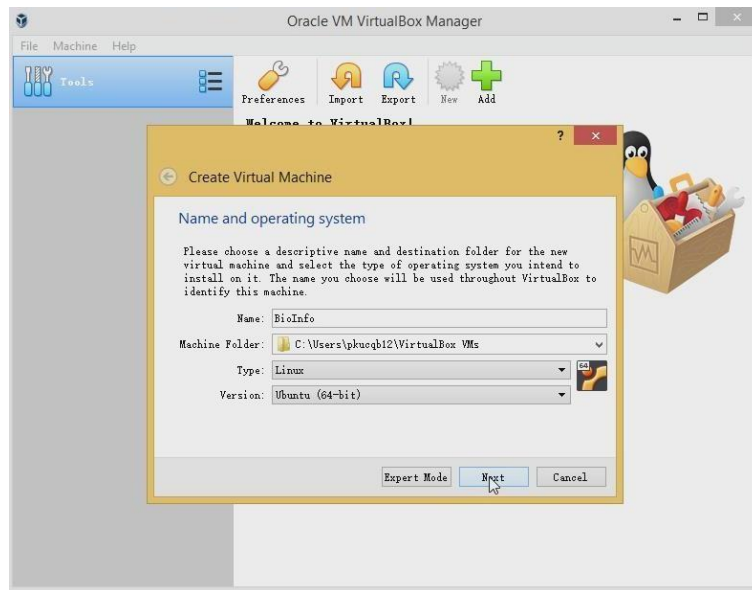
**Step 1:** download the “VM\_Bioinfo.vdi.7z” file from ([http://cqb.pku.edu.cn/ZhuLab/PPR\\_Meta/VM\\_Bioinfo.vdi.7z](http://cqb.pku.edu.cn/ZhuLab/PPR_Meta/VM_Bioinfo.vdi.7z)). The “7z” file can easily be decompressed using a current compressing software, such as “WinRAR”, “WinZip”, and “7-Zip”.

**Step 2:** download the VirtualBox software form <https://www.virtualbox.org> and install the VirtualBox. The VirtualBox is easy to install, you just need to select an installation folder and click the “next” button in each step. The version of the VirtualBox we used was 6.0.4.

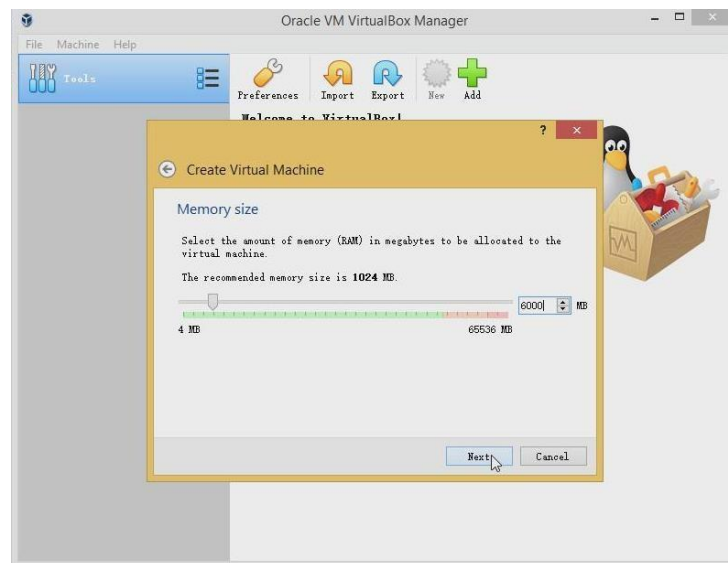
**Step 3:** Open VirtualBox, click the “New” button to create virtual machine.



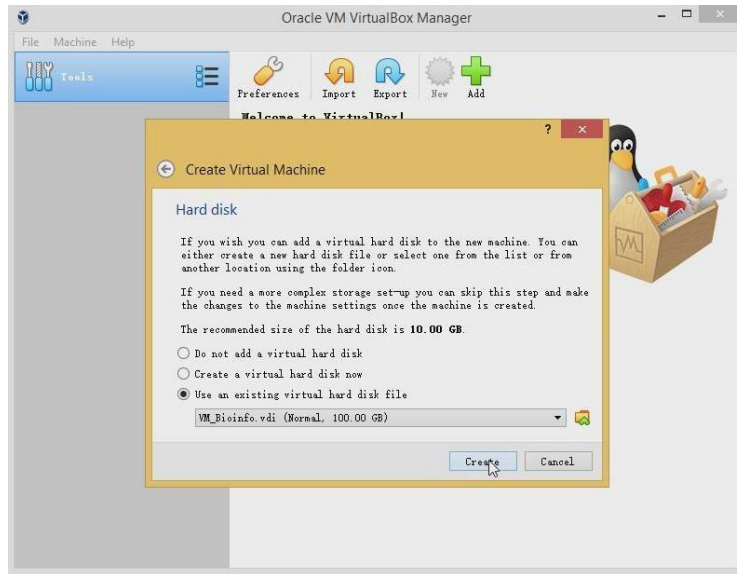
**Step 4:** Specify a name, select the “Linux” as the operating system and select “Ubuntu” as the version of the operating system. Then, click “Next”.



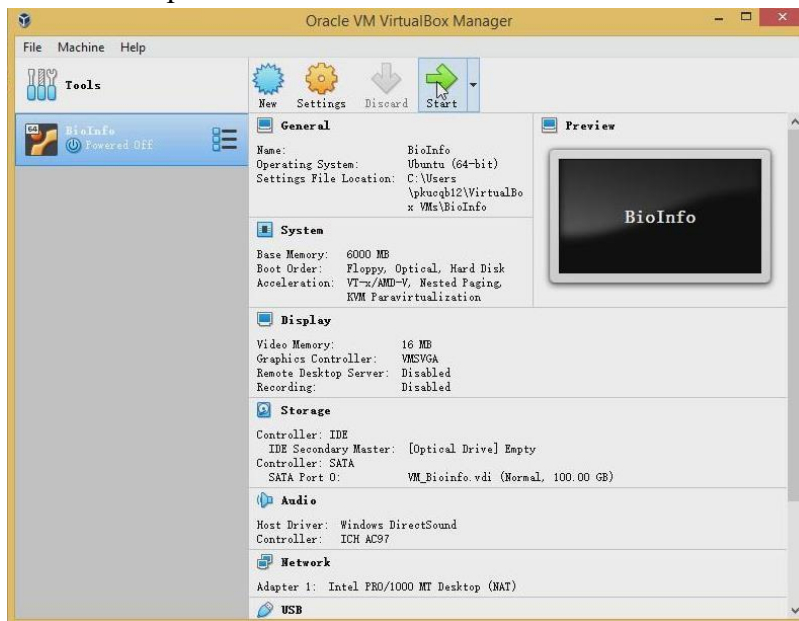
**Step 5:** If possible, allocate a larger amount of memory to the virtual machine. Click “next”.



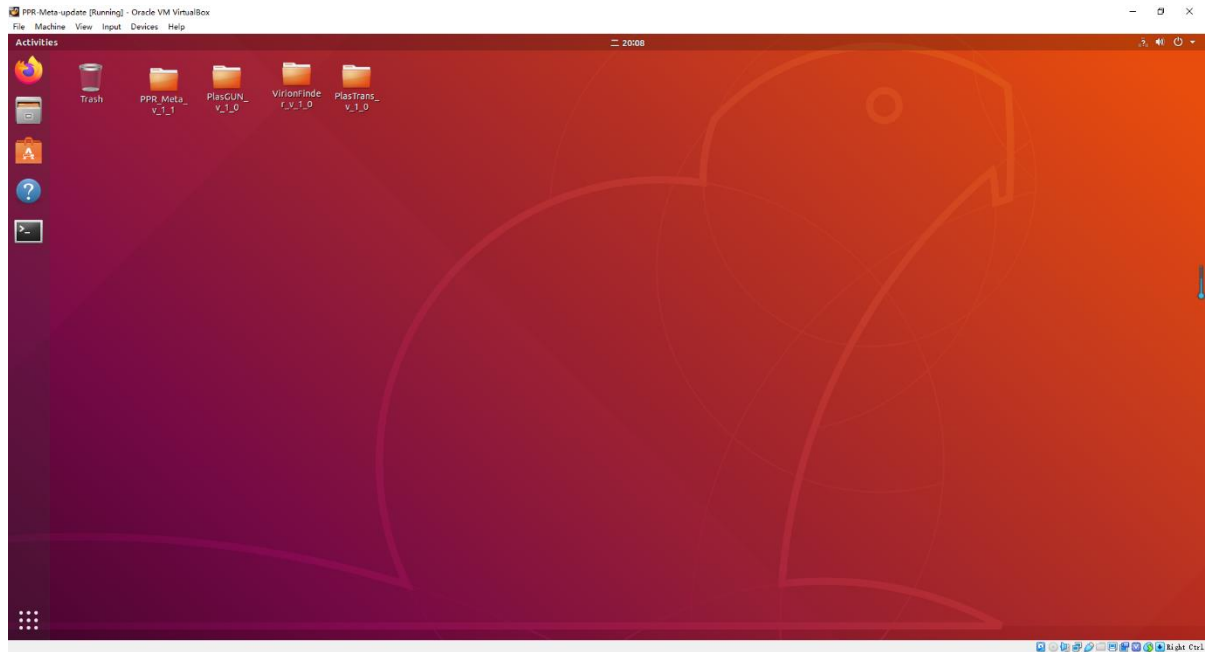
**Step 6:** Select “Use an existing virtual hard disk file”, and specify the “VM\_Bioinfo.vdi” file downloaded from Step 1. Click “Create”.



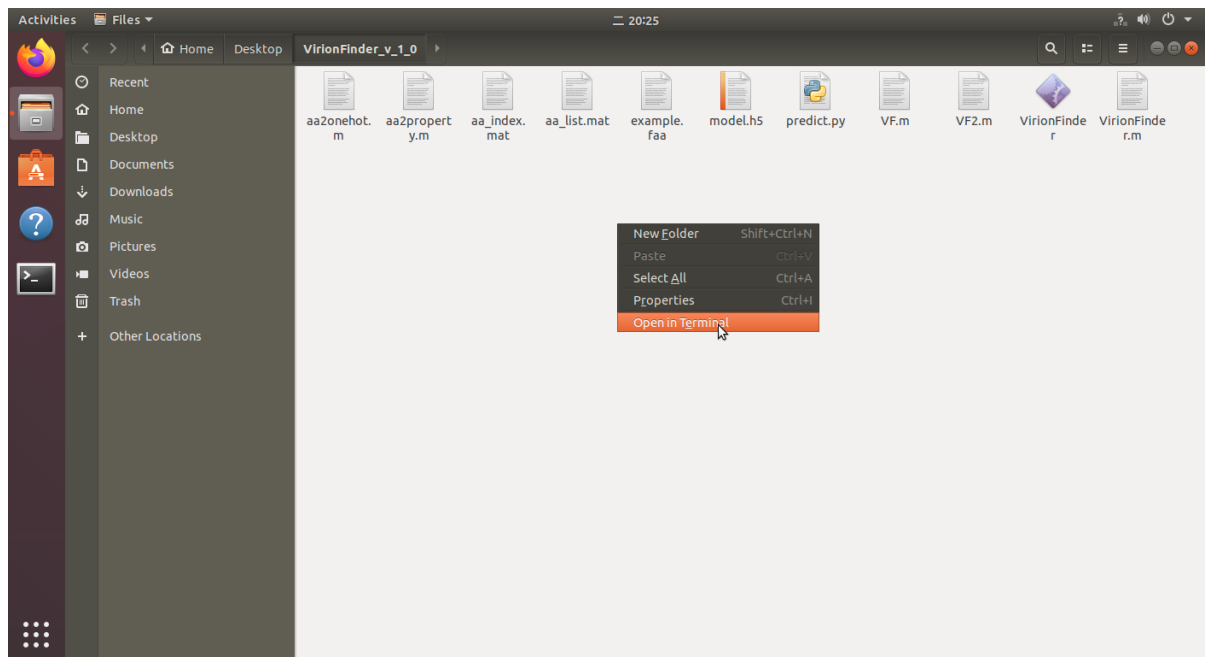
**Step 7:** Click “start” to open the machine.



**Step 8:** The VirionFinder is on the desktop.

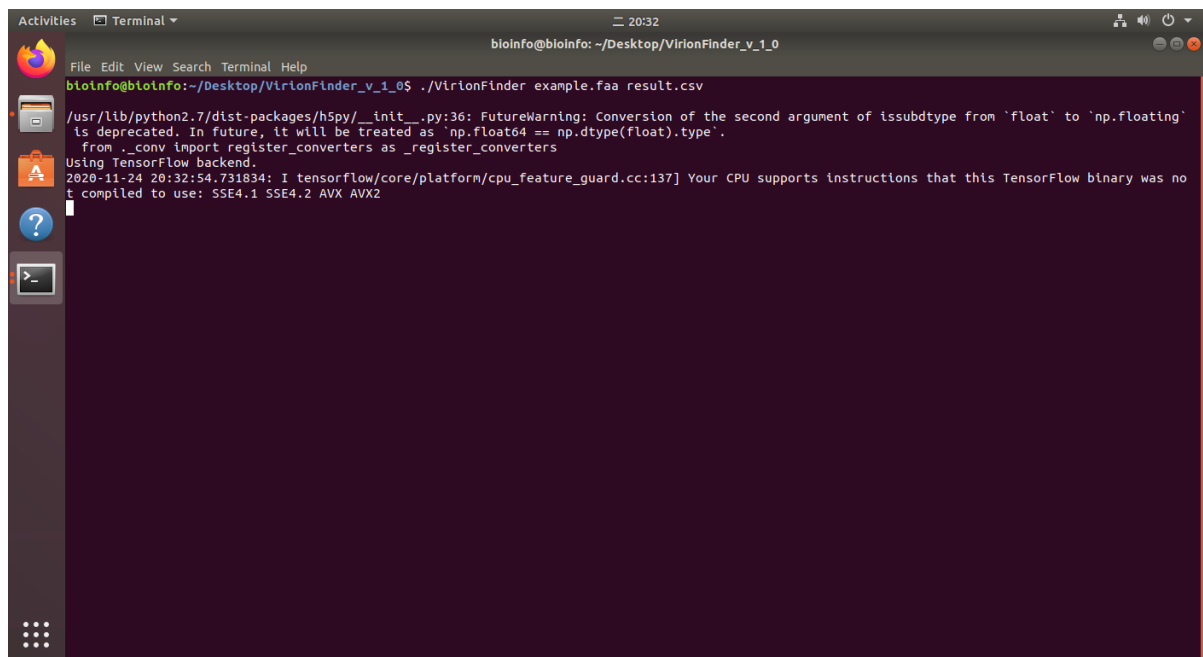


**Step 9:** Go into the “VirionFinder\_v\_1\_0” folder, click the right click and open the terminal.





**Step 10:** Now you can run VirionFinder (see also part I , section 4.1 and 4.3). You can ignore the “FutureWarning” and the information about the CPU.

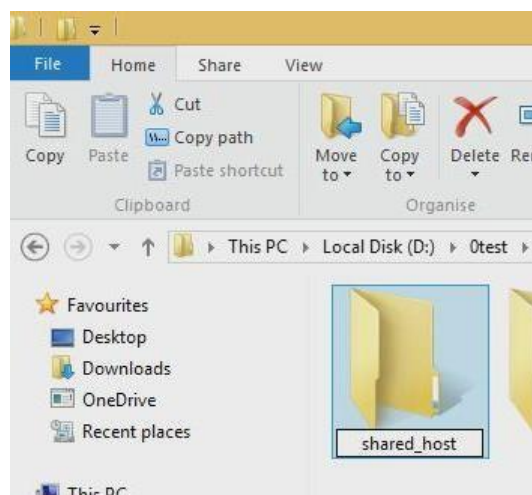


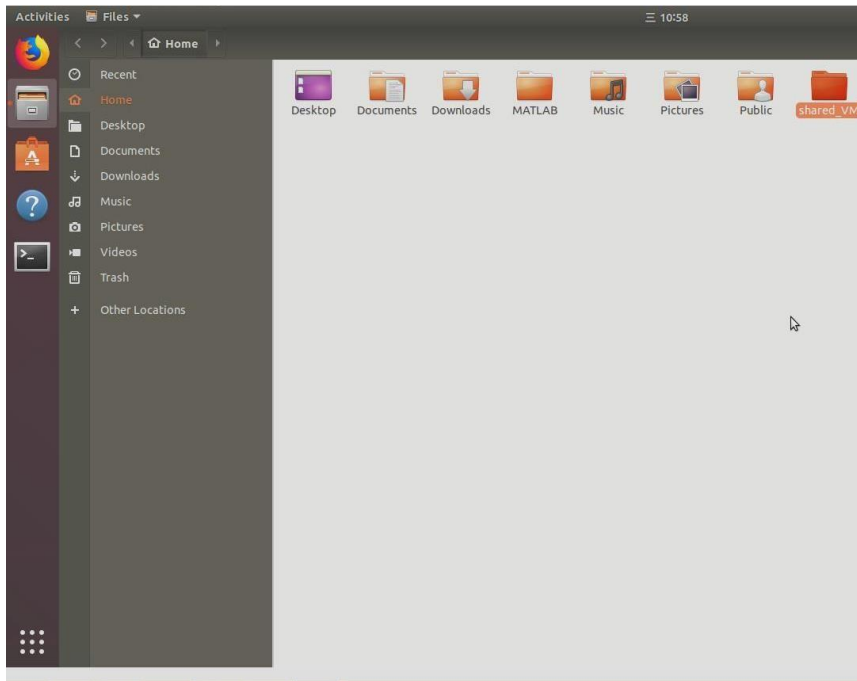
```
bioinfo@bioinfo: ~/Desktop/VirionFinder_v_1_0
bioinfo@bioinfo:~/Desktop/VirionFinder_v_1_0$ ./VirionFinder example.faa result.csv

/usr/lib/python2.7/dist-packages/h5py/__init__.py:36: FutureWarning: Conversion of the second argument of issubdtype from 'float' to 'np.floating'
is deprecated. In future, it will be treated as 'np.float64 == np.dtype(float).type'.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2020-11-24 20:32:54.731834: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports instructions that this TensorFlow binary was no
t compiled to use: SSE4.1 SSE4.2 AVX AVX2
```

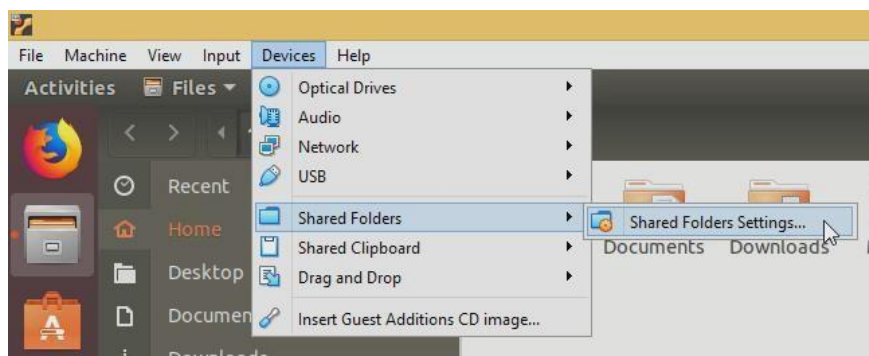
## 2. Exchange file between physical host and virtual machine.

**Step 1:** Create shared folders in both physical host (shared\_host) and virtual machine (shared\_VM).

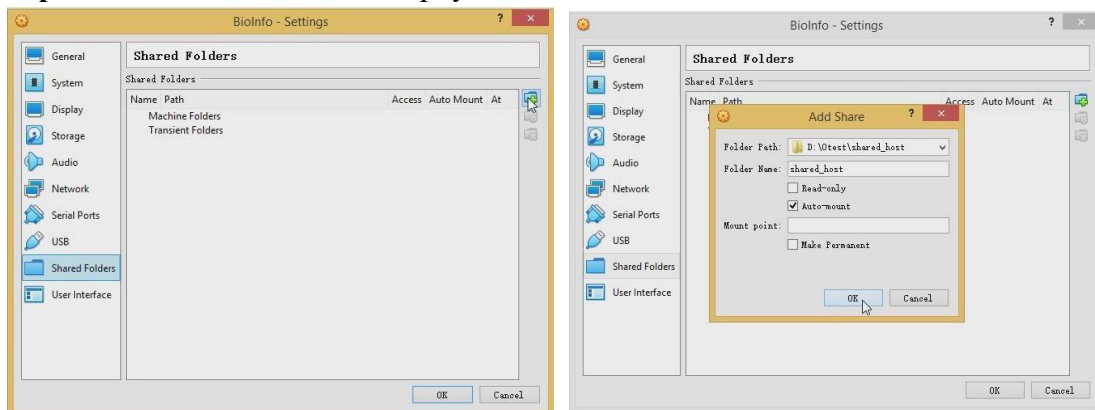




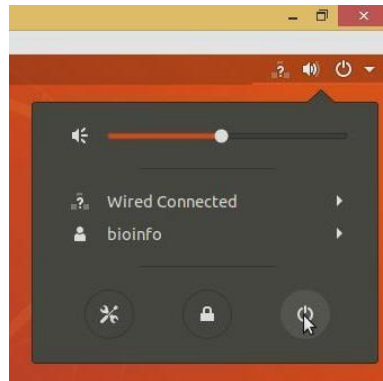
**Step 2:** In the window of VirtualBox, click “Devices”, “Shared Folder”, “Shared Folders Settings”.



**Step 3:** Add shared folder of the physical host and select “Auto-mount”.



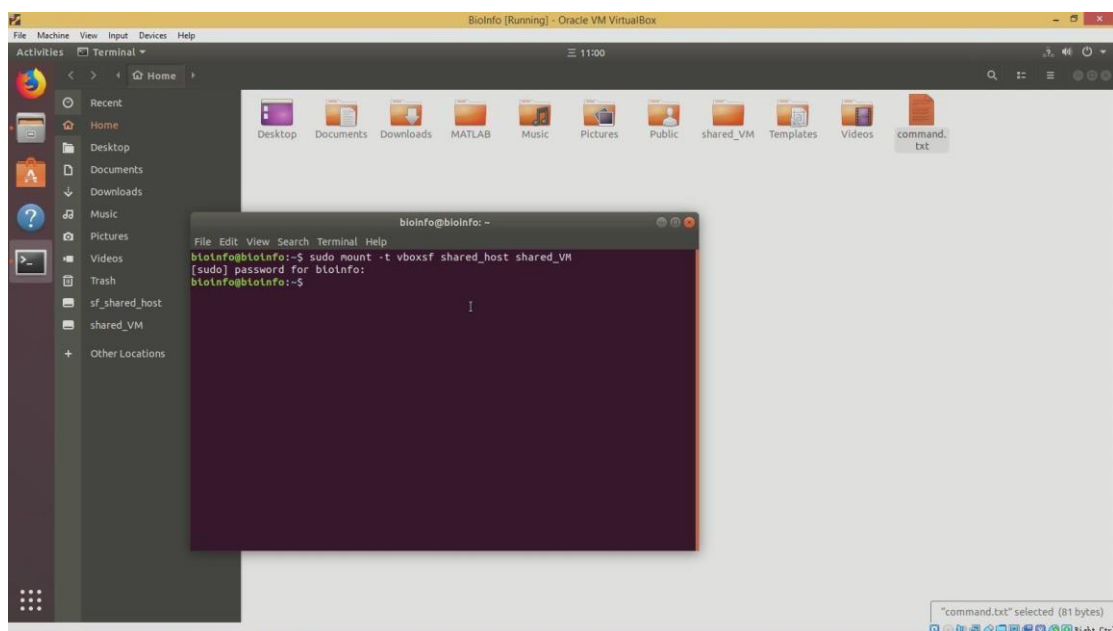
**Step 4:** Restart the virtual machine.



**Step 5:** Go to the parent folder of the shared folder in the virtual machine, click the right click and open the terminal. Copy the command in the “command.txt” file to the terminal:

**`sudo mount -t vboxsf shared_host shared_VM`**

Note, you should replace “shared\_host” and “shared\_VM” to the shared folders’ name that you specify. **The password of the virtual machine is 1.**



**Step 6:** Now, you can exchange files between virtual machine and physical host. For example, you can copy the file in the virtual machine to the “shared\_VM” folder, and this file will also exist in the “shared\_host” folder in the physical host, vice versa. If you want to know more about the file exchange, click “Help” -> “Contents” -> “Guest Additions” -> “Shared folders” in the VirtualBox window for more details.

**Note:**

The program allows run mutiple tasks in parallel. However, running mutiple same tasks (with the same input file under the same '-b' setting) will throw error.