连接虚拟机

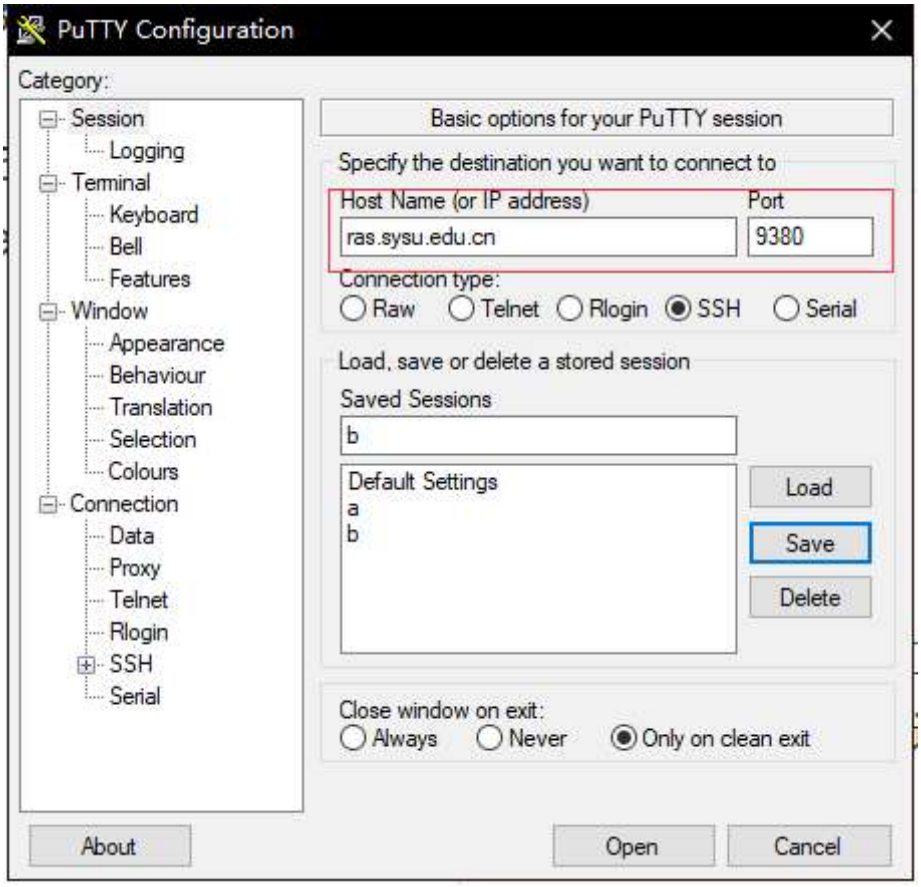
首先，我的服务器分配的IP是**10.0.3.153**，端口是**9380**至**9389**，服务器的域名为ras.sysu.edu.cn

JinZli	RUNNING	9380~9389	10.0.3.153	00:00:01

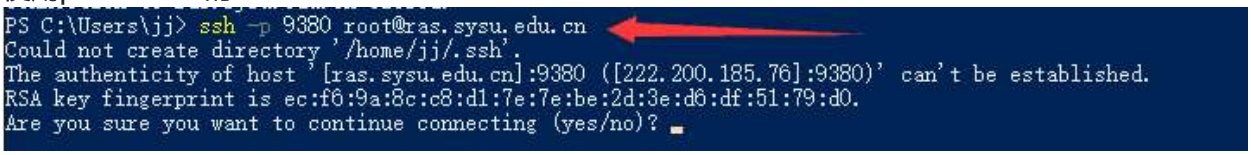
有两种方法：

- 使用**ssh**连接 此时，登陆的端口号为从**9380**至**9389**分配的10个端口号中的**第一个端口号**(9380)

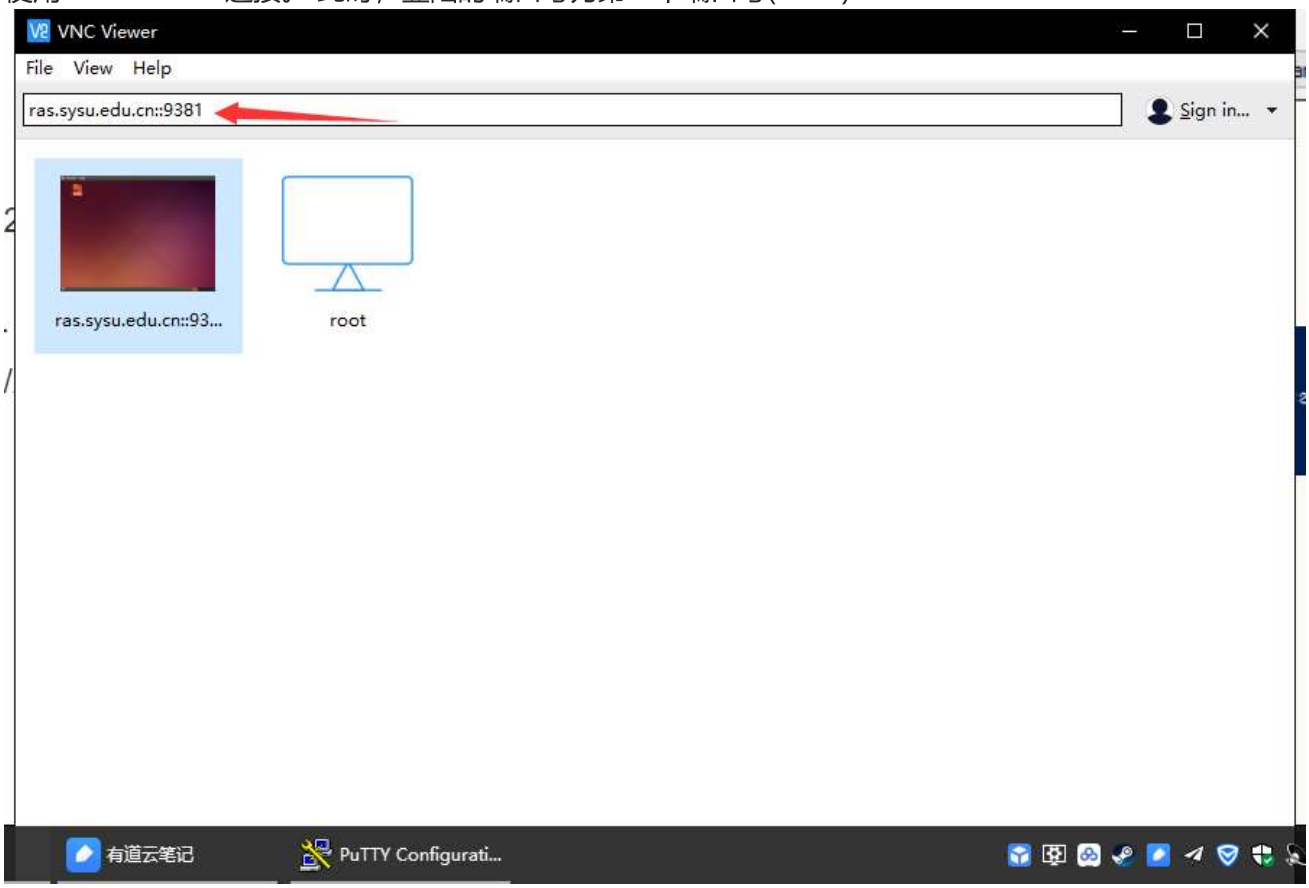
i. 使用putty



ii. 使用powershell的ssh



- 使用**VNC viewer**连接。 此时，登陆的端口号为第二个端口号(9381)



至此，你可以通过多种方法登陆到实验室的虚拟机上了

配置翻墙(未成功)

1. 试图下载shadowsocks。认为还是直接设置浏览器或系统全局代理更方便，放弃了此方案

2. 试图设置代理，似乎相关功能被屏蔽了

安装anaconda

1. 登陆anaconda官网 https://www.anaconda.com/download/#linux
2. 下载linux版的installer
3. sh xxx.sh

4. 为anaconda添加环境变量 遇到了无法添加环境变量的困难，解决方法见文末**[遇到的问题2]**
5. 修改镜像源 https://blog.csdn.net/mtj66/article/details/57074986

安装Tensorflow(GPU版本)

1. 检查是否满足最低安装要求。于 https://www.tensorflow.org/install/install_linux#nvidia_requirements_to_run_tensorflow_with_gpu_support有描述。

   If you have an earlier version of the preceding packages, please upgrade to the specified versions. If upgrading is not possible, then you may still run TensorFlow with GPU support, but only if you do the following:

   - Install TensorFlow from sources as documented in Installing TensorFlow from Sources.
   - Install or upgrade to at least the following NVIDIA versions:
     - CUDA toolkit 7.0 or greater
     - cuDNN v3 or greater
     - GPU card with CUDA Compute Capability 3.0 or higher.

   - 检查CUDA版本

     ```
     (tensorflow) -> root@JinZili:Public $ cat /usr/local/cuda/version.txt
     CUDA Version 7.5.18
     ```

     版本为7.5，**大于7.0**
   - 检查cudnn版本

     ```
     (tensorflow) ->  root@JinZili:Public $ cat /usr/local/cuda/include/cudnn.h | grep
     CUDNN_MAJOR -A 2
     #define CUDNN_MAJOR      6
     #define CUDNN_MINOR      0
     #define CUDNN_PATCHLEVEL 20
     --
     #define CUDNN_VERSION    (CUDNN_MAJOR * 1000 + CUDNN_MINOR * 100 +    CUDNN_PATCHLEVEL)

     #include "driver_types.h"
     ```

     版本为6，**大于3**
   - 检查GPU的CUDA compability 从服务器虚拟机使用指南查到Amax的GPU版本为Tesla K40c。从GPU版本与对应的Compute Capability的查阅网址查到，Tesla K40c对应的Compute Capability为3.5，**大于3.0**

> 在第四步不要按照它的做法，因为国内连不上外面的连接。应当改用 `conda install tensorflow-gpu==1.3` 。此处故意要用低版本的1.3，原因见文末。

4. Issue a command of the following format to install TensorFlow inside your conda environment:

```
(tensorflow)$ pip install --ignore-installed --upgrade tfBinaryURL
```

改用conda install tensorflow-gpu==1.3

where *tfBinaryURL* is the URL of the TensorFlow Python package. For example, the following command installs the CPU-only version of TensorFlow for Python 3.4:

```
(tensorflow)$ pip install --ignore-installed --upgrade \
https://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-1.6.0-cp34-cp34m-linux_x86_64.wh
```

2. 安装libcupti-dev 根据安装说明，我们还需要执行 `apt-get install libcupti-dev`

NVIDIA requirements to run TensorFlow with GPU support

If you are installing TensorFlow with GPU support using one of the mechanisms described in this guide, then the following NVIDIA software must be installed on your system:

- CUDA® Toolkit 9.0. For details, see NVIDIA's documentation. Ensure that you append the relevant Cuda pathnames to the `LD_LIBRARY_PATH` environment variable as described in the NVIDIA documentation.
- The NVIDIA drivers associated with CUDA Toolkit 9.0.
- cuDNN v7.0. For details, see NVIDIA's documentation. Ensure that you create the `CUDA_HOME` environment variable as described in the NVIDIA documentation.
- GPU card with CUDA Compute Capability 3.0 or higher for building from source and 3.5 or higher for our binaries. See NVIDIA documentation for a list of supported GPU cards.
- The libcupti-dev library, which is the NVIDIA CUDA Profile Tools Interface. This library provides advanced profiling support. To install this library, issue the following command for CUDA Toolkit >= 8.0:

```
$ sudo apt-get install cuda-command-line-tools
```

and add its path to your `LD_LIBRARY_PATH` environment variable:

```
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/extras/CUPTI/lib64
```

For CUDA Toolkit <= 7.5 do:

```
$ sudo apt-get install libcupti-dev
```

3. 成功安装并执行示例代码

```
# Python
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
print(sess.run(hello))
```

```
>>> print(sess.run(hello))
b'Hello, TensorFlow!'
```

限制TensorFlow使用的GPU资源。

在代码中加入以下片段

```
# Assume that you have 12GB of GPU memory and want to allocate ~4GB:
gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=0.333)

sess = tf.Session(config=tf.ConfigProto(gpu_options=gpu_options))
```

了解深度学习

什么是深度学习?

> 定义一个深度学习模型，通常需要解决3个问题： 1）激活函数，也就是先对于输入神经元的激活值。一般的有 logistic 、tanh、以及ReLU。 2）代价函数。一般学习过程都是优化问题。代价函数一般采用欧式距离。 3）优化策略。最简单的用梯度下降。

深度学习Deep Learning（01）_CNN卷积神经网络

人工智能、机器学习和深度学习的区别?

遇到的问题

1. vnc服务器重启失败
   - 问题如下

```
]# vncserver -kill :1
Can't find file /root/.vnc/xxx.xx.xx.xx:1.pid  // IP已隐去，用xxx.xx.xx.xx代替
You'll have to kill the Xvnc process manually
```

   - 解决方法 vncserver Can't find file /root/.vnc/192.168.1.3:1.pid You'll have to kill the Xvnc process manually

2. zsh: command not found: conda (未添加环境变量)
   - 探索路径: 试图修改 `/etc/profile` 来修改环境变量，但失败了
   - 解决方法: 在zsh下添加环境变量，参考 https://blog.csdn.net/qq_16735407/article/details/50561144

```
vi /root/.zshrc
```

在最后一行加上这一段

```
PATH=/root/anaconda3/bin:$PATH;

export PATH
```

按Esc，再按 : 和 w，q保存退出。激活文件

```
source /root/.zshrc
```

3. pip下载速度过慢

   - 解决方法: 替换为清华源(临时用)：在使用pip的时候加参数 -i https://pypi.tuna.tsinghua.edu.cn/simple

4. conda下载速度过慢

   - 解决方法: 替换为清华源

```
conda config --add channels      https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/

conda config --set show_channel_urls yes
```

5. 安装后试图运行 `import tensorflow as tf` 失败

   - 参考 ImportError: libcublas.so.9.0: cannot open shared object file: No such file or directory 该文认为，这种情况应该是**TensorFlow版本过高**导致的，虚拟机安装的CUDA版本较低，各种lib的版本落后，因此不兼容。
   - 问题定位 输入 `ls /usr/local/cuda/lib64` 后，发现lib的版本都低于8.0 比如 `libcudart.so` ，虚拟机只有7.5版本，而tensorflow1.6要求9.0

```
(tensorflow) ➜ root@JinZili:Public $ ls /usr/local/cuda/lib64
libcublas_device.a   libcuinj64.so          libnppi.so.7.5.18
libcublas.so         libcuinj64.so.7.5      libnppi_static.a
libcublas.so.7.5     libcuinj64.so.7.5.18   libnpps.so
libcublas.so.7.5.18  libculibos.a           libnpps.so.7.5
libcublas_static.a   libcurand.so           libnpps.so.7.5.18
libcudadevrt.a       libcurand.so.7.5       libnpps_static.a
libcudart.so         libcurand.so.7.5.18    libnvblas.so
libcudart.so.7.5     libcurand_static.a     libnvblas.so.7.5
libcudart.so.7.5.18  libcusolver.so         libnvblas.so.7.5.18
libcudart_static.a   libcusolver.so.7.5     libnvrtc-builtins.so
libcudnn.so          libcusolver.so.7.5.18  libnvrtc-builtins.so.7.5
libcudnn.so.6        libcusolver_static.a   libnvrtc-builtins.so.7.5.18
libcudnn.so.6.0.20   libcusparse.so         libnvrtc.so
libcudnn_static.a    libcusparse.so.7.5     libnvrtc.so.7.5
libcufft.so          libcusparse.so.7.5.18  libnvrtc.so.7.5.17
libcufft.so.7.5      libcusparse_static.a   libnvToolsExt.so
libcufft.so.7.5.18   libnppc.so             libnvToolsExt.so.1
libcufft_static.a    libnppc.so.7.5         libnvToolsExt.so.1.0.0
libcufftw.so         libnppc.so.7.5.18      libOpenCL.so
libcufftw.so.7.5     libnppc_static.a       stubs
libcufftw.so.7.5.18  libnppi.so
```

   - 解决方法 在conda的虚拟环境下运行 `conda install tensorflow-gpu==1.3` ，然后就能导入Tensorflow了

```
(tensorflow) ➜ root@JinZili:Public $ python
Python 3.6.2 |Continuum Analytics, Inc.| (default, Jul 20 2017, 13:51:32)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow
>>>
```

6. 虚拟机无法连接google

    - 解决方法 根据虚拟机翻墙指南配置。此外，还要自行修改代理服务器的ip为 `192.168.0.11` (指南留下的ip配置是 `192.168.0.1`)。

心得

- 以前试图为自己翻墙的服务器**配置图形界面**，但悟性不足，未能解决。看到实验室的虚拟机都自带vnc连接的方式，下决心研究了一波，终于成功给自己的服务器配置了**vnc server**。心得已总结在单独的pdf中。
- 没有仔细阅读注意事项。其实Public文件夹下有很多预留的安装包可以双击安装。
- 由于自己用的是conda，安装时应坚持用 `conda install xxx` 而不是 `pip install xxx` 。因为conda是倾向于为特殊环境的包管理服务的，而pip倾向于为整个系统的python包管理服务。