# Windows

## CMD

切换目录：cd /d E:\root\data-tmp

设置代理（Ubuntu: export）：

set http\_proxy=http://127.0.0.1:1080

set https\_proxy=http://127.0.0.1:1080

恢复：

set http\_proxy=

set https\_proxy=

创建文件：type nul >文件名.后缀

生成文件夹目录结构：tree /f>structure.txt

# Pytorch 使用

import torch

查看版本：torch.\_\_version\_\_

是否支持GPU计算：torch.cuda.is\_available()

# CUDA

## 查看CUDA版本

cuda默认安装在/usr/local目录，可以使用ls -l /usr/local | grep cuda查看该目录下有哪些cuda版本

假设有如下输出：

lrwxrwxrwx  1 root root    8 Apr 26  2019 cuda -> cuda-9.0

drwxr-xr-x 11 root root 4096 Apr 26  2019 cuda-10.0

drwxr-xr-x 11 root root 4096 Apr 26  2019 cuda-8.0

drwxr-xr-x 11 root root 4096 Apr 26  2019 cuda-9.0

则表示当前机器上安装了8.0、9.0、10.0三个cuda版本，**/usr/local/cuda是一个软链接，链接到/usr/local/cuda-9.0目录**，表示当前使用的是cuda-9.0版本

如果要查看详细的cuda版本号，可以：

cat /usr/local/cuda/version.txt

如果当前机器上安装了多个cuda版本，可以使用修改软链接的方式来修改系统使用的cuda版本，命令如下：

sudo ln -snf /usr/local/cuda-8.0 /usr/local/cuda

## 安装位置

/usr/local/cuda/

# Anaconda

## 查看Anaconda版本

conda -V

conda --version

## 查看Python环境

conda info --env可以看到所有python环境，前面有个‘\*’的代表当前环境

## 创建Python环境

conda create --name python35 python=3.5 代表创建一个python3.5的环境，把它命名为python35

## 管理和使用python环境

使用conda activate python35 来激活刚才创建的环境（source activate）

pip install numpy来给环境安装numpy包，也可以用conda install numpy来安装

使用conda remove -n python35 --all删除环境python35

使用conda create -n new\_env --clone exist\_env

new\_env是新创建的Conda环境，exist\_env是被复制的Conda环境

## 查看环境中安装的packages

pip list

# Linux环境变量

## 读取环境变量

export命令显示当前系统定义的所有环境变量 OR set OR env

echo $ENV\_VAR\_NAME 输出环境变量ENV\_VAR\_NAME的值

echo $PATH命令输出当前的PATH环境变量的值→定义了运行命令的查找路径，以冒号:分割不同的路径，使用export定义的时候可加双引号也可不加

## 常用的环境变量

PATH 决定了shell将到哪些目录中寻找命令或程序

HOME 当前用户主目录

HISTSIZE　历史记录数

LOGNAME 当前用户的登录名

HOSTNAME　指主机的名称

SHELL 当前用户Shell类型

LANGUGE 　语言相关的环境变量，多语言可以修改此环境变量

MAIL　当前用户的邮件存放目录

PS1　基本提示符，对于root用户是#，对于普通用户是$

## 设置环境变量

### 方法1：export（临时）

export VAR\_NAME=content

### 方法2：修改~/.bashrc OR ~/.bash\_profile文件（长久）

先用vim打开

在最后一行插入：export VAR\_NAME=content

## 删除环境变量

unset VAR\_NAME

## 增加PATH中的路径

### 方法1：export PATH（临时）

export PATH=t\_path/bin:$PATH

或者把PATH放在前面

export PATH=$PATH:t\_path/bin

**注意事项：**

生效时间：立即生效

生效期限：当前终端有效，窗口关闭后无效

生效范围：仅对当前用户有效

配置的环境变量中不要忘了加上原来的配置，即$PATH部分，避免覆盖原来配置

### 方法2：vim ~/.bashrc OR vim ~/.bash\_profile（长久）

通过修改用户目录下的~/.bashrc OR ~/.bash\_profile文件进行配置

vim ~/.bashrc

# 在最后一行加上

export PATH=$PATH:/home/uusama/mysql/bin

**注意事项：**

生效时间：使用相同的用户打开新的终端时生效，或者手动**source ~/.bashrc**生效

生效期限：永久有效

生效范围：仅对当前用户有效

如果有后续的环境变量加载文件覆盖了PATH定义，则可能不生效

如果没有~/.bash\_profile文件，则可以编辑~/.profile文件或者新建一个

# Tmux

## 安装

# Ubuntu

$ sudo apt-get install tmux

## 启动与退出

安装完成后，键入tmux命令，就进入了 Tmux 窗口。

tmux

上面命令会启动 Tmux 窗口，底部有一个状态栏。状态栏的左侧是窗口信息（编号和名称），右侧是系统信息。

按下Ctrl+d或者显式输入exit命令，就可以退出 Tmux 窗口。

exit

## 前缀键

Tmux 窗口有大量的快捷键。所有快捷键都要通过前缀键唤起。默认的前缀键是Ctrl+b，即先按下Ctrl+b，快捷键才会生效。

举例来说，帮助命令的快捷键是Ctrl+b ?。它的用法是，在 Tmux 窗口中，先按下Ctrl+b，再按下?，就会显示帮助信息。

然后，按下 ESC 键或q键，就可以退出帮助。

## 会话管理

### 新建会话

第一个启动的 Tmux 窗口，编号是0，第二个窗口的编号是1，以此类推。这些窗口对应的会话，就是 0 号会话、1 号会话。

使用编号区分会话，不太直观，更好的方法是为会话起名。

tmux new -s <session-name>

上面命令新建一个指定名称的会话。

### 分离会话

在 Tmux 窗口中，按下Ctrl+b d或者输入tmux detach命令，就会将当前会话与窗口分离。

**tmux detach**

上面命令执行后，就会退出当前 Tmux 窗口，但是会话和里面的进程仍然在后台运行。

### 查看会话

tmux ls命令可以查看当前所有的 Tmux 会话。

tmux ls

# OR

tmux list-session

### 进入会话

tmux attach命令用于重新接入某个已存在的会话。

使用会话编号

tmux attach -t 0

使用会话名称

tmux attach -t <session-name>

### 杀死会话

tmux kill-session命令用于杀死某个会话。

# 使用会话编号

tmux kill-session -t 0

# 使用会话名称

tmux kill-session -t <session-name>

### 切换会话

tmux switch命令用于切换会话

# 使用会话编号

tmux switch -t 0

# 使用会话名称

tmux switch -t <session-name>

### 重命名会话

tmux rename-session命令用于重命名会话

tmux rename-session -t 0 <new-name>

上面命令将0号会话重命名

### 会话快捷键

下面是一些会话相关的快捷键。

Ctrl+b d：分离当前会话。

Ctrl+b s：列出所有会话。

Ctrl+b $：重命名当前会话。

## 最简操作流程

综上所述，以下是 Tmux 的最简操作流程。

新建会话tmux new -s my\_session。

在 Tmux 窗口运行所需的程序。

按下快捷键Ctrl+b d将会话分离。

下次使用时，重新连接到会话tmux attach-session -t my\_session。

五、窗格操作

Tmux 可以将窗口分成多个窗格（pane），每个窗格运行不同的命令。以下命令都是在 Tmux 窗口中执行。

5.1 划分窗格

tmux split-window命令用来划分窗格。

# 划分上下两个窗格

$ tmux split-window

# 划分左右两个窗格

$ tmux split-window -h

5.2 移动光标

tmux select-pane命令用来移动光标位置。

# 光标切换到上方窗格

$ tmux select-pane -U

# 光标切换到下方窗格

$ tmux select-pane -D

# 光标切换到左边窗格

$ tmux select-pane -L

# 光标切换到右边窗格

$ tmux select-pane -R

5.3 交换窗格位置

tmux swap-pane命令用来交换窗格位置。

# 当前窗格上移

$ tmux swap-pane -U

# 当前窗格下移

$ tmux swap-pane -D

5.4 窗格快捷键

下面是一些窗格操作的快捷键。

Ctrl+b %：划分左右两个窗格。

Ctrl+b "：划分上下两个窗格。

Ctrl+b <arrow key>：光标切换到其他窗格。<arrow key>是指向要切换到的窗格的方向键，比如切换到下方窗格，就按方向键↓。

Ctrl+b ;：光标切换到上一个窗格。

Ctrl+b o：光标切换到下一个窗格。

Ctrl+b {：当前窗格与上一个窗格交换位置。

Ctrl+b }：当前窗格与下一个窗格交换位置。

Ctrl+b Ctrl+o：所有窗格向前移动一个位置，第一个窗格变成最后一个窗格。

Ctrl+b Alt+o：所有窗格向后移动一个位置，最后一个窗格变成第一个窗格。

Ctrl+b x：关闭当前窗格。

Ctrl+b !：将当前窗格拆分为一个独立窗口。

Ctrl+b z：当前窗格全屏显示，再使用一次会变回原来大小。

Ctrl+b Ctrl+<arrow key>：按箭头方向调整窗格大小。

Ctrl+b q：显示窗格编号。

六、窗口管理

除了将一个窗口划分成多个窗格，Tmux 也允许新建多个窗口。

6.1 新建窗口

tmux new-window命令用来创建新窗口。

$ tmux new-window

# 新建一个指定名称的窗口

$ tmux new-window -n <window-name>

6.2 切换窗口

tmux select-window命令用来切换窗口。

# 切换到指定编号的窗口

$ tmux select-window -t <window-number>

# 切换到指定名称的窗口

$ tmux select-window -t <window-name>

6.3 重命名窗口

tmux rename-window命令用于为当前窗口起名（或重命名）。

$ tmux rename-window <new-name>

6.4 窗口快捷键

下面是一些窗口操作的快捷键。

Ctrl+b c：创建一个新窗口，状态栏会显示多个窗口的信息。

Ctrl+b p：切换到上一个窗口（按照状态栏上的顺序）。

Ctrl+b n：切换到下一个窗口。

Ctrl+b <number>：切换到指定编号的窗口，其中的<number>是状态栏上的窗口编号。

Ctrl+b w：从列表中选择窗口。

Ctrl+b ,：窗口重命名。

七、其他命令

下面是一些其他命令。

# 列出所有快捷键，及其对应的 Tmux 命令

$ tmux list-keys

# 列出所有 Tmux 命令及其参数

$ tmux list-commands

# 列出当前所有 Tmux 会话的信息

$ tmux info

# 重新加载当前的 Tmux 配置

$ tmux source-file ~/.tmux.conf

# Linux

## 文件操作

### 查看文件数量

查看文件数量：ls -l |grep "^-"|wc -l

查看文件夹数量：ls -l |grep "^d"|wc –l

查看当前文件夹中文件的数量：ls –l | wc –l

查看当前目录下有多少个文件及文件夹

ls | wc -w

查看当前目录下有多少个文件

ls | wc -c

查看当前文件夹下有多少个文件，多少个子目录需在终端输入

/bin/ls -l |grep ^-|wc -l

### 查看文件大小

du -h --max-depth=1 file\_name

du -h --max-depth=0

其中，”--max-depth=“指定深入目录的层数（如果不使用该参数，则会循环列出当前目录下所有文件及文件夹使用的空间大小，文件多时会很乱）

(1) --max-depth=0：查看当前目录使用的总空间大小；

(2) --max-depth=1：查看当前目录使用总空间的大小以及当前目录下一级文件及文件夹各自使用的总空间大小；

### 删除文件

remove命令用于删除一个文件或者目录。

语法 ----> **remove -r name**

rm [options] name...

参数：

-i 删除前逐一询问确认

-f 即使原档案属性设为唯读，亦直接删除，无需逐一确认

-r 将目录及以下之档案亦逐一删除

### 下载文件——wget

#### 启动参数

-V, –version 显示wget的版本后退出

-h, –help 打印语法帮助

-b, –background 启动后转入后台执行

-e, –execute=COMMAND 执行`.wgetrc’格式的命令，wgetrc格式参见/etc/wgetrc或~/.wgetrc

#### 记录和输入文件参数

-o, –output-file=FILE 把记录写到FILE文件中

-a, –append-output=FILE 把记录追加到FILE文件中

-d, –debug 打印调试输出

-q, –quiet 安静模式(没有输出)

-v, –verbose 冗长模式(这是缺省设置)

-nv, –non-verbose 关掉冗长模式，但不是安静模式

-i, –input-file=FILE 下载在FILE文件中出现的URLs

-F, –force-html 把输入文件当作HTML格式文件对待

-B, –base=URL 将URL作为在-F -i参数指定的文件中出现的相对链接的前缀

–sslcertfile=FILE 可选客户端证书

–sslcertkey=KEYFILE 可选客户端证书的KEYFILE

–egd-file=FILE 指定EGD socket的文件名

#### 下载参数

–bind-address=ADDRESS 指定本地使用地址(主机名或IP，当本地有多个IP或名字时使用)

**-t, –tries=NUMBER 设定最大尝试链接次数(0 表示无限制).**

-O –output-document=FILE 把文档写到FILE文件中

-nc, –no-clobber 不要覆盖存在的文件或使用.#前缀

**-c, –continue 接着下载没下载完的文件**

**–progress=TYPE 设定进程条标记**

**-N, –timestamping 不要重新下载文件除非比本地文件新**

-S, –server-response 打印服务器的回应

–spider 不下载任何东西

-T, –timeout=SECONDS 设定响应超时的秒数

-w, –wait=SECONDS 两次尝试之间间隔SECONDS秒

–waitretry=SECONDS 在重新链接之间等待1…SECONDS秒

–random-wait 在下载之间等待0…2\*WAIT秒

**-Y, –proxy=on/off 打开或关闭代理**

-Q, –quota=NUMBER 设置下载的容量限制

–limit-rate=RATE 限定下载输率

#### 目录参数

-nd –no-directories 不创建目录

-x, –force-directories 强制创建目录

-nH, –no-host-directories 不创建主机目录

**-P, –directory-prefix=PREFIX 将文件保存到目录 PREFIX/…**

–cut-dirs=NUMBER 忽略 NUMBER层远程目录

#### 实例

实例1：使用wget下载单个文件

wget http://www.minjieren.com/wordpress-3.1-zh\_CN.zip

说明：从网络下载一个文件并保存在当前目录，在下载的过程中会显示进度条，包含（下载完成百分比，已经下载的字节，当前下载速度，剩余下载时间）。

实例2：使用wget -O下载并以不同的文件名保存

命令：

wget -O wordpress.zip http://www.minjieren.com/download.aspx?id=1080

说明：

wget默认会以最后一个符合”/”的后面的字符来命令，对于动态链接的下载通常文件名会不正确。

错误：下面的例子会下载一个文件并以名称download.aspx?id=1080保存

wget http://www.minjieren.com/download?id=1

即使下载的文件是zip格式，它仍然以download.php?id=1080命令。

正确：为了解决这个问题，我们可以使用参数-O来指定一个文件名：

**wget -O wordpress.zip http://www.minjieren.com/download.aspx?id=1080**

实例3：使用wget –limit -rate限速下载

命令：

wget --limit-rate=300k http://www.minjieren.com/wordpress-3.1-zh\_CN.zip

说明：

当你执行wget的时候，它默认会占用全部可能的宽带下载。但是当你准备下载一个大文件，而你还需要下载其它文件时就有必要限速了。

**实例4：使用wget -c断点续传**

命令：

wget -c http://www.minjieren.com/wordpress-3.1-zh\_CN.zip

说明：

使用wget -c重新启动下载中断的文件，对于我们下载大文件时突然由于网络等原因中断非常有帮助，我们可以继续接着下载而不是重新下载一个文件。需要继续中断的下载时可以使用-c参数。

**实例5：使用wget -b后台下载**

命令：

wget -b http://www.minjieren.com/wordpress-3.1-zh\_CN.zip

说明：

对于下载非常大的文件的时候，我们可以使用参数-b进行后台下载。

wget -b http://www.minjieren.com/wordpress-3.1-zh\_CN.zip

Continuing in background, pid 1840.

Output will be written to `wget-log'.

你可以使用以下命令来察看下载进度：

tail -f wget-log

实例6：伪装代理名称下载

命令：

wget --user-agent="Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US) AppleWebKit/534.16 (KHTML, like Gecko) Chrome/10.0.648.204 Safari/534.16" http://www.minjieren.com/wordpress-3.1-zh\_CN.zip

说明：

有些网站能通过根据判断代理名称不是浏览器而拒绝你的下载请求。不过你可以通过–user-agent参数伪装。

**实例8：使用wget –tries增加重试次数**

命令：

wget --tries=40 URL

说明：

如果网络有问题或下载一个大文件也有可能失败。wget默认重试20次连接下载文件。如果需要，你可以使用–tries增加重试次数。

**实例9：使用wget -i下载多个文件**

命令：

wget -i filelist.txt

说明：

首先，保存一份下载链接文件

cat > filelist.txt

url1

url2

url3

url4

接着使用这个文件和参数-i下载

实例12：使用wget -o把下载信息存入日志文件

命令：wget -o download.log URL

说明：

不希望下载信息直接显示在终端而是在一个日志文件，可以使用

## 快捷键

### 移动光标

Ctrl + a将光标移到所在行首

Ctrl + e将光标移到所在行尾

Ctrl + f 光标向前移动一个字符

Ctrl + b 光标向后移动一个字符

Alt + f 光标向前移动一个单词

Alt + b 光标向后移动一个单词

Esc + b移动到【当前单词】的开头

Esc + f 移动到【当前单词】的结尾

### 删除

Ctrl + d 删除光标【所在处】字符

Ctrl + h 删除光标【所在处前】一个字符

Ctrl + w擦除从【光标】位置到词首的全部字母（即擦除光标位置到前面最近的一个空格之间的内容）

Ctrl + u擦除从【当前光标】位置到【行首】的全部内容

Ctrl + k擦除从【当前光标】位置到【行尾】的全部内容

Ctrl + y粘贴【使用了 Ctrl+u、Ctrl+k 、 Ctrl+w 快捷键擦除】的文本

### 历史记录

Ctrl + p 上一条记录

Ctrl + n 下一条记录

Ctrl + r 搜索历史命令

## 后台运行

Ctrl + z 挂起当前任务（&）

Ctrl + c 结束任务

kill n 结束任务

jobs 查看任务，返回任务编号n和进程号

ps -aux | grep # a: 显示所有程序 u: 以用户为主的格式来显示 x: 显示所有程序，不以终端机来区分 grep查找进程很方便，最后一行总是会grep自己

bg n 将编号为n的任务转后台运行

fg n 将编号为n的任务转前台运行

前台执行→Ctrl + z →bg n

nohup shell命令 &（输出信息附加在当前目录的 nohup.out 文件中）

nohup的重定向

只输出错误信息到日志文件

nohup shell命令 2>log &

什么信息也不要

### 命令

nohup shell命令 > /data-tmp/log 2>&1 &

### 查看

tail -10 nohup.out (查看最后10行日志文本）

## 软件安装

### 安装位置

/var/lib/program\_name

### 安装软件

dpkg -l | grep 软件名

### 安装说明

/usr系统级的目录，理解为C:/Windows/

/usr/lib理解为C:/Windows/System32  
/usr/local：用户级的程序目录，理解为C:/Progrem Files/，用户自己编译的软件默认会安装到这个目录下  
/opt：用户级的程序目录，理解为D:/Software，opt有可选的意思，这里可以用于放置第三方大型软件（或游戏），当你不需要时，直接rm -rf掉即可。在硬盘容量不够时，也可将/opt单独挂载到其他磁盘上使用。

源码放哪里？  
/usr/src：系统级的源码目录。

/usr/local/src：用户级的源码目录

## 电脑信息

系统版本

lsb\_release -a

内核版本

cat /proc/version

**df -h 查看数据目录使用情况（data\_tmp/data\_input/data\_user）**

nvidia-smi 查看显卡状态

## 脚本

### 转义符

把\放在一个命令行表示换行

### 创建/编辑

vim/touch

i|a→进入编辑模式

Esc→退出编辑模式

:wq→保存并退出

:q→直接退出

### 运行

bash

### echo 打印命令

echo "字符串" → 在屏幕上打印 字符串。

echo “内容” > OR echo >> file.sh 导入内容到file里

例如：

echo 'hello linux' >> /data/hello.txt 单行内容追加到文件结尾

注：

一个大于号>：覆盖重定向，清除文件里的所有以前数据，增加新数据

两个大于号>>：追加重定向，文件结尾加入内容，不删除已有文件内容

### 查看

cat

### while语句

i=1

while [ $i -le 5 ];

do

let i+=1

done

### Windows .sh → Linux .sh

vim file.sh

:set fileformat=unix

:wq

# AI极链云

/data-user 是OSS个人共享目录，存放数据集压缩包

/data-share 是公共数据集目录

tensorboard 读取/data-output内文件

## 目录

### /data-tmp

**←可存放程序读写的数据（短期存储）→/data-input（长期存储）**

优势：**速度快、空间大**

劣势：1.临时盘，不宜长期存储数据。2.关机后文件有一定概率丢失。3.数据不跟随实例迁移

这个文件夹在SSD硬盘空间不足的时候（剩余空间小于5GB），会保留最近登录此机器的前5个人的/data-tmp 数据，其余人的/data-tmp 数据会被清除。

如果您有重要的数据，请在关机之前把他们复制到/data-input中，并保持账户余额不会欠费，这样您开任何一台机器都可以访问到您的这些数据。

### /data-output

**←tensorboard需要的log文件**

优势：仅支持可视化的文件夹

劣势：1.可视化文件以外的数据文件不要放在此盘。 2.受**30G**限制

tensorboard监视的文件夹，您把tensorboard要求的log输出到这个文件夹就可以使用tensorboard了

### /data-input

**←经常用的数据，个人文档，训练数据**

优势：**空间大**、可长期存放不会丢失、安全性高

劣势：速度一般

推荐使用

经常用的数据可以放到这里，比如**个人的文档，模型，训练数据等，不支持web访问。**

### /data-user

**←数据管理上传的数据**

优势：**空间很大**，安全性高，可靠性极高

劣势：速度一般

推荐使用

存放从“数据管理”页面上传的所有数据的文件夹

可通过控制台或 **vcloud-oss-cli** 命令工具上传/下载数据。**只用于存储数据集压缩包，不能解压到这里，速度极慢**

### /data-share

**→官方提供的模型和数据**

优势：**空间很大**，**可下载数据←vcloud-oss-cli 命令工具下载数据集**

劣势：不能上传数据

官方提供的一些模型和样本数据（存放提供的公共数据集或工具）

## 数据上传与下载

/data-user 用来中转数据，该目录容量巨大且数据不会丢失，不要直接在它里面压缩或解压，速度会非常慢→

用于解压和压缩两个目录（（从 /data-user 拷贝到 /data-tmp 或 /data-input 文件夹可以使用 rsync，能断点续传与查看进度）

（首选）/data-tmp 用来临时存放训练需要的输入数据（迁移实例会丢失）

/data-input 用来长期存放用户的目录，文档/模型等所有经常访问的数据 （迁移实例会保留）

**利用 Xftp 工具可以通过 SSH 来进行文件下载和上传**

**Example**

通过 rsync 将压缩包传输到 SSD 文件夹，然后再在 SSD 文件里进行解压

apt-get update

apt-get install rsync -y

rsync -ah --progress /data-user/test.gz /data-tmp/ && gzip -d /data-tmp/test.gz

### 上传命令

rsync -ah --progress /data-user/file\_name.tar /data-tmp/ && cd /data-tmp && tar -xf /data-tmp/file\_name.tar

tar -xf /data-tmp/file\_name.tar -C /data-input

rsync -ah --progress /data-user/test.zip /data-tmp/ && unzip -oq /data-tmp/test.zip -d /data-input

如果不想下次使用时再次解压，可通过命令：cp -rp /data-tmp/test /data-input/将解压好的数据移动到/data-input下（/data-input 比/data-user 访问速度快，但是比/data-tmp 访问速度慢）

将训练结果压缩后存入/data-user

# 后者要用相对路径

tar -zcf /data-user/backup.tar.gz /data-input

【tar调用gzip←gzip是GNU组织开发的一个压缩程序，.gz结尾的文件就是gzip压缩的结果。与gzip相对的解压程序是gunzip。tar中使用-z这个参数来调用gzip。例：

# tar -czf all.tar.gz \*.jpg这条命令是将所有.jpg的文件打成一个tar包，并且将其用gzip压缩，生成一个gzip压缩过的包，包名为all.tar.gz

# tar -xzf all.tar.gz这条命令是将上面产生的包解开。】

**Vcloud文件上传和下载-常用命令**

登陆注销 login/logout：使用前需要账号进行登陆。

# 使用极链云官网的账号登陆，用户名使用注册的手机号或邮箱

vc login --user=13657257512 --password=101679

# 如需删除认证则要注销账号

vci logout

浏览文件 ls：列出目录下的文件和文件夹

# 浏览个人目录下所有的文件和文件夹

# -d 表示非递归显示，不显示子目录下的所有目录

vc ls oss:// -d

# 列出全部公共数据集

**# --share 表示操作目录为公共数据集目录**

vc ls oss:// --share -d

# 查看某个公共数据集的全部文件

vci ls oss://dir\_name/ --share -s

# 创建文件夹 mkdir：创建文件夹

# 在个人目录的空间创建一个名为 folder 的文件夹

vc mkdir oss://folder/

上传文件 cp：上传文件或文件夹到 OSS 个人目录，仅支持压缩包格式

# 上传单个文件到 OSS 个人目录根下

# -u 代表文件不存在或修改时间不同时才会上传

vc cp a.zip oss:// -u

# 上传 dir 目录下文件b.zip到个人 OSS 目录的 dataset 文件夹下

**# -r 代表目录递归**

vc cp dir/b.zip oss://dataset/ -r -u

下载公共数据集 cp：从公共数据集中下载文件

# 下载公共数据集指定目录的所有文件到当前目录

vc cp oss://dir\_name/val/images . --share -r

删除文件 rm：删除文件或文件夹

# 删除文件

vc rm oss://dir\_name/file\_name

# 删除文件夹

vc rm oss://dir\_name/ -r

**实例镜像**

实例内置软件：

CUDA

cuDNN 7（GPU加速库）

Anaconda

Python

Jupyter Notebook

TensorBoard

CUDA 的相关环境变量：

CUDA\_HOME=/usr/local/cudaLD\_LIBRARY\_PATH=/usr/local/nvidia/lib:/usr/local/nvidia/lib64

Python、框架的环境均使用 conda 安装

python 的命令组已经指向创建时选择的 Python 版本

可以使用如命令

**ll /opt/conda/bin | grep "python"**

查看 conda 的 python 命令

例如：代码保存为 example.py，之后使用 python3.6 ./example.py 来执行

PIP 均配好国内镜像源，可以快速安装软件包

**PyCharm 远程连接**

IP: 39.104.69.33

Host: net.videojj.com

Port: 38687

Username: root

Interpreter: /opt/conda/bin/python

←Sync folders: /data-tmp/pycharm\_project\_\*

主机

39.104.69.33

net.videojj.com

53445

9X4R37ZYy5qKAw4yW8bYTaQZEYpwPx7s