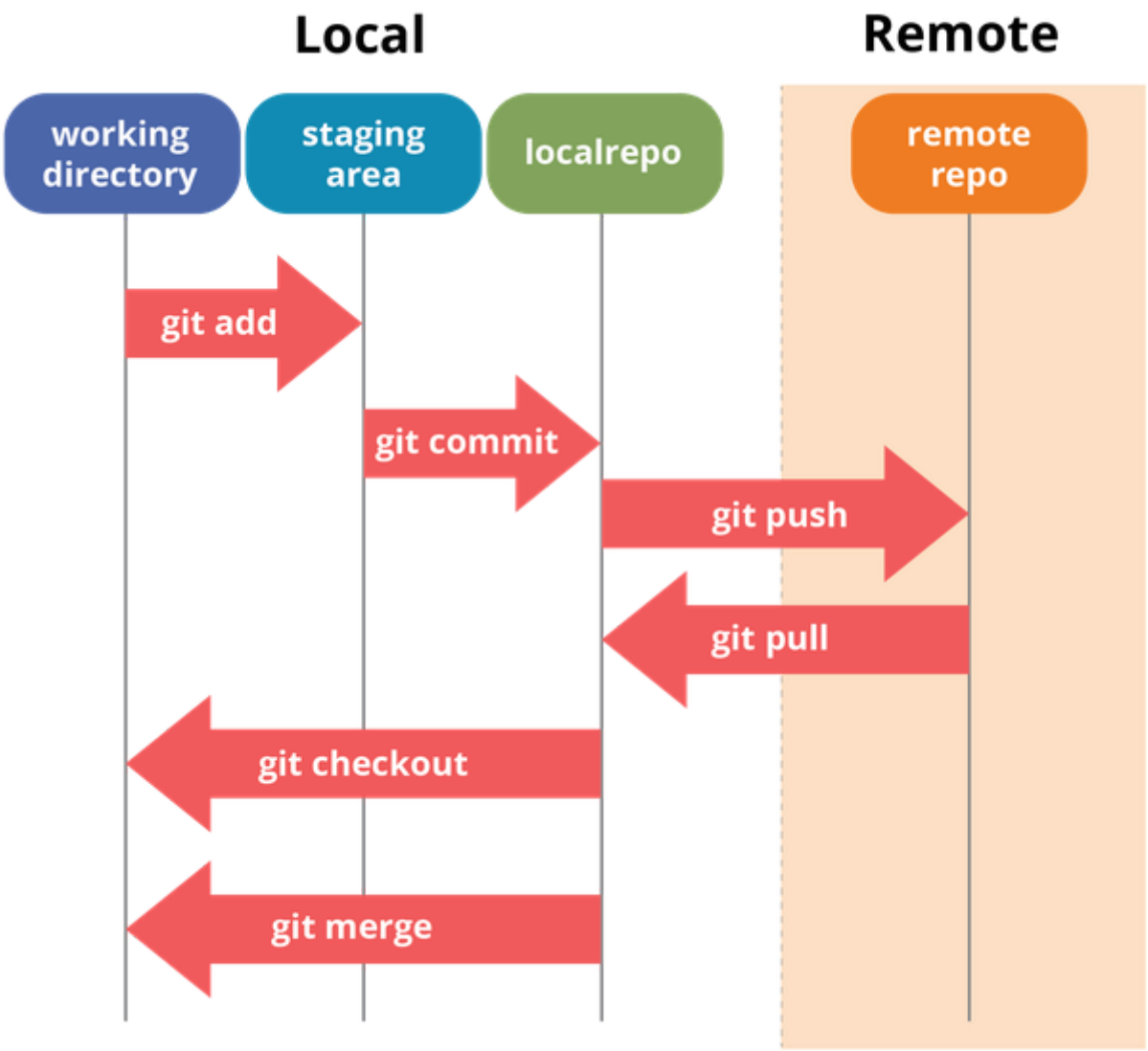

git
DVC


Local
Remote
working directory
staging area
localrepo
remote repo
git add
git commit
git push
git pull
git checkout
git merge

零、安裝

python

git

Install Conda & pip

```
$ conda install -c conda-forge dvc
```

```
$ pip install dvc
```

一、初始化

```
$ git init
```

- DVC初始化
- 初始化後，.dvc/ 將使用 config 和 .gitignore文件 和 cache目錄 創建一個新目錄。

```
$ dvc init
```

```
git commit -m "初始化 DVC"
```

- 將 .dvc/config 和 .dvc/.gitignore 文件（DVC內部）置於Git控制之下

二、配置remote(遠端)

DVC 目前支援以下七種remote類型:

1. local - 本地目錄
2. s3 - Amazon S3
3. gs - Google 雲端

4. azure - Azure Blob
5. ssh
6. hdfs - Hadoop分散式文件系統
7. http - HTTP & HTTPS協議

```
$ dvc remote add -d myremote "D:\storage"
```

查看remote 配置

```
$ dvc remote list
```

- 設定remote -d → default

```
$ git commit .dvc/config -m "Init configure local remote"
```

- 將DVC remote配置透過git託管

三、託管DATA

```
$ dvc add data
```

- 將data資料夾內的資料透過dvc託管

```
$ python train.py
```

- 訓練後產生model.h5

```
$ dvc add model.h5
```

- 託管model.h5

```
$ dvc push
```

- 將data model.h5 push 到 storage

```
$ git add data.dvc model.h5.dvc metrics.csv .gitignore train.py
$ git commit -m "First model, trained with 1000 images"
$ git tag -a "v1.0" -m "model v1.0, 1000 images"
```

四、第二版本兩千張

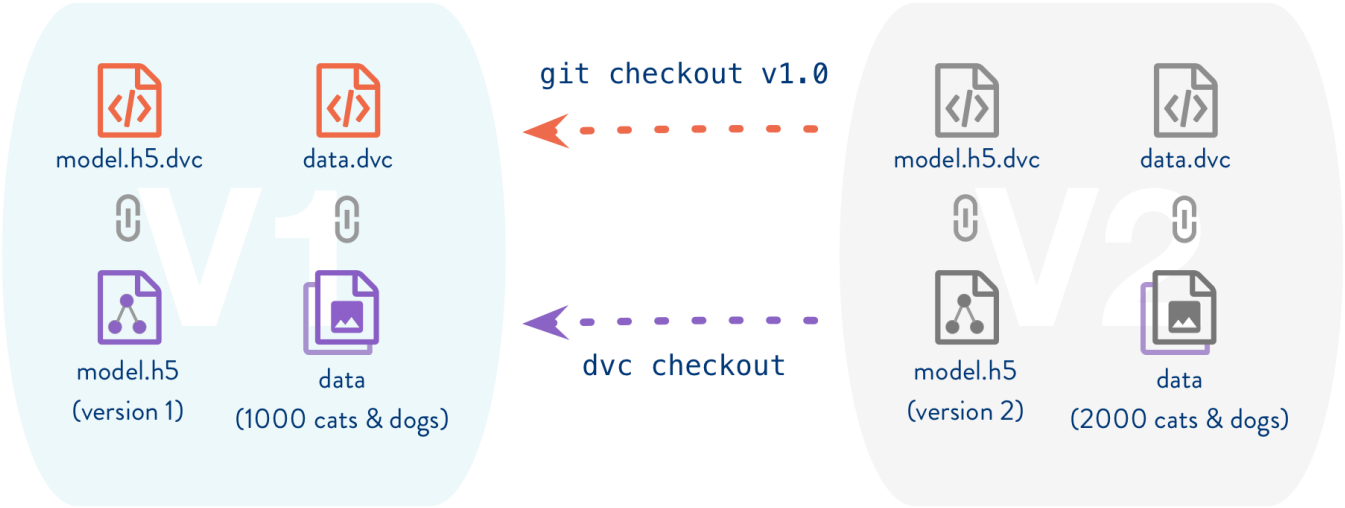
增加training data

```
$ dvc add data
$ python train.py
$ dvc add model.h5
```

```
$ git add data.dvc model.h5.dvc metrics.csv train.py
$ git commit -m "Second model, trained with 2000 images"
$ git tag -a "v2.0" -m "model v2.0, 2000 images"
```

```
$ git tag
$ git log
$ git dog
```

- 我們已經追蹤了 dataset(DVC), model(DVC), metrics(git)




```
$ git checkout v1.0
$ dvc checkout -f
$ git dog
```

- -f 刪除檔案時不提示

五、切換回 v1.0 1000 imgs

```
$ git checkout v1.0
$ dvc checkout -f
```

```
$ git dog
```

保留現在的程式，只有dataset回上一版本

```
$ git checkout v1.0 data.dvc
$ dvc checkout data.dvc
```

參考資料

> https://dvc.org/doc
> https://git-scm.com/