Data Version Control (DVC) in Machine Learning Projects

Nhúm MLOps

Ngịy 18 thòng 10 nm 2025

Ni dung v Data Version Control (DVC) c chia thịnh 5 phn chờnh:

- **Phn 1: Tng quan v AI, MLOps vị Data Versioning**
- **Phn 2: Thòch thc trong Qun lỳ D liu vị Code**
- **Phn 3: Gii thiu v Data Version Control (DVC)**
- **Phn 4: Case Study: Trin khai DVC cho Dataset MNIST**
- **Phn 5: T ng húa Pipelines vị Còc khòi nim Versioning**

Phn 1: Tng quan v AI, MLOps vị Phiển bn húa D liu (Data Versioning)

1.1 Data Versioning lị gơ ?

Data Versioning (phiển bn húa d liu) lị mt h thng dứng qun lỳ vị theo dủi nhng thay i ca d liu vị mũ hơnh trong sut vùng i d òn Machine Learning (ML).
T Nú khũng ch lu li còc phiển bn khòc nhau ca b d liu (dataset), mị cùn qun lỳ c còc file cu hơnh (Config), tham s (parameters) vị kt qu òinh giò (Eval result). V c bn, nú hot ng nh "Git cho d liu", giữp kt ni mt phiển bn code c th vi mt phiển bn d liu chờnh xòc mị code ú õ s dng hun luyn vị to ra mũ hơnh.

1.2 Vơ sao cn Data Versioning?

1.2.1 Gúc nhơn nghip v (Business Perspective)

T gúc nhơn nghip v, AI c xem lị mt chc nng (function) gii quyt còc bịi toòn c th, vờ d nh phón tờch cm xũc (Sentiment Analysis) hay xóy dng h thng gi ỳ (Recommendation) trong mt h thng phn mm ln nh Thng mi in t.

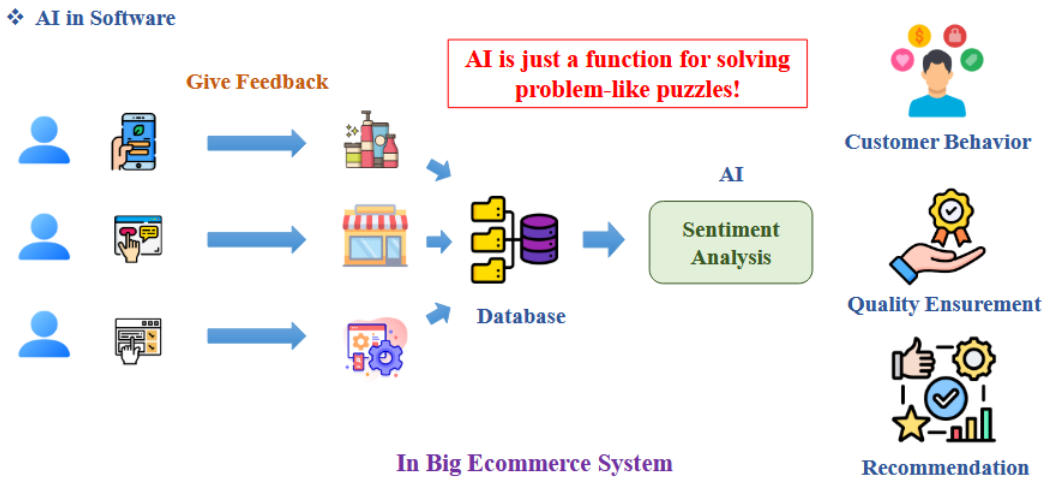


Figure 1: Vai trù ca AI trong mt h thng Thng mi in t

Toịn b quy trơnh ML bt u t **Yổu cu Nghip v (Business Require)** vị i qua mt vùng i (life cycle) hoịn chnh. m bo cht lng (Quality Ensurement) vị giòm sòt (Monitoring) mũ hơnh AI mt còch hiu qu, doanh nghip phi cú kh nng tr li còc cóu hi:

- Mũ hơnh ang chy trổn production c hun luyn t d liu nịo?
- Khi mũ hơnh d oòn sai, lịm th nịo tòi lp (reproduce) li ú?
- Nu d liu mi lịm gim hiu sut, lịm sao quay li phiổn bn mũ hơnh n nh trc ú?

Data versioning cung cp kh nng **truy xut ngun gc (lineage)** nịy, cho phỗp theo dủi chờnh xòc code nịo, d liu nịo, tham s nịo õ to ra mũ hơnh nịo.

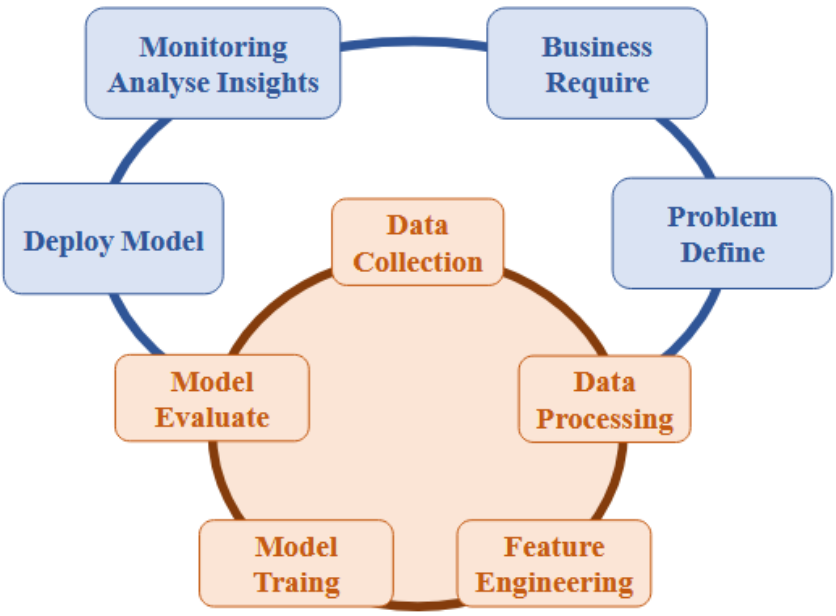


Figure 2: Vùng i Machine Learning (ML Life Cycle)

1.2.2 Gúc nhơn MLOps

MLOps tp trung vịo vic t ng húa vị tinh gn quy trơnh (pipeline) chuyn giao mũ hơnh t mũi trng nghiổn cu (Research Environment) sang mũi trng trin khai (AI Service). Quy trơnh niy bao gm còc bc lp i lp li nh X lỳ d liu (Data Handling), Hun luyn mũ hơnh (Model Training), vị ònh giò mũ hơnh (Model Evaluation).

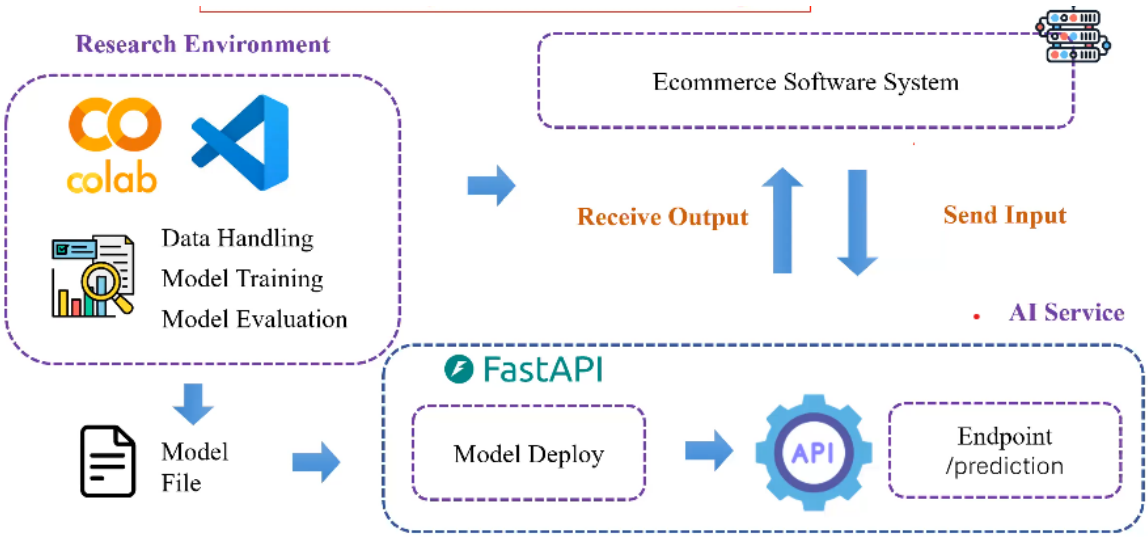


Figure 3: Quy trơnh MLOps c bn

Data versioning lị thịnh phn quan trng tr li cóu hi " óu vị ti sao chững ta cn phiổn bn húa d liu ?". Nú giữp t ng húa vic qun lỳ còc "u ra" (artifacts) ca mi bc (nh d liu õ x lỳ, file mũ hơnh, file ch s) vị m bo tờnh nht quòn gia còc mũi trng, c bit lị khi lịm vic nhúm.

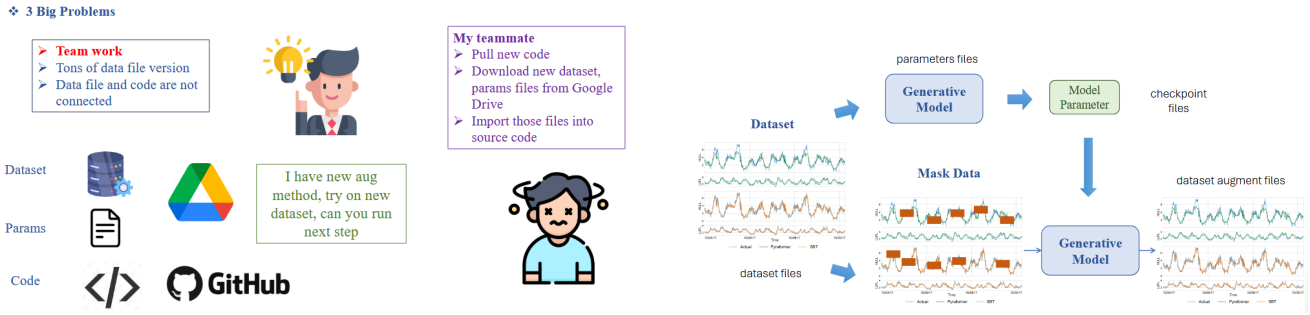
Phn 2: Thòch thc trong Qun lỳ D liu vị Code

Khi thc nghim trổn 1 tp d liu ln, còc vn nh còc thịnh viổn trong nhúm "mt kt ni / ngng hiu" vi d liu, code, tham s vị phiổn bn mũ hơnh lị iu d thy trong 1 nhúm. Nht lị i vi d liu dng TimeSeries va ln mị va phi cp nht liổn tc. nhơn rủ hn v vn niy, mơnh s i qua 2 vờ d khi CÚ vị KHÛNG CÚ data versioning pipeline.

2.1 Vờ d thc t: Th nghim mũ hơnh Time Series

2.1.1 Khi KHÛNG cú Data Versioning (Còch lịm th cũng)

Gi s nhúm ca bn ang phòt trin mt mũ hơnh d bòo chui thi gian (time-series) phc tp. Quy trơnh niy to ra vũ s file: ‘dataset files‘ (d liu gc), ‘parameters files‘ (tham s), ‘checkpoint files‘ (im kim tra mũ hơnh), vị ‘dataset augment files‘ (d liu tng cng).



(a) Vn khi lịm vic nhúm th cũng (b) Pipeline sinh ra nhiu phiển bn file

Figure 4: S hn lon khi qun lỳ th cũng còc file d liu vị th nghim

Khi mt thịnh viển trong nhúm ("My teammate") mun chy th nghim mi, quy trơnh ca h rt th cũng vị lịm chm quò trơnh phòt trin:

1. **Pull new code**: Ti code mi nht t GitHub.
2. **Download new dataset/params**: Vơ Github khũng cho lu file ln hn 100mb, húm truy cp Google Drive, tơm ững file d liu, file tham s mi nht vị ti v th cũng.
3. **Import files**: Sao chỗp còc file nịy vịo ững th mc trong source code.

iu nịy dn n 3 vn ln: (1) Cú quò nhiu phiển bn file d liu, tham s ca còc mũ hơnh khòc nhau. (2) Code vị d liu ca mi ngi khũng c ng b, vị (3) Góy cn tr khi lịm vic nhúm. Mi ngi s bi ri t hi "file `cnn_ettm1_mix1_mask025.pkl` nịy c hun luyn t d liu, mũ hơnh, thi im nịo ?".

$\longrightarrow$ Cn 1 quy trơnh (pipeline) thng nht kim soòt còc phiển bn mũ hơnh, d liu vị code c team cú th hot ng 1 còch ng b mị khũng phi thũng bòo, gi Google Meet trao i nhiu ch xòc nhn nhng vn nh nht cú tờnh lp li c nổu trển.

2.1.2 Khi CÚ Data Versioning (vi DVC)

Khi s dng mt cũng c nh DVC, quy trơnh trển c n gin húa trit .

1. **git pull**: Thịnh viển nhúm ti code mi. Ln nịy, code mi bao gm còc file `.dvc` (siểu d liu) nh nh tr n d liu.
2. **dvc pull**: H ch cn chy lnh nịy. DVC s c còc file `.dvc`, t ng tơm vị ti v ững phiển bn d liu, tham s, vị mũ hơnh tng ng vi phiển bn code ú t b lu tr (vờ d: S3, Google Drive, hoc SSH).

Bng còch nịy, DVC gii quyt vn "disconnected" bng còch liển kt code vị d liu mt còch cht ch thũng qua còc `commit`. Mi th nghim u c ghi li, nht quòn vị cú th **tòi lp (reproducible)** mt còch chờnh xòc bt c lữc nịo.

Phn 3: Gii thiu v Data Version Control (DVC)

DVC (Data Version Control) lị mt cũng c mõ ngun m c thit k qun lỳ d liu vị còc d òn Machine Learning, hot ng song song b tr cho Git [Ite25].

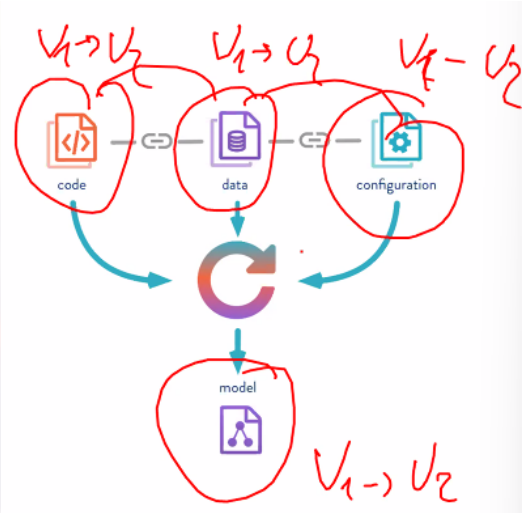


Figure 5: DvC ng b b húa code, d liu, cu hơnh vị mũ hơnh cho mi phiển bn, mi khi cú s thay i sau Hun Luyn

3.1 Workflow Qun lỳ Phiển bn D liu

Mt workflow phiển bn húa d liu (Data Versioning) lị mt vùng lp qun lỳ liển tc. Quy trơnh nịy bt u t **Model Training** (Hun luyn Mũ hơnh), s dng mt b `Data Config` (cu hơnh d liu) vị `Model (params)` (tham s mũ hơnh) c th. Toịn b tịi sn nịy (d liu, tham s, kt qu ònh giò) c h thng **Data & Model Management** theo dủi vị lu tr.

Khi cú d liu mi (`New Data`) hoc thay i cu hơnh (`Diff Config`), quy trơnh **Continuos Training** (Hun luyn Liển tc) s c kòch hot to ra mũ hơnh mi (`New Model`) vị kt qu ònh giò mi (`New Eval result`). Phiển bn mi nịy li tip tc c h thng qun lỳ, hoc sn sịng cho vic **Model Deployment** (Trin khai).

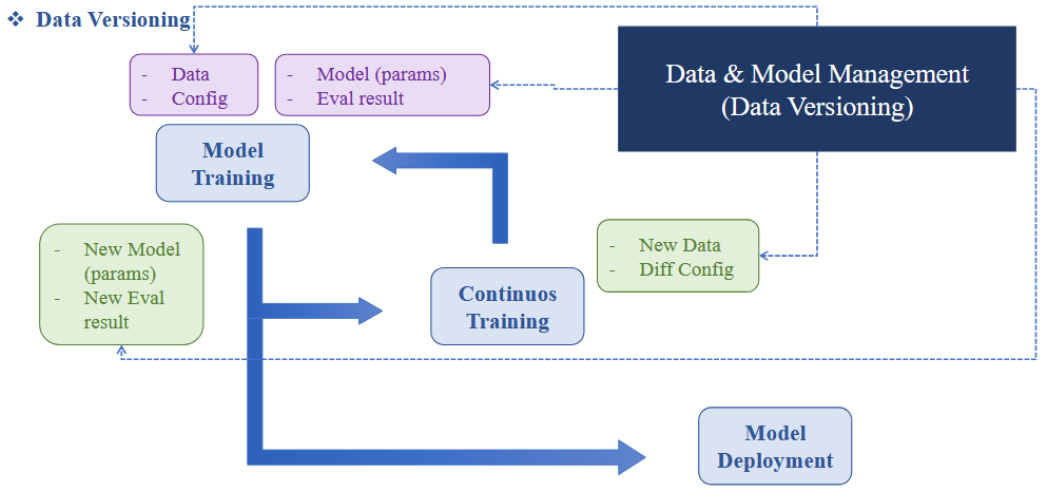


Figure 6: Workflow qun lỳ d liu vị mũ hơnh trong MLOps

3.2 So sònh DVC vị Gii thiu chi tit

Trển th trng cú nhiu cũng c phiển bn húa d liu nh Delta Lake, Pachyderm, vị DVC. Tuy nhiển, DVC tr nển rt ph bin vơ còc lỳ do chònh:

- **Mõ ngun m**: S dng giy phỗp Apache 2.0.
- **c lp vi nh dng**: **Data Format Agnostic**, ngha lị nú cú th qun lỳ bt k loi file nịo (model, .csv, .parquet, hơnh nh...).
- **c lp vi lu tr**: **Cloud/Storage Agnostic**, h tr hu ht còc nn tng lu tr ph bin nh S3, GCP, Azure, SSH.
- **D s dng**: **Simple to Use**, vơ cú còc lnh tng t Git.

	Open Source	Data Format Agnostic	Cloud/Storage Agnostic* (Supports most common cloud and storage types)	Simple to Use	Easy Support for BIG Data
DVC	✅ (Apache 2.0)	✅	✅	✅	❌
DELTA LAKE	✅ (Apache 2.0)	❌	✅	❌	✅
Git Large File Storage	✅ (MIT)	✅	❌	✅	❌
Pachyderm	✅⚠ (Non-standard license)	✅	✅	❌	✅
DOLT	✅ (Apache 2.0)	❌	❌	❌	❌
lakeFS	✅ (Apache 2.0)	✅	❌	❌	✅

DAGsHub

Figure 7: So sònh còc cũng c Data Version Control ph bin

V c bn, DVC lị mt nn tng khoa hc d liu cho phỗp bn liển kt `code`, `data` (d liu), vị `configuration` (cu hơnh) to ra còc `model` (mũ hơnh) cú th tòi lp (reproducible).

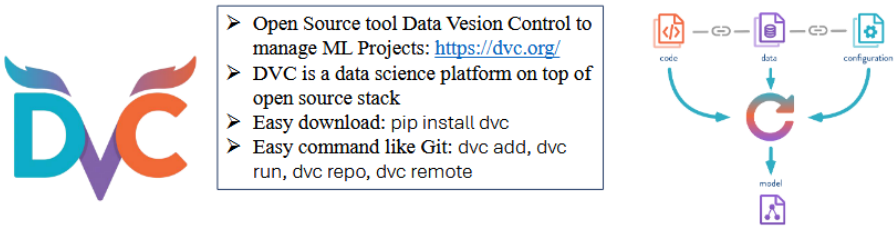


Figure 8: DVC liển kt Code, Data, vị Configuration qun lỳ Model

3.2.1 DVC khòc Git nh th nịo?

im khòc bit ct lủi lị: **Git qun lỳ Code** (mõ ngun), trong khi **DVC qun lỳ Data** (d liu).

Git khũng c thit k x lỳ còc file ln (vờ d: mũ hơnh 500MB). Khi bn dứng DVC, quy trơnh lịm vic (Local/Remote) s c tòch bit rủ rịng:

- **Git (Code)**: Bn dứng `git push/pull` ng b code (vờ d: `train.py`) vị còc file `.dvc` siểu d liu (ch nng vịi KB) lển mòy ch Git (nh GitHub, GitLab).

- **DVC (Data)**: Bn dứng `dvc push/pull` ng b còc file d liu ln thc t (vờ d: `model.pkl` 500MB) lổn mt mòy ch lu tr (Storage) riổng bit (nh S3, Azure, Google Cloud, SSH).

file `model.pkl.dvc` (1KB) mị Git theo dủi ch lị mt "con tr" (pointer) tr n file `model.pkl` (500MB) thc s c DVC qun lỳ. iu nịy gi cho kho Git ca bn luũn nh gn.

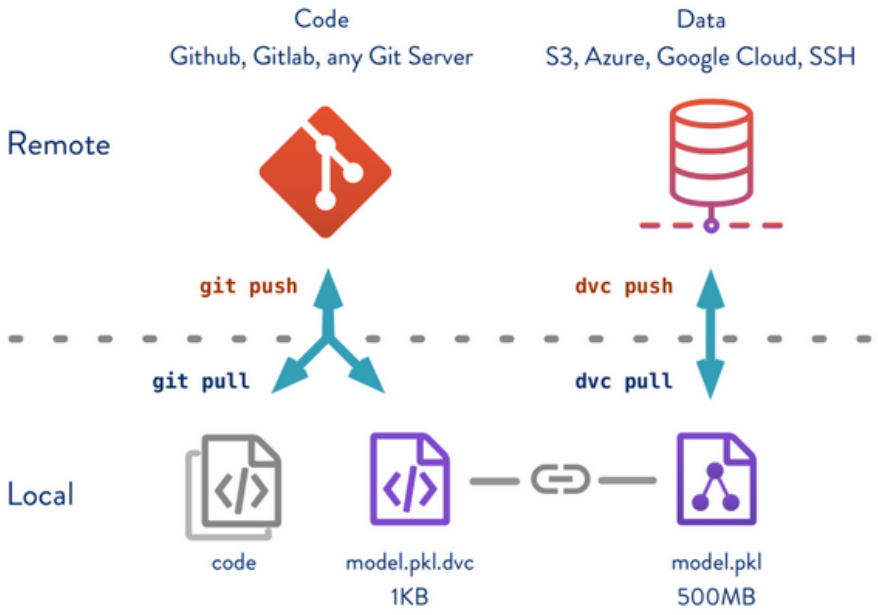


Figure 9: DVC vị Git hot ng song song: Git qun lỳ Code, DVC qun lỳ Data

DvC cú tng ng 1-1 so vi Git v cóu lnh (vờ d: `dvc add`, `dvc push`, `dvc pull`), giữp bt kơ ai quen thuc vi version control ca git hiu ngay khi s dng.

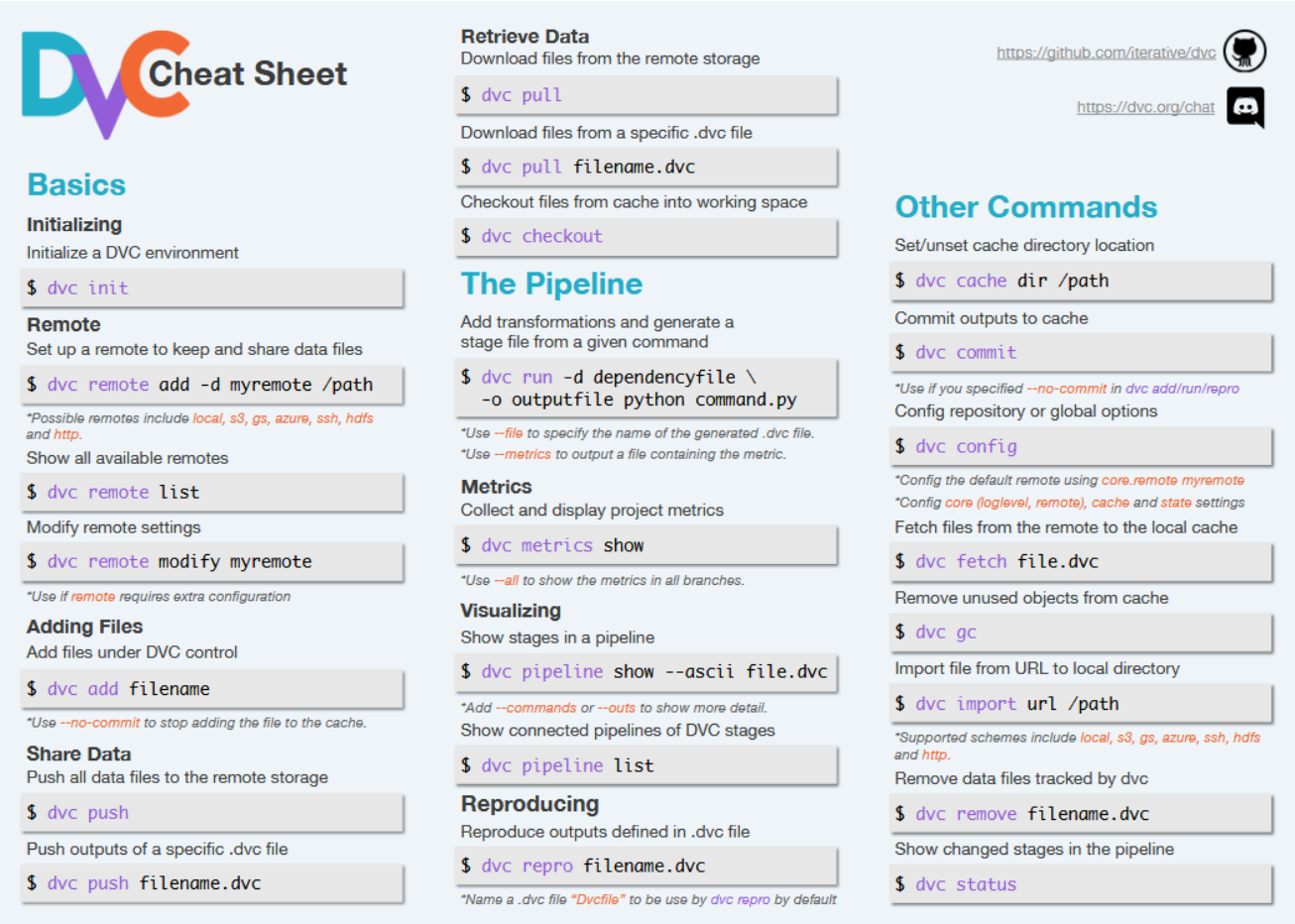


Figure 10: DVC Cheat Sheet cho thy s tng ng vi còc lnh Git

3.3 Quy trơnh DVC Pipeline hoịn chnh

Mt DVC pipeline hoịn chnh cho phỗp bn t ng húa toịn b quy trơnh ML vị m bo tờnh tòi lp (reproducibility). B nõo hay cùn gi lị cu hơnh (config) ca pipeline lị file `dvc.yaml`.

3.3.1 nh ngha Pipeline (dvc.yaml)

file `dvc.yaml` lị ni bn nh ngha tt c còc **stages** (giai on) ca quy trơnh ML. Mi stage ging nh mt bc trong "cũng thc" ca bn. Mt stage thng bao gm:

- `cmd`: Lnh thc thi (vờ d: `python train.py`).
- `deps`: Còc file ph thuc (dependencies) lị u vịo ca stage, vờ d nh code (‘train.py‘) hoc d liu thũ (‘data/raw.csv‘).
- `params`: Còc tham s (parameters), thng c nh ngha trong file `params.yaml` (vờ d: learning rate, s epochs).
- `outs`: Còc file u ra (outputs) mị stage nịy to ra, vờ d nh mũ hơnh õ hun luyn (`model.pkl`) hoc d liu õ x lỳ.

3.3.2 Thc thi Pipeline (dvc repro)

Khi bn chy lnh `dvc repro`, DVC s thc hin mt vic rt thũng minh: Nú kim tra file `dvc.yaml` vị so sònh **hash** (mt chui nh danh duy nht) ca còc file `deps` vị `params` hin ti vi thũng tin c lu trong file `dvc.lock`.

- file `dvc.lock` lu li "du vón tay" (hash) ca còc file `deps`, `params`, vị `outs` t ln chy thịnh cũng trc ú.
- Nu hash ca bt k file `deps` (vờ d: bn sa code `train.py`) hoc bt k `params` nịo thay i, DVC s nhn ra stage ú lị "li thi" (outdated) vị **ch chy li stage ú** cứng còc stage ph thuc vịo nú.
- Nu khũng cú gơ thay i, DVC s khũng chy gơ c, giữp tit kim thi gian tờnh toòn.

óy chờnh lị còch DVC m bo rng mũ hơnh ca bn luũn nht quòn vi code vị d liu õ to ra nú.

3.4 Còc khòi nim DVC ct lủi cn nh

óy lị nhng ghi chữ quan trng vị còc khòi nim cn nhc li (t còc cóu hi quiz bui Th 5 Module 5 Week 1) trc khi i vịo thc hịnh.

3.4.1 Quy trơnh lịm vic chun: DVC trc, Git sau

óy lị quy tc quan trng nht. Khi bn cú mt file d liu ln (vờ d: `data.zip`):

1. **‘dvc add data.zip‘**: Lnh nịy bo DVC bt u theo dủi `data.zip`. DVC sao chỗp file nịy vịo kho lu tr cc b (`.dvc/cache`) vị to ra mt file siổu d liu (pointer) nh tổn lị `data.zip.dvc`.
2. **‘git add data.zip.dvc‘**: Bn dứng Git theo dủi file pointer `.dvc` (ch vịi KB), **KHŨNG BAO GI** ‘git add‘ file `data.zip` gc.
3. **‘git commit -m "Track new data"‘**: Lu li trng thòi ca file pointer.

Nu bn ‘git add‘ file ln trc, kho Git ca bn s b phơnh to vị DVC s khũng th qun lỳ file ú.

3.4.2 git push vs. dvc push

óy lị im khòc bit ct lủi (nh trong hơnh `git_v_dvc.png`):

- **‘git push‘**: Ch y code (‘.py‘) vị còc file siổu d liu/con tr (‘.dvc‘, ‘dvc.yaml‘) lổn mòy ch Git (nh GitHub).
- **‘dvc push‘**: y còc file d liu ln thc t (mị DVC ang theo dủi) t cache cc b (`.dvc/cache`) lổn kho lu tr t xa (Remote Storage) nh AWS S3, GCS, hoc SSH.

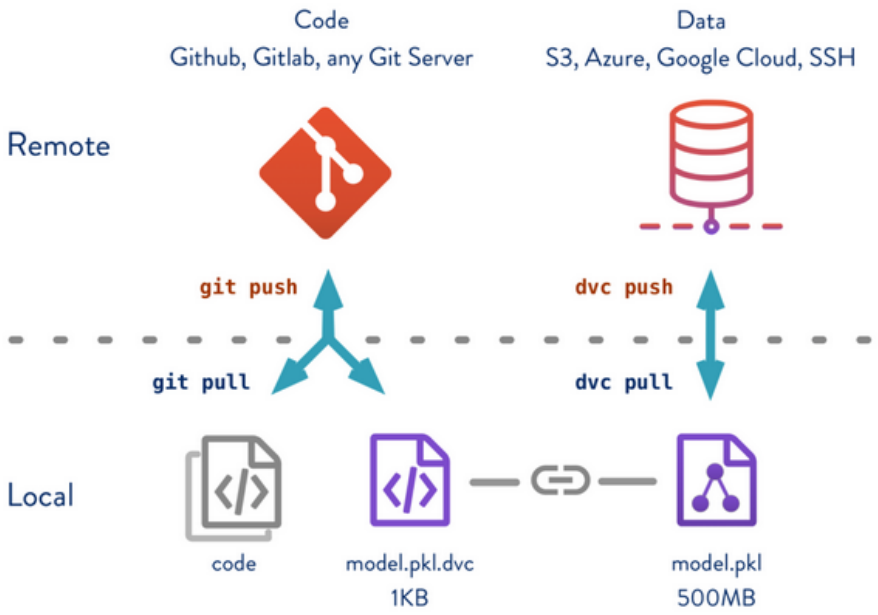


Figure 11: Git qun lý Code/Metadata, DVC qun lý Data thc t

3.4.3 dvc pull vs. dvc checkout

Khi lịm vic nhóm, bn s dứng hai lnh nịy:

- **‘dvc pull‘**: Ti d liu t Remote Storage (vờ d: S3) v kho lu tr cc b (Local Cache) ca bn (th mc `.dvc/cache`).
- **‘dvc checkout‘**: ng b húa d liu t Local Cache ra th mc lịm vic (workspace) ca bn. Lnh nịy c còc file `.dvc` trong workspace vị to liển kt (symlink) n còc file tng ng trong cache. Bn thng chy lnh nịy sau khi `git checkout` mt branch mi.

3.4.4 iu kin tiển quyt cho DVC Pipeline

Trc khi cú th chy `dvc repro`, bn cn m bo:

1. D òn õ c khi to lị mt kho Git (`git init`).
2. õ cịi t còc th vin cn thit (vờ d: `pip install dvc dvc-s3`).
3. õ nh ngha còc stage trong file `dvc.yaml`.
4. (Tứy chn) õ kt ni vi kho lu tr AWS t xa bng lnh `dvc remote add`, vờ d: `dvc remote add -d my-s3-storage s3://my-bucket/dvc-storage`. Hng dn setup AWS ca TA trển Notion

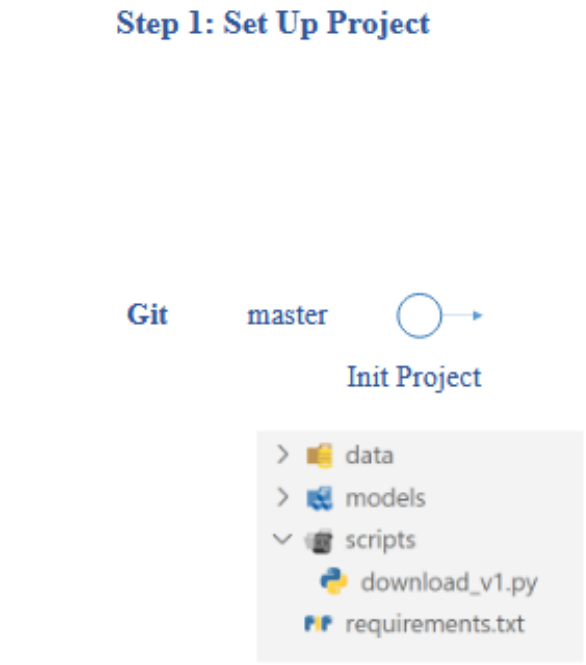
Phn 4: Case Study: Trin khai DVC cho Dataset MNIST

Chững ta s thc hin mt case study tng bc thy DVC vị Git hot ng song song nh th nịo qun lý còc phiển bn th nghim (d liu vị mũ hơnh).

4.1 Bc 1: Thit lp D òn vị Git

u tiổn, chững ta to th mc d òn, cu trữc th mc con, mũi trng `conda`, vị quan trng nht lị khi to kho Git.

```
$ # 1. Create project directory and basic structure
$ mkdir dvc-mnist-demo
$ cd dvc-mnist-demo
$ mkdir data/raw models scripts

$ # 2. Create conda environment and install libraries
$ conda create -n dvc_mnist python=3.11
$ conda activate dvc_mnist
$ pip install -r requirements.txt

$ # 3. Initialize Git and make the first commit
$ git init
$ git add .
$ git commit -m "Init␣project"
```



à à à à à à

Figure 12: Trng thòi Git vị cu trữc file ban u

4.2 Bc 2: Ti Dataset V1 vị Theo dủi bng DVC

Tip theo, chững ta ti phiổn bn u tiổn ca dataset (Full MNIST) vị dứng DVC bt u theo dủi nú.

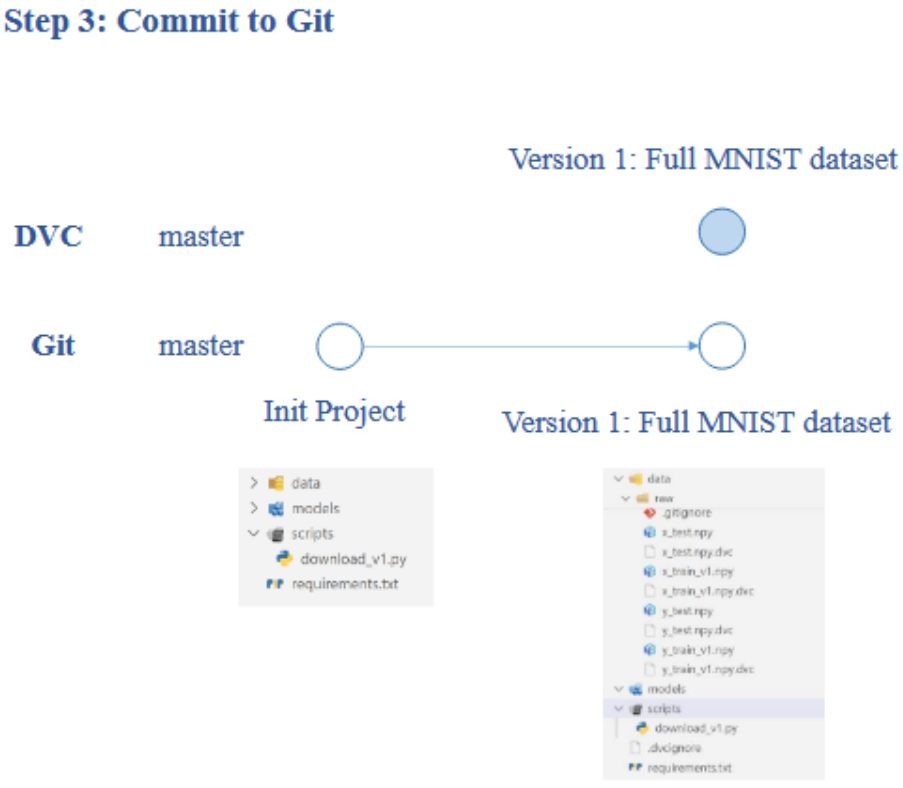
```
$ # 1. Download data (60000 samples)
$ python scripts/download\_v1.py

$ # 2. Initialize DVC in the project
$ dvc init

$ # 3. Ask DVC to track the large data files
$ dvc add data/raw/x_train\_v1.npy data/raw/y_train\_v1.npy data/raw/x_test.npy data/
    raw/y_test.npy
```

Lnh `dvc add` s to ra còc tp `.dvc` (con tr) siổu d liu vị t ng thổm còc tp `.npy` ln vịo `.gitignore`.

4.3 Bc 3: Commit Phiổn bn D liu V1

Gi chững ta commit còc tp `.dvc` (con tr) vịo Git lu li "phiổn bn" d liu niy.

```
$ # Add .dvc files (pointers) and the .gitignore file updated by DVC
$ git add data/raw/.gitignore data/raw/x_train\_v1.npy.dvc data/raw/y_train\_v1.npy.
    dvc data/raw/x_test.npy.dvc data/raw/y_test.npy.dvc

$ # Commit to finalize Version V1
$ git commit -m "Version 1: Full MNIST dataset"
```



à à à à à à

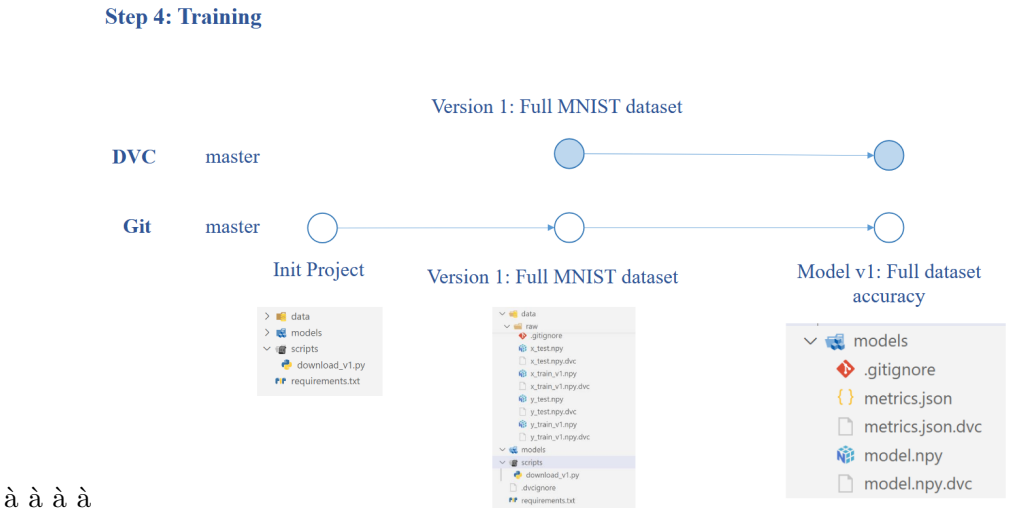
Figure 13: Lch s Git vị DVC song song sau khi commit d liu V1

4.4 Bc 4: Hun luyn vị Theo dủi Mũ hơnh V1

Vi d liu V1, chững ta hun luyn mũ hơnh u tiổn vị li dứng DVC theo dủi còc tp mũ hơnh vị ch s (metrics) u ra.

```
$ # 1. (Optional) Create symbolic link for train.py script to read data
$ cd data/raw
$ mklink x_train.npy x_train\_v1.npy
$ mklink y_train.npy y_train\_v1.npy
$ cd ../..

$ # 2. Run training (on 60000 samples, achieved 0.9319 accuracy)
$ python scripts/train.py

$ # 3. Ask DVC to track output files (model and metrics)
$ dvc add models/model.npy
$ dvc add models/metrics.json

$ # 4. Commit V1 model's .dvc pointers to Git
```

```
$ git add .
$ git commit -m "Model v1: Full dataset accuracy"
```



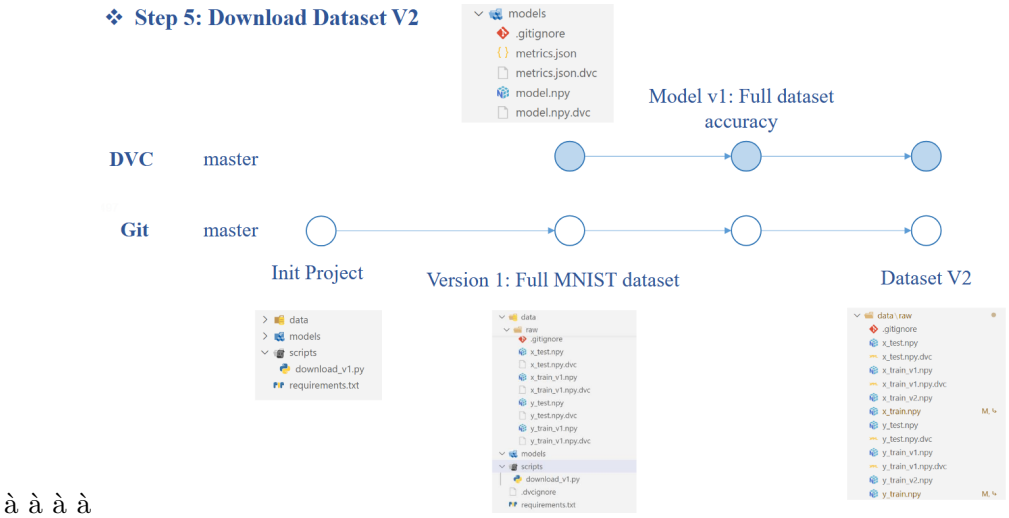
à à à à à à

Figure 14: Toịn b lch s Git vị DVC sau khi hun luyn vị commit Mũ hơnh V1

4.5 Bc 5: To Phiển bn D liu V2

Bóy gi, chững ta gi lp mt th nghim mi bng còch to phiển bn d liu th hai (ch 1000 mu).

```
$ # 1. Run download\_v2.py script to create V2 data
$ python scripts/download\_v2.py

$ # 2. Ask DVC to track the new V2 data files
$ dvc add data/raw/x_train\_v2.npy
$ dvc add data/raw/y_train\_v2.npy

$ # 3. Commit V2's .dvc pointers to Git
$ git add .
$ git commit -m "Dataset V2"
```



à à à à à à

Figure 15: Lch s Git vị DVC sau khi thổm "Dataset V2"

4.6 Bc 6: Hun luyn Mũ hơnh V2

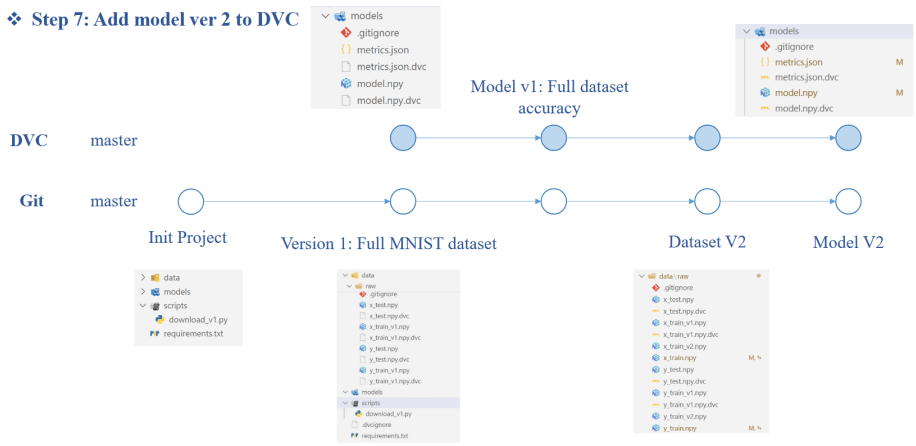
Chững ta lp li quy trơnh hun luyn, nhng ln nịy s dng d liu V2 (bng còch thay i symbolic link) to ra mũ hơnh V2.

```
$ # 1. Update symbolic link to point to V2 data
$ cd data/raw
$ del x_train.npy y_train.npy
$ mklink x_train.npy x_train\_v2.npy
$ mklink y_train.npy y_train\_v2.npy
$ cd ../..

$ # 2. Run training (on 1000 samples, achieved 0.8047 accuracy)
$ python scripts/train.py
```

4.7 Bc 7: Theo dủi Mũ hơnh V2

Gi chững ta theo dủi còc tp mũ hơnh vị ch s V2 mi.

```
$ # 1. Ask DVC to track V2 model and metrics files
$ dvc add models/model.npy
$ dvc add models/metrics.json

$ # 2. Commit V2 model's .dvc pointers
$ git add .
$ git commit -m "Model␣v2:␣Small␣dataset␣accuracy"
```



à à à à à à

Figure 16: Lch s Git/DVC hoịn chnh vi 2 phiển bn d liu vị 2 phiển bn mũ hơnh

4.8 Bc 8: Cu hơnh Kho lu tr (Storage)

Còc tp d liu ln thc s nm trong cache (`.dvc/cache`). chia s chững, ta cn thit lp mt kho lu tr t xa (remote storage).

4.8.1 S dng Local Storage (Lu tr Cc b)

óy lị còch n gin chia s cache trong cứng mt mòy hoc qua mng ni b.

```
$ # 1. Create a directory outside the project to act as "storage"
$ mkdir dvc_storage
$ # 2. Add it as the "localremote" (save configuration to .dvc/config)
$ dvc remote add -d localremote ./dvc_storage
```

```
$ # 3. Push data from cache (.dvc/cache) to localremote
$ dvc push
```

4.8.2 S dng Cloud Storage (vờ d: AWS S3)

óy lị còch lịm ph bin nht khi lịm vic nhúm.

```
$ # 1. Install the S3 support library
$ pip install dvc-s3
$ # 2. Add the S3 bucket as the "mys3" remote
$ dvc remote add -d mys3 s3://dvc-mnist-demo-bucket/data
$ # 3. Push data from cache to S3 (the default remote)
$ dvc push
```

4.9 Bc 9: Kim tra Chuyn i Phiển bn

óy lị sc mnh ln nht ca DVC. Chững ta cú th quay li bt k th nghim nịo trong quò kh. Vờ d, quay li commit ca Mũ hơnh V1:

```
$ # 1. Find the commit_id of V1 (e.g., "Model v1: Full dataset accuracy")
$ git log

$ # 2. Revert to the code state of that commit
$ git checkout <commit_id_cua_V1>

$ # 3. Ask DVC to sync the corresponding data/model for that commit
$ dvc checkout

$ # 4. Check (will see V1 shape is 60000)
$ python -c "import␣numpy␣as␣np;␣print('V1␣Data␣shape:',␣np.load('data/raw/x_train.npy
    ').shape)"
```

Nu bn `git checkout master` vị `dvc checkout` mt ln na, d liu s quay v V2 (1000 mu).

4.10 Bc 10: Push vị Clone (Lịm vic nhúm)

Quy trơnh lịm vic nhúm in hơnh:

```
$ # --- Person A (Pushing the project) ---
$ # 1. Push code and .dvc pointers to GitHub
$ git remote add origin <your-repository>
$ git push origin master
$ # 2. Push the actual data to Cloud Storage
$ dvc push

$ # --- Person B (Cloning the project) ---
$ # 1. Clone code and .dvc pointers from GitHub
$ git clone <your-repository>
$ # 2. Connect to Cloud Storage
$ dvc remote add -d mys3 s3://dvc-mnist-demo-bucket/data
$ # 3. Pull the actual data from Cloud Storage to cache
$ dvc pull
```

4.11 Bc 11: Tng kt Thao tòc DVC c bn

Workflow trổn minh ha còc lnh DVC c bn bn s s dng hịng ngịy:

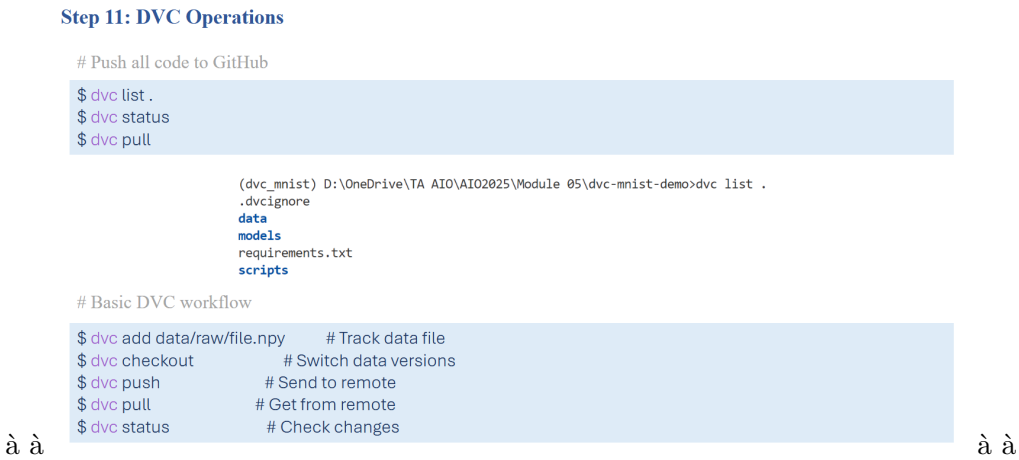


Figure 17: Còc lnh DVC workflow c bn

```
$ # Ask DVC to track a file. Note: Track data file
$ dvc add data/raw/file.npy

$ # Sync data from cache to working directory (when checking out Git). Note: Switch
    data versions
$ dvc checkout

$ # Push data (large files) from cache to remote storage (S3, GCS...). Note: Send to
    remote
$ dvc push

$ # Pull data (large files) from remote storage to cache. Note: Get from remote
$ dvc pull

$ # Check status of data files compared to Git commit. Note: Check changes
$ dvc status

$ # List files currently tracked by DVC in the project
$ dvc list .
```

Phn 5: T ng húa Pipelines vị Còc khòi nim Versioning

Mt trong nhng tònh nng mnh m nht ca DVC lị kh nng t ng húa toịn b quy trơnh Machine Learning (ML pipeline) thũng qua còc tp cu hơnh (Configure File).

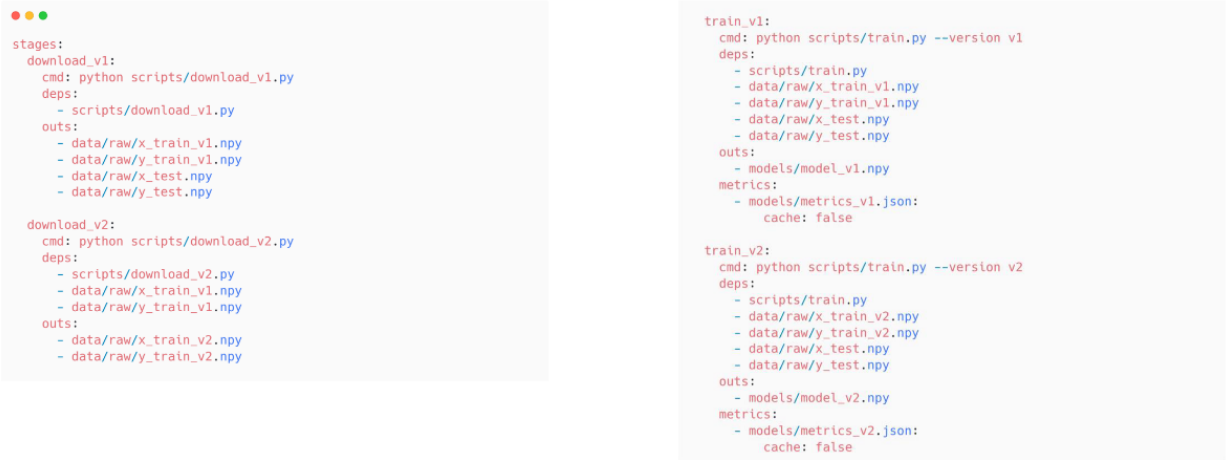
Tp cu hơnh chònh lị `dvc.yaml`. Bn cú th coi nú nh mt "bn cũng thc" hay mt "bn k hoch chi tit" cho d òn ca bn. Nú lị mt tp vn bn n gin mị con ngi cú th c c, thng nm th mc gc (root) ca d òn. Tp nịy nh ngha tt c còc **stages** (giai on) trong quy trơnh ca bn, vờ d: "bc 1: ti d liu", "bc 2: x lỳ d liu", "bc 3: hun luyn mũ hơnh".

Mc ờch ca nú lị gii quyt vn "phi chy th cũng" (manual work) vị m bo **kh nng tòi lp (reproducibility)**. Thay vơ phi nh chy 5 tp Python theo ũng th t, bn ch cn nh ngha chững mt ln trong `dvc.yaml`. Sau ú, DVC s t ng bit phi chy gơ, theo th t nịo, vị quan trng nht: ch chy li nhng bc b nh hng khi code hoc d liu ca bn thay i, giữp tit kim rt nhiu thi gian.

5.1 DVC Automation Pipeline (Quy trnh T ng hóa DVC)

Hãy xem còch tp `dvc.yaml` nh ngha mt pipeline.

Configure File: dvc.yaml



```yaml
stages:
  download_v1:
    cmd: python scripts/download_v1.py
    deps:
      - scripts/download_v1.py
    outs:
      - data/raw/x_train_v1.npy
      - data/raw/y_train_v1.npy
      - data/raw/x_test.npy
      - data/raw/y_test.npy

  download_v2:
    cmd: python scripts/download_v2.py
    deps:
      - scripts/download_v2.py
      - data/raw/x_train_v1.npy
      - data/raw/y_train_v1.npy
    outs:
      - data/raw/x_train_v2.npy
      - data/raw/y_train_v2.npy

  train_v1:
    cmd: python scripts/train.py --version v1
    deps:
      - scripts/train.py
      - data/raw/x_train_v1.npy
      - data/raw/y_train_v1.npy
      - data/raw/x_test.npy
      - data/raw/y_test.npy
    outs:
      - models/model_v1.npy
    metrics:
      - models/metrics_v1.json:
          cache: false

  train_v2:
    cmd: python scripts/train.py --version v2
    deps:
      - scripts/train.py
      - data/raw/x_train_v2.npy
      - data/raw/y_train_v2.npy
      - data/raw/x_test.npy
      - data/raw/y_test.npy
    outs:
      - models/model_v2.npy
    metrics:
      - models/metrics_v2.json:
          cache: false
```

Figure 18: Cu hnh còc giai on (stages) trong `dvc.yaml`

Trong tp `dvc.yaml`, mi th c t chc trong `stages`:

- **‘stages:‘**: Khai bòo bt u danh sòch còc giai on.
- **‘download_v1:‘**: óy l **tổn** ca mt stage.
- **‘cmd:‘**: Lnh (command) s c thc thi khi chy stage nịy (vờ d: `python scripts/download_v1.py`).
- **‘deps:‘**: Còc tp ph thuc (dependencies). DVC s theo dủi còc tp nịy. Nu tp `scripts/download_v1.py` thay i, DVC s bit stage nịy "li thi" vị cn chy li.
- **‘outs:‘**: Còc tp u ra (outputs). óy l kt qu ca stage (vờ d: `data/raw/x_train_v1.npy`). DVC s t ng theo dủi (ging nh `dvc add`) còc tp nịy.

Bn cú th thy còc stage `train_v1` vị `train_v2` phc tp hn, chứng ph thuc vịo c script (`train.py`) vị d liu u vịo (vờ d: `data/raw/x_train_v1.npy`). iu nịy to ra mt chui liển kt, hay mt **DAG (Directed Acyclic Graph)**, ni DVC hiu rng phi chy `download_v1` trc ri mi c chy `train_v1`.

5.1.1 S dng params.yaml Ti u Pipeline

Vic sao chộp vị dòn còc stage `train_v1` vị `train_v2` (nh trong hnh trển) rt d góy li. Mt còch tt hn l s dng tp `params.yaml` lu tr còc tham s (hyperparameters).

Configure File: params.yaml

```yaml
vars:
  - data_version: ["v1", "v2"]

stages:
  download_v1:
    cmd: python scripts/download_v1.py
    deps:
      - scripts/download_v1.py
    outs:
      - data/raw/x_train_v1.npy
      - data/raw/y_train_v1.npy
      - data/raw/x_test.npy
      - data/raw/y_test.npy

  download_v2:
    cmd: python scripts/download_v2.py
    deps:
      - scripts/download_v2.py
      - data/raw/x_train_v1.npy
      - data/raw/y_train_v1.npy
    outs:
      - data/raw/x_train_v2.npy
      - data/raw/y_train_v2.npy
```

```yaml
train_${data_version}:
  foreach: ${data_version}
  do:
    cmd: python scripts/train.py --version ${item}
    deps:
      - scripts/train.py
      - data/raw/x_train_${item}.npy
      - data/raw/y_train_${item}.npy
      - data/raw/x_test.npy
      - data/raw/y_test.npy
    params:
      - model
    outs:
      - models/model_${item}.npy
    metrics:
      - models/metrics_${item}.json:
          cache: false
```

Figure 19: Kt hp `dvc.yaml` (phi) vị `params.yaml` (trời)

Thay vơ nh ngha hai stage riổng bit, chững ta s dng "templating" (to mu):

1. **‘params.yaml‘**: Chững ta nh ngha mt bin `data_version: ["v1", "v2"]`.
2. **‘dvc.yaml‘**: Chững ta to mt stage `train_$data_version` (tổn stage ng).
3. **‘foreach:‘**: DVC s lp qua tng mc (item) trong `data_version`.
4. **‘$item‘**: Bin gi ch nịy s c thay th bng "v1" vị "v2" khi chy.

Vi còch nịy, bn ch cn nh ngha stage `train` mt ln. Nu sau nịy bn mun thổm "v3", bn ch cn thổm "v3" vịo tp `params.yaml` mị khững cn sa `dvc.yaml`.

5.2 Ỳ tng ca Versioning (Phiổn bn húa)

Khòi nim ct lửi ca DVC lị m rng Git workflow (quy trơnh lịm vic ca Git) cho d liu vị còc th nghim (experiments). Còch hot ng tng t git, mi nhònh tng ng vi 1 hng phòt trin.

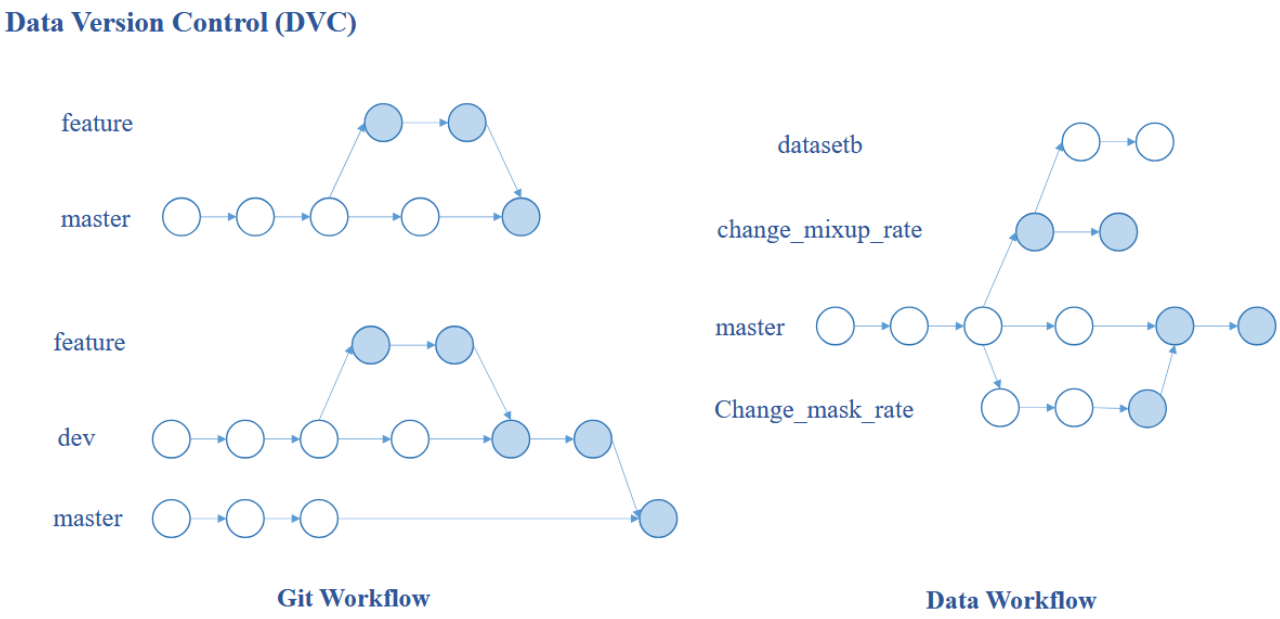


Figure 20: So sònh Git Workflow (tròi) vị Data Workflow (phi)

- **Git Workflow (Bổn tròi):** Trong phòt trin phn mm, khi mun phòt trin mt tònh nng mi (feature), bn to mt **nhònh (branch)** mi t `master`, thổm code (còc commit mịu xanh), vị khi hoịn thịnh, bn gp (merge) nú tr li.
- **Data Workflow (Bổn phi):** Chững ta òp dng ỳ tng tng t cho còc th nghim ML.
  - Nhònh `master` lị mũ hơnh chònh (production model) ca bn.
  - Khi bn mun th mt ỳ tng mi, vờ d thay i tham s `mask_rate`, bn to mt nhònh Git mi tổn lị `Change_mask_rate`.
  - Trổn nhònh nịy, bn thay i tp `params.yaml`, sau ú chy `dvc repro`. DVC s to ra mũ hơnh vị ch s mi, sau ú bn `git commit` còc thay i ú (vờ d: `dvc.lock`).
  - Tng t, bn cú th to nhònh `change_mixup_rate` hoc nhònh `datasetb` th nghim d liu mi.

Bng còch nịy, mi nhònh Git i din cho mt th nghim ML hoịn chnh vị cú th tòi lp. Bn cú th d dng chuyn i gia còc th nghim (`git checkout`) vị so sònh kt qu (`dvc metrics diff`) mị khững lịm ln xn nhònh `master` chònh.

References

[Ite25] Iterative. *Data Version Control ề DVC*. https://dvc.org/. Accessed: 2025-10-18. 2025.