Structuring Machine Learning Projects

Sayak Paul, Deep Learning Associate at PyImageSearch

DevFest Jaipur

September 8, 2019

MTX Group Inc

Agenda

- Machine learning: From a software engineer’s keystrokes
- Why care about the structure?
- A **structured** approach to **structuring**
  - A polished directory structure
  - Workspace setup
  - Building mental models of the project’s flow
  - Experimentation
  - ....
- Taking the next steps
- Guiding lights

Life-cycle of a machine learning project

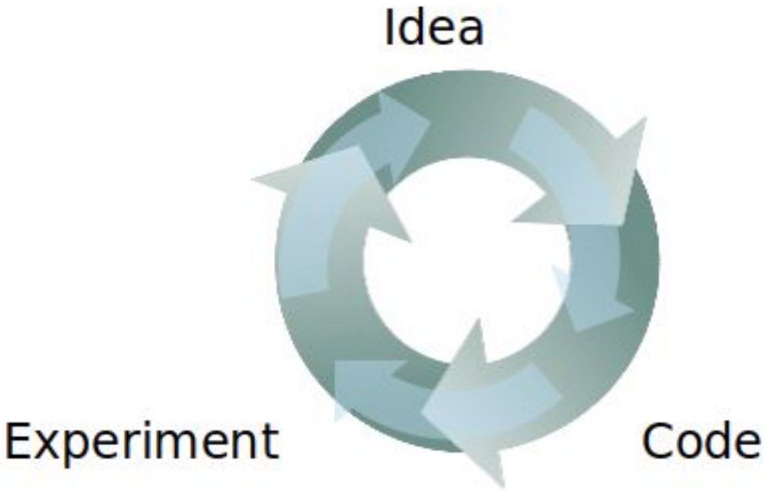
- Problem understanding
- Data collection, wrangling and so on
- Understanding of the data
- Beginning the modeling process
- Evaluate, tune, repeat
- Model deployment, monitoring and so on...

Note that the above steps are very summarized

Structuring machine learning projects: The need

Applied machine (and deep) learning is iterative

- Lots of experimentations make it hard to keep track
- Hyperparameter tuning
- Dataset reconstructions
- KPI: Time-Cost tradeoff



Source

Reproducibility crisis

- **Stochasticity** in machine learning models (neural nets specifically)
- Code breakdown due to dependency mismatches
- Difference in machine configurations

Versioning data and codebase

- Training data might change over time (less frequent)

Versioning data and codebase

- Training data might change over time (less frequent) ← Pain point!

Versioning data and codebase

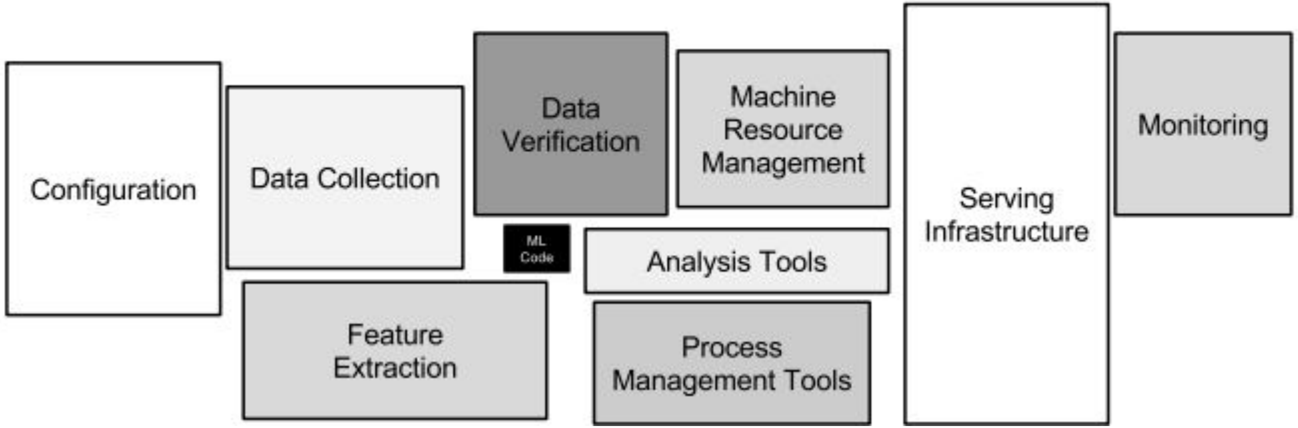
- Training data might change over time (less frequent)
- Changes in project's codebase are far more frequent

Regularity in checkpointing

Decrease in validation loss was steady during the training, but what if that model is lost?

A directory structure

- A full-fledged ML project is **not just about models**



[Source]

A high-interest technical debt

- Lots of ideas, lots of experiments - gives birth to **technical debt**
- Not everyone in the team understands everything
- Lack of *documentation* as it is considered *not-so-cool*

A high-interest technical debt

- Lots of ideas, lots of experiments - gives birth to **technical debt**
- Not everyone in the team understands everything
- Lack of *documentation* as it is considered *not-so-cool*

This hurts the overall development of the project!

Quick summary

- Applied machine learning is an iterative process
- Reproducibility crisis
- Versioning data and codebase
- Regularity in checkpointing
- Directory structures
- Technical debt

Structuring Machine Learning projects: A definitive approach

Starting with a polished directory structure

- Data (along with scripts to *download it* and *preprocess it*)
- Experimentations
- Web backend
- Utility scripts
- Model building and model training scripts

A reference directory structure

```
apparel_classifier/              # Package that can be deployed as a self-contained prediction system
  __init__.py

  apparel_predictor.py      # Takes a raw image and obtains a prediction

  datasets/                     # Code for loading datasets
      __init__.py
      dataset.py                # Base class for datasets - logic for downloading data (if not available)
      fmnist_dataset.py
      fmnist_essentials.json
      dataset_sequence.py

  models/                       # Code for instantiating models, including data preprocessing and loss functions
      __init__.py
      base.py                   # Base class for models
      image_model.py

  networks/          # Code for building neural networks (i.e., 'dumb' input->output mappings) used by models
      __init__.py
      mlp.py

  tests/
      support/                          # Raw data used by tests
      test_apparel_predictor.py       # Test model on a few key examples

  weights/                              # Weights for production model
      Image_Model_FMNIST_Dataset_weights.h5

  util.py

training/                        # Code for running training experiments and selecting the best model.
    run_experiment.py            # Parse experiment config and launch training.
    util.py                      # Logic for training a model with a given config
```

Workspace setup

- Use of environment management tools - **pipenv**, **virtualenv** etc
- Incorporating containers - **Docker**, **Kubernetes** etc
- Use of unified machine learning platforms like FloydHub

Keeping track of the experiments

Machine learning experiments have a lot of components:

- Loss/Accuracy metrics
- Model parameters
- CPU, GPU and disk usage

And so on ...

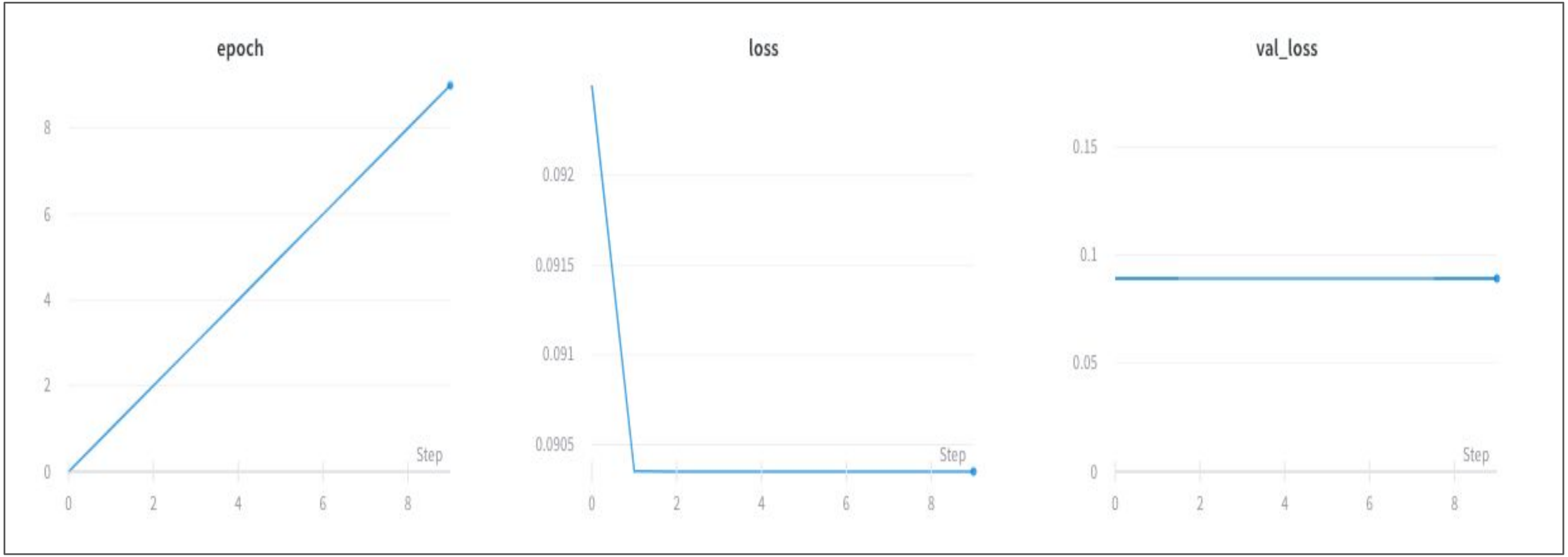
Keeping track of the experiments

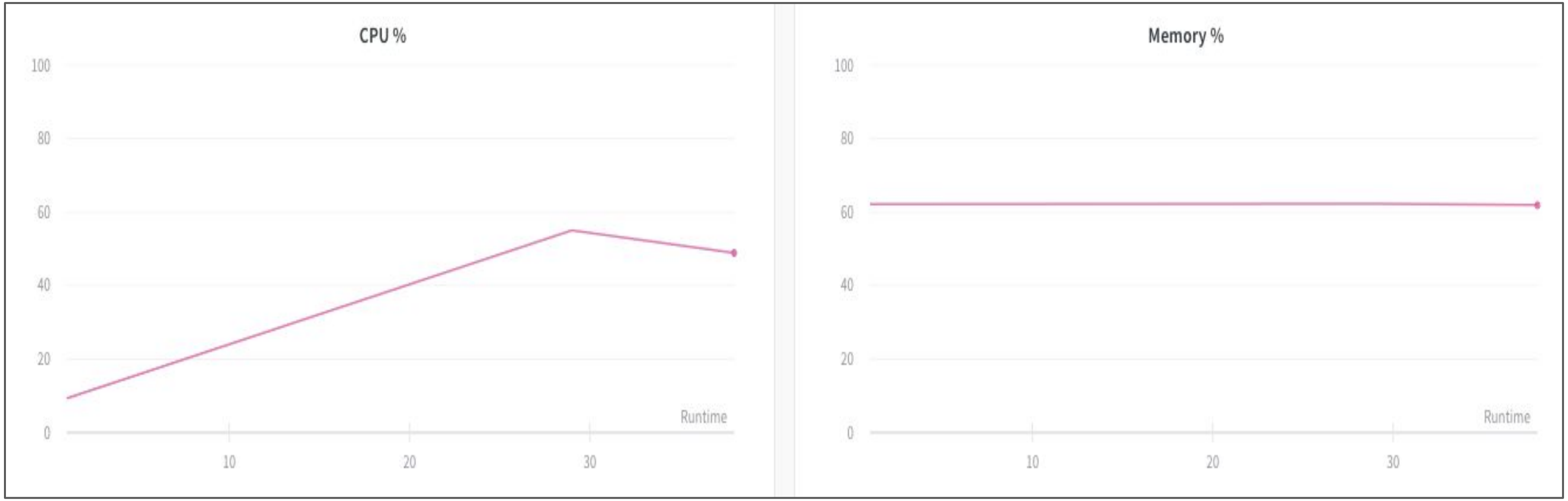
Machine learning experiments have a lot of components:

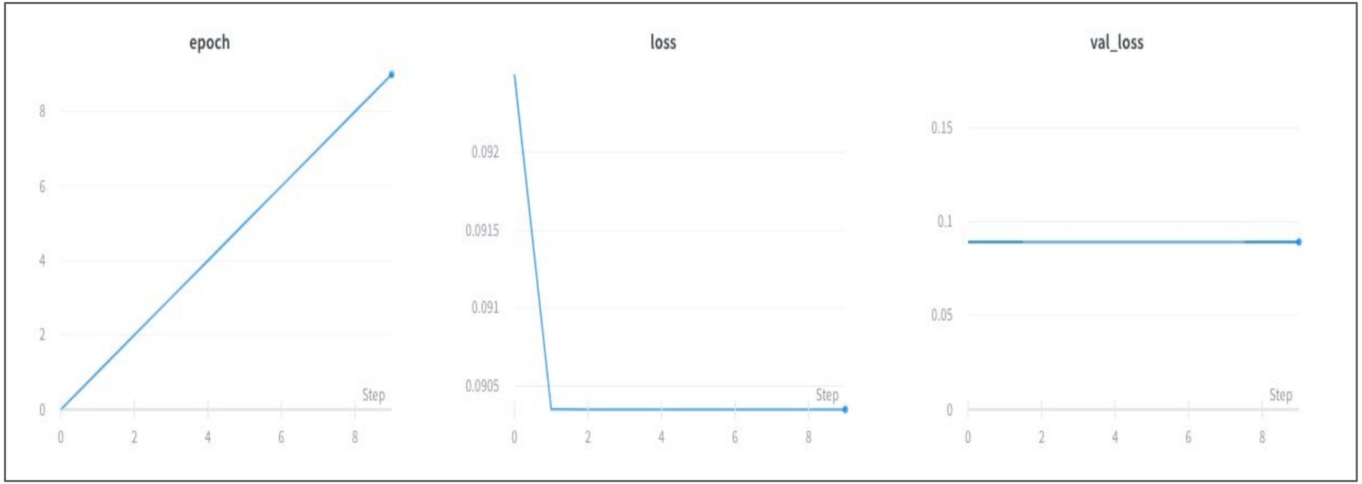
- Loss/Accuracy metrics
- Model parameters
- CPU, GPU and disk usage

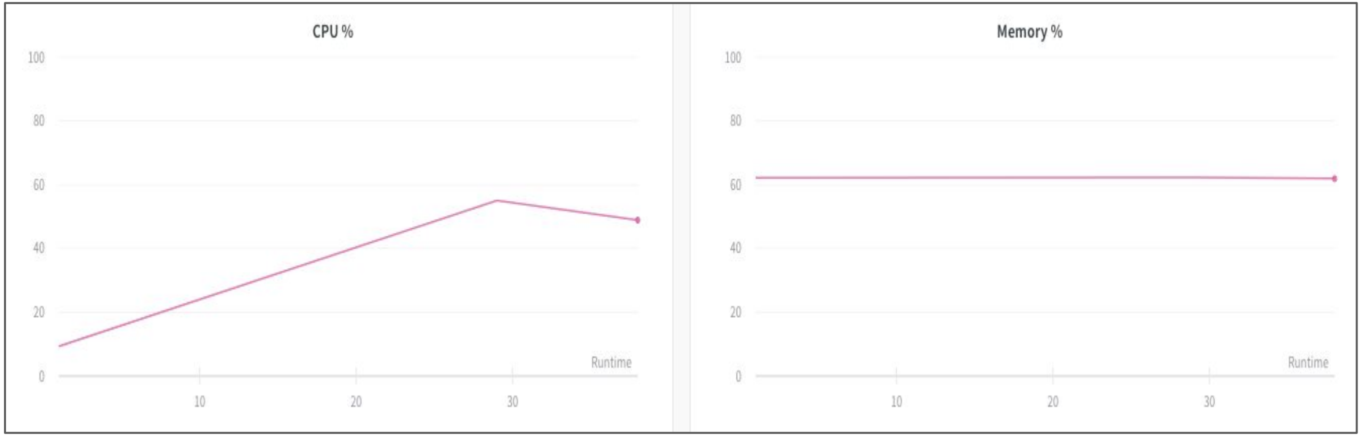
And so on ...

Use **Weights and Biases** to keep track of these for you on the **cloud**!


epoch
loss
val_loss
Step
0
2
4
6
8
0.092
0.0915
0.091
0.0905
0.15
0.1
0.05


CPU %
100
80
60
40
20
0
Runtime
10
20
30
Memory %
100
80
60
40
20
0
Runtime
10
20
30





And many things more!

Building a mental image of the execution flow

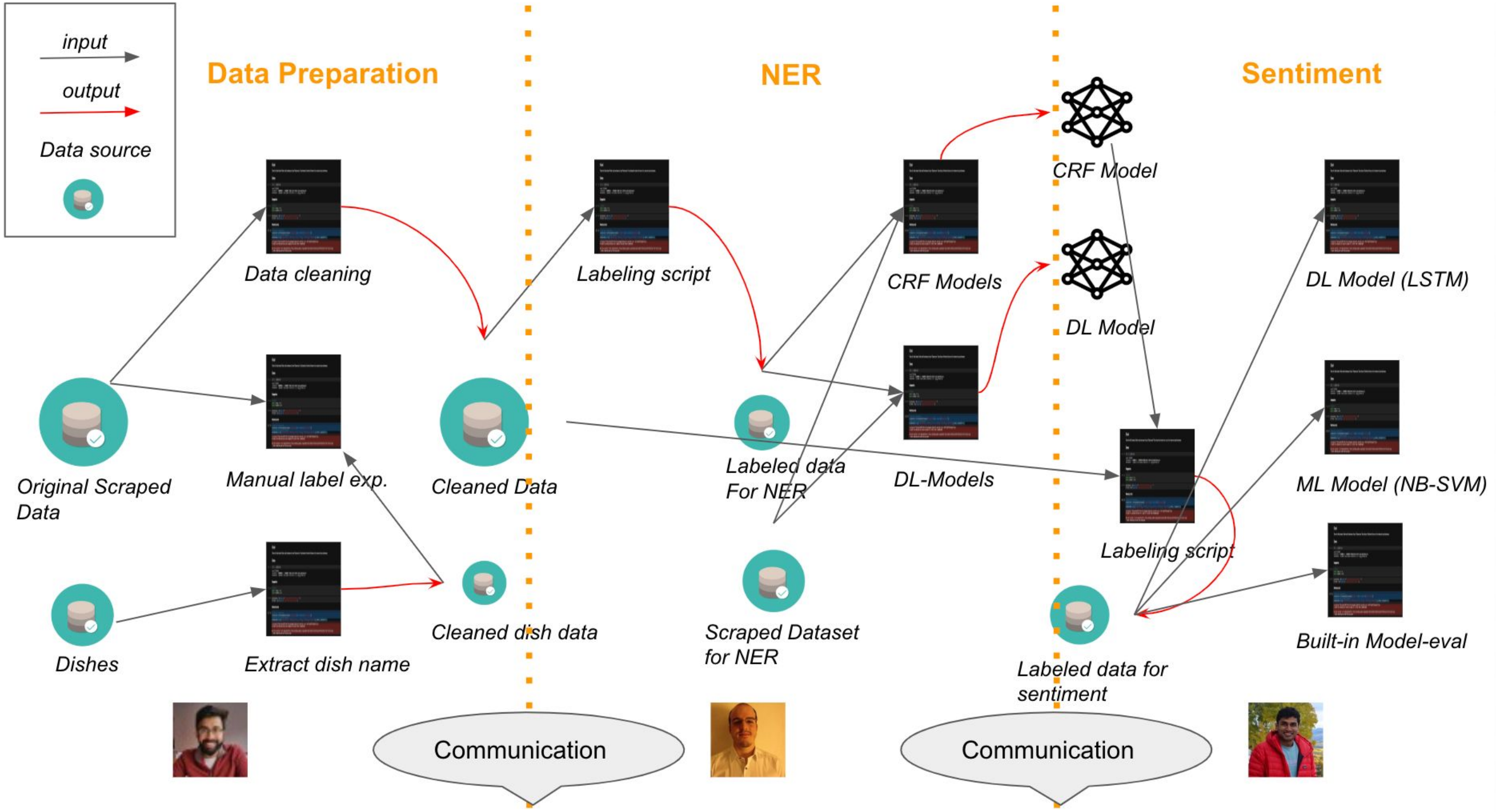
- There will be a lot of inter-connected blocks in a project
  - A block can have several sub-blocks too

Building a mental image of the execution flow

- There will be a lot of inter-connected blocks in a project
    - A block can have several sub-blocks too

Hence a mental model to keep track of these lego blocks always helps!

A specimen of a mental model



sayak.dev

Version controlling data and codebase *separately*

Here are some situations:

- Need to replace the older images with newer ones
- Need to add new images to an already existing training set
- Decision to incorporate active learning to select interesting test data points to manually label them and add to the existing training set

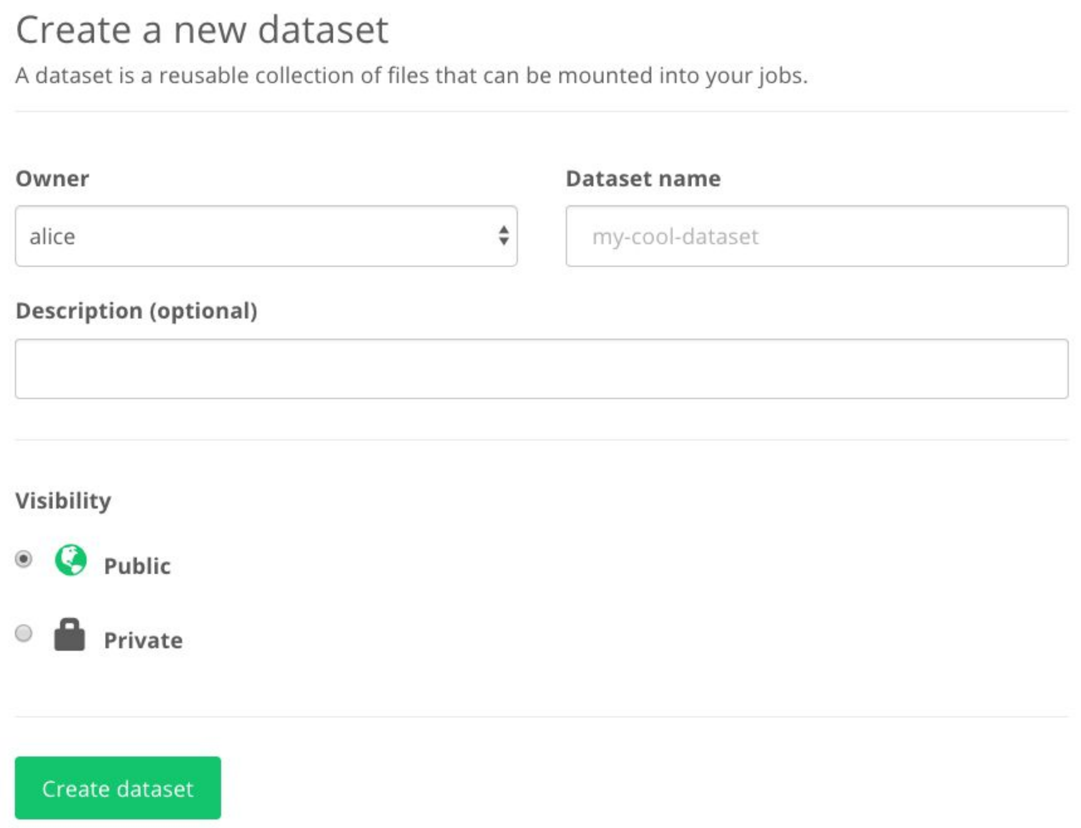
Version controlling data and codebase *separately*

Here are some situations:

- Need to replace the older images with newer ones
- Need to add new images to an already existing training set
- Decision to incorporate active learning to select interesting test data points to manually label them and add to the existing training set

Version control of codebase remains traditional!

FloydHub datasets



FloydHub datasets are a good way to version control your data!

Quick summary

- Maintaining a healthy directory structure
- Setting up the workspace
- Keeping track of your experiments
- Building a mental model of the project flow
- Version control of data and codebase

What did we *not* cover?

- Test cases for machine learning projects
- Model deployment

What did we *not* cover?

- Test cases for machine learning projects
- Model deployment

Maybe next time :)

References

- [How to plan and execute your ML and DL projects]
- [Troubleshooting Deep Neural Networks]
- [Production Machine Learning Systems]

See you next time

Find me here:
sayak.dev

Thank you very much :)

Experts

pyimagesearch