

1. Why we rock

Starting a project takes time...

bootstrap.pytorch
helps you focus on
dataset and model only

and it is

- Scalable
- Modular
- Shareable
- Extendable
- Uncomplicated
- Built for reproducibility
- Easy to log and plot anything

Not a wrapper,

an extension

```
exp:
  dir: logs/mnist/default
  resume:
dataset:
  import: mnist.datasets.factory
  name: mnist
  dir: data/mnist
  train_split: train
  eval_split: val
  nb_threads: 4
  batch_size: 64
model:
  name: simple
  network:
    import: mnist.models.networks.factory
    name: lenet
  criterion:
    name: nll
  metric:
    name: accuracy
    topk: [1,5]
optimizer:
  name: sgd
  lr: 0.01
engine:
  name: default
  nb_epochs: 10
  saving_criteria:
    - loss:min
    - acctop1:max
view:
  - logs:train_epoch.loss
  - logs:eval_epoch.acctop1
```

2. Running experiments

Yaml options are parsed

```
python -m bootstrap.run
-o mnist/options/sgd.yaml
-h
```

Overwrite options from CLI

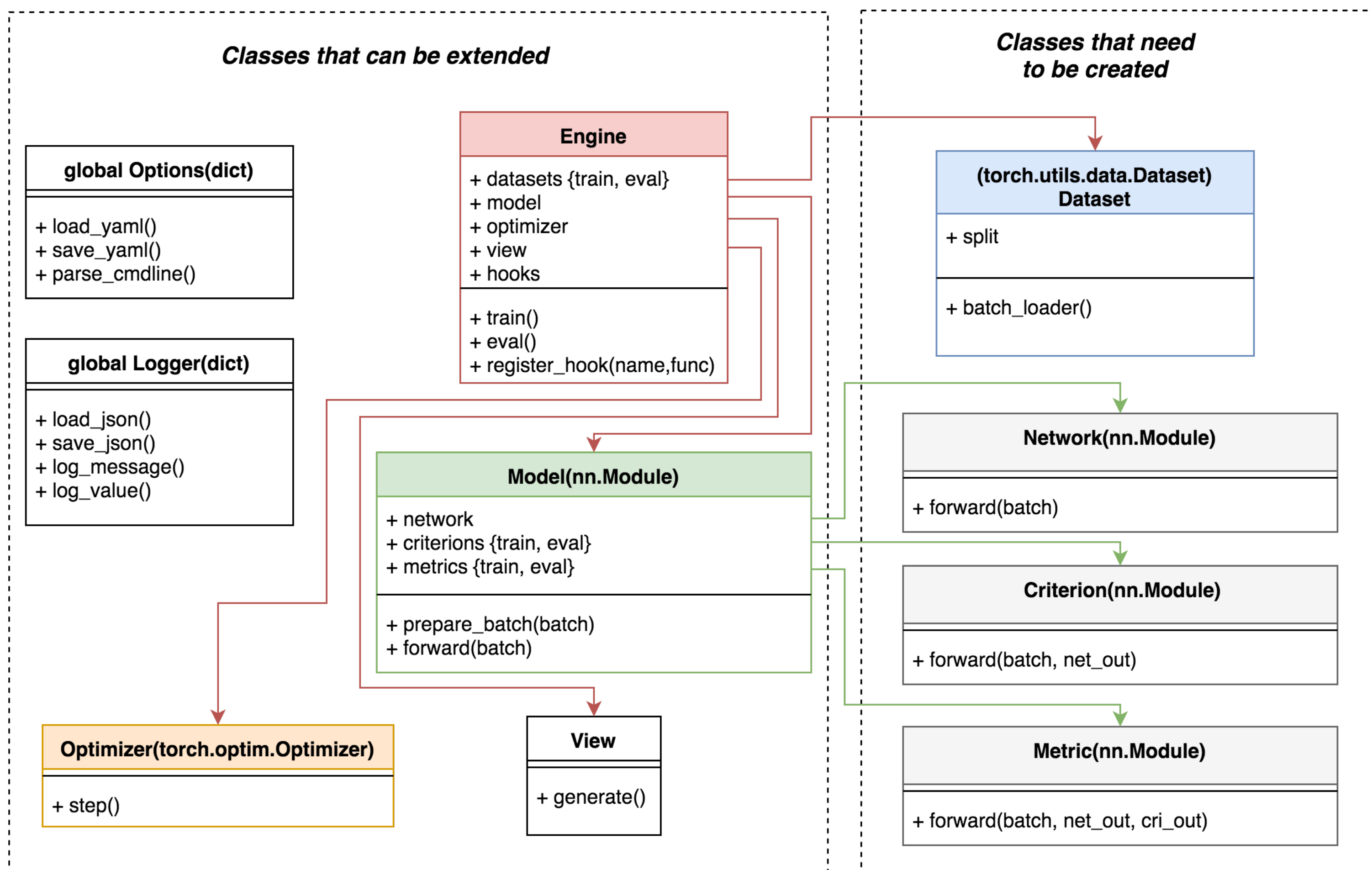
```
python -m bootstrap.run
-o mnist/options/sgd.yaml
--exp.dir logs/mnist/sgd
--model.metric.topk 1 2 3
```

Loading a checkpoint is easy

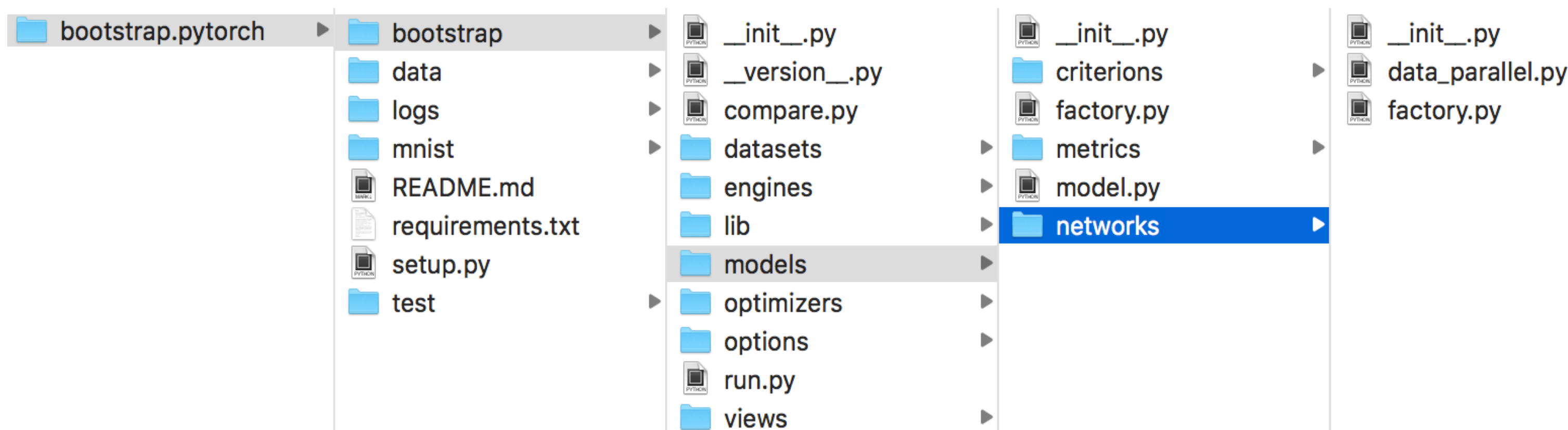
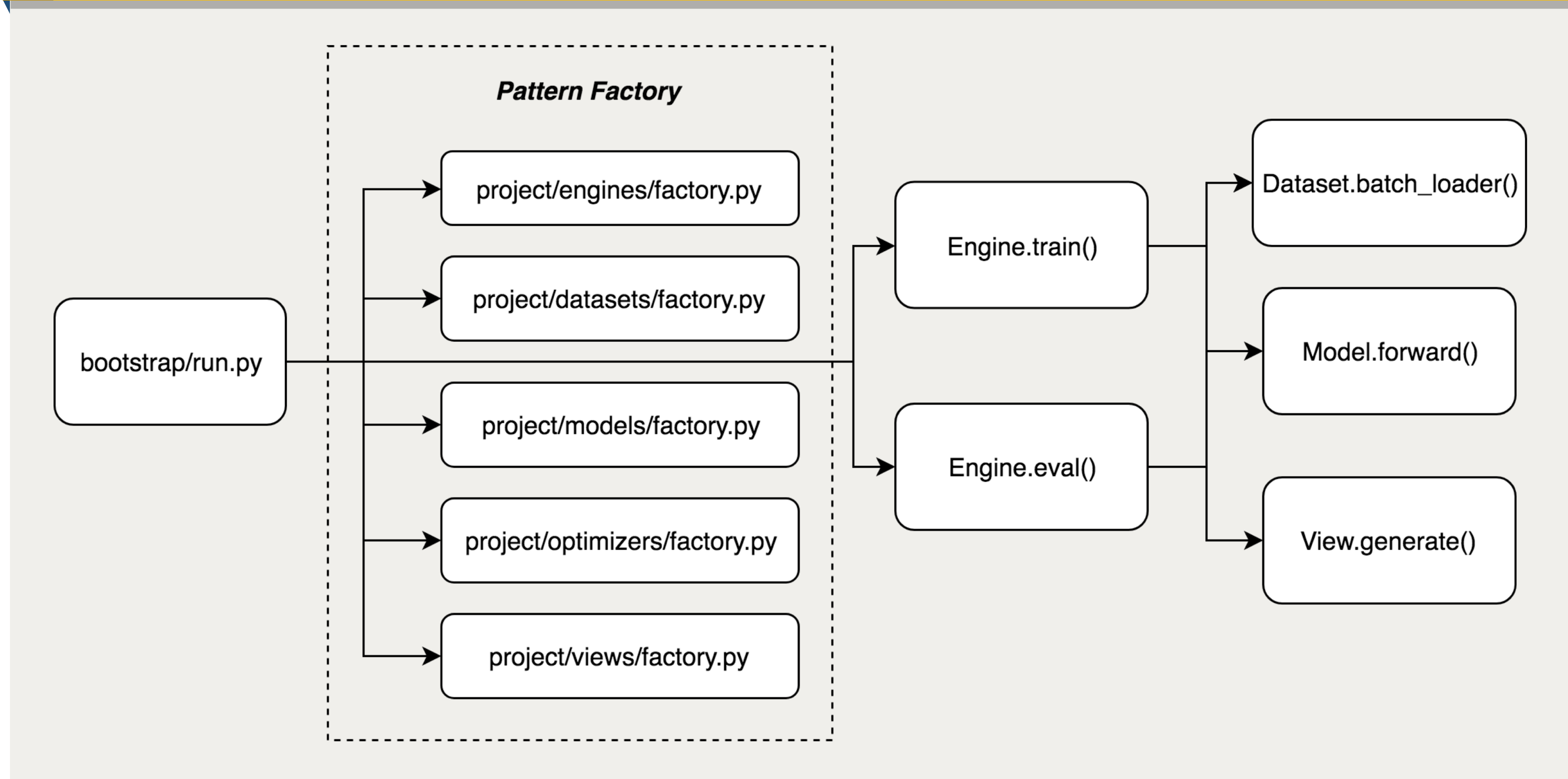
```
python -m bootstrap.run
-o logs/mnist/sgd/options.yaml
--exp.resume best_acctop1
```

Customizations are possible

```
python -m myproject.run
-o logs/mnist/sgd/options.yaml
```



3. Behind the scenes



4. Everything in one folder

```
$ ls logs/mnist
ckpt_last_engine.pth.tar
ckpt_last_model.pth.tar
ckpt_last_optimizer.pth.tar
ckpt_best_acctop1_engine.pth.tar
ckpt_best_acctop1_model.pth.tar
ckpt_best_acctop1_optimizer.pth.tar
logs.json
logs.txt
options.yaml
view.html
```

5. Come and get us

github.com/Cadene/bootstrap.pytorch

Also on github...

mnist.bootstrap.pytorch
imclassif.bootstrap.pytorch
recipe1m.bootstrap.pytorch
vqa.bootstrap.pytorch
nas.bootstrap.pytorch
rel.bootstrap.pytorch

