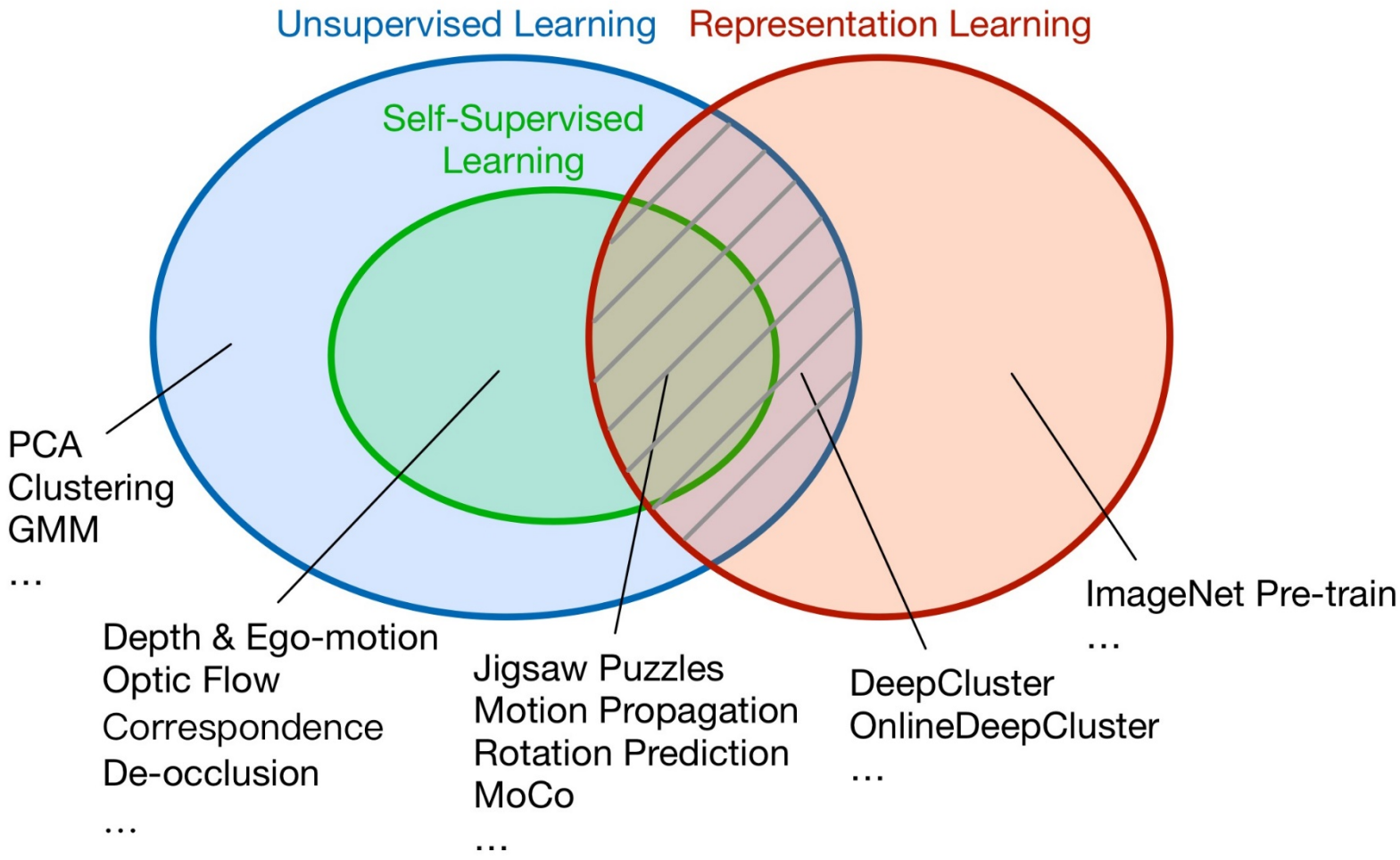
自监督学习算法库OpenSelfSup解析与开发实践

詹晓航

香港中文大学MMLab

Unsupervised Representation Learning



What is Self-Supervised Learning (SSL)?

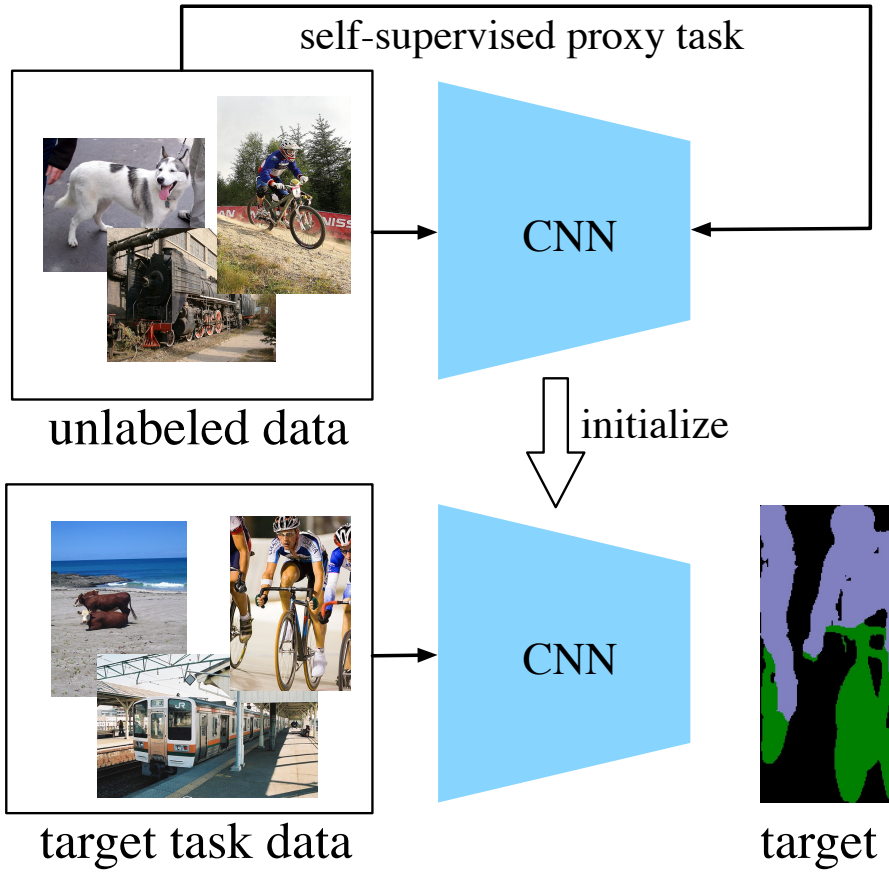
Self-Supervised

Learning

Annotation free

To learn something new

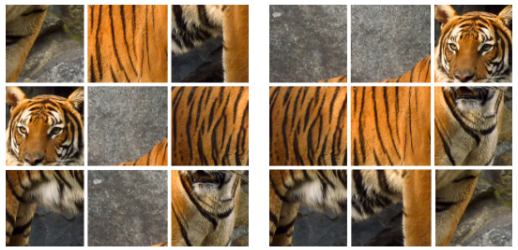
Does image in-painting belong to self-supervised learning?



A typical pipeline

Self-Supervised Proxy/Pretext Tasks

Image Colorization



Solving Jigsaw Puzzles

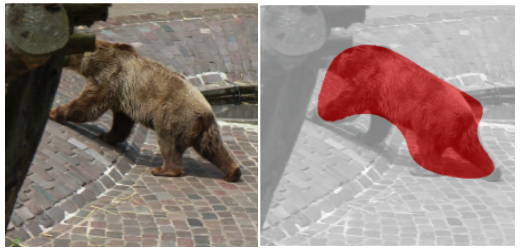
Image In-painting

Rotation Prediction

Instance Discrimination

Counting

Motion prediction



Moving foreground segmentation

Motion propagation

Essence: 1. Prior

- Appearance prior

Image Colorization

Image In-painting

- Physics prior

Rotation Prediction

- Motion tendency prior

Motion prediction
(Fine-tuned for seg: 39.7% mIoU)

- Kinematics prior

Motion propagation
(Fine-tuned for seg: 44.5% mIoU)

Low-entropy priors are more predictable.

Essence: 2. Coherence

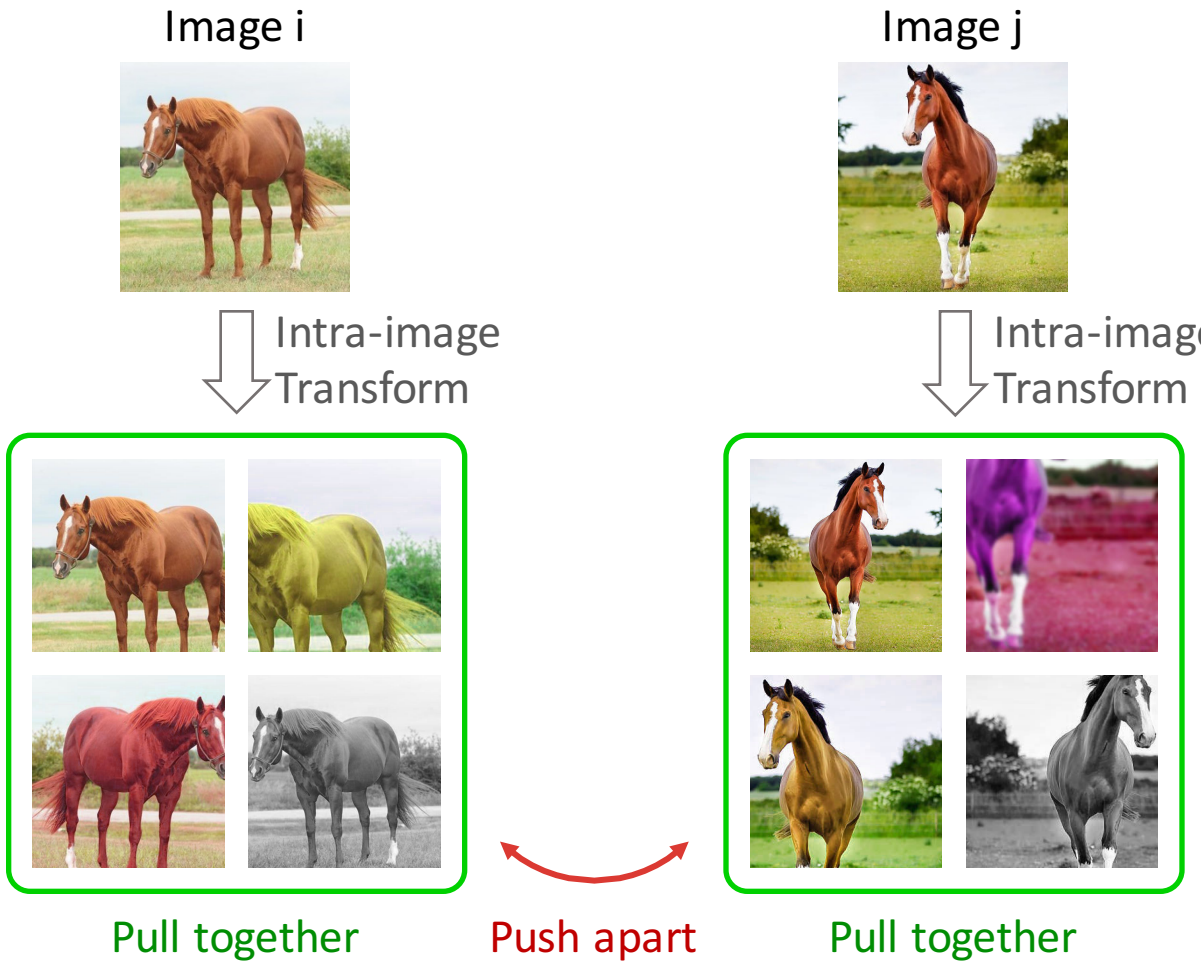
- Spatial coherence

Solving Jigsaw Puzzles

- Temporal coherence

Temporal order verification

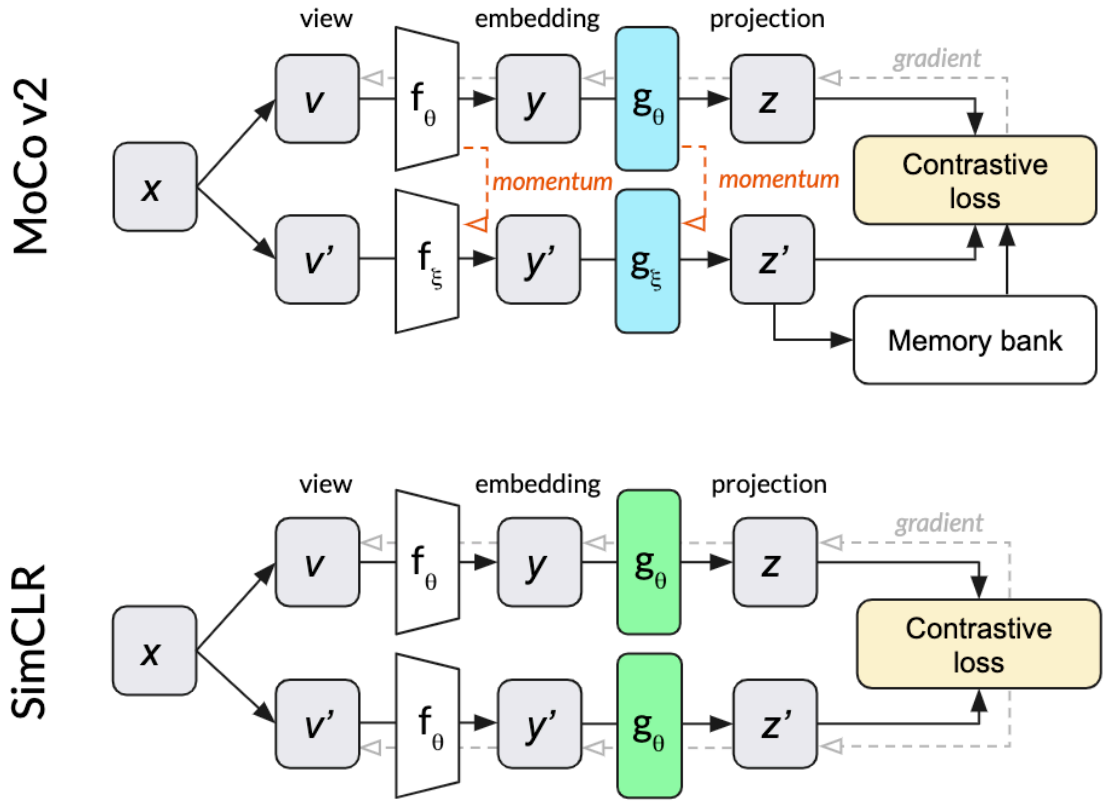
Essence: 3. Structure of Data

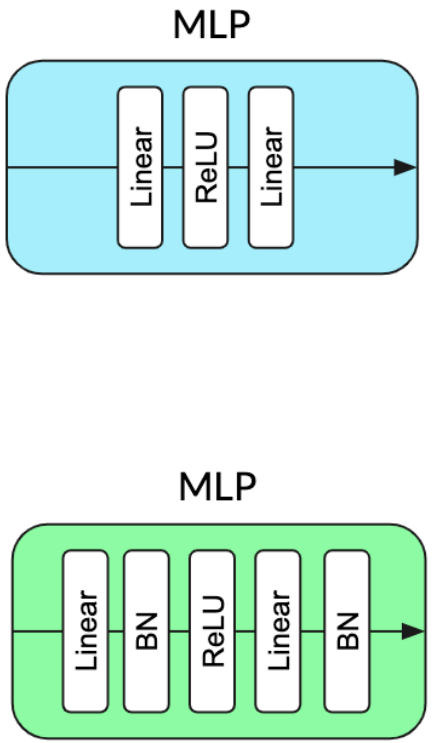


Instance Discrimination (Contrastive Learning)

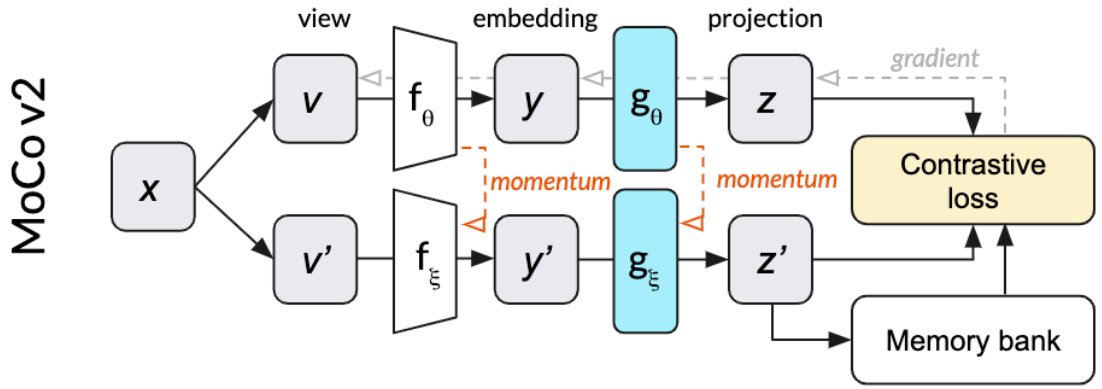
- NIPD
- CPC
- MoCo
- SimCLR
- …

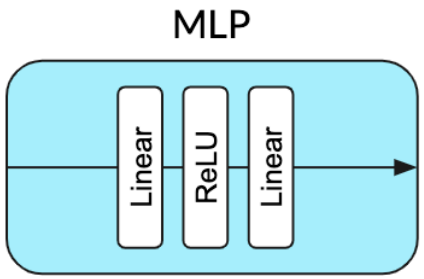
Typical Contrastive-Based SSL

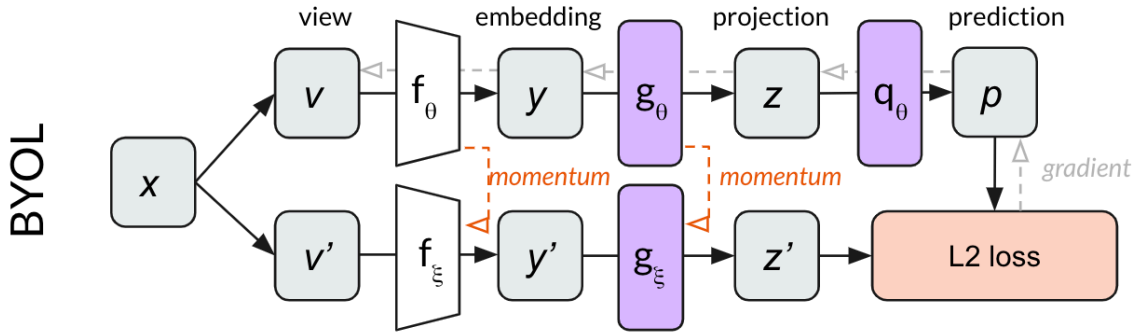


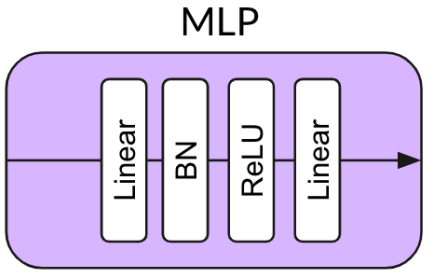


Typical Contrastive-Based SSL





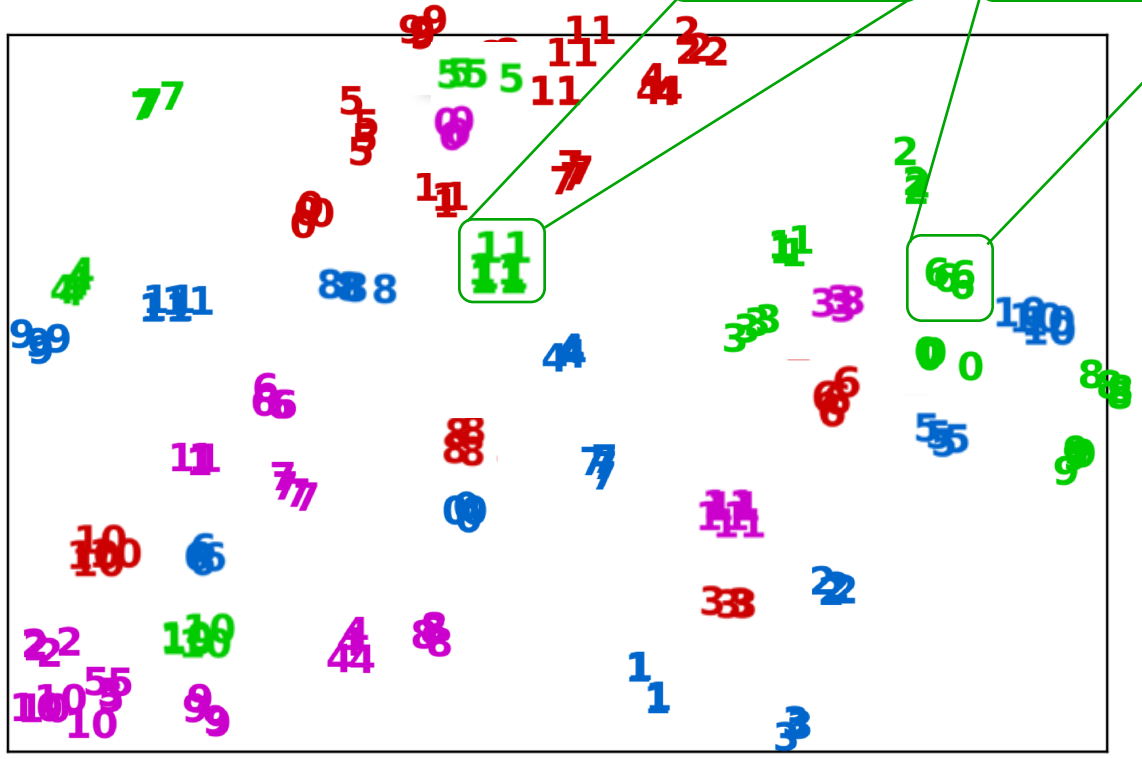




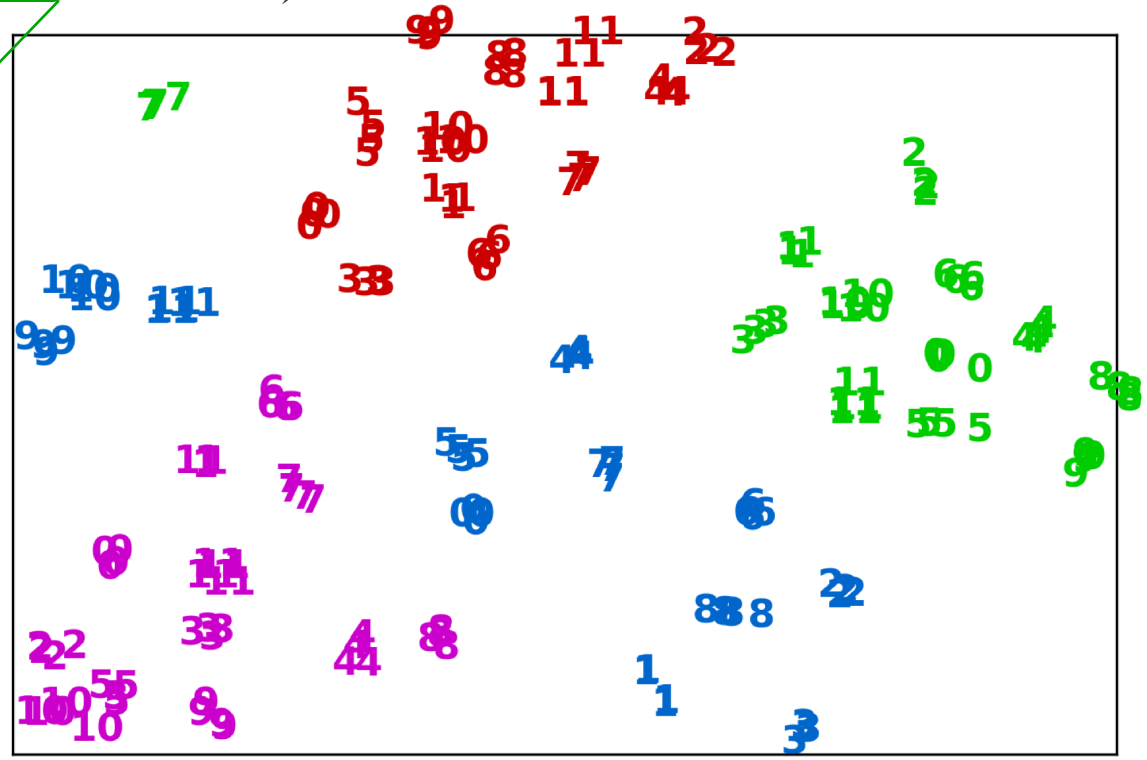
Essence: 3. Structure of Data

Color: category
Number: image index

Both are optimal for Instance Discrimination. Why does the final optimized feature space look like the second case?

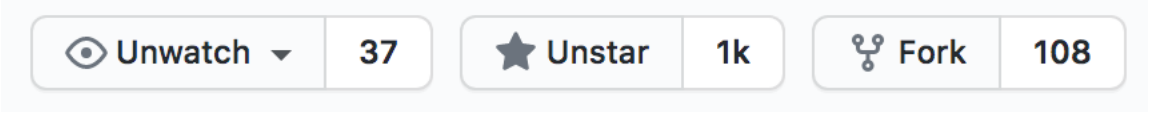


Optimal solution (suppose)



Optimal solution (actual)

open-mmlab/OpenSelfSup



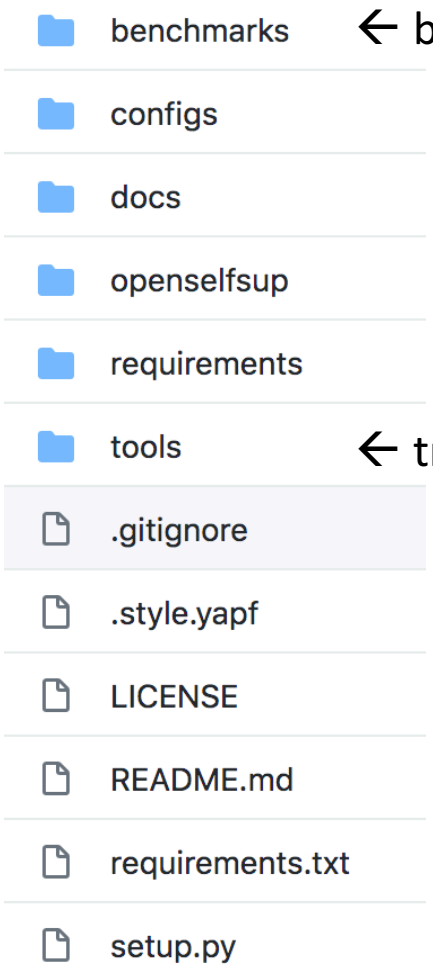
- High-efficiency
  - ➢Distributed & Mixed Precision Training
- Integrity and Extensibility
  - ➢All methods in one framework

Relative Location	Rotation Prediction	Deep Clustering	NPID
ODC	MoCo	SimCLR	BYOL

- Fair Comparisons
  - ➢Standardized benchmarks

Linear classification	Semi-supervised classification	SVM & Low-shot SVM	Object detection

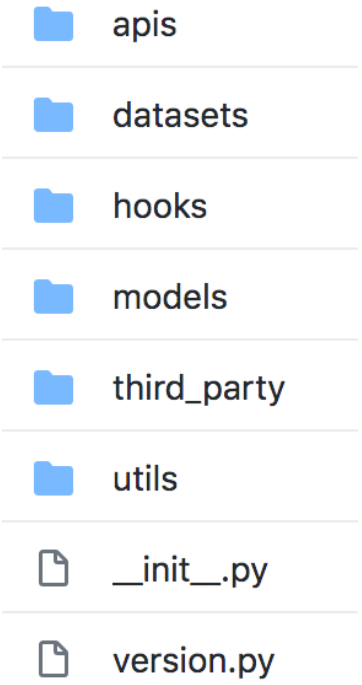
OpenSelfSup: Architecture



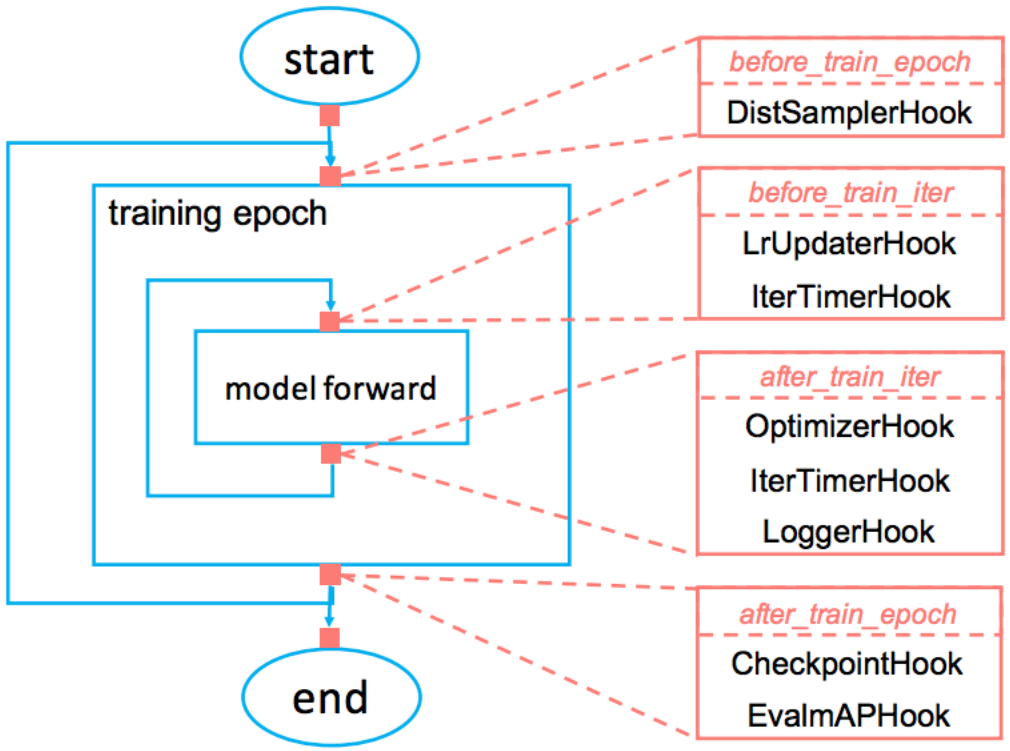
← benchmark scripts

← training scripts

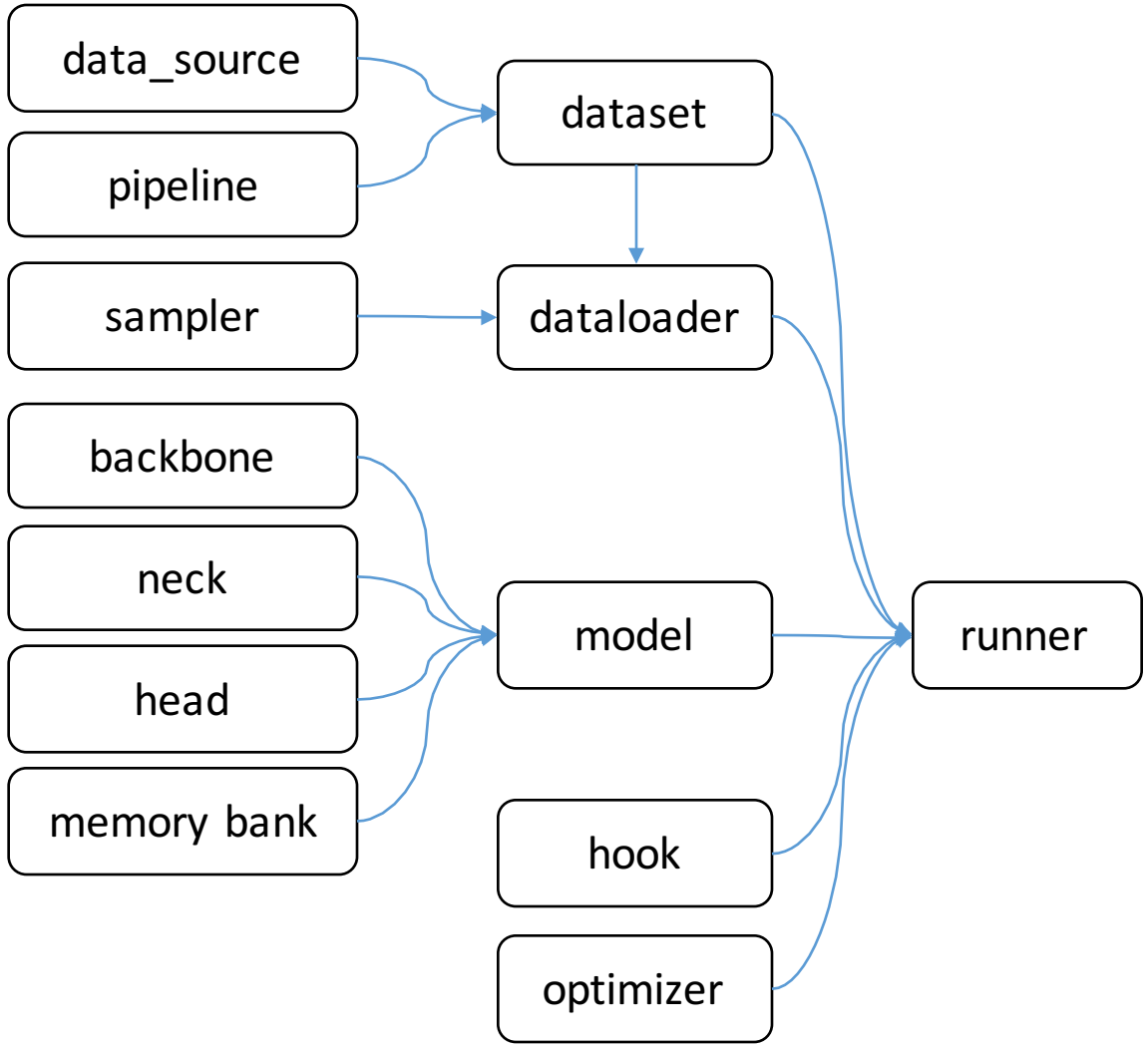
openselfsup/



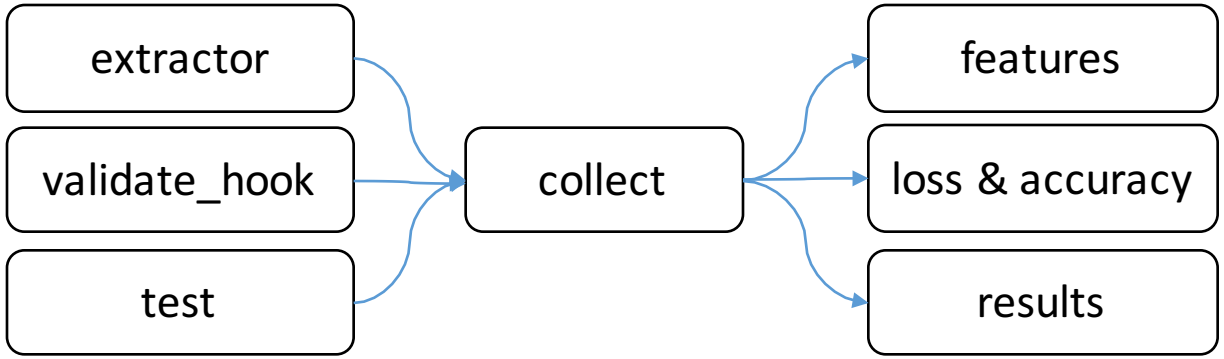
OpenSelfSup: Architecture



runner



Distributed Collect




```
def extract(self, model, data_loader, distributed=False):
    model.eval()
    func = lambda **x: self._forward_func(model, **x)
    if distributed:
        rank, world_size = get_dist_info()
        results = dist_forward_collect(func, data_loader, rank,
                                       len(data_loader.dataset))
    else:
        results = nondist_forward_collect(func, data_loader,
                                          len(data_loader.dataset))
    return results
```

OpenSelfSup: Scripts

- Train

Train with single/multiple GPUs

```
bash tools/dist_train.sh ${CONFIG_FILE} ${GPUS} [optional arguments]
```

Optional arguments are:

- `--resume_from ${CHECKPOINT_FILE}`: Resume from a previous checkpoint file.
- `--pretrained ${PRETRAIN_WEIGHTS}`: Load pretrained weights for the backbone.
- `--deterministic`: Switch on "deterministic" mode which slows down training but the results are reproducible.

```
#!/bin/bash
bash tools/dist_train.sh configs/selfsup/moco/r50_v2.py 8 \
    --resume_from work_dirs/selfsup/moco/r50_v2/epoch_100.pth \
    --deterministic
```

work_dirs/selfsup/moco/r50_v2/: *.log.json train_*.log epoch_*.pth tf_logs/events.*

OpenSelfSup: Scripts

- Train

Launch multiple jobs on a single machine

```
CUDA_VISIBLE_DEVICES=0,1,2,3 PORT=29500 bash tools/dist_train.sh ${CONFIG_FILE} 4
CUDA_VISIBLE_DEVICES=4,5,6,7 PORT=29501 bash tools/dist_train.sh ${CONFIG_FILE} 4
```

tools/dist_train.sh

```
#!/usr/bin/env bash
PYTHON=${PYTHON:-"python"}

CFG=$1
GPUS=$2
PY_ARGS=${@:3}
PORT=${PORT:-29500}

WORK_DIR=$(echo ${CFG%.*} | sed -e "s/configs/work_dirs/g")/

$PYTHON -m torch.distributed.launch --nproc_per_node=$GPUS --master_port=$PORT \
    tools/train.py $CFG --work_dir $WORK_DIR --seed 0 --launcher pytorch ${PY_ARGS}
```

OpenSelfSup: Scripts

- Evaluation

VOC07 Linear SVM & Low-shot Linear SVM

```
# test by epoch (only applicable to experiments trained with OpenSelfSup)
bash benchmarks/dist_test_svm_epoch.sh ${CONFIG_FILE} ${EPOCH} ${FEAT_LIST} ${GPUS}
# test a pretrained model (applicable to any pre-trained models)
bash benchmarks/dist_test_svm_pretrain.sh ${CONFIG_FILE} ${PRETRAIN} ${FEAT_LIST} ${GPUS}
```

ImageNet / Places205 Linear Classification

```
# train
bash benchmarks/dist_train_linear.sh ${CONFIG_FILE} ${WEIGHT_FILE} [optional arguments]
# test (unnecessary if have validation in training)
bash tools/dist_test.sh ${CONFIG_FILE} ${GPUS} ${CHECKPOINT}
```

OpenSelfSup: Scripts

- Evaluation

ImageNet Semi-Supervised Classification

```
# train
bash benchmarks/dist_train_semi.sh ${CONFIG_FILE} ${WEIGHT_FILE} [optional arguments]
# test (unnecessary if have validation in training)
bash tools/dist_test.sh ${CONFIG_FILE} ${GPUS} ${CHECKPOINT}
```

VOC07+12 / COCO17 Object Detection

```
conda activate detectron2 # use detectron2 environment here, otherwise use open-mmlab environment
cd benchmarks/detection
python convert-pretrain-to-detectron2.py ${WEIGHT_FILE} ${OUTPUT_FILE} # must use .pkl as the output extension.
bash run.sh ${DET_CFG} ${OUTPUT_FILE}
```

Benchmarks

- Refer to MODEL_ZOO.md

Tools

Count number of parameters

```
python tools/count_parameters.py ${CONFIG_FILE}
```

Publish a model

Compute the hash of the weight file and append the hash id to the filename. The output file is the input file name with a hash suffix.

```
python tools/publish_model.py ${WEIGHT_FILE}
```

Arguments:

- `WEIGHT_FILE`: The extracted backbone weights extracted aforementioned.

Reproducibility

If you want to make your performance exactly reproducible, please switch on `--deterministic` to train the final model to be published. Note that this flag will switch off `torch.backends.cudnn.benchmark` and slow down the training speed.

OpenSelfSup: Configs

configs/selfsup/

- byol
- deepcluster
- moco
- npid
- odc
- relative_loc
- rotation_pred
- simclr

configs/benchmarks/

- linear_classification
- semi_classification

configs/classification/

- cifar10
- imagenet

BYOL: a Practice

Thank You!