Outline

- Outline
- Build environment using docker
    - Create container
    - Compile with cython to accelerate evalution
- Training on custom dataset
    - *Prepare custom dataset (Step 1)*
        - Dataset hierarchy
        - Naming rule of the data
    - *Register custom dataset (Step 2)*
    - *Import dataset (Step 3)*
    - *Customize your training configuration (Step 4)*
    - *Training & Evaluation (Step 5)*
- Training configuration
- Visualization
- Convert to TorchScript Model
- TorchScript Model Evaluation
- Appendix

Build environment using docker

Create container

```
# build image
docker build -f fast-reid.dockerfile -t fsat-reid-dev .

# create container
docker run -tdi -v $(pwd):/home/workspace --gpus all --shm-size 1g --name fast-
reid {image name}
```

Compile with cython to accelerate evalution

```
cd fastreid/evaluation/rank_cylib
make all
```

Training on custom dataset

Prepare custom dataset (Step 1)

Dataset hierarchy

```
custom-data
   |- bounding_box_test
   |- bounding_box_train
   |- query
```

- bounding_box_test : Used for testing.
- bounding_box_train : Used for training.
- query : The query images. Each of them is from different identities in different cameras.

Naming rule of the data

Example :

> 0002_c1_f0044158.jpg

- 0002: person's identity
- c1: means the image from Camera 1
- f0044158: is the 46985th frame in the video of Camera 1

Register custom dataset (Step 2)

When use the custom dataset to do training, you need to tell fastreid framework how to obtain your dataset.

Here is an example script for registering a custom dataset located at `customDataset/register/custom_example.py`, you can use it directly or as a reference to create your own register class.

```python
import glob
import os.path as osp
import re
from .bases import ImageDataset
from ..datasets import DATASET_REGISTRY

@DATASET_REGISTRY.register()
class CustomDataset(ImageDataset):
```

```
    """Custom dataset
    This dataset register class is used for training on custom dataset.
    The dataset hierarchy is mandatory as below:
        -custom-data
            |-bounding_box_test
            |-bounding_box_train
            |-query
    Naming rule of the data:
        0002_c1_f0044158.jpg
        - 0002: label
        - c1: camera number
        - f0044158: the 0044158th frame in the camera c1

    The parameter 'camera_count' means the number of the camera in your dataset,

    in default, the parameter value is set to 8, please specify the actual value
according to your dataset.

    To make the register effactive, please import the class in train_net.py.
    """

    camera_count = 8
    dataset_dir = 'custom-data'
    dataset_name = "custom"

    def __init__(self, root='datasets', **kwargs):
        self.root = root
        self.dataset_dir = osp.join(self.root, self.dataset_dir)
        self.train_dir = osp.join(self.dataset_dir, 'bounding_box_train')
        self.query_dir = osp.join(self.dataset_dir, 'query')
        self.gallery_dir = osp.join(self.dataset_dir, 'bounding_box_test')

        required_files = [
            self.dataset_dir,
            self.train_dir,
            self.query_dir,
            self.gallery_dir,
        ]
        self.check_before_run(required_files)

        train = self.process_dir(self.train_dir)
        query = self.process_dir(self.query_dir, is_train=False)
        gallery = self.process_dir(self.gallery_dir, is_train=False)

        super(CustomDataset, self).__init__(train, query, gallery, **kwargs)

    def process_dir(self, dir_path, is_train=True):
        img_paths = glob.glob(osp.join(dir_path, '*.jpg'))
        pattern = re.compile(r'([-\d]+)_c(\d)')

        data = []
```

```
        for img_path in img_paths:
            pid, camid = map(int, pattern.search(img_path).groups())
            assert 1 <= camid <= self.camera_count
            camid -= 1  # index starts from 0
            if is_train:
                pid = self.dataset_name + "_" + str(pid)
                camid = self.dataset_name + "_" + str(camid)
            data.append((img_path, pid, camid))
        return data
```

For more information please reference How to train Custom Dataset

Import dataset (Step 3)

Aftre registering the custom dataset, you need to import it in `tools/train_net.py` to make it effective.

```
# use the example class above
from customDataset.register.custom_example import CustomDataset
```

Customize your training configuration (Step 4)

The fastreid provides 4 kind of training configuration including BoT, AGW, MGN and SBS. You can use one of these as a base configuration and then extend your own configuration.

The experimental results of each method please see : FastReID Model Zoo and Baselines

Example:
`configs/custom_config/bagtricks_R50.yml` demonstrates how to set up the configuration based on BoT_R50 for the custom dataset.

```
_BASE_: ../Base-bagtricks.yml

DATASETS:
  NAMES: ("CustomDataset",) # the class name of the custom register class
  TESTS: ("CustomDataset",)

OUTPUT_DIR: logs/custom/bagtricks_R50
```

Training & Evaluation (Step 5)

- Training

```
python3 tools/train_net.py
        --config-file /path/to/your/config_file
        MODEL.DEVICE "cuda:0"
```

The directory hierarchy will be:

```
example:
        market1501
                |-bagtricks_R50
                        |-config.yaml
                        |-model_best.pth
                        |-log.txt
```

PS. In the rest of this document, `/path/to/your/model/config_file` means config.yaml here.

- Evaluation

```
python3 tools/train_net.py
        --config-file /path/to/your/model/config_file
        --eval-only
        MODEL.WEIGHTS /path/to/your/model_file
        MODEL.DEVICE "cuda:0"
```

For more infomation please see : Training & Evaluation

Training configuration

- use pre-train model

```
python3 tools/train_net.py
        --config-file /path/to/your/model/config_file
        MODEL.BACKBONE.PRETRAIN_PATH /path/to/pre-train_model
```

The official pre-train model can download from FastReID Model Zoo and Baselines

- change feature dimension

```
python3 tools/train_net.py
        --config-file ./configs/custom_config/bagtricks_R50.yml
        MODEL.HEADS.EMBEDDING_DIM 2048 # the dimension you want
```

- training on GPU

```
python3 tools/train_net.py
        --config-file ./configs/custom_config/bagtricks_R50.yml
        MODEL.DEVICE "cuda:0" # cuda:{gpu_id}
```

- training on CPU

```
python3 tools/train_net.py
        --config-file ./configs/custom_config/bagtricks_R50.yml
        MODEL.DEVICE "cpu"
```

- use multiple gpus

```
python3 tools/train_net.py
        --config-file ./configs/custom_config/bagtricks_R50.yml
        --num-gpus 4
```

Visualization

```
python demo/visualize_result.py
        --config-file /path/to/your/model/config_file
        --parallel
        --vis-label
        --dataset-name your_data_register_class_name
        --output /path/to/output
        --opts MODEL.WEIGHTS /path/to/your/model_file
```

Blue : False recognition
Red : Corrct recognition

Convert to TorchScript Model

Currently, Vaidio used the javacpp pytorch bind to support pytorch, which only allow the model with 'torchscript' format. You can use `torchscript_tools/pth_to_torchscript.py` to convert the model directly.

```
python3 torchscript_tools/pth_to_torchscript.py
        --config-file /path/to/your/model/config_file
        --image /path/to/image
        --output /path/to/output
        --opts MODEL.WEIGHTS /path/to/your/model_file
```

- --image: The script uses the TRACE mode to convert the model which needs to provide an example input to trace the forward propagation of the model. Based on that requirement, please provide an image that comes from your training dataset.

TorchScript Model Evaluation

- Visualization

```
python torchscript_tools/torchscript_visuallize.py
        --config-file /path/to/your/model/config_file
        --vis-label
        --dataset-name your_data_register_class_name
        --output /path/to/output
        --opts MODEL.WEIGHTS /path/to/your/ts_model_file
```

- Evaluation

```
python3 torchscript_tools/torchscript_eval.py
        --config-file /path/to/your/model/config_file
        MODEL.WEIGHTS /path/to/your/ts_model_file
```

Appendix

- Market1501 dataset : http://188.138.127.15:81/Datasets/Market-1501-v15.09.15.zip
- DukeMTMC : https://drive.google.com/open?id=1jjE85dRCMOgRtvJ5RQV9-Afs-2_5dY3O
- Fast-reid : https://github.com/JDAI-CV/fast-reid