

Компьютерное зрение

Практический курс
Савельева Юлия Олеговна
i.o.saveleva.kpfu@gmail.com
2-й семестр, 19.02.2022 г.



Организационная информация

- **Оценки за практику:**

<https://docs.google.com/spreadsheets/d/1YELGWuu07wWdp0mi-9N3UWZHrboDgtx00ogU8sUDQ78/edit?usp=sharing>

- **Ссылка на материалы курса:**

https://github.com/IuliiaSaveleva/Computer_Vision_course_students_2022

- **Мой username на Bitbucket:**

IuliiaSaveleva

Image Retrieval

Stanford Online Products

1) Скачать датасет **Stanford Online Products** (SOP):

https://cvgl.stanford.edu/projects/lifted_struct/

Структура датасета:

SOP/<class_name>_final/<product_id>_<img_no>.JPG

2) Перестроить датасет на основе pickle файла разбиения на train, valid и test ([Practice1/SOP_train_valid_test_split.pickle](#) на github).

Структура пикла:

```
▼ ■ 'bicycle_final' (4586555248) = {dict} {'train': {'product_labels': ['1_331601205444', '1_331601205444', '1_331601205444'], '__len__': {int} 3}
▼ ■ 'test' (4584425264) = {dict} {'product_labels': ['1_151395559989', '1_151395559989', '1_151395559989'], 'category_labels': ['1_151395559989', '1_151395559989', '1_151395559989'], 'paths': ['1_151395559989', '1_151395559989', '1_151395559989'], 'product_labels': ['1_151395559989', '1_151395559989', '1_151395559989'], '__len__': {int} 3}
▶ ■ 'category_labels' (4581356848) = {list} <class 'list'>: <Too big to print. Len: 1672>
▶ ■ 'paths' (4462522528) = {list} <class 'list'>: <Too big to print. Len: 1672>
▶ ■ 'product_labels' (4581267248) = {list} <class 'list'>: <Too big to print. Len: 1672>
▶ ■ 'train' (4462522416) = {dict} {'product_labels': ['1_331601205444', '1_331601205444', '1_331601205444'], 'category_labels': ['1_331601205444', '1_331601205444', '1_331601205444'], 'paths': ['1_331601205444', '1_331601205444', '1_331601205444'], 'product_labels': ['1_331601205444', '1_331601205444', '1_331601205444'], '__len__': {int} 3}
▶ ■ 'valid' (4581227800) = {dict} {'product_labels': ['1_252038663284', '1_252038663284', '1_252038663284'], 'category_labels': ['1_252038663284', '1_252038663284', '1_252038663284'], 'paths': ['1_252038663284', '1_252038663284', '1_252038663284'], 'product_labels': ['1_252038663284', '1_252038663284', '1_252038663284'], '__len__': {int} 3}
▶ ■ 'cabinet_final' (4462031216) = {dict} {'train': {'product_labels': ['0_191643174063', '0_191643174063', '0_191643174063'], '__len__': {int} 3}}
▶ ■ 'chair_final' (4595755440) = {dict} {'train': {'product_labels': ['2_371198230125', '2_371198230125', '2_371198230125'], '__len__': {int} 3}}
```

Image Retrieval

Stanford Online Products

▼ `'paths' (4462522528) = {list} <class 'list'>: <Too big to print. Len: 1672>`

`__len__ = {int} 1672`

`0000 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_10.JPG`

`0001 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_4.JPG`

`0002 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_7.JPG`

`0003 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_1.JPG`

`0004 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_0.JPG`

`0005 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_2.JPG`

`0006 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_5.JPG`

`0007 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_8.JPG`

`0008 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_6.JPG`

`0009 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_9.JPG`

`0010 = {str_} /Stanford_Online_Products/bicycle_final/151395559989_3.JPG`

`0011 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_6.JPG`

`0012 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_2.JPG`

`0013 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_7.JPG`

`0014 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_10.JPG`

`0015 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_8.JPG`

`0016 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_3.JPG`

`0017 = {str_} /Stanford_Online_Products/bicycle_final/400967143545_4.JPG`

▼ `'product_labels' (4581267248) = {lis`

`__len__ = {int} 1672`

`0000 = {str} '1_151395559989'`

`0001 = {str} '1_151395559989'`

`0002 = {str} '1_151395559989'`

`0003 = {str} '1_151395559989'`

`0004 = {str} '1_151395559989'`

`0005 = {str} '1_151395559989'`

`0006 = {str} '1_151395559989'`

`0007 = {str} '1_151395559989'`

`0008 = {str} '1_151395559989'`

`0009 = {str} '1_151395559989'`

`0010 = {str} '1_151395559989'`

`0011 = {str} '1_400967143545'`

`0012 = {str} '1_400967143545'`

`0013 = {str} '1_400967143545'`

`0014 = {str} '1_400967143545'`

`0015 = {str} '1_400967143545'`

`0016 = {str} '1_400967143545'`

`0017 = {str} '1_400967143545'`

Сформировать датасет со структурой:

SOP_retrieval/<set>/<product_label>/<class_name>/<class_name>_<product_id>_<img_no>.JPG

Например: SOP_retrieval/test/1_151395559989/bicycle_final/bicycle_final_151395559989_10.JPG

Image Retrieval

Архитектура

3) Забрать код ResNet:

<https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py>

Мы будем использовать ResNet50, предобученный на ImageNet, но необходимо предусмотреть возможность подменять последний полносвязный слой на слой с нужным нам количеством классов, который будет случайно инициализирован.

```
def _resnet(
    arch: str,
    block: Type[Union[BasicBlock, Bottleneck]],
    layers: List[int],
    pretrained: bool,
    progress: bool,
    **kwargs: Any
) -> ResNet:
    model = ResNet(block, layers, **kwargs)
    if pretrained:
        state_dict = load_state_dict_from_url(model_urls[arch],
                                              progress=progress)
        model.load_state_dict(state_dict)
    return model
```

Можно модифицировать метод `_resnet()` так, чтобы при выполнении условия *pretrained* из *state_dict* с предобученными весами удалялась информация о весах для тех слоев, чьи размеры с размерами слоя текущей модели не совпадают:

```
model_dict = model.state_dict()
state_dict = load_state_dict_from_url(...)
state_dict = {k: v for k, v in
state_dict.items() if k in model_dict and
v.size() == model_dict[k].size()}
```


Image Retrieval

Препроцессинг

from torchvision **import** transforms, datasets

4) Использовать следующий препроцессинг на этапе тестирования сети:

https://pytorch.org/hub/pytorch_vision_resnet/

```
preprocess = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

На этапе обучения необходимо подменить 2 первые операции:

```
transforms.RandomResizedCrop(224),
transforms.RandomHorizontalFlip()
```

5) Создайте обучающий датасет:

datasets.ImageFolder(**root**=.../SOP_retrieval/train, **transform**=train_preprocess)

Дедлайн 26.02.2022 00:00