Tutorial 1: Image Classification

In the tutorial, you will learn

- how to use mmclassification to classify images
- how to train a flowers classifier in practice
- how to evaluate your model

Step 1. Installation

The enviroument for the classification is as follow Run on the RTX 2080 cuda=='11.0' python=='3.7' pytorch=='1,7,1' torchvision=='0.8.2' mmcv=='1.3.14' mmcls=='0.15.0'

Note that the conda enviroument named 'open-mmlab' has already been installed before, so you may not need to install it by yourself on this node. The following shows how to install mmclassification from scratch. We Create a conda virtual environment and activate it.

```
conda create -n open-mmlab python=3.7 -y
conda activate open-mmlab
conda install pytorch cudatoolkit=11.0 torchvision -c pytorch
```

Install mmcv

```
pip install mmcv
```

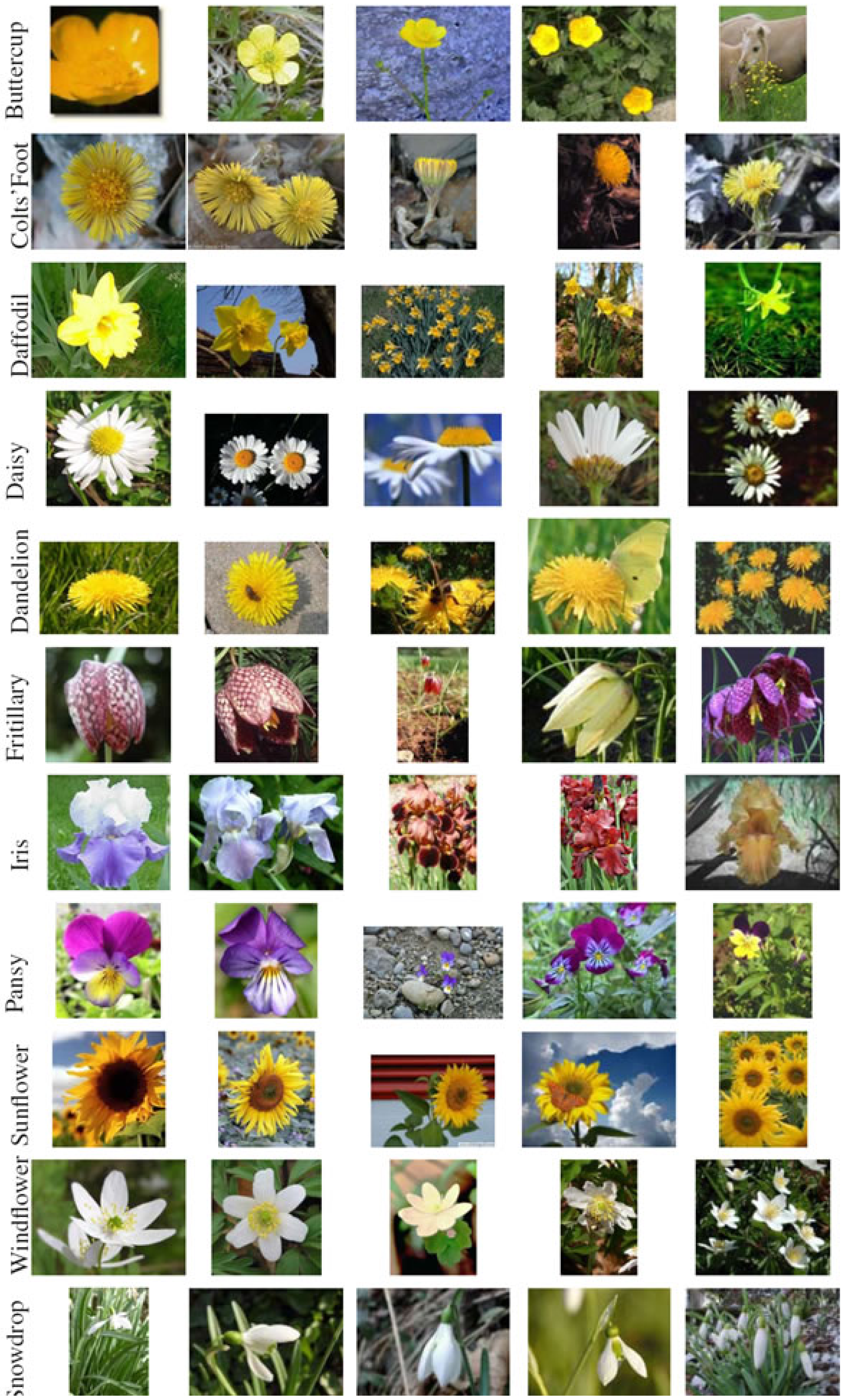
Install build requirements and install MMClassification.

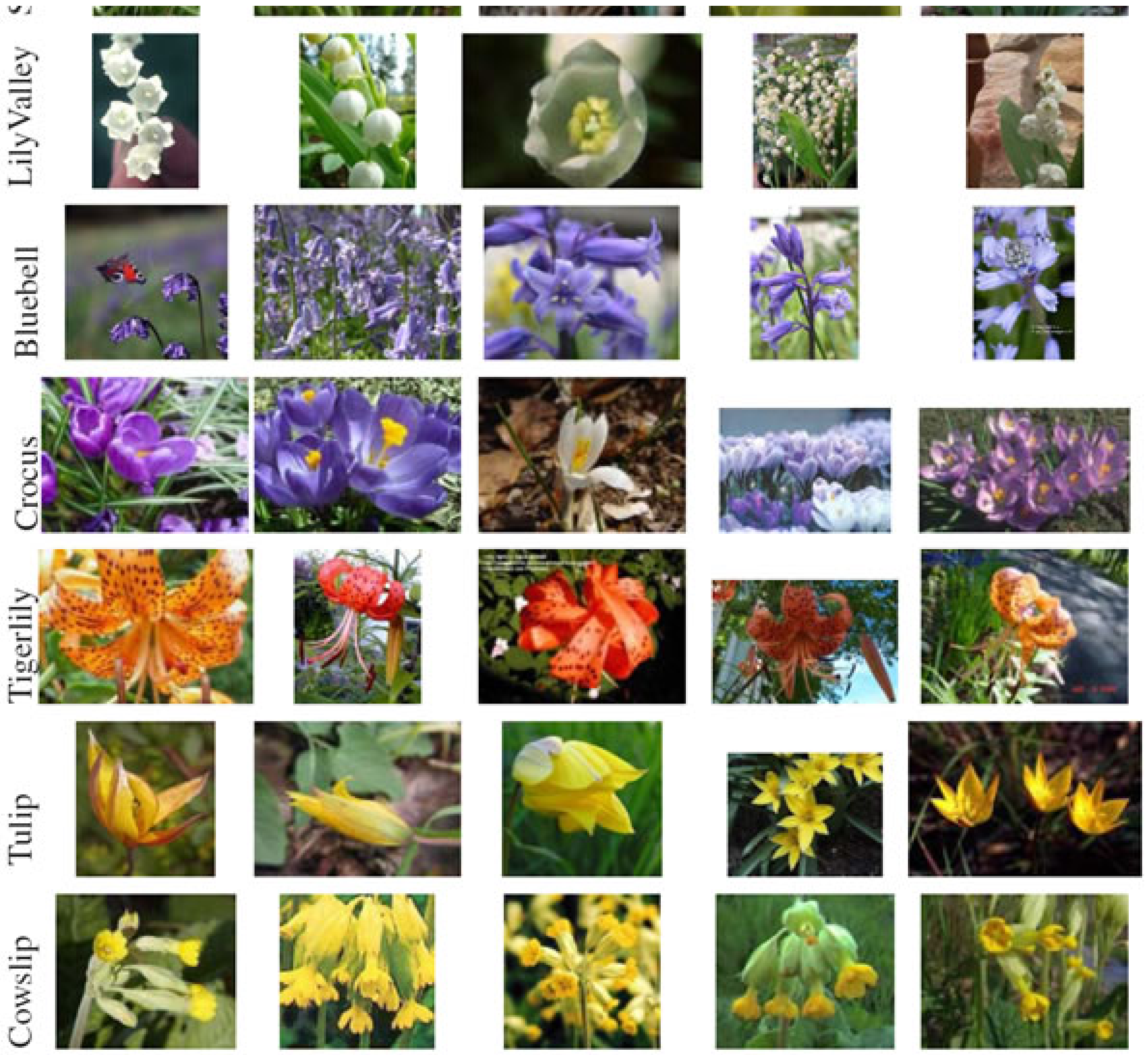
```
git clone https://github.com/open-mmlab/mmclassification.git
cd mmclassification
pip install -e .  # or "python setup.py develop"
```

If you run 'python' and 'import mmcls' without returning error messages, then the repo has been installed successfully!

Step 2. Get data and prepare it before setting config

We get the data from https://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html (https://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html). It is a task of flower classification with 1360 images totally. There are 17 categories, and each category has 80 images. Here is the data example. From the example, we can observe that some flowers are not very easy classified easily by human eyes.


Buttercup
Colts'Foot
Daffodil
Daisy
Dandelion
Fritillary
Iris
Pansy
Sunflower
Windflower
Snowdrop



Download data

```
wget https://www.robots.ox.ac.uk/~vgg/data/flowers/17/17flowers.tgz
tar zxvf 17flowers.tgz
mkdir data
mv 17flowers data/flowers
```

rename the '17flowers' to 'flowers', and put the directory into `mmclassification/data/flowers`

Split data

To fine-tuning on a customized dataset (i.e flowers), the fastest way is to follow the configuration of [Imagenet (https://github.com/open-mmlab/mmclassification/blob/master/docs/tutorials/finetune.md)](https://github.com/open-mmlab/mmclassification/blob/master/docs/tutorials/finetune.md).

We need to split the data into training set and validation set. The directory should be like that

```
flowers
     |--train
          |--class_0
                |--image_xxxx.jpg
                |--image_xxxx.jpg
          |--class_1
                |--image_xxxx.jpg
                |--image_xxxx.jpg
          ...
          |--class_16
                |--image_xxxx.jpg
                |--image_xxxx.jpg
     |--val
          |--class_0
                |--image_xxxx.jpg
                |--image_xxxx.jpg
          |--class_1
                |--image_xxxx.jpg
                |--image_xxxx.jpg
          ...
          |--class_16
                |--image_xxxx.jpg
                |--image_xxxx.jpg
    |--meta
        |--train.txt
        |--val.txt
```

We first split the data to the format above. Now the directory is `mmclassification/data/flowers`

```
python split.py
```

Then, we need to generate **meta files** for both training data and validation data, to describle (image_path, image_class), which will be used in the config files. The example of a line in the meta file is: 'class_0/image_0005.jpg 0' 'class_2/image_0195.jpg 2'

We generate the meta files by

```
mkdir meta
python generate_meta.py
```

In `mmclassification/mmcls` , we follow the file 'imagenet.py', which writes the dataset class of Imagenet, to write a dataset class file 'flowers.py'. Since we follow the data directory of Imagenet above, we just need to copy 'imagenet.py' and re-write the **CLASSES** here.

```
@DATASETS.register_module()
class Flowers(BaseDataset):

    IMG_EXTENSIONS = ('.jpg', '.jpeg', '.png', '.ppm', '.bmp', '.pg
m', '.tif')
    CLASSES = [
        'daffodil', 'snowdrop', 'lilyValley', 'bluebell', 'crocys',
'iris', 'tigerlily', 'tulip', 'fritillary', 'sunflower', 'daisy',
'colts foot', 'dandelion', 'cowslip', 'buttercup', 'wind flower',
'pansy'
    ]

    def load_annotations(self):
        if self.ann_file is None:
            folder_to_idx = find_folders(self.data_prefix)
            samples = get_samples(
                self.data_prefix,
                folder_to_idx,
                extensions=self.IMG_EXTENSIONS)
            if len(samples) == 0:
                raise (RuntimeError('Found 0 files in subfolders o
f: '
                                    f'{self.data_prefix}. '
                                    'Supported extensions are: '
                                    f'{",".join(self.IMG_EXTENSIONS
)}'))

            self.folder_to_idx = folder_to_idx
        elif isinstance(self.ann_file, str):
            with open(self.ann_file) as f:
                samples = [x.strip().rsplit(' ', 1) for x in f.read
lines()]
        else:
            raise TypeError('ann_file must be a str or None')
        self.samples = samples

        data_infos = []
        for filename, gt_label in self.samples:
            info = {'img_prefix': self.data_prefix}
            info['img_info'] = {'filename': filename}
            info['gt_label'] = np.array(gt_label, dtype=np.int64)
            data_infos.append(info)
        return data_infos
```

We also need to import 'Flower' class in the '_*init*_.py'

Step 3. Set config files

Now we are at 'mmclassification/configs/*base*' directory

set data config

a. In `_base_/datasets` , we add `flowers_bs32.py` to describle the basic config about data. Note that we need to claim the data meta path in it like that

```
data = dict(
samples_per_gpu=32,
workers_per_gpu=1,
train=dict(
    type=dataset_type,
    data_prefix='data/flowers/train',
    ann_file='data/flowers/meta/train.txt',
    pipeline=train_pipeline),
val=dict(
    type=dataset_type,
    data_prefix='data/flowers/val',
    ann_file='data/flowers/meta/val.txt',
    pipeline=test_pipeline),
test=dict(
    # replace `data/val` with `data/test` for standard test
    type=dataset_type,
    data_prefix='data/flowers/val',
    ann_file='data/flowers/meta/val.txt',
    pipeline=test_pipeline))
evaluation = dict(interval=1, metric='accuracy')
```

set model config

b. In `_base_/models` , we add `resnet18_flowers.py` to describle the basic config about model.

```
type='ImageClassifier',
backbone=dict(
    type='ResNet',
    depth=18,
    num_stages=4,
    out_indices=(3, ),
    style='pytorch'),
neck=dict(type='GlobalAveragePooling'),
head=dict(
    type='LinearClsHead',
    num_classes=17,
    in_channels=512,
    loss=dict(type='CrossEntropyLoss', loss_weight=1.0),
))
```

Note we use the 'resnet-18' model, and set 'num_classes' as 17 since the number of flower categories is 17.

set training config

c. In `_base_/schedules` , we add `flowers_bs32.py` to describle the basic config about schedule. Here, we set the SGD optimizer with initial learning rate 0.02, and step learning scheduler, which decreases the learning rate (default 10x smaller) at 100th epoch and 150th epoch only.

```
    # optimizer, modified from cifar10_bs128.py
    optimizer = dict(type='SGD', lr=0.02, momentum=0.9, weight_deca
y=0.0001)
    optimizer_config = dict(grad_clip=None)
    # learning policy
    lr_config = dict(policy='step', step=[100, 150])
    runner = dict(type='EpochBasedRunner', max_epochs=200)
```

set saving config

d. In the `_base_/default_runtime.py` , we set the config about checkpoint saving and log file.

```
    # checkpoint saving
    checkpoint_config = dict(interval=1)
    # yapf:disable
    log_config = dict(
        interval=100,
        hooks=[
            dict(type='TextLoggerHook'),
            # dict(type='TensorboardLoggerHook')
        ])
    # yapf:enable

    dist_params = dict(backend='nccl')
    log_level = 'INFO'
    load_from = None
    resume_from = None
    workflow = [('train', 1)]
```

And then, we new a config file in `mmclassification/configs/resnet/resnet18_flowers_bs128.py` we add these config .py files into this file as

```
_base_ = [
    '../_base_/models/resnet18_flowers.py', '../_base_/datasets/flo
wers_bs32.py',
    '../_base_/schedules/flowers_bs32.py', '../_base_/default_runti
me.py'
]
```

Step 4. train

In the directory of 'mmclassification', we run

```
python tools/train.py \
  --config 'configs/resnet/resnet18_flowers_bs128.py' \
  --work_dir 'output/resnet18_flowers_bs128'
```

NOTE: should run in the 'open-mmlab' conda environment!

Here we need to indicate the `config` 'configs/resnet/resnet18_flowers_bs128.py', and `work_dir` is used for model and log file saving. In this case, the trained model and log are saved in `output/resnet18_flowers_bs128` . Now we change directory to `output/resnet18_flowers_bs128` , we are supposed to get saved model and log files. Example log is as follow, the classifiation accuacy arrives 84.7% at 200th epoch.

```
2021-09-25 21:45:52,472 - mmcls - INFO - Environment info:
------------------------------------------------------------
sys.platform: linux
Python: 3.7.11 (default, Jul 27 2021, 14:32:16) [GCC 7.5.0]
CUDA available: True
GPU 0: GeForce RTX 2080 Ti
CUDA_HOME: /usr/local/cuda
NVCC: Build cuda_11.0_bu.TC445_37.28845127_0
GCC: gcc (Ubuntu 7.5.0-3ubuntu1~18.04) 7.5.0
PyTorch: 1.7.1
PyTorch compiling details: PyTorch built with:
  - GCC 7.3
  - C++ Version: 201402
  - Intel(R) oneAPI Math Kernel Library Version 2021.3-Product Build
20210617 for Intel(R) 64 architecture applications
  - Intel(R) MKL-DNN v1.6.0 (Git Hash
5ef631a030a6f73131c7789204104280 5a06064f)
  - OpenMP 201511 (a.k.a. OpenMP 4.5)
  - NNPACK is enabled
  - CPU capability usage: AVX2
  - CUDA Runtime 11.0
  - NVCC architecture flags: -gencode;arch=compute_37,code=sm_37;-
gencode;arch=compute_50,code=sm_50;-gencode;arch=compute_60,code=sm_60;-
gencode;arch=compute_61,code=sm_61;-gencode;arch=compute_70,code=sm_70;-
gencode;arch=compute_75,code=sm_75;-gencode;arch=compute_80,code=sm_80;-
gencode;arch=compute_37,code=compute_37
  - CuDNN 8.0.5
  - Magma 2.5.2
  - Build settings: BLAS=MKL, BUILD_TYPE=Release, CXX_FLAGS= -Wno-
deprecated -fvisibility-inlines-hidden -DUSE_PTHREADPOOL -fopenmp -DNDEBUG
-DUSE_FBGEMM -DUSE_QNNPACK -DUSE_PYTORCH_QNNPACK -DUSE_XNNPACK -
DUSE_VULKAN_WRAPPER -O2 -fPIC -Wno-narrowing -Wall -Wextra -Werror=return-
type -Wno-missing-field-initializers -Wno-type-limits -Wno-array-bounds -
Wno-unknown-pragmas -Wno-sign-compare -Wno-unused-parameter -Wno-unused-
variable -Wno-unused-function -Wno-unused-result -Wno-unused-local-
typedefs -Wno-strict-overflow -Wno-strict-aliasing -Wno-error=deprecated-
declarations -Wno-stringop-overflow -Wno-psabi -Wno-error=pedantic -Wno-
error=redundant-decls -Wno-error=old-style-cast -fdiagnostics-color=always
-faligned-new -Wno-unused-but-set-variable -Wno-maybe-uninitialized -fno-
math-errno -fno-trapping-math -Werror=format -Wno-stringop-overflow,
PERF_WITH_AVX=1, PERF_WITH_AVX2=1, PERF_WITH_AVX512=1, USE_CUDA=ON,
USE_EXCEPTION_PTR=1, USE_GFLAGS=OFF, USE_GLOG=OFF, USE_MKL=ON,
USE_MKLDNN=ON, USE_MPI=OFF, USE_NCCL=ON, USE_NNPACK=ON, USE_OPENMP=ON,

TorchVision: 0.8.2
OpenCV: 4.5.3
MMCV: 1.3.14
MMCV Compiler: n/a
MMCV CUDA Compiler: n/a
MMClassification: 0.15.0+617932d
------------------------------------------------------------

2021-09-25 21:45:52,473 - mmcls - INFO - Distributed training: False
2021-09-25 21:45:52,657 - mmcls - INFO - Config:
model = dict(
    type='ImageClassifier',
    backbone=dict(
        type='ResNet',
        depth=18,
        num_stages=4,
        out_indices=(3, ),
        style='pytorch'),
    neck=dict(type='GlobalAveragePooling'),
    head=dict(
        type='LinearClsHead',
        num_classes=17,
        in_channels=512,
        loss=dict(type='CrossEntropyLoss', loss_weight=1.0)))
dataset_type = 'Flowers'
img_norm_cfg = dict(
    mean=[123.675, 116.28, 103.53], std=[58.395, 57.12, 57.375],
to_rgb=True)
train_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(type='RandomResizedCrop', size=224),
    dict(type='RandomFlip', flip_prob=0.5, direction='horizontal'),
    dict(
        type='Normalize',
        mean=[123.675, 116.28, 103.53],
        std=[58.395, 57.12, 57.375],
        to_rgb=True),
    dict(type='ImageToTensor', keys=['img']),
    dict(type='ToTensor', keys=['gt_label']),
    dict(type='Collect', keys=['img', 'gt_label'])
]
```

```
test_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(type='Resize', size=(256, -1)),
    dict(type='CenterCrop', crop_size=224),
    dict(
        type='Normalize',
        mean=[123.675, 116.28, 103.53],
        std=[58.395, 57.12, 57.375],
        to_rgb=True),
    dict(type='ImageToTensor', keys=['img']),
    dict(type='Collect', keys=['img'])
]
data = dict(
    samples_per_gpu=32,
    workers_per_gpu=1,
    train=dict(
        type='Flowers',
        data_prefix='data/flowers/train',
        ann_file='data/flowers/meta/train.txt',
        pipeline=[
            dict(type='LoadImageFromFile'),
            dict(type='RandomResizedCrop', size=224),
            dict(type='RandomFlip', flip_prob=0.5,
direction='horizontal'),
            dict(
                type='Normalize',
                mean=[123.675, 116.28, 103.53],
                std=[58.395, 57.12, 57.375],
                to_rgb=True),
            dict(type='ImageToTensor', keys=['img']),
            dict(type='ToTensor', keys=['gt_label']),
            dict(type='Collect', keys=['img', 'gt_label'])
        ]),
    val=dict(
        type='Flowers',
        data_prefix='data/flowers/val',
        ann_file='data/flowers/meta/val.txt',
        pipeline=[
            dict(type='LoadImageFromFile'),
            dict(type='Resize', size=(256, -1)),
            dict(type='CenterCrop', crop_size=224),
            dict(
                type='Normalize',
                mean=[123.675, 116.28, 103.53],
                std=[58.395, 57.12, 57.375],
                to_rgb=True),
            dict(type='ImageToTensor', keys=['img']),
            dict(type='Collect', keys=['img'])
        ]),
    test=dict(
        type='Flowers',
        data_prefix='data/flowers/val',
        ann_file='data/flowers/meta/val.txt',
        pipeline=[
            dict(type='LoadImageFromFile'),
            dict(type='Resize', size=(256, -1)),
            dict(type='CenterCrop', crop_size=224),
            dict(
                type='Normalize',
                mean=[123.675, 116.28, 103.53],
                std=[58.395, 57.12, 57.375],
                to_rgb=True),
            dict(type='ImageToTensor', keys=['img']),
            dict(type='Collect', keys=['img'])
        ]))
evaluation = dict(interval=1, metric='accuracy')
optimizer = dict(type='SGD', lr=0.02, momentum=0.9, weight_decay=0.0001)
optimizer_config = dict(grad_clip=None)
lr_config = dict(policy='step', step=[100, 150])
runner = dict(type='EpochBasedRunner', max_epochs=200)
checkpoint_config = dict(interval=1)
log_config = dict(interval=100, hooks=[dict(type='TextLoggerHook')])
dist_params = dict(backend='nccl')
log_level = 'INFO'
load_from = None
resume_from = None
workflow = [('train', 1)]
work_dir = 'output/resnet18_flowers_bs128'
gpu_ids = range(0, 1)

2021-09-25 21:45:52,793 - mmcls - INFO - initialize ResNet with init_cfg
[{'type': 'Kaiming', 'layer': ['Conv2d']}, {'type': 'Constant', 'val': 1,
'layer': ['_BatchNorm', 'GroupNorm']}]
2021-09-25 21:45:52,957 - mmcls - INFO - initialize LinearClsHead with
init_cfg {'type': 'Normal', 'layer': 'Linear', 'std': 0.01}
Name of parameter - Initialization information

backbone.conv1.weight - torch.Size([64, 3, 7, 7]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.bn1.weight - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.bn1.bias - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer1.0.conv1.weight - torch.Size([64, 64, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer1.0.bn1.weight - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer1.0.bn1.bias - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer1.0.conv2.weight - torch.Size([64, 64, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0
```

```
backbone.layer1.0.bn2.weight - torch.Size([64]):
Initialized by user-defined `init_weights` in ResNet

backbone.layer1.0.bn2.bias - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer1.1.conv1.weight - torch.Size([64, 64, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer1.1.bn1.weight - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer1.1.bn1.bias - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer1.1.conv2.weight - torch.Size([64, 64, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer1.1.bn2.weight - torch.Size([64]):
Initialized by user-defined `init_weights` in ResNet

backbone.layer1.1.bn2.bias - torch.Size([64]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer2.0.conv1.weight - torch.Size([128, 64, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer2.0.bn1.weight - torch.Size([128]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer2.0.bn1.bias - torch.Size([128]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer2.0.conv2.weight - torch.Size([128, 128, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer2.0.bn2.weight - torch.Size([128]):
Initialized by user-defined `init_weights` in ResNet

backbone.layer2.0.bn2.bias - torch.Size([128]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer2.0.downsample.0.weight - torch.Size([128, 64, 1, 1]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0
```

```
backbone.layer2.0.downsample.1.weight - torch.Size([128]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer2.0.downsample.1.bias - torch.Size([128]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer2.1.conv1.weight - torch.Size([128, 128, 3, 3]): 
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal, 
bias=0 

backbone.layer2.1.bn1.weight - torch.Size([128]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer2.1.bn1.bias - torch.Size([128]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer2.1.conv2.weight - torch.Size([128, 128, 3, 3]): 
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal, 
bias=0 

backbone.layer2.1.bn2.weight - torch.Size([128]): 
Initialized by user-defined `init_weights` in ResNet  

backbone.layer2.1.bn2.bias - torch.Size([128]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer3.0.conv1.weight - torch.Size([256, 128, 3, 3]): 
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal, 
bias=0 

backbone.layer3.0.bn1.weight - torch.Size([256]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer3.0.bn1.bias - torch.Size([256]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer3.0.conv2.weight - torch.Size([256, 256, 3, 3]): 
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal, 
bias=0 

backbone.layer3.0.bn2.weight - torch.Size([256]): 
Initialized by user-defined `init_weights` in ResNet  

backbone.layer3.0.bn2.bias - torch.Size([256]): 
The value is the same before and after calling `init_weights` of 
ImageClassifier  

backbone.layer3.0.downsample.0.weight - torch.Size([256, 128, 1, 1]): 
```

```
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer3.0.downsample.1.weight - torch.Size([256]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer3.0.downsample.1.bias - torch.Size([256]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer3.1.conv1.weight - torch.Size([256, 256, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer3.1.bn1.weight - torch.Size([256]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer3.1.bn1.bias - torch.Size([256]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer3.1.conv2.weight - torch.Size([256, 256, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer3.1.bn2.weight - torch.Size([256]):
Initialized by user-defined `init_weights` in ResNet

backbone.layer3.1.bn2.bias - torch.Size([256]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.0.conv1.weight - torch.Size([512, 256, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer4.0.bn1.weight - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.0.bn1.bias - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.0.conv2.weight - torch.Size([512, 512, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer4.0.bn2.weight - torch.Size([512]):
Initialized by user-defined `init_weights` in ResNet

backbone.layer4.0.bn2.bias - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier
```

```
backbone.layer4.0.downsample.0.weight - torch.Size([512, 256, 1, 1]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer4.0.downsample.1.weight - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.0.downsample.1.bias - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.1.conv1.weight - torch.Size([512, 512, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer4.1.bn1.weight - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.1.bn1.bias - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

backbone.layer4.1.conv2.weight - torch.Size([512, 512, 3, 3]):
KaimingInit: a=0, mode=fan_out, nonlinearity=relu, distribution =normal,
bias=0

backbone.layer4.1.bn2.weight - torch.Size([512]):
Initialized by user-defined `init_weights` in ResNet

backbone.layer4.1.bn2.bias - torch.Size([512]):
The value is the same before and after calling `init_weights` of
ImageClassifier

head.fc.weight - torch.Size([17, 512]):
NormalInit: mean=0, std=0.01, bias=0

head.fc.bias - torch.Size([17]):
NormalInit: mean=0, std=0.01, bias=0
2021-09-25 21:45:55,765 - mmcls - INFO - Start running, host:
u3007305@gpu2-comp-104, work_dir:
/userhome/cs/u3007305/mmclassification/output/resnet18_flowers_bs128
2021-09-25 21:45:55,766 - mmcls - INFO - Hooks will be executed in the
following order:
before_run:
(VERY_HIGH   ) StepLrUpdaterHook
(NORMAL      ) CheckpointHook
(NORMAL      ) EvalHook
(VERY_LOW    ) TextLoggerHook
 --------------------
before_train_epoch:
(VERY_HIGH   ) StepLrUpdaterHook
(NORMAL      ) EvalHook
(LOW         ) IterTimerHook
(VERY_LOW    ) TextLoggerHook
 --------------------
```

```
before_train_iter:
(VERY_HIGH   ) StepLrUpdaterHook
(NORMAL      ) EvalHook
(LOW         ) IterTimerHook
 --------------------
after_train_iter:
(ABOVE_NORMAL) OptimizerHook
(NORMAL      ) CheckpointHook
(NORMAL      ) EvalHook
(LOW         ) IterTimerHook
(VERY_LOW    ) TextLoggerHook
 --------------------
after_train_epoch:
(NORMAL      ) CheckpointHook
(NORMAL      ) EvalHook
(VERY_LOW    ) TextLoggerHook
 --------------------
before_val_epoch:
(LOW         ) IterTimerHook
(VERY_LOW    ) TextLoggerHook
 --------------------
before_val_iter:
(LOW         ) IterTimerHook
 --------------------
after_val_iter:
(LOW         ) IterTimerHook
 --------------------
after_val_epoch:
(VERY_LOW    ) TextLoggerHook
 --------------------
2021-09-25 21:45:55,766 - mmcls - INFO - workflow: [('train', 1)], max:
200 epochs
2021-09-25 21:46:08,232 - mmcls - INFO - Saving checkpoint at 1 epochs
2021-09-25 21:46:10,198 - mmcls - INFO - Epoch(val) [1][6]  accuracy_top-
1: 35.2941, accuracy_top-5: 83.5294
2021-09-25 21:46:22,483 - mmcls - INFO - Saving checkpoint at 2 epochs
2021-09-25 21:46:24,429 - mmcls - INFO - Epoch(val) [2][6]  accuracy_top-
1: 41.1765, accuracy_top-5: 90.0000
2021-09-25 21:46:36,170 - mmcls - INFO - Saving checkpoint at 3 epochs
2021-09-25 21:46:38,110 - mmcls - INFO - Epoch(val) [3][6]  accuracy_top-
1: 41.1765, accuracy_top-5: 90.0000
2021-09-25 21:46:50,259 - mmcls - INFO - Saving checkpoint at 4 epochs
2021-09-25 21:46:52,174 - mmcls - INFO - Epoch(val) [4][6]  accuracy_top-
1: 53.5294, accuracy_top-5: 90.5882
2021-09-25 21:47:03,984 - mmcls - INFO - Saving checkpoint at 5 epochs
2021-09-25 21:47:05,888 - mmcls - INFO - Epoch(val) [5][6]  accuracy_top-
1: 48.8235, accuracy_top-5: 90.0000
2021-09-25 21:47:17,288 - mmcls - INFO - Saving checkpoint at 6 epochs
2021-09-25 21:47:19,108 - mmcls - INFO - Epoch(val) [6][6]  accuracy_top-
1: 52.9412, accuracy_top-5: 91.1765
2021-09-25 21:47:31,301 - mmcls - INFO - Saving checkpoint at 7 epochs
2021-09-25 21:47:33,251 - mmcls - INFO - Epoch(val) [7][6]  accuracy_top-
1: 56.4706, accuracy_top-5: 95.2941
2021-09-25 21:47:45,448 - mmcls - INFO - Saving checkpoint at 8 epochs
2021-09-25 21:47:47,432 - mmcls - INFO - Epoch(val) [8][6]  accuracy_top-
1: 54.7059, accuracy_top-5: 94.1176
2021-09-25 21:47:59,343 - mmcls - INFO - Saving checkpoint at 9 epochs
```

```
2021-09-25 21:48:01,200 - mmcls - INFO - Epoch(val) [9][6]  accuracy_top-
1: 60.0000, accuracy_top-5: 92.9412
2021-09-25 21:48:13,148 - mmcls - INFO - Saving checkpoint at 10 epochs
2021-09-25 21:48:15,086 - mmcls - INFO - Epoch(val) [10][6] accuracy_top-
1: 62.9412, accuracy_top-5: 93.5294
2021-09-25 21:48:27,054 - mmcls - INFO - Saving checkpoint at 11 epochs
2021-09-25 21:48:29,015 - mmcls - INFO - Epoch(val) [11][6] accuracy_top-
1: 54.7059, accuracy_top-5: 91.7647
2021-09-25 21:48:41,061 - mmcls - INFO - Saving checkpoint at 12 epochs
2021-09-25 21:48:43,021 - mmcls - INFO - Epoch(val) [12][6] accuracy_top-
1: 57.0588, accuracy_top-5: 92.9412
2021-09-25 21:48:54,942 - mmcls - INFO - Saving checkpoint at 13 epochs
2021-09-25 21:48:56,878 - mmcls - INFO - Epoch(val) [13][6] accuracy_top-
1: 64.1176, accuracy_top-5: 93.5294
2021-09-25 21:49:08,998 - mmcls - INFO - Saving checkpoint at 14 epochs
2021-09-25 21:49:10,928 - mmcls - INFO - Epoch(val) [14][6] accuracy_top-
1: 67.6471, accuracy_top-5: 97.0588
2021-09-25 21:49:23,051 - mmcls - INFO - Saving checkpoint at 15 epochs
2021-09-25 21:49:24,949 - mmcls - INFO - Epoch(val) [15][6] accuracy_top-
1: 57.0588, accuracy_top-5: 94.7059
2021-09-25 21:49:36,952 - mmcls - INFO - Saving checkpoint at 16 epochs
2021-09-25 21:49:38,900 - mmcls - INFO - Epoch(val) [16][6] accuracy_top-
1: 64.1176, accuracy_top-5: 94.7059
2021-09-25 21:49:50,834 - mmcls - INFO - Saving checkpoint at 17 epochs
2021-09-25 21:49:52,767 - mmcls - INFO - Epoch(val) [17][6] accuracy_top-
1: 58.8235, accuracy_top-5: 95.2941
2021-09-25 21:50:04,928 - mmcls - INFO - Saving checkpoint at 18 epochs
2021-09-25 21:50:06,855 - mmcls - INFO - Epoch(val) [18][6] accuracy_top-
1: 67.0588, accuracy_top-5: 95.2941
2021-09-25 21:50:19,092 - mmcls - INFO - Saving checkpoint at 19 epochs
2021-09-25 21:50:21,057 - mmcls - INFO - Epoch(val) [19][6] accuracy_top-
1: 65.8824, accuracy_top-5: 95.2941
2021-09-25 21:50:33,324 - mmcls - INFO - Saving checkpoint at 20 epochs
2021-09-25 21:50:35,253 - mmcls - INFO - Epoch(val) [20][6] accuracy_top-
1: 62.3529, accuracy_top-5: 95.8824
2021-09-25 21:50:47,544 - mmcls - INFO - Saving checkpoint at 21 epochs
2021-09-25 21:50:49,454 - mmcls - INFO - Epoch(val) [21][6] accuracy_top-
1: 59.4118, accuracy_top-5: 96.4706
2021-09-25 21:51:01,706 - mmcls - INFO - Saving checkpoint at 22 epochs
2021-09-25 21:51:03,587 - mmcls - INFO - Epoch(val) [22][6] accuracy_top-
1: 57.6471, accuracy_top-5: 96.4706
2021-09-25 21:51:15,750 - mmcls - INFO - Saving checkpoint at 23 epochs
2021-09-25 21:51:17,655 - mmcls - INFO - Epoch(val) [23][6] accuracy_top-
1: 62.9412, accuracy_top-5: 95.8824
2021-09-25 21:51:29,859 - mmcls - INFO - Saving checkpoint at 24 epochs
2021-09-25 21:51:31,815 - mmcls - INFO - Epoch(val) [24][6] accuracy_top-
1: 64.1176, accuracy_top-5: 96.4706
2021-09-25 21:51:43,729 - mmcls - INFO - Saving checkpoint at 25 epochs
2021-09-25 21:51:45,594 - mmcls - INFO - Epoch(val) [25][6] accuracy_top-
1: 67.0588, accuracy_top-5: 97.0588
2021-09-25 21:51:57,405 - mmcls - INFO - Saving checkpoint at 26 epochs
2021-09-25 21:51:59,370 - mmcls - INFO - Epoch(val) [26][6] accuracy_top-
1: 57.6471, accuracy_top-5: 94.7059
2021-09-25 21:52:11,365 - mmcls - INFO - Saving checkpoint at 27 epochs
2021-09-25 21:52:13,361 - mmcls - INFO - Epoch(val) [27][6] accuracy_top-
1: 63.5294, accuracy_top-5: 96.4706
2021-09-25 21:52:25,251 - mmcls - INFO - Saving checkpoint at 28 epochs
```

```
2021-09-25 21:52:27,213 - mmcls - INFO - Epoch(val) [28][6] accuracy_top-
1: 67.6471, accuracy_top-5: 95.8824
2021-09-25 21:52:39,201 - mmcls - INFO - Saving checkpoint at 29 epochs
2021-09-25 21:52:41,142 - mmcls - INFO - Epoch(val) [29][6] accuracy_top-
1: 57.0588, accuracy_top-5: 98.2353
2021-09-25 21:52:52,951 - mmcls - INFO - Saving checkpoint at 30 epochs
2021-09-25 21:52:54,910 - mmcls - INFO - Epoch(val) [30][6] accuracy_top-
1: 69.4118, accuracy_top-5: 98.2353
2021-09-25 21:53:06,872 - mmcls - INFO - Saving checkpoint at 31 epochs
2021-09-25 21:53:08,827 - mmcls - INFO - Epoch(val) [31][6] accuracy_top-
1: 66.4706, accuracy_top-5: 97.0588
2021-09-25 21:53:20,673 - mmcls - INFO - Saving checkpoint at 32 epochs
2021-09-25 21:53:22,646 - mmcls - INFO - Epoch(val) [32][6] accuracy_top-
1: 67.0588, accuracy_top-5: 92.9412
2021-09-25 21:53:34,589 - mmcls - INFO - Saving checkpoint at 33 epochs
2021-09-25 21:53:36,518 - mmcls - INFO - Epoch(val) [33][6] accuracy_top-
1: 64.7059, accuracy_top-5: 93.5294
2021-09-25 21:53:48,468 - mmcls - INFO - Saving checkpoint at 34 epochs
2021-09-25 21:53:50,436 - mmcls - INFO - Epoch(val) [34][6] accuracy_top-
1: 68.8235, accuracy_top-5: 97.6471
2021-09-25 21:54:02,537 - mmcls - INFO - Saving checkpoint at 35 epochs
2021-09-25 21:54:04,494 - mmcls - INFO - Epoch(val) [35][6] accuracy_top-
1: 74.7059, accuracy_top-5: 96.4706
2021-09-25 21:54:16,358 - mmcls - INFO - Saving checkpoint at 36 epochs
2021-09-25 21:54:18,330 - mmcls - INFO - Epoch(val) [36][6] accuracy_top-
1: 68.2353, accuracy_top-5: 97.6471
2021-09-25 21:54:30,129 - mmcls - INFO - Saving checkpoint at 37 epochs
2021-09-25 21:54:32,066 - mmcls - INFO - Epoch(val) [37][6] accuracy_top-
1: 68.8235, accuracy_top-5: 97.6471
2021-09-25 21:54:44,300 - mmcls - INFO - Saving checkpoint at 38 epochs
2021-09-25 21:54:46,269 - mmcls - INFO - Epoch(val) [38][6] accuracy_top-
1: 71.7647, accuracy_top-5: 96.4706
2021-09-25 21:54:58,442 - mmcls - INFO - Saving checkpoint at 39 epochs
2021-09-25 21:55:00,373 - mmcls - INFO - Epoch(val) [39][6] accuracy_top-
1: 76.4706, accuracy_top-5: 97.0588
2021-09-25 21:55:12,532 - mmcls - INFO - Saving checkpoint at 40 epochs
2021-09-25 21:55:14,485 - mmcls - INFO - Epoch(val) [40][6] accuracy_top-
1: 73.5294, accuracy_top-5: 96.4706
2021-09-25 21:55:26,588 - mmcls - INFO - Saving checkpoint at 41 epochs
2021-09-25 21:55:28,497 - mmcls - INFO - Epoch(val) [41][6] accuracy_top-
1: 69.4118, accuracy_top-5: 95.8824
2021-09-25 21:55:40,656 - mmcls - INFO - Saving checkpoint at 42 epochs
2021-09-25 21:55:42,561 - mmcls - INFO - Epoch(val) [42][6] accuracy_top-
1: 76.4706, accuracy_top-5: 97.0588
2021-09-25 21:55:54,521 - mmcls - INFO - Saving checkpoint at 43 epochs
2021-09-25 21:55:56,452 - mmcls - INFO - Epoch(val) [43][6] accuracy_top-
1: 70.5882, accuracy_top-5: 98.8235
2021-09-25 21:56:08,588 - mmcls - INFO - Saving checkpoint at 44 epochs
2021-09-25 21:56:10,501 - mmcls - INFO - Epoch(val) [44][6] accuracy_top-
1: 71.7647, accuracy_top-5: 97.0588
2021-09-25 21:56:22,591 - mmcls - INFO - Saving checkpoint at 45 epochs
2021-09-25 21:56:24,557 - mmcls - INFO - Epoch(val) [45][6] accuracy_top-
1: 72.3529, accuracy_top-5: 98.2353
2021-09-25 21:56:36,760 - mmcls - INFO - Saving checkpoint at 46 epochs
2021-09-25 21:56:38,719 - mmcls - INFO - Epoch(val) [46][6] accuracy_top-
1: 70.5882, accuracy_top-5: 98.2353
2021-09-25 21:56:50,930 - mmcls - INFO - Saving checkpoint at 47 epochs
```

```
2021-09-25 21:56:52,854 - mmcls - INFO - Epoch(val) [47][6] accuracy_top-
1: 70.5882, accuracy_top-5: 97.0588
2021-09-25 21:57:04,883 - mmcls - INFO - Saving checkpoint at 48 epochs
2021-09-25 21:57:06,821 - mmcls - INFO - Epoch(val) [48][6] accuracy_top-
1: 70.0000, accuracy_top-5: 98.2353
2021-09-25 21:57:18,946 - mmcls - INFO - Saving checkpoint at 49 epochs
2021-09-25 21:57:20,872 - mmcls - INFO - Epoch(val) [49][6] accuracy_top-
1: 76.4706, accuracy_top-5: 98.2353
2021-09-25 21:57:32,716 - mmcls - INFO - Saving checkpoint at 50 epochs
2021-09-25 21:57:34,586 - mmcls - INFO - Epoch(val) [50][6] accuracy_top-
1: 77.0588, accuracy_top-5: 98.8235
2021-09-25 21:57:46,528 - mmcls - INFO - Saving checkpoint at 51 epochs
2021-09-25 21:57:48,467 - mmcls - INFO - Epoch(val) [51][6] accuracy_top-
1: 74.7059, accuracy_top-5: 98.8235
2021-09-25 21:58:00,559 - mmcls - INFO - Saving checkpoint at 52 epochs
2021-09-25 21:58:02,521 - mmcls - INFO - Epoch(val) [52][6] accuracy_top-
1: 70.5882, accuracy_top-5: 97.6471
2021-09-25 21:58:14,733 - mmcls - INFO - Saving checkpoint at 53 epochs
2021-09-25 21:58:16,681 - mmcls - INFO - Epoch(val) [53][6] accuracy_top-
1: 80.5882, accuracy_top-5: 97.0588
2021-09-25 21:58:28,899 - mmcls - INFO - Saving checkpoint at 54 epochs
2021-09-25 21:58:30,825 - mmcls - INFO - Epoch(val) [54][6] accuracy_top-
1: 74.7059, accuracy_top-5: 96.4706
2021-09-25 21:58:42,736 - mmcls - INFO - Saving checkpoint at 55 epochs
2021-09-25 21:58:44,608 - mmcls - INFO - Epoch(val) [55][6] accuracy_top-
1: 70.5882, accuracy_top-5: 97.0588
2021-09-25 21:58:56,318 - mmcls - INFO - Saving checkpoint at 56 epochs
2021-09-25 21:58:58,222 - mmcls - INFO - Epoch(val) [56][6] accuracy_top-
1: 74.7059, accuracy_top-5: 98.2353
2021-09-25 21:59:10,393 - mmcls - INFO - Saving checkpoint at 57 epochs
2021-09-25 21:59:12,343 - mmcls - INFO - Epoch(val) [57][6] accuracy_top-
1: 72.9412, accuracy_top-5: 98.8235
2021-09-25 21:59:24,531 - mmcls - INFO - Saving checkpoint at 58 epochs
2021-09-25 21:59:26,516 - mmcls - INFO - Epoch(val) [58][6] accuracy_top-
1: 75.2941, accuracy_top-5: 98.2353
2021-09-25 21:59:38,255 - mmcls - INFO - Saving checkpoint at 59 epochs
2021-09-25 21:59:40,180 - mmcls - INFO - Epoch(val) [59][6] accuracy_top-
1: 70.0000, accuracy_top-5: 95.8824
2021-09-25 21:59:51,965 - mmcls - INFO - Saving checkpoint at 60 epochs
2021-09-25 21:59:53,899 - mmcls - INFO - Epoch(val) [60][6] accuracy_top-
1: 74.7059, accuracy_top-5: 97.6471
2021-09-25 22:00:06,119 - mmcls - INFO - Saving checkpoint at 61 epochs
2021-09-25 22:00:08,071 - mmcls - INFO - Epoch(val) [61][6] accuracy_top-
1: 77.0588, accuracy_top-5: 99.4118
2021-09-25 22:00:20,087 - mmcls - INFO - Saving checkpoint at 62 epochs
2021-09-25 22:00:21,970 - mmcls - INFO - Epoch(val) [62][6] accuracy_top-
1: 72.9412, accuracy_top-5: 97.0588
2021-09-25 22:00:33,348 - mmcls - INFO - Saving checkpoint at 63 epochs
2021-09-25 22:00:35,175 - mmcls - INFO - Epoch(val) [63][6] accuracy_top-
1: 77.0588, accuracy_top-5: 97.6471
2021-09-25 22:00:47,007 - mmcls - INFO - Saving checkpoint at 64 epochs
2021-09-25 22:00:48,940 - mmcls - INFO - Epoch(val) [64][6] accuracy_top-
1: 77.0588, accuracy_top-5: 97.6471
2021-09-25 22:01:01,014 - mmcls - INFO - Saving checkpoint at 65 epochs
2021-09-25 22:01:02,923 - mmcls - INFO - Epoch(val) [65][6] accuracy_top-
1: 75.2941, accuracy_top-5: 98.8235
2021-09-25 22:01:14,916 - mmcls - INFO - Saving checkpoint at 66 epochs
```

```
2021-09-25 22:01:16,831 - mmcls - INFO - Epoch(val) [66][6] accuracy_top-
1: 77.0588, accuracy_top-5: 98.8235
2021-09-25 22:01:28,832 - mmcls - INFO - Saving checkpoint at 67 epochs
2021-09-25 22:01:30,664 - mmcls - INFO - Epoch(val) [67][6] accuracy_top-
1: 75.8824, accuracy_top-5: 99.4118
2021-09-25 22:01:42,479 - mmcls - INFO - Saving checkpoint at 68 epochs
2021-09-25 22:01:44,309 - mmcls - INFO - Epoch(val) [68][6] accuracy_top-
1: 79.4118, accuracy_top-5: 98.2353
2021-09-25 22:01:56,558 - mmcls - INFO - Saving checkpoint at 69 epochs
2021-09-25 22:01:58,591 - mmcls - INFO - Epoch(val) [69][6] accuracy_top-
1: 72.3529, accuracy_top-5: 97.6471
2021-09-25 22:02:10,865 - mmcls - INFO - Saving checkpoint at 70 epochs
2021-09-25 22:02:12,804 - mmcls - INFO - Epoch(val) [70][6] accuracy_top-
1: 70.5882, accuracy_top-5: 97.6471
2021-09-25 22:02:25,026 - mmcls - INFO - Saving checkpoint at 71 epochs
2021-09-25 22:02:27,068 - mmcls - INFO - Epoch(val) [71][6] accuracy_top-
1: 77.0588, accuracy_top-5: 97.6471
2021-09-25 22:02:39,197 - mmcls - INFO - Saving checkpoint at 72 epochs
2021-09-25 22:02:41,170 - mmcls - INFO - Epoch(val) [72][6] accuracy_top-
1: 74.7059, accuracy_top-5: 97.6471
2021-09-25 22:02:53,881 - mmcls - INFO - Saving checkpoint at 73 epochs
2021-09-25 22:02:55,915 - mmcls - INFO - Epoch(val) [73][6] accuracy_top-
1: 79.4118, accuracy_top-5: 97.6471
2021-09-25 22:03:08,513 - mmcls - INFO - Saving checkpoint at 74 epochs
2021-09-25 22:03:10,518 - mmcls - INFO - Epoch(val) [74][6] accuracy_top-
1: 80.0000, accuracy_top-5: 98.2353
2021-09-25 22:03:22,891 - mmcls - INFO - Saving checkpoint at 75 epochs
2021-09-25 22:03:24,939 - mmcls - INFO - Epoch(val) [75][6] accuracy_top-
1: 75.8824, accuracy_top-5: 97.6471
2021-09-25 22:03:37,315 - mmcls - INFO - Saving checkpoint at 76 epochs
2021-09-25 22:03:39,350 - mmcls - INFO - Epoch(val) [76][6] accuracy_top-
1: 81.1765, accuracy_top-5: 98.8235
2021-09-25 22:03:51,803 - mmcls - INFO - Saving checkpoint at 77 epochs
2021-09-25 22:03:53,876 - mmcls - INFO - Epoch(val) [77][6] accuracy_top-
1: 78.2353, accuracy_top-5: 98.8235
2021-09-25 22:04:06,471 - mmcls - INFO - Saving checkpoint at 78 epochs
2021-09-25 22:04:08,469 - mmcls - INFO - Epoch(val) [78][6] accuracy_top-
1: 78.8235, accuracy_top-5: 98.8235
2021-09-25 22:04:20,963 - mmcls - INFO - Saving checkpoint at 79 epochs
2021-09-25 22:04:22,962 - mmcls - INFO - Epoch(val) [79][6] accuracy_top-
1: 68.8235, accuracy_top-5: 97.6471
2021-09-25 22:04:35,354 - mmcls - INFO - Saving checkpoint at 80 epochs
2021-09-25 22:04:37,243 - mmcls - INFO - Epoch(val) [80][6] accuracy_top-
1: 71.1765, accuracy_top-5: 97.6471
2021-09-25 22:04:49,733 - mmcls - INFO - Saving checkpoint at 81 epochs
2021-09-25 22:04:51,727 - mmcls - INFO - Epoch(val) [81][6] accuracy_top-
1: 79.4118, accuracy_top-5: 98.8235
2021-09-25 22:05:04,178 - mmcls - INFO - Saving checkpoint at 82 epochs
2021-09-25 22:05:06,200 - mmcls - INFO - Epoch(val) [82][6] accuracy_top-
1: 79.4118, accuracy_top-5: 98.2353
2021-09-25 22:05:18,249 - mmcls - INFO - Saving checkpoint at 83 epochs
2021-09-25 22:05:20,212 - mmcls - INFO - Epoch(val) [83][6] accuracy_top-
1: 80.0000, accuracy_top-5: 99.4118
2021-09-25 22:05:32,725 - mmcls - INFO - Saving checkpoint at 84 epochs
2021-09-25 22:05:34,780 - mmcls - INFO - Epoch(val) [84][6] accuracy_top-
1: 75.8824, accuracy_top-5: 97.0588
2021-09-25 22:05:47,306 - mmcls - INFO - Saving checkpoint at 85 epochs
```

```
2021-09-25 22:05:49,307 - mmcls - INFO - Epoch(val) [85][6] accuracy_top-
1: 75.8824, accuracy_top-5: 98.8235
2021-09-25 22:06:01,933 - mmcls - INFO - Saving checkpoint at 86 epochs
2021-09-25 22:06:03,988 - mmcls - INFO - Epoch(val) [86][6] accuracy_top-
1: 71.1765, accuracy_top-5: 95.8824
2021-09-25 22:06:16,541 - mmcls - INFO - Saving checkpoint at 87 epochs
2021-09-25 22:06:18,573 - mmcls - INFO - Epoch(val) [87][6] accuracy_top-
1: 75.2941, accuracy_top-5: 98.8235
2021-09-25 22:06:31,203 - mmcls - INFO - Saving checkpoint at 88 epochs
2021-09-25 22:06:33,230 - mmcls - INFO - Epoch(val) [88][6] accuracy_top-
1: 78.2353, accuracy_top-5: 97.6471
2021-09-25 22:06:45,356 - mmcls - INFO - Saving checkpoint at 89 epochs
2021-09-25 22:06:47,363 - mmcls - INFO - Epoch(val) [89][6] accuracy_top-
1: 74.1176, accuracy_top-5: 98.2353
2021-09-25 22:06:59,891 - mmcls - INFO - Saving checkpoint at 90 epochs
2021-09-25 22:07:01,895 - mmcls - INFO - Epoch(val) [90][6] accuracy_top-
1: 78.2353, accuracy_top-5: 99.4118
2021-09-25 22:07:14,417 - mmcls - INFO - Saving checkpoint at 91 epochs
2021-09-25 22:07:16,364 - mmcls - INFO - Epoch(val) [91][6] accuracy_top-
1: 77.0588, accuracy_top-5: 99.4118
2021-09-25 22:07:28,880 - mmcls - INFO - Saving checkpoint at 92 epochs
2021-09-25 22:07:30,882 - mmcls - INFO - Epoch(val) [92][6] accuracy_top-
1: 80.0000, accuracy_top-5: 98.2353
2021-09-25 22:07:43,426 - mmcls - INFO - Saving checkpoint at 93 epochs
2021-09-25 22:07:45,386 - mmcls - INFO - Epoch(val) [93][6] accuracy_top-
1: 81.7647, accuracy_top-5: 98.8235
2021-09-25 22:07:58,012 - mmcls - INFO - Saving checkpoint at 94 epochs
2021-09-25 22:07:59,977 - mmcls - INFO - Epoch(val) [94][6] accuracy_top-
1: 78.8235, accuracy_top-5: 98.8235
2021-09-25 22:08:12,163 - mmcls - INFO - Saving checkpoint at 95 epochs
2021-09-25 22:08:14,100 - mmcls - INFO - Epoch(val) [95][6] accuracy_top-
1: 77.6471, accuracy_top-5: 98.8235
2021-09-25 22:08:26,124 - mmcls - INFO - Saving checkpoint at 96 epochs
2021-09-25 22:08:28,093 - mmcls - INFO - Epoch(val) [96][6] accuracy_top-
1: 81.1765, accuracy_top-5: 98.8235
2021-09-25 22:08:40,696 - mmcls - INFO - Saving checkpoint at 97 epochs
2021-09-25 22:08:42,728 - mmcls - INFO - Epoch(val) [97][6] accuracy_top-
1: 78.2353, accuracy_top-5: 98.8235
2021-09-25 22:08:55,280 - mmcls - INFO - Saving checkpoint at 98 epochs
2021-09-25 22:08:57,241 - mmcls - INFO - Epoch(val) [98][6] accuracy_top-
1: 82.9412, accuracy_top-5: 98.8235
2021-09-25 22:09:09,334 - mmcls - INFO - Saving checkpoint at 99 epochs
2021-09-25 22:09:11,369 - mmcls - INFO - Epoch(val) [99][6] accuracy_top-
1: 82.9412, accuracy_top-5: 97.6471
2021-09-25 22:09:23,919 - mmcls - INFO - Saving checkpoint at 100 epochs
2021-09-25 22:09:25,890 - mmcls - INFO - Epoch(val) [100][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:09:38,263 - mmcls - INFO - Saving checkpoint at 101 epochs
2021-09-25 22:09:40,289 - mmcls - INFO - Epoch(val) [101][6]
accuracy_top-1: 85.2941, accuracy_top-5: 99.4118
2021-09-25 22:09:52,298 - mmcls - INFO - Saving checkpoint at 102 epochs
2021-09-25 22:09:54,234 - mmcls - INFO - Epoch(val) [102][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:10:06,681 - mmcls - INFO - Saving checkpoint at 103 epochs
2021-09-25 22:10:08,639 - mmcls - INFO - Epoch(val) [103][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:10:20,739 - mmcls - INFO - Saving checkpoint at 104 epochs
```

```
2021-09-25 22:10:22,732 - mmcls - INFO - Epoch(val) [104][6]
accuracy_top-1: 84.1176, accuracy_top-5: 99.4118
2021-09-25 22:10:34,823 - mmcls - INFO - Saving checkpoint at 105 epochs
2021-09-25 22:10:36,838 - mmcls - INFO - Epoch(val) [105][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:10:48,936 - mmcls - INFO - Saving checkpoint at 106 epochs
2021-09-25 22:10:50,905 - mmcls - INFO - Epoch(val) [106][6]
accuracy_top-1: 82.9412, accuracy_top-5: 99.4118
2021-09-25 22:11:02,931 - mmcls - INFO - Saving checkpoint at 107 epochs
2021-09-25 22:11:04,821 - mmcls - INFO - Epoch(val) [107][6]
accuracy_top-1: 85.2941, accuracy_top-5: 99.4118
2021-09-25 22:11:17,288 - mmcls - INFO - Saving checkpoint at 108 epochs
2021-09-25 22:11:19,240 - mmcls - INFO - Epoch(val) [108][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:11:31,784 - mmcls - INFO - Saving checkpoint at 109 epochs
2021-09-25 22:11:33,664 - mmcls - INFO - Epoch(val) [109][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:11:46,196 - mmcls - INFO - Saving checkpoint at 110 epochs
2021-09-25 22:11:48,225 - mmcls - INFO - Epoch(val) [110][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:12:00,337 - mmcls - INFO - Saving checkpoint at 111 epochs
2021-09-25 22:12:02,294 - mmcls - INFO - Epoch(val) [111][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:12:14,994 - mmcls - INFO - Saving checkpoint at 112 epochs
2021-09-25 22:12:16,960 - mmcls - INFO - Epoch(val) [112][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:12:29,533 - mmcls - INFO - Saving checkpoint at 113 epochs
2021-09-25 22:12:31,441 - mmcls - INFO - Epoch(val) [113][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:12:43,895 - mmcls - INFO - Saving checkpoint at 114 epochs
2021-09-25 22:12:45,840 - mmcls - INFO - Epoch(val) [114][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:12:57,925 - mmcls - INFO - Saving checkpoint at 115 epochs
2021-09-25 22:12:59,958 - mmcls - INFO - Epoch(val) [115][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:13:11,985 - mmcls - INFO - Saving checkpoint at 116 epochs
2021-09-25 22:13:13,977 - mmcls - INFO - Epoch(val) [116][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:13:26,557 - mmcls - INFO - Saving checkpoint at 117 epochs
2021-09-25 22:13:28,511 - mmcls - INFO - Epoch(val) [117][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:13:40,798 - mmcls - INFO - Saving checkpoint at 118 epochs
2021-09-25 22:13:42,802 - mmcls - INFO - Epoch(val) [118][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:13:54,806 - mmcls - INFO - Saving checkpoint at 119 epochs
2021-09-25 22:13:56,774 - mmcls - INFO - Epoch(val) [119][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:14:08,965 - mmcls - INFO - Saving checkpoint at 120 epochs
2021-09-25 22:14:10,897 - mmcls - INFO - Epoch(val) [120][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:14:23,255 - mmcls - INFO - Saving checkpoint at 121 epochs
2021-09-25 22:14:25,268 - mmcls - INFO - Epoch(val) [121][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:14:37,779 - mmcls - INFO - Saving checkpoint at 122 epochs
2021-09-25 22:14:39,783 - mmcls - INFO - Epoch(val) [122][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:14:52,212 - mmcls - INFO - Saving checkpoint at 123 epochs
```

```
2021-09-25 22:14:54,223 - mmcls - INFO - Epoch(val) [123][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:15:06,659 - mmcls - INFO - Saving checkpoint at 124 epochs
2021-09-25 22:15:08,601 - mmcls - INFO - Epoch(val) [124][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:15:21,033 - mmcls - INFO - Saving checkpoint at 125 epochs
2021-09-25 22:15:22,992 - mmcls - INFO - Epoch(val) [125][6]
accuracy_top-1: 84.7059, accuracy_top-5: 99.4118
2021-09-25 22:15:35,462 - mmcls - INFO - Saving checkpoint at 126 epochs
2021-09-25 22:15:37,479 - mmcls - INFO - Epoch(val) [126][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:15:50,024 - mmcls - INFO - Saving checkpoint at 127 epochs
2021-09-25 22:15:52,014 - mmcls - INFO - Epoch(val) [127][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:16:03,950 - mmcls - INFO - Saving checkpoint at 128 epochs
2021-09-25 22:16:05,943 - mmcls - INFO - Epoch(val) [128][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:16:18,461 - mmcls - INFO - Saving checkpoint at 129 epochs
2021-09-25 22:16:20,415 - mmcls - INFO - Epoch(val) [129][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:16:32,398 - mmcls - INFO - Saving checkpoint at 130 epochs
2021-09-25 22:16:34,387 - mmcls - INFO - Epoch(val) [130][6]
accuracy_top-1: 82.3529, accuracy_top-5: 99.4118
2021-09-25 22:16:46,703 - mmcls - INFO - Saving checkpoint at 131 epochs
2021-09-25 22:16:48,688 - mmcls - INFO - Epoch(val) [131][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:17:00,802 - mmcls - INFO - Saving checkpoint at 132 epochs
2021-09-25 22:17:02,803 - mmcls - INFO - Epoch(val) [132][6]
accuracy_top-1: 81.7647, accuracy_top-5: 100.0000
2021-09-25 22:17:14,966 - mmcls - INFO - Saving checkpoint at 133 epochs
2021-09-25 22:17:17,037 - mmcls - INFO - Epoch(val) [133][6]
accuracy_top-1: 84.1176, accuracy_top-5: 99.4118
2021-09-25 22:17:29,675 - mmcls - INFO - Saving checkpoint at 134 epochs
2021-09-25 22:17:31,677 - mmcls - INFO - Epoch(val) [134][6]
accuracy_top-1: 82.3529, accuracy_top-5: 99.4118
2021-09-25 22:17:44,486 - mmcls - INFO - Saving checkpoint at 135 epochs
2021-09-25 22:17:46,436 - mmcls - INFO - Epoch(val) [135][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:17:58,911 - mmcls - INFO - Saving checkpoint at 136 epochs
2021-09-25 22:18:00,892 - mmcls - INFO - Epoch(val) [136][6]
accuracy_top-1: 81.1765, accuracy_top-5: 100.0000
2021-09-25 22:18:13,487 - mmcls - INFO - Saving checkpoint at 137 epochs
2021-09-25 22:18:15,517 - mmcls - INFO - Epoch(val) [137][6]
accuracy_top-1: 82.3529, accuracy_top-5: 99.4118
2021-09-25 22:18:28,162 - mmcls - INFO - Saving checkpoint at 138 epochs
2021-09-25 22:18:30,170 - mmcls - INFO - Epoch(val) [138][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:18:42,852 - mmcls - INFO - Saving checkpoint at 139 epochs
2021-09-25 22:18:44,851 - mmcls - INFO - Epoch(val) [139][6]
accuracy_top-1: 81.7647, accuracy_top-5: 99.4118
2021-09-25 22:18:57,565 - mmcls - INFO - Saving checkpoint at 140 epochs
2021-09-25 22:18:59,635 - mmcls - INFO - Epoch(val) [140][6]
accuracy_top-1: 81.7647, accuracy_top-5: 100.0000
2021-09-25 22:19:12,266 - mmcls - INFO - Saving checkpoint at 141 epochs
2021-09-25 22:19:14,341 - mmcls - INFO - Epoch(val) [141][6]
accuracy_top-1: 81.7647, accuracy_top-5: 100.0000
2021-09-25 22:19:27,010 - mmcls - INFO - Saving checkpoint at 142 epochs
```

```
2021-09-25 22:19:29,016 - mmcls - INFO - Epoch(val) [142][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:19:41,763 - mmcls - INFO - Saving checkpoint at 143 epochs
2021-09-25 22:19:43,851 - mmcls - INFO - Epoch(val) [143][6]
accuracy_top-1: 84.7059, accuracy_top-5: 100.0000
2021-09-25 22:19:56,546 - mmcls - INFO - Saving checkpoint at 144 epochs
2021-09-25 22:19:58,535 - mmcls - INFO - Epoch(val) [144][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:20:11,199 - mmcls - INFO - Saving checkpoint at 145 epochs
2021-09-25 22:20:13,249 - mmcls - INFO - Epoch(val) [145][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:20:25,952 - mmcls - INFO - Saving checkpoint at 146 epochs
2021-09-25 22:20:27,944 - mmcls - INFO - Epoch(val) [146][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:20:40,534 - mmcls - INFO - Saving checkpoint at 147 epochs
2021-09-25 22:20:42,576 - mmcls - INFO - Epoch(val) [147][6]
accuracy_top-1: 81.7647, accuracy_top-5: 100.0000
2021-09-25 22:20:55,149 - mmcls - INFO - Saving checkpoint at 148 epochs
2021-09-25 22:20:57,154 - mmcls - INFO - Epoch(val) [148][6]
accuracy_top-1: 81.1765, accuracy_top-5: 100.0000
2021-09-25 22:21:09,722 - mmcls - INFO - Saving checkpoint at 149 epochs
2021-09-25 22:21:11,789 - mmcls - INFO - Epoch(val) [149][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:21:24,465 - mmcls - INFO - Saving checkpoint at 150 epochs
2021-09-25 22:21:26,521 - mmcls - INFO - Epoch(val) [150][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:21:39,197 - mmcls - INFO - Saving checkpoint at 151 epochs
2021-09-25 22:21:41,255 - mmcls - INFO - Epoch(val) [151][6]
accuracy_top-1: 81.7647, accuracy_top-5: 100.0000
2021-09-25 22:21:53,741 - mmcls - INFO - Saving checkpoint at 152 epochs
2021-09-25 22:21:55,704 - mmcls - INFO - Epoch(val) [152][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:22:07,981 - mmcls - INFO - Saving checkpoint at 153 epochs
2021-09-25 22:22:09,918 - mmcls - INFO - Epoch(val) [153][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:22:22,042 - mmcls - INFO - Saving checkpoint at 154 epochs
2021-09-25 22:22:23,998 - mmcls - INFO - Epoch(val) [154][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:22:36,217 - mmcls - INFO - Saving checkpoint at 155 epochs
2021-09-25 22:22:38,170 - mmcls - INFO - Epoch(val) [155][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:22:50,228 - mmcls - INFO - Saving checkpoint at 156 epochs
2021-09-25 22:22:52,195 - mmcls - INFO - Epoch(val) [156][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:23:04,620 - mmcls - INFO - Saving checkpoint at 157 epochs
2021-09-25 22:23:06,579 - mmcls - INFO - Epoch(val) [157][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:23:18,868 - mmcls - INFO - Saving checkpoint at 158 epochs
2021-09-25 22:23:20,824 - mmcls - INFO - Epoch(val) [158][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:23:33,124 - mmcls - INFO - Saving checkpoint at 159 epochs
2021-09-25 22:23:35,091 - mmcls - INFO - Epoch(val) [159][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:23:47,332 - mmcls - INFO - Saving checkpoint at 160 epochs
2021-09-25 22:23:49,307 - mmcls - INFO - Epoch(val) [160][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:24:01,504 - mmcls - INFO - Saving checkpoint at 161 epochs
```

```
2021-09-25 22:24:03,512 - mmcls - INFO - Epoch(val) [161][6]
accuracy_top-1: 81.7647, accuracy_top-5: 100.0000
2021-09-25 22:24:15,492 - mmcls - INFO - Saving checkpoint at 162 epochs
2021-09-25 22:24:17,423 - mmcls - INFO - Epoch(val) [162][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:24:29,251 - mmcls - INFO - Saving checkpoint at 163 epochs
2021-09-25 22:24:31,264 - mmcls - INFO - Epoch(val) [163][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:24:43,410 - mmcls - INFO - Saving checkpoint at 164 epochs
2021-09-25 22:24:45,399 - mmcls - INFO - Epoch(val) [164][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:24:57,455 - mmcls - INFO - Saving checkpoint at 165 epochs
2021-09-25 22:24:59,426 - mmcls - INFO - Epoch(val) [165][6]
accuracy_top-1: 82.9412, accuracy_top-5: 99.4118
2021-09-25 22:25:11,461 - mmcls - INFO - Saving checkpoint at 166 epochs
2021-09-25 22:25:13,409 - mmcls - INFO - Epoch(val) [166][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:25:25,439 - mmcls - INFO - Saving checkpoint at 167 epochs
2021-09-25 22:25:27,462 - mmcls - INFO - Epoch(val) [167][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:25:39,388 - mmcls - INFO - Saving checkpoint at 168 epochs
2021-09-25 22:25:41,347 - mmcls - INFO - Epoch(val) [168][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:25:53,644 - mmcls - INFO - Saving checkpoint at 169 epochs
2021-09-25 22:25:55,618 - mmcls - INFO - Epoch(val) [169][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:26:07,788 - mmcls - INFO - Saving checkpoint at 170 epochs
2021-09-25 22:26:09,732 - mmcls - INFO - Epoch(val) [170][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:26:21,889 - mmcls - INFO - Saving checkpoint at 171 epochs
2021-09-25 22:26:23,853 - mmcls - INFO - Epoch(val) [171][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:26:35,739 - mmcls - INFO - Saving checkpoint at 172 epochs
2021-09-25 22:26:37,705 - mmcls - INFO - Epoch(val) [172][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:26:49,642 - mmcls - INFO - Saving checkpoint at 173 epochs
2021-09-25 22:26:51,574 - mmcls - INFO - Epoch(val) [173][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:27:03,847 - mmcls - INFO - Saving checkpoint at 174 epochs
2021-09-25 22:27:05,798 - mmcls - INFO - Epoch(val) [174][6]
accuracy_top-1: 83.5294, accuracy_top-5: 99.4118
2021-09-25 22:27:18,022 - mmcls - INFO - Saving checkpoint at 175 epochs
2021-09-25 22:27:19,970 - mmcls - INFO - Epoch(val) [175][6]
accuracy_top-1: 82.3529, accuracy_top-5: 99.4118
2021-09-25 22:27:31,697 - mmcls - INFO - Saving checkpoint at 176 epochs
2021-09-25 22:27:33,699 - mmcls - INFO - Epoch(val) [176][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:27:45,951 - mmcls - INFO - Saving checkpoint at 177 epochs
2021-09-25 22:27:47,915 - mmcls - INFO - Epoch(val) [177][6]
accuracy_top-1: 82.9412, accuracy_top-5: 99.4118
2021-09-25 22:27:59,846 - mmcls - INFO - Saving checkpoint at 178 epochs
2021-09-25 22:28:01,849 - mmcls - INFO - Epoch(val) [178][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:28:13,855 - mmcls - INFO - Saving checkpoint at 179 epochs
2021-09-25 22:28:15,857 - mmcls - INFO - Epoch(val) [179][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:28:27,818 - mmcls - INFO - Saving checkpoint at 180 epochs
```

```
2021-09-25 22:28:29,783 - mmcls - INFO - Epoch(val) [180][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:28:41,979 - mmcls - INFO - Saving checkpoint at 181 epochs
2021-09-25 22:28:43,912 - mmcls - INFO - Epoch(val) [181][6]
accuracy_top-1: 82.9412, accuracy_top-5: 99.4118
2021-09-25 22:28:56,168 - mmcls - INFO - Saving checkpoint at 182 epochs
2021-09-25 22:28:58,110 - mmcls - INFO - Epoch(val) [182][6]
accuracy_top-1: 82.3529, accuracy_top-5: 99.4118
2021-09-25 22:29:10,063 - mmcls - INFO - Saving checkpoint at 183 epochs
2021-09-25 22:29:11,975 - mmcls - INFO - Epoch(val) [183][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:29:24,126 - mmcls - INFO - Saving checkpoint at 184 epochs
2021-09-25 22:29:26,084 - mmcls - INFO - Epoch(val) [184][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:29:38,279 - mmcls - INFO - Saving checkpoint at 185 epochs
2021-09-25 22:29:40,241 - mmcls - INFO - Epoch(val) [185][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:29:52,477 - mmcls - INFO - Saving checkpoint at 186 epochs
2021-09-25 22:29:54,404 - mmcls - INFO - Epoch(val) [186][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:30:06,247 - mmcls - INFO - Saving checkpoint at 187 epochs
2021-09-25 22:30:08,207 - mmcls - INFO - Epoch(val) [187][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:30:20,035 - mmcls - INFO - Saving checkpoint at 188 epochs
2021-09-25 22:30:22,007 - mmcls - INFO - Epoch(val) [188][6]
accuracy_top-1: 81.1765, accuracy_top-5: 100.0000
2021-09-25 22:30:33,891 - mmcls - INFO - Saving checkpoint at 189 epochs
2021-09-25 22:30:35,809 - mmcls - INFO - Epoch(val) [189][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:30:47,735 - mmcls - INFO - Saving checkpoint at 190 epochs
2021-09-25 22:30:49,690 - mmcls - INFO - Epoch(val) [190][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:31:01,792 - mmcls - INFO - Saving checkpoint at 191 epochs
2021-09-25 22:31:03,769 - mmcls - INFO - Epoch(val) [191][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:31:15,621 - mmcls - INFO - Saving checkpoint at 192 epochs
2021-09-25 22:31:17,556 - mmcls - INFO - Epoch(val) [192][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:31:29,580 - mmcls - INFO - Saving checkpoint at 193 epochs
2021-09-25 22:31:31,597 - mmcls - INFO - Epoch(val) [193][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:31:43,843 - mmcls - INFO - Saving checkpoint at 194 epochs
2021-09-25 22:31:45,806 - mmcls - INFO - Epoch(val) [194][6]
accuracy_top-1: 84.1176, accuracy_top-5: 100.0000
2021-09-25 22:31:57,699 - mmcls - INFO - Saving checkpoint at 195 epochs
2021-09-25 22:31:59,622 - mmcls - INFO - Epoch(val) [195][6]
accuracy_top-1: 82.3529, accuracy_top-5: 99.4118
2021-09-25 22:32:11,494 - mmcls - INFO - Saving checkpoint at 196 epochs
2021-09-25 22:32:13,427 - mmcls - INFO - Epoch(val) [196][6]
accuracy_top-1: 82.9412, accuracy_top-5: 100.0000
2021-09-25 22:32:25,429 - mmcls - INFO - Saving checkpoint at 197 epochs
2021-09-25 22:32:27,437 - mmcls - INFO - Epoch(val) [197][6]
accuracy_top-1: 83.5294, accuracy_top-5: 100.0000
2021-09-25 22:32:39,733 - mmcls - INFO - Saving checkpoint at 198 epochs
2021-09-25 22:32:41,714 - mmcls - INFO - Epoch(val) [198][6]
accuracy_top-1: 82.3529, accuracy_top-5: 100.0000
2021-09-25 22:32:53,650 - mmcls - INFO - Saving checkpoint at 199 epochs
```

```
2021-09-25 22:32:55,614 - mmcls - INFO - Epoch(val) [199][6]
accuracy_top-1: 82.9412, accuracy_top-5: 99.4118
2021-09-25 22:33:07,780 - mmcls - INFO - Saving checkpoint at 200 epochs
2021-09-25 22:33:09,788 - mmcls - INFO - Epoch(val) [200][6]
accuracy_top-1: 84.7059, accuracy_top-5: 100.0000
```

Step 5. Evaluation

Demo

Before demo, you need mmcv-full.

```
pip install mmcv-full
```

Here we choose one image from the validation set `image_0005.jpg` and put it in the `demo` directory. And we run this following command to classify this image by our trained model, which is saved at `output/resnet18_flowers_bs128/` .

The image we choose is

Run the model!

```
python demo/image_demo.py \
  --img 'demo/image_0005.jpg'\
  --config 'configs/resnet/resnet18_flowers_bs128.py' \
  --checkpoint 'output/resnet18_flowers_bs128/epoch_199.pth'
```

Example output: {'pred_label': 0, 'pred_score': 0.9722172021865845, 'pred_class': 'daffodil'}

From the output, the trained model successfully classify the demo image as 'daffodil' with over 97% confidence.

Test

You can test the trained model by running the following command

```
python tools/test.py \
  --config 'configs/resnet/resnet18_flowers_bs128.py' \
  --checkpoint 'output/resnet18_flowers_bs128/epoch_199.pth' \
  --out 'output/resnet18_flowers_bs128/test.json'
```

The output file will be saved in the `--out`, which contains the prediction results for all individual dat samples.

Extension Questions

1. What points can be improved to increase classification accuacy?
2. What is the influence of data split on the model?

```
In [ ]:
```