Last Updated:20231020

复现精度

文章精度：

Variations	cls loss	distilling loss	bias removal	FC retrain	20	40	60	80	100
baseline-1	✓				84.40	68.30	55.10	48.52	39.83
baseline-2	✓	✓			**85.05**	72.22	59.41	50.43	40.34
BiC(Ours)	✓	✓	✓		84.00	**74.69**	**67.93**	**61.25**	**56.69**
upper bound	✓	✓		✓	84.39	76.15	69.51	64.03	60.93

Table 3. Incremental learning results on CIFAR-100 with a batch of 20 classes. baseline-1 uses the classification loss alone. baseline-2 uses both the distilling loss and the classification loss. BiC corrects the bias in FC layer of baseline-2. Upper bound retrains the last FC layer using all samples from both old and new classes after learning the model of baseline-2. The best results are marked in bold.

训练集和验证集按不同比例划分时的精度：

$train_{old}$:val_{old}	20	40	60	80	100
9:1	84.00	**74.69**	**67.93**	**61.25**	**56.69**
8:2	84.50	73.19	65.01	58.68	54.31
7:3	**84.70**	71.60	63.68	58.12	53.74
6:4	83.33	68.84	62.21	56.00	51.17

Table 4. Incremental learning results on CIFAR-100 with a batch of 20 classes for different training/validation split on exemplars from old classes. The training set is used to learn the feature and classifier layers, and the validation set is used to learn the bias correction layer. The best results are marked in bold.

精度对比：

结果参照：`log/bic-cifarresnet-epoch100-23-10-19-19-04-26.log`

	20	40	60	80	100
原文	0.84	0.747	0.679	0.613	0.567
Ours（复现精度）	0.63	0.515	0.410	0.320	0.260

一些问题

目前的代码与原始论文中所述模型的差别

1.stage2还是用**训练集**中的数据来来训练，没有验证集，而且stage2不是在更新bias correction这一层（**模型中没有bias correction这一层**）的参数，它还是在更新全局参数（相比stage1的区别是损失函数不太一样)

2.目前bic的训练过程是

每个task：

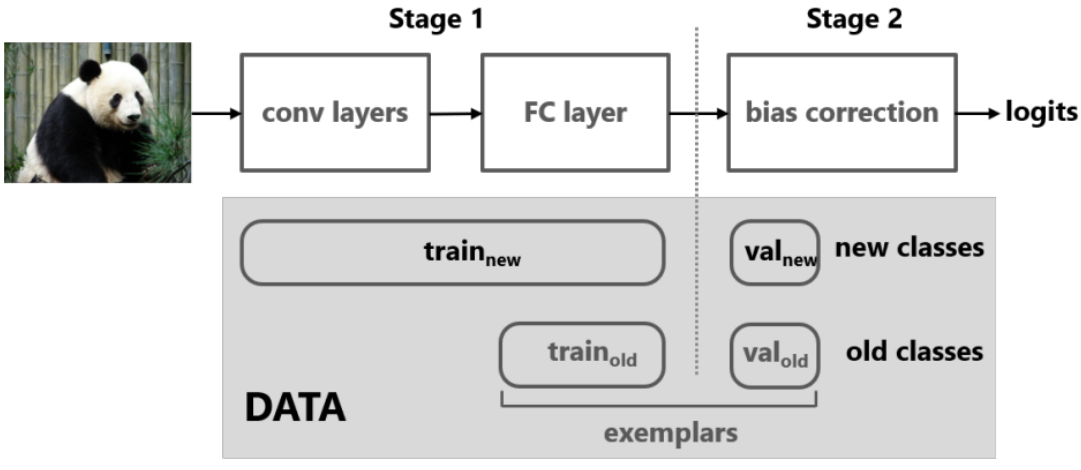
before_task（stage1+stage2(task_idx>0)）+ train(250 epoch)+ after_task(存旧模型)

更新buffer后继续下一task

原文对应的模型训练应该没有中间这一过程（？）

3.训练过程中，task_idx>0时，before_task的每次train_acc和test_acc都趋近于0（几乎都为0）

```
stage : training, => Task 1, Epoch 1/5 => Loss 1.843, train_acc : 0.000, test_acc : 0.000
stage : training, => Task 1, Epoch 2/5 => Loss 1.755, train_acc : 0.000, test_acc : 0.000
stage : training, => Task 1, Epoch 3/5 => Loss 1.727, train_acc : 0.000, test_acc : 0.000
stage : training, => Task 1, Epoch 4/5 => Loss 1.708, train_acc : 0.000, test_acc : 0.000
stage : training, => Task 1, Epoch 5/5 => Loss 1.699, train_acc : 0.000, test_acc : 0.000
stage : bias_correction, => Task 1, Epoch 1/5 => Loss 4.619, train_acc : 0.000, test_acc : 0.000
stage : bias_correction, => Task 1, Epoch 2/5 => Loss 4.615, train_acc : 0.000, test_acc : 0.000
stage : bias_correction, => Task 1, Epoch 3/5 => Loss 4.583, train_acc : 0.062, test_acc : 0.062
stage : bias_correction, => Task 1, Epoch 4/5 => Loss 4.478, train_acc : 0.281, test_acc : 0.281
stage : bias_correction, => Task 1, Epoch 5/5 => Loss 4.423, train_acc : 0.219, test_acc : 0.219
```



主要修改内容

一.self.network()无bias_layers这一属性

问题：`core/model/replay/bic.py`中`_stage1_training`和`_stage2_bias_correction`用到：

```
def _stage1_training(self, train_loader, test_loader):
    """
    if self.cur_task_id == 0:
        loaded_dict = torch.load('./dict_0.pkl')
        self.network.load_state_dict(loaded_dict['model_state_dict'])
        self.network.to(self.device)
        return
    """

    ignored_params = list(map(id, self.network.bias_layers.parameters()))
```

但self.network()无bias_layers这一属性

解决方法:

注释了ignored_params 、base_params 、ignored_params, network_params，把`optim.SGD`的第一个参数由`network_params`改成了`self.network.parameters`

```
def _stage1_training(self, train_loader, test_loader):
    """
    if self.cur_task_id == 0:
        loaded_dict = torch.load('./dict_0.pkl')
        self.network.load_state_dict(loaded_dict['model_state_dict'])
        self.network.to(self.device)
        return
    """
    '''
    ignored_params = list(map(id, self.network.bias_layers.parameters()))
    base_params = filter(lambda p: id(p) not in ignored_params, self.network.parameters())
    network_params = [
        {"params": base_params, "lr": self.lr, "weight_decay": self.weight_decay},
        {
            "params": self.network.bias_layers.parameters(),
            "lr": 0,
            "weight_decay": 0,
        },]
    '''
    #optimizer = optim.SGD(network_params, lr=self.lr, momentum=self.momentum, weight_decay=self.weight_
    optimizer = optim.SGD(self.network.parameters(),lr=self.lr, momentum=self.momentum, weight_decay=sel
    scheduler = optim.lr_scheduler.MultiStepLR(optimizer=optimizer, milestones=self.milestones, gamma=se

    self.network.to(self.device)
    if self.old_network is not None:
        self.old_network.to(self.device)

    self._run(train_loader, test_loader, optimizer, scheduler, stage="training")
```

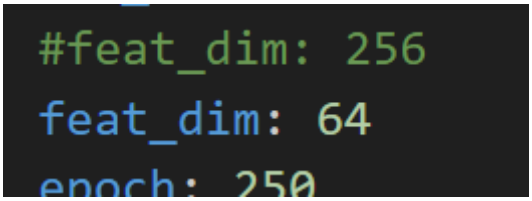
`core/mode/backbone/resnet.py`中：

```python
class CifarResNet(nn.Module):
    """
    ResNet optimized for the Cifar Dataset, as specified in
    https://arxiv.org/abs/1512.03385.pdf
    """

    def __init__(self, block, depth, channels=3):
        super(CifarResNet, self).__init__()

        # Model type specifies number of layers for CIFAR-10 and CI
        assert (depth - 2) % 6 == 0, 'depth should be one of 20, 32
        layer_blocks = (depth - 2) // 6

        self.conv_1_3x3 = nn.Conv2d(channels, 16, kernel_size=3, st
        self.bn_1 = nn.BatchNorm2d(16)
```

二.cifarresnet的输出特征维度与bic.py中定义的classifier的feat_dim不匹配

解决方法:

`bic.yaml`里把`feat_dim`改为64



三.bic类初始化时kwargs里无epoch等属性

解决方法：把epoch等属性再抄一遍放在bic.yaml的classifier的kwargs里

```yaml
classifier:
  name: bic
  kwargs:
    num_class: 100
    #feat_dim: 256
    feat_dim: 64
    epoch: 250
    init_cls_num: 10
    inc_cls_num: 10
    task_num: 10
    gamma: 0.5
    lr: 0.1
    weight_decay: 2e-4
    momentum: 0.9
    milestones: [60, 100, 140]
```

四. File "/home/bzx_yjy/LibContinual/core/model/replay/bic.py", line 178, in _run image, label = image.to(self.device), label.to(self.device) AttributeError: 'str' object has no attribute 'to'

解决方法:

将原先的

```
for i,(image,label) in enumerate(train_loader):
    image, label = image.to(self.device), label.to(self.device)
```

改为

```
for i, batch in enumerate(train_loader):
                image = batch['image']
                label = batch['label']
                image, label = image.to(self.device), label.to(self.device)
```

五.inference里传的是一个不支持索引的数据加载器

解决方法:

```
  File "/home/bzx_yjy/LibContinual/core/model/replay/bic.py", line 216, in _run
    _, train_acc = self.inference(train_loader)
  File "/home/bzx_yjy/LibContinual/core/model/replay/bic.py", line 113, in inference
    x, y = data['image'], data['label']
TypeError: 'DataLoader' object is not subscriptable
```

修改216行的推理过程传递的内容，把train_loader换成batch，batch的含义如下：

```
for i, batch in enumerate(train_loader):

    image = batch['image']
    label = batch['label']
    image, label = image.to(self.device), label.to(self.device)
    logits = self.network(image)
```

六.buffer.reduce_old_data(self.cur_task_id, self.accu_cls_num) AttributeError: 'LinearBuffer' object has no attribute 'reduce_old_data'

解决方法：

`bic.py`中：

```
def after_task(self, task_idx, buffer, train_loader, test_loaders):
    # freeze old network as KD teacher
    self.old_network = deepcopy(self.network)
    self.old_network.eval()
    self.prev_cls_num = self.accu_cls_num
    '''
    # update buffer
    buffer.reduce_old_data(self.cur_task_id, self.accu_cls_num)
    val_transform = test_loaders[0].dataset.trfms
    buffer.update(self.network, train_loader, val_transform,
                  self.cur_task_id, self.accu_cls_num, self.cur_cls_indexes,
                  self.device)


    # compute class mean vector via samples in buffer
    self.class_means = self.calc_class_mean(buffer,
                                            train_loader,
                                            val_transform,
                                            self.device).to(self.device)
    '''
    self.cur_task_id += 1
    self.lamda = self.prev_cls_num / self.accu_cls_num
```

七. File "/home/bzx_yjy/LibContinual/core/model/replay/bic.py", line 201, in _run distill_loss = -torch.mean(torch.sum(hat_pai_k * log_pai_k, dim=1))

RuntimeError: The size of tensor a (100) must match the size of tensor b (10) at non-singleton dimension 1

解决方法：

在bic.py的_run中修改hat_pai_k

```
#hat_pai_k = F.softmax(old_logits / self.T, dim=1)
hat_pai_k = F.softmax(old_logits[:, : self.prev_cls_num] / self.T, dim=1)
log_pai_k = F.log_softmax(logits[:, : self.prev_cls_num] / self.T, dim=1)
distill_loss = -torch.mean(torch.sum(hat_pai_k * log_pai_k, dim=1))
```

部分结果

这是before_task跑了250个epoch的截图（不是很快）

```
===============Task 0 Start!================
stage : training, => Task 0, Epoch 1/250 => Loss 2.399, train_acc : 0.219, test_acc : 0.219
stage : training, => Task 0, Epoch 2/250 => Loss 1.794, train_acc : 0.312, test_acc : 0.312
stage : training, => Task 0, Epoch 3/250 => Loss 1.605, train_acc : 0.375, test_acc : 0.375
stage : training, => Task 0, Epoch 4/250 => Loss 1.459, train_acc : 0.469, test_acc : 0.469
stage : training, => Task 0, Epoch 5/250 => Loss 1.319, train_acc : 0.688, test_acc : 0.688
stage : training, => Task 0, Epoch 6/250 => Loss 1.227, train_acc : 0.500, test_acc : 0.500
stage : training, => Task 0, Epoch 7/250 => Loss 1.162, train_acc : 0.594, test_acc : 0.594
stage : training, => Task 0, Epoch 8/250 => Loss 1.084, train_acc : 0.625, test_acc : 0.625
stage : training, => Task 0, Epoch 9/250 => Loss 1.041, train_acc : 0.719, test_acc : 0.719
stage : training, => Task 0, Epoch 10/250 => Loss 0.951, train_acc : 0.844, test_acc : 0.844
stage : training, => Task 0, Epoch 11/250 => Loss 0.904, train_acc : 0.875, test_acc : 0.875
stage : training, => Task 0, Epoch 12/250 => Loss 0.813, train_acc : 0.781, test_acc : 0.781
stage : training, => Task 0, Epoch 13/250 => Loss 0.741, train_acc : 0.750, test_acc : 0.750
stage : training, => Task 0, Epoch 14/250 => Loss 0.691, train_acc : 0.875, test_acc : 0.875
stage : training, => Task 0, Epoch 15/250 => Loss 0.638, train_acc : 0.875, test_acc : 0.875
stage : training, => Task 0, Epoch 16/250 => Loss 0.592, train_acc : 0.875, test_acc : 0.875
stage : training, => Task 0, Epoch 17/250 => Loss 0.504, train_acc : 0.906, test_acc : 0.906
stage : training, => Task 0, Epoch 18/250 => Loss 0.481, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 19/250 => Loss 0.461, train_acc : 0.938, test_acc : 0.938
stage : training, => Task 0, Epoch 20/250 => Loss 0.358, train_acc : 0.938, test_acc : 0.938
stage : training, => Task 0, Epoch 21/250 => Loss 0.416, train_acc : 0.812, test_acc : 0.812
stage : training, => Task 0, Epoch 22/250 => Loss 0.381, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 23/250 => Loss 0.363, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 24/250 => Loss 0.287, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 25/250 => Loss 0.264, train_acc : 0.906, test_acc : 0.906
stage : training, => Task 0, Epoch 26/250 => Loss 0.295, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 27/250 => Loss 0.273, train_acc : 0.906, test_acc : 0.906
stage : training, => Task 0, Epoch 28/250 => Loss 0.233, train_acc : 0.938, test_acc : 0.938
stage : training, => Task 0, Epoch 29/250 => Loss 0.229, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 30/250 => Loss 0.186, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 31/250 => Loss 0.222, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 32/250 => Loss 0.169, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 33/250 => Loss 0.206, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 34/250 => Loss 0.205, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 35/250 => Loss 0.169, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 36/250 => Loss 0.211, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 37/250 => Loss 0.163, train_acc : 0.969, test_acc : 0.969
stage : training, => Task 0, Epoch 38/250 => Loss 0.165, train_acc : 0.938, test_acc : 0.938
stage : training, => Task 0, Epoch 39/250 => Loss 0.150, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 40/250 => Loss 0.131, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 41/250 => Loss 0.179, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 42/250 => Loss 0.184, train_acc : 1.000, test_acc : 1.000
stage : training, => Task 0, Epoch 43/250 => Loss 0.166, train_acc : 1.000, test_acc : 1.000
```

不小心中止了训练，为节约时间，把before_task里的epoch设成5跑

```
)
================Task 0 Start!================
stage : training, => Task 0, Epoch 1/5 => Loss 2.399, train_acc : 0.219, test_acc : 0.219
stage : training, => Task 0, Epoch 2/5 => Loss 1.794, train_acc : 0.312, test_acc : 0.312
stage : training, => Task 0, Epoch 3/5 => Loss 1.605, train_acc : 0.375, test_acc : 0.375
stage : training, => Task 0, Epoch 4/5 => Loss 1.459, train_acc : 0.469, test_acc : 0.469
stage : training, => Task 0, Epoch 5/5 => Loss 1.319, train_acc : 0.688, test_acc : 0.688
SGD (
Parameter Group 0
    dampening: 0
    differentiable: False
    foreach: None
    initial_lr: 0.1
    lr: 0.03333333333333333
    maximize: False
    momentum: 0.9
    nesterov: False
    weight_decay: 0.0002
)
================Task 0 Training!================
```

5趟后开始训练，task0中部分训练截图如下所示：

```
learning rate: [0.025]
=============== Train on the train set ===============
100%|████████████████████████████████████████████████████████████| 156/156 [00:05<00:00, 29.17it/s]
Epoch [22/250] |       Loss: 0.000     Average Acc: 0.990
=============== Test on the test set ===============
 * Average Acc: 0.750 Best acc 0.750
 Per-Task Acc:[0.75]
learning rate: [0.025]
=============== Train on the train set ===============
100%|████████████████████████████████████████████████████████████| 156/156 [00:05<00:00, 29.12it/s]
Epoch [23/250] |       Loss: 0.000     Average Acc: 0.998
=============== Test on the test set ===============
 * Average Acc: 0.740 Best acc 0.750
 Per-Task Acc:[0.74]
learning rate: [0.025]
=============== Train on the train set ===============
100%|████████████████████████████████████████████████████████████| 156/156 [00:05<00:00, 29.26it/s]
Epoch [24/250] |       Loss: 0.000     Average Acc: 0.999
=============== Test on the test set ===============
 * Average Acc: 0.740 Best acc 0.750
 Per-Task Acc:[0.74]
learning rate: [0.025]
=============== Train on the train set ===============
100%|████████████████████████████████████████████████████████████| 156/156 [00:05<00:00, 27.28it/s]
Epoch [25/250] |       Loss: 0.000     Average Acc: 0.999
=============== Test on the test set ===============
 * Average Acc: 0.740 Best acc 0.750
 Per-Task Acc:[0.74]
learning rate: [0.025]
=============== Train on the train set ===============
100%|████████████████████████████████████████████████████████████| 156/156 [00:06<00:00, 23.69it/s]
Epoch [26/250] |       Loss: 0.000     Average Acc: 1.000
=============== Test on the test set ===============
 * Average Acc: 0.750 Best acc 0.750
 Per-Task Acc:[0.75]
learning rate: [0.025]
=============== Train on the train set ===============
 54%|████████████████████████████████                            | 84/156 [00:03<00:02, 24.76it/s]
```