HW4: Implement a Deep Residual Neural Network for CIFAR100

Zhenye Na (zna2)

Part 1: Implementation of ResNet in Pytorch

Hyper-parameters

Hyper-parameters	Description
lr=0.01	learning rate
momentum=0.9	momentum factor
weight_decay=1e-5	weight decay (L2 penalty)
epochs=500	Number of epochs to train
batch_size_train=256	Batch size of training set
batch_size_test=256	Batch size of test set

Optimizer and Loss function

- torch.optim.SGD()
- CrossEntropyLoss()

Data Augmentation

```
# Normalize training set together with augmentation
transform_train = transforms.Compose([
    transforms.RandomCrop(32, padding=4),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.507, 0.487, 0.441], std=[0.267, 0.256, 0.276])
])

# Normalize test set same as training set without augmentation
transform_test = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.507, 0.487, 0.441], std=[0.267, 0.256, 0.276])
])
```

Channel downsampling

```
downsample = nn.Sequential(
    conv3x3(self.in_channels, out_channels, stride=stride),
    nn.BatchNorm2d(num_features=out_channels)
)

def conv3x3(in_channels, out_channels, stride=1):
    """3x3 kernel size with padding convolutional layer in ResNet BasicBlock."""
    return nn.Conv2d(
        in_channels=in_channels,
        out_channels=out_channels,
        kernel_size=3,
        stride=stride,
        padding=1,
        bias=False)
```

Result

I have used SGD+momentum optimizer, with 100 epochs, it reached 63-64% accuracy on CIFAR100 test set.

```
==> Building new ResNet model ...
==> Initialize CUDA support for ResNet model ...
==> Data Augmentation ...
==> Preparing CIFAR100 dataset ...
Files already downloaded and verified
Files already downloaded and verified
==> Start training ...
Iteration: 1 | Loss: 4.104180923530033 | Training accuracy: 14.472% | Test accuracy: 14.85%
==> Saving model ...
Iteration: 2 | Loss: 3.4602092735621395 | Training accuracy: 21.066% | Test accuracy: 20.77%
Iteration: 3 | Loss: 3.1336532514922473 | Training accuracy: 26.2% | Test accuracy: 25.07%
Iteration: 4 | Loss: 2.880905361808076 | Training accuracy: 29.676% | Test accuracy: 28.47%
Iteration: 5 | Loss: 2.6510908907773545 | Training accuracy: 34.976% | Test accuracy: 32.88%
Iteration: 6 | Loss: 2.481336920845265 | Training accuracy: 37.614% | Test accuracy: 34.41%
Iteration: 7 | Loss: 2.319791035384548 | Training accuracy: 42.072% | Test accuracy: 38.09%
Iteration: 8 | Loss: 2.1693926453590393 | Training accuracy: 45.586% | Test accuracy: 41.59%
```

```
Iteration: 9 | Loss: 2.0416611147170163 | Training accuracy: 47.214% | Test accuracy: 43.04%
Iteration: 10 | Loss: 1.9338786614184478 | Training accuracy: 50.044% | Test accuracy: 45.35%
Iteration: 11 | Loss: 1.830668755331818 | Training accuracy: 52.016% | Test accuracy: 47.05%
Iteration: 12 | Loss: 1.7460169713107907 | Training accuracy: 55.38% | Test accuracy: 48.44%
Iteration: 13 | Loss: 1.6628405780208355 | Training accuracy: 56.55% | Test accuracy: 49.71%
Iteration: 14 | Loss: 1.5798143872192927 | Training accuracy: 57.216% | Test accuracy: 49.22%
Iteration: 15 | Loss: 1.5135374920708793 | Training accuracy: 59.196% | Test accuracy: 51.76%
Iteration: 16 | Loss: 1.4557876057770787 | Training accuracy: 60.756% | Test accuracy: 51.58%
Iteration: 17 | Loss: 1.397268416930218 | Training accuracy: 62.26% | Test accuracy: 53.51%
Iteration: 18 | Loss: 1.3465026586639637 | Training accuracy: 64.048% | Test accuracy: 53.08%
Iteration: 19 | Loss: 1.2904698045886294 | Training accuracy: 64.964% | Test accuracy: 54.2%
Iteration: 20 | Loss: 1.2304265331857058 | Training accuracy: 66.884% | Test accuracy: 55.15%
Iteration: 21 | Loss: 1.192518736026725 | Training accuracy: 68.66% | Test accuracy: 55.48%
Iteration: 22 | Loss: 1.1429028416774711 | Training accuracy: 67.996% | Test accuracy: 55.17%
Iteration: 23 | Loss: 1.0980666112534854 | Training accuracy: 69.424% | Test accuracy: 56.26%
Iteration: 24 | Loss: 1.057483225756762 | Training accuracy: 71.148% | Test accuracy: 57.85%
Iteration: 25 | Loss: 1.032663247719103 | Training accuracy: 71.622% | Test accuracy: 57.16%
Iteration: 26 | Loss: 0.9889624885150364 | Training accuracy: 72.68% | Test accuracy: 57.52%
Iteration: 27 | Loss: 0.9433630595401842 | Training accuracy: 73.674% | Test accuracy: 56.65%
Iteration: 28 | Loss: 0.9149068362858831 | Training accuracy: 74.494% | Test accuracy: 57.6%
Iteration: 29 | Loss: 0.8813325060265405 | Training accuracy: 74.864% | Test accuracy: 57.12%
Iteration: 30 | Loss: 0.8572023571753988 | Training accuracy: 77.112% | Test accuracy: 59.18%
Iteration: 31 | Loss: 0.8264880502710537 | Training accuracy: 76.842% | Test accuracy: 58.53%
Iteration: 32 | Loss: 0.7944095457086757 | Training accuracy: 78.034% | Test accuracy: 59.13%
Iteration: 33 | Loss: 0.7609095737642172 | Training accuracy: 78.102% | Test accuracy: 58.32%
Iteration: 34 | Loss: 0.730701387536769 | Training accuracy: 79.234% | Test accuracy: 59.07%
Iteration: 35 | Loss: 0.7049194653423465 | Training accuracy: 79.648% | Test accuracy: 58.29%
Iteration: 36 | Loss: 0.6780204106958545 | Training accuracy: 81.206% | Test accuracy: 60.43%
Iteration: 37 | Loss: 0.6612185776537779 | Training accuracy: 81.446% | Test accuracy: 59.32%
Iteration: 38 | Loss: 0.629750130736098 | Training accuracy: 82.106% | Test accuracy: 59.46%
Iteration: 39 | Loss: 0.6031098405317384 | Training accuracy: 83.31% | Test accuracy: 60.11%
Iteration: 40 | Loss: 0.5774347835353443 | Training accuracy: 83.256% | Test accuracy: 59.33%
Iteration: 41 | Loss: 0.564434007418399 | Training accuracy: 83.934% | Test accuracy: 59.97%
Iteration: 42 | Loss: 0.5355604668052829 | Training accuracy: 85.076% | Test accuracy: 60.77%
Iteration: 43 | Loss: 0.5126350830708232 | Training accuracy: 85.768% | Test accuracy: 59.78%
Iteration: 44 | Loss: 0.5005355355690937 | Training accuracy: 84.766% | Test accuracy: 58.71%
Iteration: 45 | Loss: 0.48476455406266816 | Training accuracy: 86.344% | Test accuracy: 60.54%
Iteration: 46 | Loss: 0.4556497615210864 | Training accuracy: 87.492% | Test accuracy: 61.03%
Iteration: 47 | Loss: 0.4387689603834736 | Training accuracy: 87.684% | Test accuracy: 60.64%
Iteration: 48 | Loss: 0.41509357033943645 | Training accuracy: 88.33% | Test accuracy: 61.16%
Iteration: 49 | Loss: 0.4069142019262119 | Training accuracy: 88.748% | Test accuracy: 61.1%
Iteration: 50 | Loss: 0.3926576251278118 | Training accuracy: 89.712% | Test accuracy: 61.49%
Iteration: 51 | Loss: 0.37341941132837414 | Training accuracy: 89.238% | Test accuracy: 61.19%
==> Saving model ...
Iteration: 52 | Loss: 0.3532286737950481 | Training accuracy: 90.372% | Test accuracy: 61.24%
Iteration: 53 | Loss: 0.3430648485616762 | Training accuracy: 90.106% | Test accuracy: 60.44%
Iteration: 54 | Loss: 0.32229845735187435 | Training accuracy: 90.802% | Test accuracy: 61.17%
Iteration: 55 | Loss: 0.3160853220187888 | Training accuracy: 91.03% | Test accuracy: 61.29%
```

```
Iteration: 56 | Loss: 0.30303438988571263 | Training accuracy: 91.784% | Test accuracy: 60.74%
Iteration: 57 | Loss: 0.28862097471648335 | Training accuracy: 91.85% | Test accuracy: 61.69%
Iteration: 58 | Loss: 0.27474444374746204 | Training accuracy: 92.214% | Test accuracy: 61.59%
Iteration: 59 | Loss: 0.26047237947279095 | Training accuracy: 92.97% | Test accuracy: 61.41%
Iteration: 60 | Loss: 0.2498370428018424 | Training accuracy: 92.774% | Test accuracy: 60.91%
Iteration: 61 | Loss: 0.24482293457401041 | Training accuracy: 93.09% | Test accuracy: 61.07%
Iteration: 62 | Loss: 0.24151663269315446 | Training accuracy: 93.188% | Test accuracy: 61.42%
Iteration: 63 | Loss: 0.2337216458910582 | Training accuracy: 93.858% | Test accuracy: 61.36%
Iteration: 64 | Loss: 0.2185495105020854 | Training accuracy: 94.138% | Test accuracy: 62.55%
Iteration: 65 | Loss: 0.21097918805115076 | Training accuracy: 94.15% | Test accuracy: 61.64%
Iteration: 66 | Loss: 0.1980812152733608 | Training accuracy: 94.666% | Test accuracy: 62.19%
Iteration: 67 | Loss: 0.19419803546399486 | Training accuracy: 94.804% | Test accuracy: 62.07%
Iteration: 68 | Loss: 0.18773984844435235 | Training accuracy: 95.182% | Test accuracy: 62.69%
Iteration: 69 | Loss: 0.17875460022110112 | Training accuracy: 95.026% | Test accuracy: 62.7%
Iteration: 70 | Loss: 0.16828414216181453 | Training accuracy: 95.162% | Test accuracy: 61.97%
Iteration: 71 | Loss: 0.16690176189401928 | Training accuracy: 94.95% | Test accuracy: 62.24%
Iteration: 72 | Loss: 0.16335826640834614 | Training accuracy: 95.586% | Test accuracy: 62.36%
Iteration: 73 | Loss: 0.1569182300293932 | Training accuracy: 95.586% | Test accuracy: 61.65%
Iteration: 74 | Loss: 0.1456173792557449 | Training accuracy: 96.01% | Test accuracy: 62.61%
Iteration: 75 | Loss: 0.13593653091514596 | Training accuracy: 96.118% | Test accuracy: 62.35%
Iteration: 76 | Loss: 0.13525629895074026 | Training accuracy: 96.424% | Test accuracy: 62.76%
Iteration: 77 | Loss: 0.13298602196939138 | Training accuracy: 95.966% | Test accuracy: 61.94%
Iteration: 78 | Loss: 0.12998289785975095 | Training accuracy: 96.416% | Test accuracy: 62.24%
Iteration: 79 | Loss: 0.1252191846772116 | Training accuracy: 96.398% | Test accuracy: 61.56%
Iteration: 80 | Loss: 0.11754128008092544 | Training accuracy: 96.624% | Test accuracy: 62.14%
Iteration: 81 | Loss: 0.11218413461607937 | Training accuracy: 96.836% | Test accuracy: 62.49%
Iteration: 82 | Loss: 0.10849060604766923 | Training accuracy: 97.238% | Test accuracy: 62.66%
Iteration: 83 | Loss: 0.11016624517814845 | Training accuracy: 96.96% | Test accuracy: 62.23%
Iteration: 84 | Loss: 0.10297167498846443 | Training accuracy: 97.152% | Test accuracy: 62.74%
Iteration: 85 | Loss: 0.09708763187637134 | Training accuracy: 97.562% | Test accuracy: 62.91%
Iteration: 86 | Loss: 0.09874775933519918 | Training accuracy: 97.346% | Test accuracy: 63.17%
Iteration: 87 | Loss: 0.09549941109227282 | Training accuracy: 97.274% | Test accuracy: 62.17%
Iteration: 88 | Loss: 0.0933245594936366 | Training accuracy: 97.3% | Test accuracy: 63.14%
Iteration: 89 | Loss: 0.09233627390420558 | Training accuracy: 97.424% | Test accuracy: 63.52%
Iteration: 90 | Loss: 0.0904448315957371 | Training accuracy: 97.86% | Test accuracy: 63.1%
Iteration: 91 | Loss: 0.08890507337922345 | Training accuracy: 97.834% | Test accuracy: 63.28%
Iteration: 92 | Loss: 0.08488734942689842 | Training accuracy: 97.658% | Test accuracy: 63.16%
Iteration: 93 | Loss: 0.08385954704135656 | Training accuracy: 97.738% | Test accuracy: 63.18%
Iteration: 94 | Loss: 0.08202194106974164 | Training accuracy: 97.78% | Test accuracy: 62.82%
Iteration: 95 | Loss: 0.07593175037098783 | Training accuracy: 97.956% | Test accuracy: 63.39%
Iteration: 96 | Loss: 0.07439591596853368 | Training accuracy: 98.03% | Test accuracy: 63.29%
Iteration: 97 | Loss: 0.07279670525494279 | Training accuracy: 98.128% | Test accuracy: 62.96%
Iteration: 98 | Loss: 0.07384988501173806 | Training accuracy: 97.792% | Test accuracy: 63.25%
Iteration: 99 | Loss: 0.06875882858448491 | Training accuracy: 98.046% | Test accuracy: 63.36%
Iteration: 100 | Loss: 0.0674783697503866 | Training accuracy: 98.44% | Test accuracy: 63.31%
Iteration: 101 | Loss: 0.06507352591321176 | Training accuracy: 98.27% | Test accuracy: 62.67%
```

Part 2: Fine-tune ResNet-18 pre-trained model on CIFAR100 dataset

Hyper-parameters

Hyper-parameters	Description
lr=0.001	learning rate
momentum=0.9	momentum factor
weight_decay=1e-5	weight decay (L2 penalty)
epochs=500	Number of epochs to train
batch_size_train=64	Batch size of training set
batch_size_test=64	Batch size of test set

Optimizer, scheduler and Loss function

- torch.optim.Adam()
- CrossEntropyLoss()
- torch.optim.lr_scheduler.ExponentialLR()

Data Augmentation

```
# Normalize training set together with augmentation
transform_train = transforms.Compose([
    transforms.RandomResizedCrop(224),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])

# Normalize test set same as training set without augmentation
transform_test = transforms.Compose([
    transforms.Resize(224),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])
```

Fine tune model

```
# Everything except the last linear layer
self.features = nn.Sequential(*list(resnet.children())[:-1])
num_ftrs = resnet.fc.in_features
self.classifier = nn.Sequential(
    nn.Linear(num_ftrs, num_classes)
)
```

Result

I have used Adam optimizer, with 14 epochs, it reached 71% accuracy on CIFAR100 test set.

```
==> Load pre-trained ResNet model ...
==> Data Augmentation ...
==> Preparing CIFAR100 dataset ...
Files already downloaded and verified
Files already downloaded and verified
==> Start training ...
Iteration: 1 | Loss: 3.102943994200138 | Training accuracy: 35.358% | Test accuracy: 42.99%
==> Saving model ...
Iteration: 2 | Loss: 2.40171246303012 | Training accuracy: 42.136% | Test accuracy: 50.52%
Iteration: 3 | Loss: 2.085792493789702 | Training accuracy: 48.434% | Test accuracy: 54.83%
Iteration: 4 | Loss: 1.8882903026802766 | Training accuracy: 52.774% | Test accuracy: 59.63%
Iteration: 5 | Loss: 1.7602971520875117 | Training accuracy: 55.094% | Test accuracy: 60.21%
Iteration: 6 | Loss: 1.6348791982206847 | Training accuracy: 58.04% | Test accuracy: 62.43%
Iteration: 7 | Loss: 1.5522922139490962 | Training accuracy: 60.128% | Test accuracy: 63.67%
Iteration: 8 | Loss: 1.4837936880948293 | Training accuracy: 62.098% | Test accuracy: 64.31%
Iteration: 9 | Loss: 1.4072782235682164 | Training accuracy: 63.582% | Test accuracy: 66.54%
Iteration: 10 | Loss: 1.3392375324593874 | Training accuracy: 64.368% | Test accuracy: 67.82%
Iteration: 11 | Loss: 1.269238745329752 | Training accuracy: 66.144% | Test accuracy: 68.66%
Iteration: 12 | Loss: 1.2129387459374597 | Training accuracy: 67.342% | Test accuracy: 69.45%
Iteration: 13 | Loss: 1.16239485723947592 | Training accuracy: 68.824% | Test accuracy: 70.31%
Iteration: 14 | Loss: 1.16239485723947592 | Training accuracy: 69.583% | Test accuracy: 71.23%
```