```
import torch
import torch.nn as nn
import torch.nn.functional as F
import numpy as np
import matplotlib.pyplot as plt
```

```
from helper_dataset import get_dataloaders_mnist
from helper_train import train_model
from helper_plotting import plot_training_loss, plot_accuracy, show_examples
```

```
RANDOM_SEED = 1
BATCH_SIZE = 64
NUM_EPOCHS = 20
DEVICE = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
```

```
train_loader, valid_loader, test_loader = get_dataloaders_mnist(batch_size=BATCH_SIZE, validation_fraction=0.1)

# Checking the dataset
for images, labels in train_loader:
    print('Image batch dimensions:', images.shape)
    print('Image label dimensions:', labels.shape)
    print('Class labels of 10 examples:', labels[:10])
    break
```

```
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz to data/MNIST/raw/train-images-idx3-ubyte.gz
100%                                        9912422/9912422 [00:00<00:00, 14684834.26it/s]
Extracting data/MNIST/raw/train-images-idx3-ubyte.gz to data/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz
Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz to data/MNIST/raw/train-labels-idx1-ubyte.gz
100%                                        28881/28881 [00:00<00:00, 538151.25it/s]
Extracting data/MNIST/raw/train-labels-idx1-ubyte.gz to data/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz
Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz to data/MNIST/raw/t10k-images-idx3-ubyte.gz
100%                                        1648877/1648877 [00:00<00:00, 16577715.61it/s]
Extracting data/MNIST/raw/t10k-images-idx3-ubyte.gz to data/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz to data/MNIST/raw/t10k-labels-idx1-ubyte.gz
100%                                        4542/4542 [00:00<00:00, 67385.17it/s]
Extracting data/MNIST/raw/t10k-labels-idx1-ubyte.gz to data/MNIST/raw

Image batch dimensions: torch.Size([64, 1, 28, 28])
Image label dimensions: torch.Size([64])
Class labels of 10 examples: tensor([2, 9, 3, 0, 8, 2, 5, 2, 3, 4])
```

```
class MLPNet(nn.Module):
      def __init__(self):
         super(MLPNet, self).__init__()
         self.fc1 = nn.Linear(784, 256)
         self.fc2 = nn.Linear(256, 256)
         self.fc3 = nn.Linear(256, 10)

      def forward(self, x):
         x = x.view(-1, 28*28)
         x = F.relu(self.fc1(x))
         x = F.relu(self.fc2(x))
         x = self.fc3(x)
         return x

      def name(self):
         return "MLP"
```

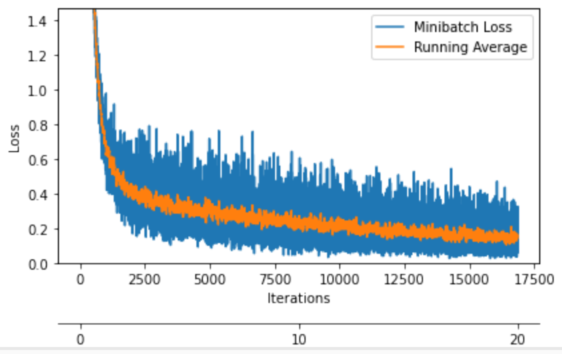
```
model = MLPNet()
```

```
optimizer = torch.optim.SGD(model.parameters(), lr=0.01, momentum=0)
```

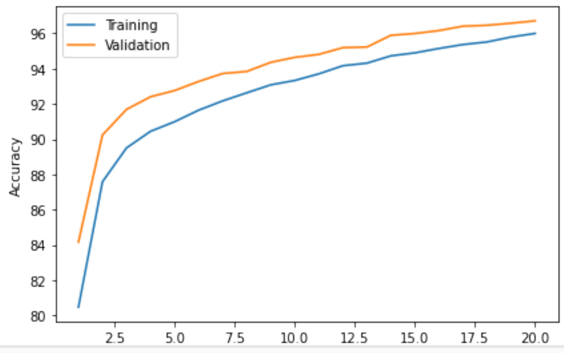
```
minibatch_loss_list, train_acc_list, valid_acc_list = train_model(
    model=model,
    num_epochs=NUM_EPOCHS,
    train_loader=train_loader,
    valid_loader=valid_loader,
    test_loader=test_loader,
    optimizer=optimizer,
    device=DEVICE)
```

```
Epoch: 001/020 | Batch 0000/0843 | Loss: 2.3064
Epoch: 001/020 | Batch 0050/0843 | Loss: 2.2820
Epoch: 001/020 | Batch 0100/0843 | Loss: 2.2714
Epoch: 001/020 | Batch 0150/0843 | Loss: 2.2457
Epoch: 001/020 | Batch 0200/0843 | Loss: 2.2197
Epoch: 001/020 | Batch 0250/0843 | Loss: 2.1419
Epoch: 001/020 | Batch 0300/0843 | Loss: 2.0879
Epoch: 001/020 | Batch 0350/0843 | Loss: 2.0785
Epoch: 001/020 | Batch 0400/0843 | Loss: 1.9656
Epoch: 001/020 | Batch 0450/0843 | Loss: 1.8305
Epoch: 001/020 | Batch 0500/0843 | Loss: 1.6400
Epoch: 001/020 | Batch 0550/0843 | Loss: 1.4777
Epoch: 001/020 | Batch 0600/0843 | Loss: 1.2426
Epoch: 001/020 | Batch 0650/0843 | Loss: 1.1712
Epoch: 001/020 | Batch 0700/0843 | Loss: 1.1498
Epoch: 001/020 | Batch 0750/0843 | Loss: 0.9303
Epoch: 001/020 | Batch 0800/0843 | Loss: 0.9504
Epoch: 001/020 | Train: 80.48% | Validation: 84.17%
Time elapsed: 0.26 min
Epoch: 002/020 | Batch 0000/0843 | Loss: 0.7361
Epoch: 002/020 | Batch 0050/0843 | Loss: 0.8466
Epoch: 002/020 | Batch 0100/0843 | Loss: 0.7648
Epoch: 002/020 | Batch 0150/0843 | Loss: 0.5560
Epoch: 002/020 | Batch 0200/0843 | Loss: 0.6794
Epoch: 002/020 | Batch 0250/0843 | Loss: 0.6891
Epoch: 002/020 | Batch 0300/0843 | Loss: 0.6461
Epoch: 002/020 | Batch 0350/0843 | Loss: 0.5024
Epoch: 002/020 | Batch 0400/0843 | Loss: 0.6198
Epoch: 002/020 | Batch 0450/0843 | Loss: 0.5461
Epoch: 002/020 | Batch 0500/0843 | Loss: 0.5128
Epoch: 002/020 | Batch 0550/0843 | Loss: 0.4064
Epoch: 002/020 | Batch 0600/0843 | Loss: 0.5421
Epoch: 002/020 | Batch 0650/0843 | Loss: 0.3653
Epoch: 002/020 | Batch 0700/0843 | Loss: 0.3928
Epoch: 002/020 | Batch 0750/0843 | Loss: 0.4154
Epoch: 002/020 | Batch 0800/0843 | Loss: 0.4329
Epoch: 002/020 | Train: 87.58% | Validation: 90.23%
Time elapsed: 0.53 min
Epoch: 003/020 | Batch 0000/0843 | Loss: 0.6005
Epoch: 003/020 | Batch 0050/0843 | Loss: 0.4965
Epoch: 003/020 | Batch 0100/0843 | Loss: 0.5014
Epoch: 003/020 | Batch 0150/0843 | Loss: 0.5441
Epoch: 003/020 | Batch 0200/0843 | Loss: 0.2967
Epoch: 003/020 | Batch 0250/0843 | Loss: 0.4320
Epoch: 003/020 | Batch 0300/0843 | Loss: 0.3660
Epoch: 003/020 | Batch 0350/0843 | Loss: 0.3337
Epoch: 003/020 | Batch 0400/0843 | Loss: 0.3882
Epoch: 003/020 | Batch 0450/0843 | Loss: 0.5051
Epoch: 003/020 | Batch 0500/0843 | Loss: 0.4188
Epoch: 003/020 | Batch 0550/0843 | Loss: 0.2438
Epoch: 003/020 | Batch 0600/0843 | Loss: 0.4691
Epoch: 003/020 | Batch 0650/0843 | Loss: 0.4528
Epoch: 003/020 | Batch 0700/0843 | Loss: 0.2964
Epoch: 003/020 | Batch 0750/0843 | Loss: 0.5412
Epoch: 003/020 | Batch 0800/0843 | Loss: 0.3299
Epoch: 003/020 | Train: 89.50% | Validation: 91.68%
Time elapsed: 0.79 min
Epoch: 004/020 | Batch 0000/0843 | Loss: 0.3632
```

```
plot_training_loss(minibatch_loss_list=minibatch_loss_list,
                   num_epochs=NUM_EPOCHS,
                   iter_per_epoch=len(train_loader),
                   results_dir=None,
                   averaging_iterations=20)
plt.show()
```



```
plot_accuracy(train_acc_list=train_acc_list,
              valid_acc_list=valid_acc_list,
              results_dir=None)
plt.show()
```



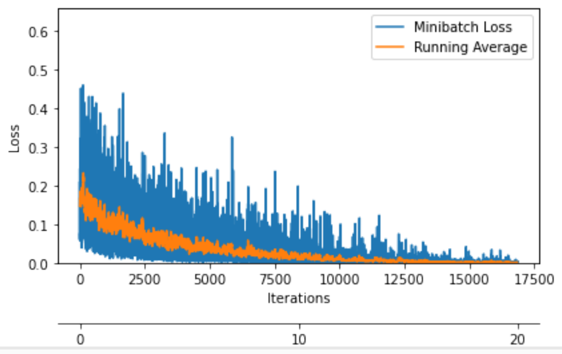
```
optimizer_sgb_momentum = torch.optim.SGD(model.parameters(), lr=0.01, momentum=0.92)
```

```
minibatch_loss_list_sgb_momentum, train_acc_list_sgb_momentum, valid_acc_list_sgb_momentum = train_model(
    model=model,
    num_epochs=NUM_EPOCHS,
    train_loader=train_loader,
    valid_loader=valid_loader,
    test_loader=test_loader,
    optimizer=optimizer_sgb_momentum,
    device=DEVICE)
```

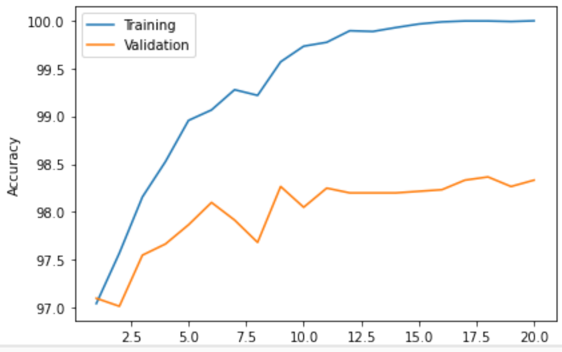
```
Epoch: 001/020 | Batch 0000/0843 | Loss: 0.0742
Epoch: 001/020 | Batch 0050/0843 | Loss: 0.1686
Epoch: 001/020 | Batch 0100/0843 | Loss: 0.2552
Epoch: 001/020 | Batch 0150/0843 | Loss: 0.2462
Epoch: 001/020 | Batch 0200/0843 | Loss: 0.0612
Epoch: 001/020 | Batch 0250/0843 | Loss: 0.1934
Epoch: 001/020 | Batch 0300/0843 | Loss: 0.1575
Epoch: 001/020 | Batch 0350/0843 | Loss: 0.1653
Epoch: 001/020 | Batch 0400/0843 | Loss: 0.0586
Epoch: 001/020 | Batch 0450/0843 | Loss: 0.0680
Epoch: 001/020 | Batch 0500/0843 | Loss: 0.2555
Epoch: 001/020 | Batch 0550/0843 | Loss: 0.2142
Epoch: 001/020 | Batch 0600/0843 | Loss: 0.1690
Epoch: 001/020 | Batch 0650/0843 | Loss: 0.0503
Epoch: 001/020 | Batch 0700/0843 | Loss: 0.2615
Epoch: 001/020 | Batch 0750/0843 | Loss: 0.0352
Epoch: 001/020 | Batch 0800/0843 | Loss: 0.2370
Epoch: 001/020 | Train: 97.05% | Validation: 97.10%
Time elapsed: 0.28 min
Epoch: 002/020 | Batch 0000/0843 | Loss: 0.1793
Epoch: 002/020 | Batch 0050/0843 | Loss: 0.2210
Epoch: 002/020 | Batch 0100/0843 | Loss: 0.0722
Epoch: 002/020 | Batch 0150/0843 | Loss: 0.0872
Epoch: 002/020 | Batch 0200/0843 | Loss: 0.1437
Epoch: 002/020 | Batch 0250/0843 | Loss: 0.1403
Epoch: 002/020 | Batch 0300/0843 | Loss: 0.0706
Epoch: 002/020 | Batch 0350/0843 | Loss: 0.0314
Epoch: 002/020 | Batch 0400/0843 | Loss: 0.1104
Epoch: 002/020 | Batch 0450/0843 | Loss: 0.0496
Epoch: 002/020 | Batch 0500/0843 | Loss: 0.0926
```

```
Epoch: 002/020 | Batch 0550/0843 | Loss: 0.0883
Epoch: 002/020 | Batch 0600/0843 | Loss: 0.1576
Epoch: 002/020 | Batch 0650/0843 | Loss: 0.0509
Epoch: 002/020 | Batch 0700/0843 | Loss: 0.1049
Epoch: 002/020 | Batch 0750/0843 | Loss: 0.1966
Epoch: 002/020 | Batch 0800/0843 | Loss: 0.0865
Epoch: 002/020 | Train: 97.57% | Validation: 97.02%
Time elapsed: 0.54 min
Epoch: 003/020 | Batch 0000/0843 | Loss: 0.0430
Epoch: 003/020 | Batch 0050/0843 | Loss: 0.0565
Epoch: 003/020 | Batch 0100/0843 | Loss: 0.1375
Epoch: 003/020 | Batch 0150/0843 | Loss: 0.1668
Epoch: 003/020 | Batch 0200/0843 | Loss: 0.0913
Epoch: 003/020 | Batch 0250/0843 | Loss: 0.0335
Epoch: 003/020 | Batch 0300/0843 | Loss: 0.0651
Epoch: 003/020 | Batch 0350/0843 | Loss: 0.0213
Epoch: 003/020 | Batch 0400/0843 | Loss: 0.0502
Epoch: 003/020 | Batch 0450/0843 | Loss: 0.0515
Epoch: 003/020 | Batch 0500/0843 | Loss: 0.0780
Epoch: 003/020 | Batch 0550/0843 | Loss: 0.0283
Epoch: 003/020 | Batch 0600/0843 | Loss: 0.1175
Epoch: 003/020 | Batch 0650/0843 | Loss: 0.1445
Epoch: 003/020 | Batch 0700/0843 | Loss: 0.0793
Epoch: 003/020 | Batch 0750/0843 | Loss: 0.2338
Epoch: 003/020 | Batch 0800/0843 | Loss: 0.0420
Epoch: 003/020 | Train: 98.16% | Validation: 97.55%
Time elapsed: 0.80 min
Epoch: 004/020 | Batch 0000/0843 | Loss: 0.0670
```

```
plot_training_loss(minibatch_loss_list=minibatch_loss_list_sgb_momentum,
                   num_epochs=NUM_EPOCHS,
                   iter_per_epoch=len(train_loader),
                   results_dir=None,
                   averaging_iterations=20)
plt.show()
```



```
plot_accuracy(train_acc_list=train_acc_list_sgb_momentum,
              valid_acc_list=valid_acc_list_sgb_momentum,
              results_dir=None)
plt.show()
```

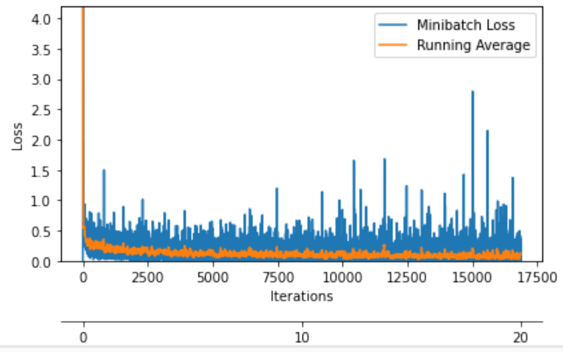


```
optimizer_rms = torch.optim.RMSprop(model.parameters())
```

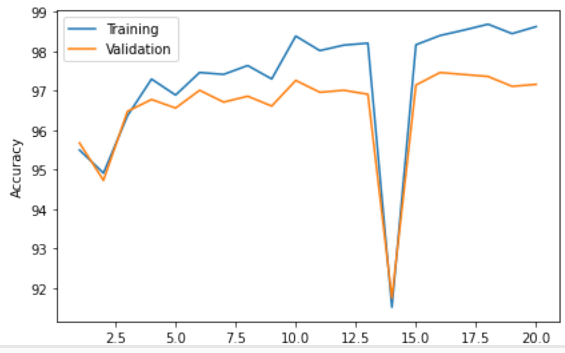
```
minibatch_loss_list_rms, train_acc_list_rms, valid_acc_list_rms = train_model(
    model=model,
    num_epochs=NUM_EPOCHS,
    train_loader=train_loader,
    valid_loader=valid_loader,
    test_loader=test_loader,
    optimizer=optimizer_rms,
    device=DEVICE)
```

```
Epoch: 006/020 | Batch 0650/0843 | Loss: 0.0819
Epoch: 006/020 | Batch 0700/0843 | Loss: 0.3162
Epoch: 006/020 | Batch 0750/0843 | Loss: 0.0842
Epoch: 006/020 | Batch 0800/0843 | Loss: 0.0732
Epoch: 006/020 | Train: 97.47% | Validation: 97.02%
Time elapsed: 1.73 min
Epoch: 007/020 | Batch 0000/0843 | Loss: 0.0119
Epoch: 007/020 | Batch 0050/0843 | Loss: 0.0805
Epoch: 007/020 | Batch 0100/0843 | Loss: 0.2219
Epoch: 007/020 | Batch 0150/0843 | Loss: 0.0493
Epoch: 007/020 | Batch 0200/0843 | Loss: 0.0462
Epoch: 007/020 | Batch 0250/0843 | Loss: 0.2187
Epoch: 007/020 | Batch 0300/0843 | Loss: 0.2451
Epoch: 007/020 | Batch 0350/0843 | Loss: 0.3057
Epoch: 007/020 | Batch 0400/0843 | Loss: 0.2495
Epoch: 007/020 | Batch 0450/0843 | Loss: 0.1984
Epoch: 007/020 | Batch 0500/0843 | Loss: 0.1722
Epoch: 007/020 | Batch 0550/0843 | Loss: 0.1303
Epoch: 007/020 | Batch 0600/0843 | Loss: 0.0221
Epoch: 007/020 | Batch 0650/0843 | Loss: 0.1270
Epoch: 007/020 | Batch 0700/0843 | Loss: 0.0039
Epoch: 007/020 | Batch 0750/0843 | Loss: 0.0125
Epoch: 007/020 | Batch 0800/0843 | Loss: 0.2498
Epoch: 007/020 | Train: 97.42% | Validation: 96.72%
Time elapsed: 2.04 min
Epoch: 008/020 | Batch 0000/0843 | Loss: 0.1346
Epoch: 008/020 | Batch 0050/0843 | Loss: 0.1335
Epoch: 008/020 | Batch 0100/0843 | Loss: 0.0342
Epoch: 008/020 | Batch 0150/0843 | Loss: 0.1120
Epoch: 008/020 | Batch 0200/0843 | Loss: 0.1735
Epoch: 008/020 | Batch 0250/0843 | Loss: 0.1324
Epoch: 008/020 | Batch 0300/0843 | Loss: 0.0278
Epoch: 008/020 | Batch 0350/0843 | Loss: 0.1847
Epoch: 008/020 | Batch 0400/0843 | Loss: 0.4127
Epoch: 008/020 | Batch 0450/0843 | Loss: 0.0970
Epoch: 008/020 | Batch 0500/0843 | Loss: 0.0736
Epoch: 008/020 | Batch 0550/0843 | Loss: 0.1048
Epoch: 008/020 | Batch 0600/0843 | Loss: 0.0749
Epoch: 008/020 | Batch 0650/0843 | Loss: 0.2858
Epoch: 008/020 | Batch 0700/0843 | Loss: 0.0166
Epoch: 008/020 | Batch 0750/0843 | Loss: 0.0904
Epoch: 008/020 | Batch 0800/0843 | Loss: 0.1155
Epoch: 008/020 | Train: 97.64% | Validation: 96.87%
Time elapsed: 2.39 min
Epoch: 009/020 | Batch 0000/0843 | Loss: 0.0703
Epoch: 009/020 | Batch 0050/0843 | Loss: 0.0187
Epoch: 009/020 | Batch 0100/0843 | Loss: 0.3137
Epoch: 009/020 | Batch 0150/0843 | Loss: 0.1655
Epoch: 009/020 | Batch 0200/0843 | Loss: 0.1347
Epoch: 009/020 | Batch 0250/0843 | Loss: 0.0044
Epoch: 009/020 | Batch 0300/0843 | Loss: 0.3080
Epoch: 009/020 | Batch 0350/0843 | Loss: 0.1854
Epoch: 009/020 | Batch 0400/0843 | Loss: 0.1072
Epoch: 009/020 | Batch 0450/0843 | Loss: 0.1098
Epoch: 009/020 | Batch 0500/0843 | Loss: 0.0317
Epoch: 009/020 | Batch 0550/0843 | Loss: 0.0052
Epoch: 009/020 | Batch 0600/0843 | Loss: 0.0054
Epoch: 009/020 | Batch 0650/0843 | Loss: 0.1117
Epoch: 009/020 | Batch 0700/0843 | Loss: 0.1366
```

```
plot_training_loss(minibatch_loss_list=minibatch_loss_list_rms,
                   num_epochs=NUM_EPOCHS,
                   iter_per_epoch=len(train_loader),
                   results_dir=None,
                   averaging_iterations=20)
plt.show()
```



```
plot_accuracy(train_acc_list=train_acc_list_rms,
              valid_acc_list=valid_acc_list_rms,
              results_dir=None)
plt.show()
```

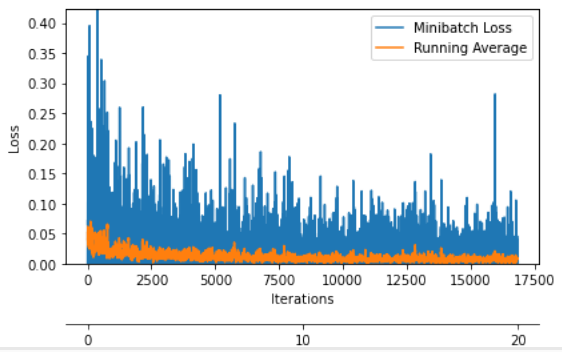


```
optimizer_adam = torch.optim.Adam(model.parameters())
```

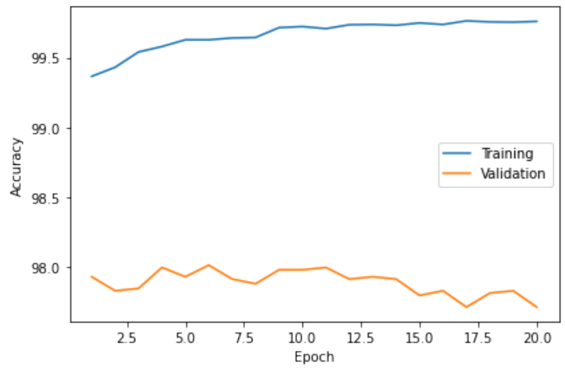
```
minibatch_loss_list_adam, train_acc_list_adam, valid_acc_list_adam = train_model(
    model=model,
    num_epochs=NUM_EPOCHS,
    train_loader=train_loader,
    valid_loader=valid_loader,
    test_loader=test_loader,
    optimizer=optimizer_adam,
    device=DEVICE)
```

```
Time elapsed: 2.82 min
Epoch: 010/020 | Batch 0000/0843 | Loss: 0.0002
Epoch: 010/020 | Batch 0050/0843 | Loss: 0.0221
Epoch: 010/020 | Batch 0100/0843 | Loss: 0.0014
Epoch: 010/020 | Batch 0150/0843 | Loss: 0.0235
Epoch: 010/020 | Batch 0200/0843 | Loss: 0.0002
Epoch: 010/020 | Batch 0250/0843 | Loss: 0.0000
Epoch: 010/020 | Batch 0300/0843 | Loss: 0.0255
Epoch: 010/020 | Batch 0350/0843 | Loss: 0.0223
Epoch: 010/020 | Batch 0400/0843 | Loss: 0.0000
Epoch: 010/020 | Batch 0450/0843 | Loss: 0.0008
Epoch: 010/020 | Batch 0500/0843 | Loss: 0.0001
Epoch: 010/020 | Batch 0550/0843 | Loss: 0.0058
Epoch: 010/020 | Batch 0600/0843 | Loss: 0.0000
Epoch: 010/020 | Batch 0650/0843 | Loss: 0.0422
Epoch: 010/020 | Batch 0700/0843 | Loss: 0.0002
Epoch: 010/020 | Batch 0750/0843 | Loss: 0.0006
Epoch: 010/020 | Batch 0800/0843 | Loss: 0.0418
Epoch: 010/020 | Train: 99.72% | Validation: 97.98%
Time elapsed: 3.18 min
Epoch: 011/020 | Batch 0000/0843 | Loss: 0.0000
Epoch: 011/020 | Batch 0050/0843 | Loss: 0.0004
Epoch: 011/020 | Batch 0100/0843 | Loss: 0.0001
Epoch: 011/020 | Batch 0150/0843 | Loss: 0.0001
Epoch: 011/020 | Batch 0200/0843 | Loss: 0.0000
Epoch: 011/020 | Batch 0250/0843 | Loss: 0.0432
Epoch: 011/020 | Batch 0300/0843 | Loss: 0.0008
```

```
plot_training_loss(minibatch_loss_list=minibatch_loss_list_adam,
                   num_epochs=NUM_EPOCHS,
                   iter_per_epoch=len(train_loader),
                   results_dir=None,
                   averaging_iterations=20)
plt.show()
```



```
plot_accuracy(train_acc_list=train_acc_list_adam,
              valid_acc_list=valid_acc_list_adam,
              results_dir=None)
plt.show()
```



Performance of optimizers

We can see that vanila SGD was able to converge although it had to do some epochs and there were some good number of oscillations too. The accuracy as well was not so good, it only had 96% of training accuracy and 95 % testing error, there wasn't much overfitting.

SGD with moment was much better than SGD, we can see that though there were oscillations at the starting, the oscillations reduced significantly as time progressed. The convergence of loss was also very good for this optimiser. There is a little overfitting here, the training accuracy shot upto 100% where as the test accuracy was 98%, overfitt but very very little.

RMS Prop optimiser was very good without many oscillations and the loss as well converged after only a few epochs. Even though there were few oscillations they were constant over time. The accuracies were good, there wasn't much overfitting as well.

The Adam optimiser even though a very good optimiser had a lot of oscillations and the generalisation error was also higher compared to the other optimisers. The training accuracy went upto 100% but test accuracy was below 98%

```
Start coding or generate with AI.
```