hw7_1

April 8, 2023

[84]:
```
import torch
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader
from torchvision import datasets, transforms
import torch.nn.functional as F
import torch.optim as optim
from tqdm import tqdm
import matplotlib.pyplot as plt
import numpy as np
import time
from skorch import NeuralNetClassifier
from sklearn.model_selection import GridSearchCV
```

[4]:
```
## this function is from Prof. Chugg's torch_fmnist_loader notebook
## https://github.com/keithchugg/ee559_spring2023/blob/main/hw_helpers/
 ↪torch_fmnist_loader.py

class FashionMNISTDataset(Dataset):
    def __init__(self, data):
        self.data = data

    def __getitem__(self, index):
        image, label = self.data[index]
        return image, label

    def __len__(self):
        return len(self.data)

transform = transforms.Compose([transforms.ToTensor(), transforms.Normalize((0.
 ↪5,), (0.5,)),])
trainset = datasets.FashionMNIST('~/.pytorch/F_MNIST_data/', download=True,␣
 ↪train=True, transform=transform)
testset = datasets.FashionMNIST('~/.pytorch/F_MNIST_data/', download=True,␣
 ↪train=False, transform=transform)
```

[5]:
```
trainset, valset = torch.utils.data.random_split(trainset, [48000, 12000])
```

```
print(f'Train set size: {len(trainset)}, Validation set size: {len(valset)}, Test set size: {len(testset)}')
```

```
Train set size: 48000, Validation set size: 12000, Test set size: 10000
```

1 ANN/MLP Model Definition

Start with =48 hidden nodes, = 0.01, = 10^-3 and = 32

[157]:
```
## this function is from Prof. Chugg's fmnist_mlp_torch notebook
## https://github.com/keithchugg/ee559_spring2023/blob/main/hw_helpers/fmnist_mlp_torch.py

# Parameters for the model
n_pixels = 28 * 28
n_classes = 10
n_hidden = 48

# Define the model
class MLP(nn.Module):
    def __init__(self, n_hidden): # Define layers in the constructor
        super().__init__()
        self.fc1 = nn.Linear(28 * 28, n_hidden)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(n_hidden, 10)

    def forward(self, x): # Define forward pass in the forward method
        x = x.view(x.shape[0], -1)  #flatten into a 784 length tensor
        x = self.relu(self.fc1(x))
        x = self.fc2(x)
        return x # note: no softmax, as this is included in the loss function in PyTorch
```

2 Train Model

[199]:
```
## this function is from Prof. Chugg's fmnist_mlp_torch notebook
## https://github.com/keithchugg/ee559_spring2023/blob/main/hw_helpers/fmnist_mlp_torch.py

# Define function to call for each training epoch (one complete pass over the training set)
def train(model, trainloader, criterion, optimizer, device):
    model.train() # set model to training mode
    running_loss = 0; running_acc = 0
#     with tqdm(total=len(trainloader), desc=f"Train", unit="batch") as pbar:
```

```
#         for n_batch, (images, labels) in enumerate(trainloader): # Iterate␣
↪over batches
#             images, labels = images.to(device), labels.to(device) # Move␣
↪batch to device
#             optimizer.zero_grad()
#             output = model(images) # Forward pass
#             loss = criterion(output, labels) # Compute loss
#             loss.backward() # Backward pass
#             optimizer.step() # Update weights
#             running_loss += loss.item()
#             running_acc += (output.argmax(1) == labels).float().mean().item()
#             pbar.set_postfix({'loss': loss.item(), 'acc': 100. * running_acc /
↪ (n_batch+1)})
#             pbar.update() # Update progress bar
    for n_batch, (images, labels) in enumerate(trainloader): # Iterate over␣
↪batches
            images, labels = images.to(device), labels.to(device) # Move batch␣
↪to device
            optimizer.zero_grad()
            output = model(images) # Forward pass
            loss = criterion(output, labels) # Compute loss
            loss.backward() # Backward pass
            optimizer.step() # Update weights
            running_loss += loss.item()
            running_acc += (output.argmax(1) == labels).float().mean().item()
    return running_loss / len(trainloader), running_acc / len(trainloader) #␣
↪return loss and accuracy for this epoch
```

[200]:

```
## this function is from Prof. Chugg's fmnist_mlp_torch notebook
## https://github.com/keithchugg/ee559_spring2023/blob/main/hw_helpers/
↪fmnist_mlp_torch.py

# Define function to call for each validation epoch (one complete pass over the␣
↪validation set)
def validate(model, valloader, criterion, device):
    model.eval() # set model to evaluation mode (e.g. turn off dropout,␣
↪batchnorm, etc.)
    running_loss = 0; running_acc = 0
    with torch.no_grad(): # no need to compute gradients for validation
#         with tqdm(total=len(valloader), desc=f"Eval", unit="batch") as pbar:
#             for n_batch, (images, labels) in enumerate(valloader): # Iterate␣
↪over batches
#                 images, labels = images.to(device), labels.to(device) # Move␣
↪batch to device
#                 output = model(images) # Forward pass
#                 loss = criterion(output, labels) # Compute loss
```

```
#                  running_loss += loss.item()
#                  running_acc += (output.argmax(1) == labels).float().mean().
↪item()
#                  pbar.set_postfix({'loss': loss.item(), 'acc': 100. *␣
↪running_acc / (n_batch+1)})
#                  pbar.update() # Update progress bar
        for n_batch, (images, labels) in enumerate(valloader): # Iterate over␣
↪batches
            images, labels = images.to(device), labels.to(device) # Move batch␣
↪to device
            output = model(images) # Forward pass
            loss = criterion(output, labels) # Compute loss
            running_loss += loss.item()
            running_acc += (output.argmax(1) == labels).float().mean().item()

    return running_loss / len(valloader), running_acc / len(valloader)  #␣
↪return loss and accuracy for this epoch
```

[173]:
```
def report_runtime(batchsize):

    runtime_history = []
    # Create a model
    model = MLP(n_hidden)
#         print(model)

    lr = 0.01   ## the learning rate in TF is part of the optimizer.  Default␣
↪is 1e-2
    reg_val = 1e-3
    criterion = nn.CrossEntropyLoss() # includes softmax (for numerical␣
↪stability)
    optimizer = optim.SGD(model.parameters(), lr=lr, weight_decay=reg_val)
    device = torch.device("cpu")
#         print(f'Using device: {device}')
    model.to(device) # Move model to device

    ite = 0
    for ite in range(5): # Run 5 times
        val_acc_checkpoint = -1
        epoch = 0
        # Shuffle the data at the start of each epoch (only useful for training␣
↪set)
        trainloader = torch.utils.data.DataLoader(trainset,␣
↪batch_size=batchsize, shuffle=True)
        valloader = torch.utils.data.DataLoader(valset, batch_size=batchsize,␣
↪shuffle=False)
```

```
        testloader = torch.utils.data.DataLoader(testset, batch_size=batchsize,␣
↪shuffle=False)

        start = time.time()
        while(val_acc_checkpoint <= 0.8):
            print(f"Epoch {epoch+1} in time {ite+1}")
            train_loss, train_acc  = train(model, trainloader, criterion,␣
↪optimizer, device) # Train
            val_loss, val_acc = validate(model, valloader, criterion, device) #␣
↪Validate
            val_acc_checkpoint = val_acc
#             train_loss_history.append(train_loss)
#             train_acc_history.append(train_acc)
#             val_loss_history.append(val_loss)
#             val_acc_history.append(val_acc)
            epoch += 1

        end = time.time()
        runtime = end - start
        runtime_history.append(runtime)
    print(f'The batch size is {batchsize}, and the mean is {np.
↪mean(runtime_history)}, the std is {np.std(runtime_history)}')
```

[176]:
```
report_runtime(32)
```

```
Epoch 1 in time 1

Train: 100%|    | 1500/1500 [00:10<00:00, 137.23batch/s, loss=0.452, acc=76.6]
Eval: 100%|     | 375/375 [00:02<00:00, 183.02batch/s, loss=0.611, acc=81.4]

Epoch 1 in time 2

Train: 100%|    | 1500/1500 [00:10<00:00, 139.36batch/s, loss=0.418, acc=82.8]
Eval: 100%|      | 375/375 [00:02<00:00, 179.11batch/s, loss=0.46, acc=83.2]

Epoch 1 in time 3

Train: 100%|    | 1500/1500 [00:10<00:00, 138.12batch/s, loss=0.516, acc=84.2]
Eval: 100%|     | 375/375 [00:02<00:00, 187.26batch/s, loss=0.494, acc=83.9]

Epoch 1 in time 4

Train: 100%|    | 1500/1500 [00:10<00:00, 142.40batch/s, loss=0.539, acc=84.9]
Eval: 100%|     | 375/375 [00:02<00:00, 183.65batch/s, loss=0.546, acc=84.3]

Epoch 1 in time 5

Train: 100%|    | 1500/1500 [00:10<00:00, 140.99batch/s, loss=0.258, acc=85.5]
Eval: 100%|     | 375/375 [00:01<00:00, 189.47batch/s, loss=0.487, acc=85.4]

The batch size is 32, and the mean is 12.781783056259155, the std is
0.15533816985575313
```

[177]:
```
report_runtime(16)
```

```
Epoch 1 in time 1

Train: 100%|    | 3000/3000 [00:15<00:00, 191.67batch/s, loss=0.348, acc=78.3]
Eval: 100%|     | 750/750 [00:02<00:00, 278.11batch/s, loss=0.684, acc=82.1]

Epoch 1 in time 2

Train: 100%|    | 3000/3000 [00:15<00:00, 192.16batch/s, loss=0.575, acc=83.7]
Eval: 100%|     | 750/750 [00:02<00:00, 282.17batch/s, loss=0.557, acc=83.9]

Epoch 1 in time 3

Train: 100%|    | 3000/3000 [00:16<00:00, 183.83batch/s, loss=0.306, acc=85.1]
Eval: 100%|     | 750/750 [00:02<00:00, 280.37batch/s, loss=0.621, acc=84.7]

Epoch 1 in time 4

Train: 100%|    | 3000/3000 [00:15<00:00, 195.21batch/s, loss=0.363, acc=85.8]
Eval: 100%|      | 750/750 [00:02<00:00, 278.45batch/s, loss=0.745, acc=86]

Epoch 1 in time 5

Train: 100%|    | 3000/3000 [00:15<00:00, 195.51batch/s, loss=0.403, acc=86.4]
Eval: 100%|      | 750/750 [00:02<00:00, 279.10batch/s, loss=0.648, acc=86]

The batch size is 16, and the mean is 18.344105577468873, the std is
0.34812015279238673
```

[174]:
```
report_runtime(64)
```

```
Epoch 1 in time 1

Train: 100%|     | 750/750 [00:07<00:00, 103.70batch/s, loss=0.543, acc=72.6]
Eval: 100%|     | 188/188 [00:01<00:00, 120.15batch/s, loss=0.714, acc=78.7]

Epoch 2 in time 1

Train: 100%|     | 750/750 [00:07<00:00, 106.36batch/s, loss=0.723, acc=80.9]
Eval: 100%|     | 188/188 [00:01<00:00, 119.93batch/s, loss=0.579, acc=81.4]

Epoch 1 in time 2

Train: 100%|     | 750/750 [00:07<00:00, 101.41batch/s, loss=0.412, acc=82.7]
Eval: 100%|     | 188/188 [00:01<00:00, 117.27batch/s, loss=0.495, acc=82.7]

Epoch 1 in time 3

Train: 100%|      | 750/750 [00:07<00:00, 104.89batch/s, loss=0.51, acc=83.6]
Eval: 100%|     | 188/188 [00:01<00:00, 120.18batch/s, loss=0.514, acc=83.3]

Epoch 1 in time 4
```

```
Train: 100%|    | 750/750 [00:07<00:00, 106.86batch/s, loss=0.469, acc=84.2]
Eval: 100%|    | 188/188 [00:01<00:00, 119.63batch/s, loss=0.485, acc=83.8]

Epoch 1 in time 5

Train: 100%|    | 750/750 [00:07<00:00, 103.82batch/s, loss=0.406, acc=84.7]
Eval: 100%|    | 188/188 [00:01<00:00, 118.31batch/s, loss=0.42, acc=84.4]

The batch size is 64, and the mean is 10.510799455642701, the std is
3.460327396677144
```

[175]:
```
report_runtime(128)
```

```
Epoch 1 in time 1

Train: 100%|    | 375/375 [00:06<00:00, 59.67batch/s, loss=0.684, acc=67.2]
Eval: 100%|    | 94/94 [00:01<00:00, 69.22batch/s, loss=0.816, acc=75.4]

Epoch 2 in time 1

Train: 100%|    | 375/375 [00:06<00:00, 59.62batch/s, loss=0.621, acc=77.4]
Eval: 100%|    | 94/94 [00:01<00:00, 67.08batch/s, loss=0.691, acc=78.4]

Epoch 3 in time 1

Train: 100%|    | 375/375 [00:06<00:00, 58.70batch/s, loss=0.52, acc=80]
Eval: 100%|    | 94/94 [00:01<00:00, 69.30batch/s, loss=0.63, acc=80.3]

Epoch 1 in time 2

Train: 100%|    | 375/375 [00:06<00:00, 59.94batch/s, loss=0.413, acc=81.5]
Eval: 100%|    | 94/94 [00:01<00:00, 68.80batch/s, loss=0.591, acc=81.3]

Epoch 1 in time 3

Train: 100%|    | 375/375 [00:06<00:00, 59.43batch/s, loss=0.576, acc=82.3]
Eval: 100%|    | 94/94 [00:01<00:00, 67.95batch/s, loss=0.558, acc=82.3]

Epoch 1 in time 4

Train: 100%|    | 375/375 [00:06<00:00, 59.20batch/s, loss=0.439, acc=83]
Eval: 100%|    | 94/94 [00:01<00:00, 67.02batch/s, loss=0.538, acc=82.7]

Epoch 1 in time 5

Train: 100%|    | 375/375 [00:06<00:00, 58.89batch/s, loss=0.426, acc=83.5]
Eval: 100%|    | 94/94 [00:01<00:00, 64.57batch/s, loss=0.504, acc=83.1]

The batch size is 128, and the mean is 10.795682621002197, the std is
6.1475982896351065
```

2.1 Using batchsize = 64 according to the result above

[158]:
```
def grid_search():
    # create model with skorch
    model = NeuralNetClassifier(
        MLP,
        criterion=nn.CrossEntropyLoss,
        optimizer=optim.SGD,
        max_epochs=30,
        batch_size=64,
        lr=1e-02,
        iterator_train__shuffle=True
    )
    # define the grid search parameters
    param_grid = {
        'optimizer__lr': [0.001, 0.01, 0.1],
        'optimizer__weight_decay': [1e-04, 1e-03, 1e-02],
        'module__n_hidden': [40, 80, 160]
    }
    grid = GridSearchCV(estimator=model, param_grid=param_grid, n_jobs=-1,␣
↪cv=3, scoring='accuracy',return_train_score = True)
    trainloader = torch.utils.data.DataLoader(trainset, batch_size=64,␣
↪shuffle=False)

    X_train = np.array([X_train.cpu().detach().numpy() for X_train, y_train in␣
↪trainloader])
    X_train = np.squeeze(np.concatenate(X_train, axis=0))
    y_train = np.array([y_train.cpu().detach().numpy() for X_train, y_train in␣
↪trainloader])
    y_train = np.squeeze(np.concatenate(y_train, axis=0))
    grid_result = grid.fit(X_train,y_train)

    return grid_result
```

[143]:
```
grid_result = grid_search()
```

```
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  ------
      1        1.5381       0.7053        1.0495
1.3321
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  ------
      1        1.4536       0.7103        1.0052
1.3613
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  ------
      1        0.7201       0.8050        0.5398
```

```
1.3827
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.4985       0.7120        1.0097
1.4039
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.5301       0.7048        1.0365
1.4641
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.4608       0.7102        0.9700
1.4135
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.5022       0.7013        1.0278
1.4805
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.4664       0.7136        0.9741
1.4221
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.4430       0.7125        0.9813
1.4953
  epoch    train_loss    valid_acc    valid_loss    dur
-------  ------------  -----------  ------------  ------
      1        1.4218       0.7234        0.9705
1.4497
      2        0.8856       0.7392        0.7859
1.1391
      2        0.8555       0.7478        0.7662
1.1109
      2        0.8328       0.7559        0.7474
1.1180
      2        0.8602       0.7459        0.7738
1.1774
      2        0.8749       0.7453        0.7770
1.1427
      2        0.8396       0.7448        0.7531
1.1224
      2        0.8309       0.7541        0.7448
1.1173
      2        0.4952       0.8239        0.4933
1.1989
      2        0.8729       0.7402        0.7855
1.1588
      2        0.8372       0.7472        0.7548
```

```
1.1601
     3       0.7183      0.7641       0.6913
1.0385
     3       0.7006      0.7738       0.6662
1.0847
     3       0.7091      0.7677       0.6816
1.1097
     3       0.7276      0.7539       0.6977
1.1669
     3       0.4517      0.8406       0.4549
1.0724
     3       0.7198      0.7650       0.6853
1.1290
     3       0.7297      0.7580       0.6983
1.0959
     3       0.7105      0.7698       0.6744
1.1406
     3       0.7022      0.7742       0.6701
1.1342
     3       0.6944      0.7739       0.6632
1.1698
     4       0.6553      0.7723       0.6479
1.1429
     4       0.6596      0.7678       0.6495
1.1301
     4       0.6427      0.7800       0.6329
1.1583
     4       0.4255      0.8417       0.4469
1.1375
     4       0.6519      0.7812       0.6329
1.1400
     4       0.6386      0.7881       0.6176
1.2097
     4       0.6660      0.7741       0.6490
1.1555
     4       0.6521      0.7830       0.6307
1.1694
     4       0.6414      0.7855       0.6242
1.1856
     4       0.6337      0.7841       0.6149
1.1553
     5       0.6165      0.7852       0.6168
1.1283
     5       0.6177      0.7809       0.6160
1.1469
     5       0.6269      0.7842       0.6145
1.1023
     5       0.5986      0.7986       0.5834
```

1.1409				
	5	0.6007	0.7906	0.5998
1.1801				
	5	0.6105	0.7922	0.5990
1.1615				
	5	0.6022	0.7945	0.5901
1.1004				
	5	0.6155	0.7923	0.5991
1.1694				
	5	0.4082	0.8453	0.4355
1.2452				
	5	0.5958	0.7987	0.5828
1.1639				
	6	0.5892	0.7917	0.5943
1.1501				
	6	0.5983	0.7963	0.5879
1.1036				
	6	0.5871	0.7841	0.5926
1.1444				
	6	0.5816	0.8017	0.5710
1.1177				
	6	0.5700	0.8084	0.5590
1.1437				
	6	0.5710	0.7992	0.5740
1.1752				
	6	0.5888	0.8044	0.5757
1.1278				
	6	0.5746	0.8031	0.5642
1.1537				
	6	0.5686	0.8072	0.5582
1.1379				
	6	0.3914	0.8489	0.4291
1.1702				
	7	0.5683	0.8000	0.5751
1.0900				
	7	0.5768	0.8033	0.5677
1.0848				
	7	0.5638	0.7956	0.5695
1.0959				
	7	0.5478	0.8159	0.5378
1.0925				
	7	0.5593	0.8086	0.5525
1.1319				
	7	0.5681	0.8136	0.5557
1.0859				
	7	0.5484	0.8042	0.5581
1.1061				
	7	0.5527	0.8117	0.5433

1.1078				
7	0.5476	0.8128	0.5402	
1.1082				
7	0.3802	0.8406	0.4455	1.1231
8	0.5514	0.8048	0.5594	
1.0834				
8	0.5598	0.8092	0.5515	
1.1175				
8	0.5442	0.8020	0.5546	
1.1437				
8	0.5309	0.8213	0.5232	
1.1548				
8	0.5306	0.8106	0.5405	
1.1093				
8	0.5419	0.8137	0.5346	
1.1305				
8	0.5518	0.8173	0.5412	
1.1426				
8	0.5351	0.8184	0.5261	
1.1162				
8	0.3697	0.8558	0.4111	
1.1059				
8	0.5308	0.8172	0.5250	
1.1423				
9	0.5278	0.8175	0.5205	
1.1206				
9	0.5457	0.8150	0.5385	
1.2550				
9	0.5163	0.8230	0.5109	
1.2016				
9	0.5286	0.8070	0.5404	
1.2575				
9	0.5380	0.8189	0.5322	
1.2047				
9	0.5376	0.8092	0.5483	
1.4088				
9	0.5159	0.8161	0.5283	
1.2508				
9	0.3607	0.8523	0.4113	1.2143
9	0.5166	0.8216	0.5133	
1.2109				
9	0.5205	0.8219	0.5135	
1.3101				
10	0.5158	0.8216	0.5106	
1.1247				
10	0.5340	0.8166	0.5306	
1.2169				
10	0.5148	0.8127	0.5278	

1.1626				
10	0.5265	0.8095	0.5396	
1.1794				
10	0.5050	0.8283	0.5000	
1.2635				
10	0.5269	0.8220	0.5200	
1.2149				
10	0.5053	0.8253	0.5029	
1.1551				
10	0.5039	0.8180	0.5203	
1.2260				
10	0.3514	0.8630	0.3964	
1.1815				
10	0.5083	0.8211	0.5053	1.2041
11	0.5059	0.8211	0.5006	1.1249
11	0.5241	0.8228	0.5191	
1.0609				
11	0.5036	0.8158	0.5192	
1.1588				
11	0.5165	0.8150	0.5306	
1.1391				
11	0.5169	0.8241	0.5116	
1.1093				
11	0.4949	0.8286	0.4913	
1.1159				
11	0.4932	0.8220	0.5098	
1.1122				
11	0.3442	0.8612	0.3978	1.1147
11	0.4953	0.8273	0.4939	
1.1540				
11	0.4984	0.8259	0.4932	
1.1425				
12	0.4973	0.8277	0.4921	
1.1592				
12	0.5160	0.8255	0.5110	
1.1647				
12	0.4933	0.8144	0.5134	1.0911
12	0.5087	0.8284	0.5037	
1.0864				
12	0.5086	0.8164	0.5235	
1.1330				
12	0.4862	0.8273	0.4882	1.1502
12	0.4869	0.8297	0.4855	
1.1113				
12	0.4843	0.8252	0.5008	
1.1545				
12	0.3357	0.8638	0.3966	1.2008
12	0.4896	0.8292	0.4856	

1.1816				
13	0.4897	0.8291	0.4853	
1.2483				
13	0.4850	0.8213	0.5042	
1.1978				
13	0.5084	0.8263	0.5051	
1.2518				
13	0.5009	0.8300	0.4972	
1.2207				
13	0.4766	0.8253	0.4944	
1.1534				
13	0.5013	0.8191	0.5166	
1.2282				
13	0.4797	0.8327	0.4794	
1.1983				
13	0.4787	0.8380	0.4774	
1.2197				
13	0.3295	0.8581	0.3962	1.1735
13	0.4817	0.8355	0.4781	
1.2687				
14	0.4824	0.8305	0.4788	
1.1573				
14	0.5021	0.8319	0.4967	
1.1486				
14	0.4690	0.8292	0.4876	
1.0888				
14	0.4774	0.8203	0.4998	1.1787
14	0.4949	0.8223	0.5128	
1.1205				
14	0.4946	0.8342	0.4921	
1.1678				
14	0.3230	0.8586	0.3969	1.1429
14	0.4719	0.8348	0.4741	1.1893
14	0.4726	0.8345	0.4741	
1.2276				
14	0.4750	0.8363	0.4717	
1.1381				
15	0.4769	0.8355	0.4747	
1.1450				
15	0.4963	0.8319	0.4921	1.1194
15	0.4706	0.8267	0.4905	
1.1409				
15	0.4627	0.8291	0.4848	1.1789
15	0.4889	0.8330	0.4856	1.1531
15	0.4895	0.8213	0.5069	1.1784
15	0.3162	0.8666	0.3801	
1.1484				
15	0.4665	0.8373	0.4680	

1.1413				
15	0.4661	0.8391	0.4692	
1.1932				
15	0.4687	0.8403	0.4662	
1.1499				
16	0.4907	0.8311	0.4895	1.0820
16	0.4712	0.8342	0.4684	1.1427
16	0.4645	0.8278	0.4855	
1.1136				
16	0.4566	0.8298	0.4807	
1.1090				
16	0.4842	0.8231	0.5027	
1.0844				
16	0.4834	0.8355	0.4817	
1.0984				
16	0.3094	0.8614	0.3954	1.1324
16	0.4607	0.8402	0.4649	
1.0858				
16	0.4612	0.8397	0.4640	
1.1464				
16	0.4628	0.8380	0.4605	1.0612
17	0.4862	0.8353	0.4836	
1.0687				
17	0.4657	0.8387	0.4630	
1.0917				
17	0.4797	0.8225	0.5027	1.0742
17	0.4585	0.8281	0.4836	
1.1208				
17	0.4518	0.8297	0.4767	1.1068
17	0.4788	0.8377	0.4771	
1.1056				
17	0.4554	0.8430	0.4594	
1.1113				
17	0.3043	0.8612	0.3943	1.1468
17	0.4561	0.8408	0.4598	
1.1482				
17	0.4578	0.8389	0.4560	1.1106
18	0.4820	0.8377	0.4798	
1.1022				
18	0.4611	0.8392	0.4603	
1.0614				
18	0.4757	0.8256	0.4953	
1.0845				
18	0.4747	0.8353	0.4751	1.0747
18	0.4532	0.8313	0.4762	
1.1128				
18	0.4465	0.8325	0.4710	
1.1305				

18	0.4506	0.8400	0.4586	1.0827
18	0.2987	0.8603	0.3975	1.0803
18	0.4512	0.8417	0.4556	1.1057
18	0.4527	0.8419	0.4527	1.0928
19	0.4781	0.8391	0.4766	1.1419
19	0.4569	0.8417	0.4566	1.1391
19	0.4719	0.8273	0.4911	1.1448
19	0.4706	0.8413	0.4703	1.1441
19	0.4416	0.8331	0.4673	1.1404
19	0.4483	0.8317	0.4720	1.1688
19	0.2945	0.8709	0.3769	1.0998
19	0.4466	0.8445	0.4529	1.1284
19	0.4481	0.8414	0.4517	1.1499
19	0.4470	0.8434	0.4533	1.2036
20	0.4524	0.8423	0.4539	1.1494
20	0.4744	0.8375	0.4738	1.1715
20	0.4667	0.8375	0.4705	1.1063
20	0.4687	0.8295	0.4888	1.1522
20	0.4441	0.8350	0.4692	1.1083
20	0.4425	0.8472	0.4494	1.0745
20	0.2899	0.8647	0.3828	1.1004
20	0.4378	0.8375	0.4620	1.1820
20	0.4439	0.8406	0.4449	1.0973
20	0.4428	0.8444	0.4488	1.1351
21	0.4487	0.8427	0.4507	1.0884
21	0.4711	0.8414	0.4715	1.1069
21	0.4636	0.8433	0.4655	1.1352
21	0.4655	0.8270	0.4890	1.1319

21	0.4387	0.8455	0.4475	1.0894
21	0.4398	0.8328	0.4692	1.1205
21	0.4339	0.8377	0.4591	1.1460
21	0.2849	0.8627	0.3934	1.1940
21	0.4396	0.8455	0.4415	1.1325
21	0.4391	0.8450	0.4468	1.0879
22	0.4449	0.8459	0.4464	1.1110
22	0.4680	0.8419	0.4677	1.1510
22	0.4603	0.8414	0.4631	1.1155
22	0.4351	0.8483	0.4440	1.0924
22	0.4357	0.8359	0.4631	1.1384
22	0.4299	0.8358	0.4585	1.1042
22	0.4626	0.8305	0.4840	1.2259
22	0.2793	0.8656	0.3823	1.1366
22	0.4360	0.8441	0.4382	1.1613
22	0.4354	0.8444	0.4451	1.1724
23	0.4418	0.8445	0.4434	1.1022
23	0.4573	0.8442	0.4595	1.1095
23	0.4653	0.8431	0.4656	1.1677
23	0.4318	0.8458	0.4412	1.1438
23	0.4317	0.8372	0.4605	1.1255
23	0.2756	0.8648	0.3867	1.1155
23	0.4264	0.8363	0.4555	1.1585
23	0.4596	0.8292	0.4830	1.2260
23	0.4321	0.8428	0.4418	1.1456
23	0.4324	0.8467	0.4373	1.2031
24	0.4383	0.8486	0.4417	1.3662
24	0.4541	0.8431	0.4586	1.2474
24	0.4285	0.8398	0.4571	1.1857
24	0.4620	0.8438	0.4642	1.2750
24	0.4287	0.8492	0.4392	1.3671
24	0.4573	0.8328	0.4793	

1.2157

24	0.4231	0.8392	0.4530	

1.2860

24	0.2707	0.8662	0.3850	1.3422
24	0.4280	0.8456	0.4393	

1.1973

24	0.4290	0.8461	0.4350	1.2493
25	0.4248	0.8366	0.4567	1.1306
25	0.4522	0.8444	0.4564	

1.1707

25	0.4597	0.8445	0.4607	

1.1303

25	0.4355	0.8448	0.4391	1.1804
25	0.4257	0.8464	0.4406	1.0523
25	0.4548	0.8323	0.4778	1.0762
25	0.4197	0.8409	0.4495	

1.1262

25	0.2655	0.8669	0.3843	1.1163
25	0.4256	0.8492	0.4352	

1.1117

25	0.4257	0.8494	0.4315	

1.1144

26	0.4218	0.8381	0.4522	1.1055
26	0.4573	0.8436	0.4600	1.0793
26	0.4227	0.8506	0.4355	

1.0644

26	0.4497	0.8445	0.4554	

1.1332

26	0.4320	0.8480	0.4366	1.1701
26	0.4524	0.8345	0.4766	

1.0785

26	0.4170	0.8400	0.4460	1.1211
26	0.2616	0.8634	0.3873	1.1094
26	0.4228	0.8486	0.4349	1.0856
26	0.4229	0.8477	0.4291	1.0786
27	0.4190	0.8389	0.4488	1.0958
27	0.4552	0.8436	0.4603	1.0980
27	0.4199	0.8464	0.4350	1.1134
27	0.4473	0.8478	0.4516	

1.0961

27	0.4292	0.8477	0.4355	1.0886
27	0.4501	0.8352	0.4744	

1.1101

27	0.4135	0.8403	0.4456	1.0776
27	0.2572	0.8658	0.3930	1.1010
27	0.4201	0.8486	0.4316	1.1318
27	0.4199	0.8502	0.4271	

1.0936

```
     28        0.4176       0.8498        0.4341  1.0635
     28        0.4531       0.8456        0.4552
1.1035
     28        0.4158       0.8398        0.4482  1.1156
     28        0.4455       0.8464        0.4515  1.1315
     28        0.4268       0.8469        0.4338  1.1158
     28        0.4483       0.8341        0.4728  1.1423
     28        0.4112       0.8427        0.4434
1.1074
     28        0.2558       0.8670        0.3851  1.1193
     28        0.4172       0.8514        0.4231
1.0867
     28        0.4170       0.8512        0.4292
1.1365
     29        0.4149       0.8494        0.4308  1.0669
     29        0.4128       0.8411        0.4481
1.0763
     29        0.4510       0.8452        0.4558  1.1160
     29        0.4430       0.8475        0.4502  1.1046
     29        0.4240       0.8472        0.4306  1.0932
     29        0.4462       0.8322        0.4735  1.0700
     29        0.4082       0.8433        0.4405
1.1457
     29        0.2509       0.8700        0.3769  1.1289
     29        0.4141       0.8511        0.4210  1.1045
     29        0.4148       0.8491        0.4281  1.1071
     30        0.4124       0.8511        0.4316  1.1742
     30        0.4105       0.8439        0.4444
1.1412
     30        0.4493       0.8483        0.4525
1.1954
     30        0.4415       0.8472        0.4480  1.1674
     30        0.4448       0.8344        0.4701  1.1583
     30        0.4211       0.8512        0.4272
1.2023
     30        0.2479       0.8681        0.3829  1.0960
     30        0.4113       0.8506        0.4216  1.1162
     30        0.4057       0.8430        0.4377  1.2076
     30        0.4123       0.8500        0.4256  1.2912
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7189       0.8114        0.5257
1.2964
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7280       0.8047        0.5345
1.3101
  epoch    train_loss    valid_acc    valid_loss     dur
```

```
-------  ------------  -----------  ------------  ------
      1        0.7231       0.8078        0.5362
1.2333
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7394       0.8061        0.5344
1.1445
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7445       0.8127        0.5319
1.2524
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7356       0.8075        0.5605
1.2339
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7347       0.8172        0.5284
1.1753
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7343       0.8045        0.5506
1.1388
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6159       0.8302        0.4710
1.1884
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6188       0.8320        0.4619
1.2794
      2        0.4947       0.8319        0.4732
1.1108
      2        0.4969       0.8311        0.4807
1.1077
      2        0.5007       0.8352        0.4636
1.2297
      2        0.4958       0.8302        0.4665
1.2394
      2        0.5121       0.8252        0.4912
1.0807
      2        0.5113       0.8223        0.5001
1.1631
      2        0.4995       0.8287        0.4838
1.1705
      2        0.5170       0.8270        0.4936
1.1566
      2        0.4706       0.8416        0.4503
```

1.1237
2 0.4679 0.8389 0.4437
1.1598
3 0.4515 0.8383 0.4518
1.0750
3 0.4572 0.8442 0.4363
1.1014
3 0.4531 0.8359 0.4645
1.1447
3 0.4520 0.8384 0.4471
1.1106
3 0.4556 0.8220 0.4735 1.0778
3 0.4760 0.8245 0.4903
1.1099
3 0.4752 0.8378 0.4624
1.1589
3 0.4806 0.8398 0.4640
1.1685
3 0.4285 0.8363 0.4703 1.1263
3 0.4253 0.8464 0.4165
1.1126
4 0.4258 0.8494 0.4241
1.0826
4 0.4314 0.8459 0.4251
1.1195
4 0.4273 0.8337 0.4551 1.1577
4 0.4261 0.8517 0.4233
1.0994
4 0.4565 0.8341 0.4645 1.0755
4 0.4575 0.8367 0.4667
1.1458
4 0.4290 0.8462 0.4315
1.1925
4 0.4605 0.8436 0.4546
1.1763
4 0.4028 0.8534 0.4148
1.1729
4 0.3945 0.8589 0.4081
1.1960
5 0.4073 0.8458 0.4282 1.1643
5 0.4132 0.8523 0.4083
1.1874
5 0.4432 0.8489 0.4419
1.1137
5 0.4090 0.8434 0.4389
1.2163
5 0.4079 0.8459 0.4227 1.2536
5 0.4464 0.8405 0.4571

1.1727				
5	0.4104	0.8545	0.4170	
1.2349				
5	0.4465	0.8538	0.4404	
1.1743				
5	0.3779	0.8536	0.4143	
1.1968				
5	0.3769	0.8342	0.4860	1.2219
6	0.3910	0.8570	0.4101	
1.3204				
6	0.3989	0.8531	0.4030	
1.2191				
6	0.3937	0.8478	0.4200	1.1368
6	0.4346	0.8453	0.4395	1.3228
6	0.3933	0.8438	0.4272	
1.3251				
6	0.4387	0.8436	0.4595	1.2772
6	0.4381	0.8462	0.4408	1.2015
6	0.3682	0.8555	0.4057	
1.2179				
6	0.3959	0.8580	0.4035	
1.3843				
6	0.3626	0.8544	0.4085	1.2590
7	0.3794	0.8541	0.4064	1.2456
7	0.3870	0.8572	0.4014	
1.1514				
7	0.3810	0.8619	0.3940	
1.1356				
7	0.3836	0.8361	0.4605	1.1163
7	0.4282	0.8500	0.4413	1.1486
7	0.4316	0.8383	0.4545	1.1656
7	0.4312	0.8539	0.4334	
1.1732				
7	0.3838	0.8614	0.3874	
1.1031				
7	0.3499	0.8472	0.4539	1.1594
7	0.3478	0.8575	0.4117	1.1133
8	0.3670	0.8633	0.3844	
1.0563				
8	0.3759	0.8648	0.3886	
1.1346				
8	0.3706	0.8598	0.3881	1.1378
8	0.3734	0.8500	0.4134	
1.1387				
8	0.4213	0.8427	0.4436	1.1588
8	0.4262	0.8363	0.4607	1.1134
8	0.4256	0.8555	0.4227	
1.0881				

8	0.3747	0.8544	0.3946	1.1323
8	0.3446	0.8548	0.4255	1.0884
8	0.3370	0.8622	0.3937	
1.1616				
9	0.3576	0.8617	0.3955	1.0715
9	0.3685	0.8642	0.3792	1.1096
9	0.3612	0.8620	0.3843	
1.0846				
9	0.4165	0.8417	0.4482	1.1175
9	0.3630	0.8530	0.4163	1.1630
9	0.4233	0.8416	0.4543	1.1103
9	0.4215	0.8478	0.4314	1.0931
9	0.3290	0.8073	0.5926	1.1086
9	0.3653	0.8589	0.3841	1.1371
9	0.3286	0.8708	0.3642	
1.1136				
10	0.3479	0.8662	0.3813	
1.1246				
10	0.3568	0.8611	0.3841	1.1998
10	0.3532	0.8669	0.3745	
1.1864				
10	0.4132	0.8530	0.4220	
1.1411				
10	0.3558	0.8605	0.4036	
1.1564				
10	0.4196	0.8423	0.4432	1.1635
10	0.4181	0.8517	0.4253	1.1328
10	0.3205	0.8534	0.4322	1.1018
10	0.3571	0.8523	0.4053	1.1500
10	0.3243	0.8666	0.3989	1.1341
11	0.3391	0.8661	0.3755	1.0853
11	0.3516	0.8652	0.3757	
1.1156				
11	0.3473	0.8641	0.3782	1.1488
11	0.4121	0.8541	0.4259	1.1261
11	0.4165	0.8462	0.4406	
1.1147				
11	0.4154	0.8544	0.4290	1.1008
11	0.3481	0.8573	0.4013	1.1617
11	0.3110	0.8539	0.4380	1.0616
11	0.3493	0.8617	0.3903	1.1186
11	0.3130	0.8666	0.4015	1.1274
12	0.3306	0.8661	0.3729	1.1416
12	0.3428	0.8673	0.3767	1.1666
12	0.3398	0.8586	0.3852	1.1601
12	0.4088	0.8516	0.4293	1.1861
12	0.4127	0.8541	0.4206	1.1401
12	0.4137	0.8333	0.4571	1.1626

12	0.3057	0.8617	0.4161	1.1396
12	0.3415	0.8520	0.4167	1.1520
12	0.3451	0.8686	0.3718	
1.0948				
12	0.3073	0.8661	0.3876	1.1036
13	0.3243	0.8705	0.3649	
1.0907				
13	0.3356	0.8702	0.3770	1.1444
13	0.3331	0.8633	0.3792	1.0805
13	0.4063	0.8561	0.4227	1.1367
13	0.4102	0.8431	0.4394	1.1103
13	0.3355	0.8617	0.3918	
1.1070				
13	0.4114	0.8536	0.4251	1.1346
13	0.2998	0.8647	0.3996	
1.1349				
13	0.3390	0.8739	0.3620	
1.1361				
13	0.3005	0.8428	0.4785	1.1012
14	0.3184	0.8714	0.3662	1.0865
14	0.3290	0.8727	0.3641	
1.1186				
14	0.3278	0.8733	0.3656	
1.1135				
14	0.3300	0.8622	0.3859	
1.0724				
14	0.4080	0.8575	0.4161	
1.0871				
14	0.4098	0.8464	0.4418	1.1239
14	0.4064	0.8444	0.4324	1.1494
14	0.2912	0.8600	0.4214	1.1463
14	0.3332	0.8686	0.3699	1.1295
14	0.2906	0.8495	0.4518	1.1397
15	0.3114	0.8730	0.3631	
1.0843				
15	0.3228	0.8606	0.3807	1.0894
15	0.3227	0.8653	0.3759	1.1204
15	0.3249	0.8506	0.4188	1.0950
15	0.4061	0.8514	0.4293	1.1341
15	0.4037	0.8581	0.4137	
1.1000				
15	0.4075	0.8505	0.4365	
1.1225				
15	0.2858	0.8586	0.4376	1.1412
15	0.3278	0.8728	0.3598	1.0858
15	0.2818	0.8520	0.4219	1.1421
16	0.3066	0.8719	0.3651	1.1069
16	0.3171	0.8727	0.3588	1.1045

16	0.3188	0.8686	0.3617	1.1232
16	0.3196	0.8650	0.3877	1.0961
16	0.4066	0.8589	0.4121	1.0950
16	0.4071	0.8438	0.4406	1.1387
16	0.2784	0.8562	0.4366	1.0790
16	0.3231	0.8695	0.3680	1.0947
16	0.4033	0.8477	0.4340	1.1864
16	0.2788	0.8625	0.4325	1.0975
17	0.3009	0.8714	0.3659	1.1265
17	0.3109	0.8716	0.3591	1.0766
17	0.3129	0.8700	0.3674	1.0944
17	0.3154	0.8653	0.3855	1.0911
17	0.4030	0.8544	0.4196	1.1055
17	0.4047	0.8517	0.4370	1.1187
17	0.2756	0.8562	0.4280	1.1158
17	0.4012	0.8527	0.4247	1.1345
17	0.3183	0.8755	0.3577	1.1403
17	0.2803	0.8570	0.4643	1.1148
18	0.2950	0.8747	0.3569	1.1100
18	0.3053	0.8716	0.3648	1.2048
18	0.3077	0.8600	0.3779	1.1818
18	0.3097	0.8672	0.3870	1.1397
18	0.4033	0.8511	0.4128	1.1766
18	0.4036	0.8462	0.4325	1.1446
18	0.3149	0.8750	0.3505	1.1290
18	0.4012	0.8595	0.4180	1.1318
18	0.2681	0.8516	0.4539	1.1809
18	0.2698	0.8759	0.3877	1.2466
19	0.2889	0.8708	0.3637	1.2218
19	0.3049	0.8647	0.3847	1.2213
19	0.3050	0.8719	0.3569	1.3268
19	0.3004	0.8762	0.3502	1.3723
19	0.4026	0.8497	0.4319	1.1819
19	0.4032	0.8617	0.4056	1.2755
19	0.2675	0.8558	0.4865	1.3232
19	0.3989	0.8600	0.4095	1.3723
19	0.3097	0.8698	0.3613	1.3761
19	0.2659	0.8666	0.3952	1.2748
20	0.2841	0.8686	0.3733	1.3334
20	0.3014	0.8612	0.3868	1.1594
20	0.2999	0.8697	0.3607	1.2086
20	0.4014	0.8462	0.4349	1.2056
20	0.4004	0.8536	0.4116	1.1831

20	0.2979	0.8706	0.3577	1.2872
20	0.2612	0.8698	0.4213	1.2087
20	0.3063	0.8730	0.3638	1.2177
20	0.3995	0.8500	0.4231	1.2413
20	0.2628	0.8655	0.4226	1.1154
21	0.2798	0.8816	0.3446	1.1393
21	0.2971	0.8653	0.3807	1.1101
21	0.4006	0.8413	0.4432	1.1003
21	0.2974	0.8739	0.3576	1.1283
21	0.3997	0.8530	0.4198	1.1233
21	0.2926	0.8791	0.3477	1.1318
21	0.2608	0.8633	0.4198	1.1154
21	0.3015	0.8728	0.3576	1.0824
21	0.3978	0.8514	0.4229	1.1206
21	0.2597	0.8692	0.3907	1.1574
22	0.2750	0.8773	0.3495	1.1498
22	0.2938	0.8644	0.3914	1.1504
22	0.2928	0.8638	0.3668	1.0701
22	0.3998	0.8480	0.4326	1.1153
22	0.4006	0.8603	0.4057	1.0881
22	0.2859	0.8716	0.3564	1.1551
22	0.2554	0.8616	0.4273	1.1379
22	0.2994	0.8708	0.3550	1.1237
22	0.2541	0.8697	0.4052	1.0765
22	0.3960	0.8520	0.4231	1.1520
23	0.2714	0.8780	0.3447	1.1099
23	0.2885	0.8705	0.3680	1.1095
23	0.2901	0.8733	0.3563	1.1073
23	0.3981	0.8413	0.4530	1.0946
23	0.3992	0.8602	0.4108	1.1132
23	0.2836	0.8736	0.3617	1.1012
23	0.2957	0.8731	0.3486	1.1096
23	0.2499	0.8598	0.4597	1.1116
23	0.2473	0.8672	0.4153	1.1337
23	0.3979	0.8602	0.4098	1.1288
24	0.2673	0.8784	0.3500	1.1111
24	0.2843	0.8697	0.3762	1.1159
24	0.3982	0.8500	0.4273	1.0744
24	0.2866	0.8661	0.3693	1.1651
24	0.2790	0.8802	0.3475	1.0591
24	0.3986	0.8605	0.4096	1.1733
24	0.2460	0.8548	0.4698	1.1040
24	0.2927	0.8777	0.3444	1.1080

24	0.2457	0.8750	0.3894	1.1450
24	0.3961	0.8550	0.4192	1.1412
25	0.2628	0.8748	0.3632	1.0777
25	0.2817	0.8719	0.3679	1.1040
25	0.3961	0.8489	0.4326	1.1082
25	0.2832	0.8730	0.3474	1.0809
25	0.2744	0.8733	0.3575	1.1273
25	0.3976	0.8544	0.4214	1.1166
25	0.2427	0.8709	0.4128	1.1173
25	0.2883	0.8669	0.3594	1.1473
25	0.3945	0.8472	0.4292	1.1109
25	0.2453	0.8617	0.4356	1.1395
26	0.2590	0.8744	0.3599	1.1534
26	0.3960	0.8545	0.4300	1.0952
26	0.2809	0.8684	0.3690	1.1474
26	0.2805	0.8752	0.3454	1.1508
26	0.3982	0.8600	0.4087	1.0951
26	0.2706	0.8812	0.3436	1.1087
26	0.2422	0.8648	0.4480	1.0858
26	0.2850	0.8745	0.3560	1.1143
26	0.3943	0.8608	0.4098	1.1151
26	0.2429	0.8744	0.3900	1.1405
27	0.2544	0.8806	0.3485	1.1093
27	0.3960	0.8484	0.4278	1.1483
27	0.2768	0.8627	0.3922	1.1705
27	0.3975	0.8608	0.4057	1.1439
27	0.2771	0.8677	0.3559	1.2123
27	0.2671	0.8747	0.3619	1.1689
27	0.2355	0.8614	0.4353	1.1877
27	0.2818	0.8742	0.3559	1.3636
27	0.3954	0.8577	0.4113	1.3350
27	0.2424	0.8655	0.4356	1.3761
28	0.2504	0.8720	0.3674	1.3686
28	0.3939	0.8534	0.4271	1.3874
28	0.2738	0.8681	0.3743	1.4723
28	0.2734	0.8750	0.3475	1.4147
28	0.3956	0.8577	0.4151	1.4136
28	0.2625	0.8789	0.3571	1.4076
28	0.2372	0.8620	0.4394	1.3009
28	0.2785	0.8755	0.3540	1.1797
28	0.3936	0.8570	0.4078	1.2716
28	0.2327	0.8578	0.4200	1.3005
29	0.2487	0.8850	0.3428	1.2256
29	0.3936	0.8444	0.4379	1.1529

```
     29       0.2701       0.8630        0.3957  1.1785
     29       0.3957       0.8531        0.4226  1.1517
     29       0.2707       0.8686        0.3575  1.1649
     29       0.2579       0.8809        0.3551  1.2062
     29       0.2325       0.8589        0.4468  1.1784
     29       0.2773       0.8658        0.3651  1.1392
     29       0.3928       0.8527        0.4163  1.1923
     29       0.2325       0.8717        0.4119  1.1092
     30       0.2450       0.8834        0.3496  1.1694
     30       0.3937       0.8548        0.4160  1.1401
     30       0.2669       0.8648        0.3765  1.1781
     30       0.3964       0.8556        0.4122  1.1481
     30       0.2680       0.8670        0.3719  1.1536
     30       0.2558       0.8784        0.3515  1.1243
     30       0.2311       0.8495        0.5186  1.1552
     30       0.2724       0.8789        0.3401
1.1115
     30       0.3939       0.8545        0.4173  1.1434
     30       0.2259       0.8698        0.4237  1.1795
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6272       0.8308        0.4754
1.1333
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6213       0.8116        0.5129
1.1747
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6237       0.8314        0.4675
1.2071
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6160       0.8394        0.4436
1.1721
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6606       0.7297        0.7176
1.1453
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6625       0.8167        0.5261
1.1864
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6682       0.8245        0.4945
1.2418
  epoch    train_loss    valid_acc    valid_loss     dur
```

```
-------  ------------  -----------  ------------  ------
      1        1.4754       0.7137        0.9868
1.3941
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4202       0.7253        0.9487
1.3232
      2        0.4743       0.8455        0.4309
1.1800
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4209       0.7383        0.9411
1.3885
      2        0.4796       0.8183        0.5109
1.1851
      2        0.4799       0.8422        0.4486
1.2436
      2        0.4819       0.8094        0.5024  1.2727
      2        0.5634       0.7889        0.5634
1.2767
      2        0.5689       0.8163        0.4990  1.2785
      2        0.5683       0.8028        0.5653  1.2187
      2        0.8375       0.7450        0.7574
1.3802
      3        0.4300       0.8602        0.3769
1.4233
      2        0.8190       0.7577        0.7355
1.5464
      2        0.8048       0.7672        0.7253
1.6231
      3        0.4410       0.8211        0.5019
1.3062
      3        0.4400       0.8270        0.4840  1.3633
      3        0.5603       0.7933        0.5967  1.2981
      3        0.4500       0.8423        0.4318
1.3693
      3        0.5541       0.7809        0.6035  1.3521
      3        0.5583       0.7858        0.5979  1.3614
      3        0.6992       0.7631        0.6768
1.4357
      4        0.4086       0.8123        0.5190  1.1493
      3        0.6899       0.7756        0.6558
1.3773
      4        0.4185       0.8370        0.4520
1.1407
      3        0.6784       0.7811        0.6512
1.4331
      4        0.4225       0.8464        0.4132
```

1.1653					
	4	0.5581	0.7648	0.6776	1.1473
	4	0.5576	0.7795	0.5858	1.2506
	4	0.4236	0.8522	0.3923	
1.2241					
	4	0.5414	0.8044	0.5507	
1.2362					
	5	0.3905	0.8603	0.3986	1.1526
	4	0.6379	0.7767	0.6328	
1.3787					
	5	0.4095	0.8441	0.4288	
1.1914					
	4	0.6298	0.7908	0.6107	
1.3266					
	5	0.4054	0.8542	0.4059	
1.1436					
	5	0.5596	0.7914	0.5627	1.1404
	5	0.5573	0.7869	0.5662	1.1274
	5	0.5479	0.8192	0.5210	1.1437
	4	0.6224	0.7937	0.6064	
1.3254					
	5	0.4071	0.8344	0.4439	1.2103
	6	0.3696	0.8497	0.4043	1.1726
	5	0.5998	0.7858	0.6034	
1.2878					
	6	0.3960	0.8377	0.4559	1.1239
	6	0.3925	0.8509	0.4175	1.1088
	6	0.5523	0.7886	0.5402	1.1524
	5	0.5909	0.8027	0.5753	
1.3929					
	6	0.5522	0.7681	0.5943	1.2123
	6	0.5448	0.8098	0.5174	1.2261
	6	0.4016	0.8472	0.4226	1.2533
	5	0.5873	0.8039	0.5763	
1.3714					
	7	0.3597	0.8527	0.4086	1.2051
	7	0.3863	0.8384	0.4661	1.1430
	7	0.3902	0.8575	0.3952	
1.1529					
	6	0.5727	0.7905	0.5815	
1.4185					
	7	0.5523	0.7597	0.6327	1.1540
	7	0.5497	0.8106	0.5174	1.2105
	7	0.5423	0.8067	0.5462	1.1942
	6	0.5632	0.8128	0.5516	
1.3445					
	7	0.3878	0.8491	0.4247	1.1878
	6	0.5617	0.8117	0.5521	

1.3904				
8	0.3434	0.8620	0.3771	1.1491
8	0.3804	0.8503	0.4326	1.1508
8	0.3821	0.8467	0.4284	1.0827
8	0.5489	0.8152	0.5145	1.1628
7	0.5516	0.7986	0.5647	
1.3615				
8	0.5494	0.8192	0.5080	1.1376
8	0.5491	0.8155	0.5134	1.1985
8	0.3840	0.8341	0.4481	1.1183
7	0.5419	0.8186	0.5327	
1.3039				
9	0.3413	0.8719	0.3640	
1.1200				
7	0.5420	0.8164	0.5339	
1.3840				
9	0.3735	0.8483	0.4279	1.2436
9	0.3794	0.8658	0.3890	
1.1336				
9	0.5528	0.8153	0.4958	1.2173
9	0.5423	0.7955	0.5576	1.1323
9	0.5462	0.8187	0.4972	
1.1485				
9	0.3787	0.8134	0.5348	1.1725
8	0.5343	0.8047	0.5499	
1.3466				
10	0.3280	0.8561	0.4116	1.1512
8	0.5247	0.8209	0.5187	
1.3092				
10	0.3753	0.8402	0.4343	1.0772
10	0.3687	0.8272	0.4840	1.1399
8	0.5263	0.8219	0.5183	
1.3720				
10	0.5526	0.8219	0.4845	1.1845
10	0.5466	0.7983	0.5579	1.1197
10	0.5431	0.8047	0.5629	1.1509
10	0.3792	0.8430	0.4369	1.1406
9	0.5209	0.8105	0.5375	
1.4130				
11	0.3213	0.8738	0.3523	
1.3264				
9	0.5110	0.8258	0.5046	
1.5113				
11	0.3681	0.8231	0.5027	1.2871
11	0.3691	0.8542	0.4092	1.3516
9	0.5133	0.8220	0.5080	
1.6428				
11	0.5400	0.7986	0.5201	1.5256

11	0.5493	0.7759	0.5819	1.5417
11	0.5429	0.7780	0.6026	1.5145
11	0.3716	0.8480	0.4024	1.5847
12	0.3111	0.8698	0.3718	1.5482
10	0.5088	0.8137	0.5264	
1.7175				
12	0.3626	0.8483	0.4213	1.4503
10	0.4993	0.8272	0.4944	
1.6823				
12	0.3639	0.8470	0.4286	1.4912
12	0.5424	0.8091	0.5217	1.2476
12	0.5501	0.7978	0.5317	1.2301
12	0.5392	0.7736	0.6019	1.2953
10	0.5027	0.8270	0.4970	
1.5554				
12	0.3715	0.8606	0.3855	
1.2213				
13	0.3069	0.8502	0.4170	1.1084
13	0.3631	0.8356	0.4600	1.1073
11	0.4986	0.8169	0.5187	
1.3590				
13	0.3637	0.8600	0.4003	1.1449
11	0.4893	0.8305	0.4861	
1.3189				
13	0.5386	0.7955	0.5423	1.0842
13	0.5487	0.8155	0.5093	1.1392
13	0.3689	0.8575	0.3831	1.1404
13	0.5478	0.8109	0.5261	1.2119
11	0.4931	0.8272	0.4906	
1.3100				
14	0.3023	0.8722	0.3726	1.1485
14	0.3616	0.8462	0.4270	1.1510
14	0.3637	0.8455	0.4310	1.1816
12	0.4896	0.8213	0.5102	
1.4387				
12	0.4803	0.8334	0.4777	
1.4773				
14	0.5433	0.7945	0.5245	1.3573
14	0.5496	0.8025	0.5378	1.3101
14	0.5477	0.8053	0.5506	1.2553
14	0.3671	0.8564	0.4014	1.3120
12	0.4851	0.8337	0.4813	
1.6345				
15	0.2934	0.8572	0.4091	1.5502
15	0.3564	0.8347	0.4396	1.5804
15	0.3555	0.8570	0.3958	1.7354
13	0.4817	0.8214	0.5039	
2.0508				

15	0.3663	0.8586	0.3910	1.6700
13	0.4730	0.8345	0.4727	
2.0302				
15	0.5514	0.7972	0.5296	1.9306
15	0.5463	0.8192	0.4927	1.9785
15	0.5412	0.7816	0.5953	1.9211
13	0.4781	0.8358	0.4744	
1.8797				
16	0.2892	0.8658	0.3819	1.8457
16	0.3588	0.8592	0.3977	1.6323
16	0.3565	0.8691	0.3724	1.5134
16	0.5500	0.8039	0.5310	1.2021
16	0.3590	0.8680	0.3727	
1.3013				
16	0.5429	0.8072	0.5367	1.2157
14	0.4746	0.8236	0.4975	
1.5222				
16	0.5475	0.7941	0.5523	1.2419
14	0.4662	0.8378	0.4645	
1.4819				
17	0.2880	0.8738	0.3650	1.2140
14	0.4714	0.8395	0.4684	
1.4138				
17	0.3501	0.8566	0.4169	1.2213
17	0.3499	0.8619	0.3906	1.1667
17	0.5486	0.8091	0.5403	1.2168
17	0.3605	0.8250	0.4668	1.2493
17	0.5416	0.7756	0.6026	1.2653
17	0.5430	0.8005	0.5390	1.2304
15	0.4676	0.8247	0.4926	
1.3636				
15	0.4597	0.8384	0.4602	
1.3379				
18	0.2806	0.8711	0.3837	1.0827
18	0.3527	0.8517	0.4137	1.1787
18	0.3540	0.8653	0.3790	1.1198
15	0.4658	0.8406	0.4631	
1.3290				
18	0.3582	0.8600	0.3854	1.1867
18	0.5471	0.7681	0.5888	1.2228
18	0.5429	0.8087	0.5324	1.2290
18	0.5413	0.7827	0.6047	1.1705
16	0.4616	0.8280	0.4900	
1.3783				
19	0.2751	0.8584	0.4204	1.2055
19	0.3532	0.8527	0.4083	1.1280
16	0.4542	0.8391	0.4584	
1.3430				

19	0.3574	0.8630	0.3850	1.1535
16	0.4601	0.8434	0.4583	
1.3695				
19	0.3581	0.8616	0.3902	1.1736
19	0.5520	0.7822	0.5975	1.1809
19	0.5388	0.8153	0.5030	1.1237
19	0.5466	0.8195	0.4965	1.1543
20	0.2703	0.8664	0.4036	1.1560
20	0.3486	0.8636	0.3987	1.1089
17	0.4566	0.8303	0.4822	
1.3645				
20	0.3515	0.8530	0.4042	1.1636
17	0.4493	0.8419	0.4534	
1.3203				
17	0.4552	0.8441	0.4547	
1.3078				
20	0.5454	0.8037	0.5375	1.1378
20	0.3583	0.8556	0.3892	1.1791
20	0.5428	0.7275	0.7727	1.1567
20	0.5413	0.8220	0.5099	1.1458
21	0.2695	0.8569	0.4045	1.1353
21	0.3462	0.8558	0.3998	1.1651
21	0.3494	0.8622	0.3992	1.1214
18	0.4508	0.8275	0.4808	1.2981
18	0.4443	0.8434	0.4482	
1.3405				
21	0.5465	0.7678	0.6492	1.0978
21	0.5417	0.8100	0.5026	1.1046
21	0.3541	0.8600	0.3999	1.1159
21	0.5371	0.7922	0.5730	1.1302
18	0.4501	0.8448	0.4505	
1.3401				
22	0.2615	0.8598	0.4099	1.0489
22	0.3471	0.8602	0.4038	1.1521
22	0.3468	0.8336	0.4916	1.1543
19	0.4465	0.8303	0.4752	1.3174
19	0.4399	0.8464	0.4452	
1.2928				
22	0.5479	0.7983	0.5558	1.1091
22	0.5449	0.8078	0.5236	1.1489
22	0.3580	0.8323	0.4543	1.1378
22	0.5450	0.8081	0.5057	1.1884
19	0.4464	0.8484	0.4468	
1.2954				
23	0.2574	0.8717	0.3790	1.1084
23	0.3439	0.8614	0.4056	1.0534
23	0.3446	0.8694	0.3692	
1.1299				

23	0.5500	0.7811	0.5744	1.1854
20	0.4419	0.8320	0.4713	
1.3595				
23	0.3519	0.8505	0.4216	1.1496
23	0.5388	0.8245	0.4893	1.1716
20	0.4356	0.8472	0.4407	
1.3039				
23	0.5454	0.8202	0.4770	1.1687
24	0.2576	0.8728	0.3730	1.1076
24	0.3444	0.8531	0.4446	1.1711
20	0.4420	0.8452	0.4453	1.3522
24	0.3457	0.8642	0.3845	1.1554
24	0.3540	0.8517	0.4100	1.0835
24	0.5481	0.7978	0.6000	1.1703
24	0.5436	0.7937	0.5702	1.1714
24	0.5387	0.8028	0.5615	1.1145
21	0.4377	0.8345	0.4675	
1.2870				
25	0.2510	0.8600	0.4326	1.0968
21	0.4321	0.8481	0.4374	
1.3077				
25	0.3432	0.8419	0.4422	1.1723
25	0.3427	0.8581	0.3836	1.2035
21	0.4382	0.8475	0.4421	1.4134
25	0.3553	0.8456	0.4175	1.1366
25	0.5442	0.7706	0.6133	1.2165
25	0.5424	0.7823	0.5934	1.1924
25	0.5399	0.7986	0.6010	1.2683
26	0.2518	0.8741	0.3687	1.2343
22	0.4336	0.8373	0.4626	
1.4111				
22	0.4286	0.8447	0.4378	1.4186
26	0.3429	0.8323	0.5080	1.2188
26	0.3401	0.8628	0.3766	1.2124
22	0.4344	0.8483	0.4374	1.4062
26	0.3542	0.8280	0.4923	1.1949
26	0.5413	0.8028	0.5276	1.1765
26	0.5496	0.8283	0.4882	1.2534
26	0.5433	0.8150	0.5259	1.2080
27	0.2459	0.8636	0.4229	1.1296
27	0.3439	0.8364	0.4770	1.1467
23	0.4301	0.8337	0.4640	1.4302
27	0.3432	0.8511	0.4365	1.1194
23	0.4247	0.8484	0.4352	
1.3914				
27	0.3540	0.8525	0.3986	1.1430
23	0.4308	0.8502	0.4342	
1.2825				

```
     27      0.5489       0.7806       0.6052  1.1125
     27      0.5481       0.8216       0.4841  1.1233
     28      0.2432       0.8723       0.4047  1.1172
     27      0.5424       0.7992       0.5462  1.1375
     28      0.3444       0.8302       0.4792  1.1327
     28      0.3419       0.8723       0.3652  1.0875
     24      0.4264       0.8394       0.4575
1.3127
     24      0.4216       0.8495       0.4299
1.3141
     28      0.3518       0.8661       0.3696  1.1687
     28      0.5409       0.8152       0.5172  1.1237
     28      0.5449       0.7878       0.5861  1.2086
     24      0.4277       0.8498       0.4321  1.3604
     29      0.2400       0.8784       0.4048  1.2150
     28      0.5419       0.7712       0.5988  1.2508
     29      0.3444       0.8509       0.4307  1.2578
     29      0.3379       0.8552       0.4160  1.1640
     25      0.4230       0.8377       0.4557  1.3344
     25      0.4185       0.8516       0.4282
1.6039
     29      0.3536       0.8605       0.3844  1.3767
     29      0.5483       0.7609       0.6992  1.3437
     29      0.5417       0.8214       0.4925  1.5014
     29      0.5442       0.7600       0.7225  1.2312
     30      0.2383       0.8716       0.3862  1.2985
     25      0.4243       0.8538       0.4322  1.6326
     30      0.3383       0.8620       0.4045  1.1788
     30      0.3407       0.8573       0.4051  1.3728
     26      0.4198       0.8394       0.4531  1.4832
     30      0.3494       0.8520       0.4057  1.2436
     30      0.5448       0.8037       0.5420  1.1540
     30      0.5412       0.7698       0.6243  1.1489
     26      0.4154       0.8517       0.4256
1.4191
     30      0.5409       0.8108       0.4944  1.1933
     26      0.4214       0.8512       0.4261  1.4109
  epoch  train_loss  valid_acc  valid_loss   dur
------- ------------ ----------- ------------ ------
      1      1.4739       0.7102       0.9749
1.3708
  epoch  train_loss  valid_acc  valid_loss   dur
------- ------------ ----------- ------------ ------
      1      1.4252       0.7228       0.9629
1.3836
     27      0.4167       0.8420       0.4499
1.3522
  epoch  train_loss  valid_acc  valid_loss   dur
```

```
-------  ------------  -----------  ------------  ------
      1        1.4555       0.7142        0.9755
1.3131
     27        0.4125       0.8519        0.4236
1.4082
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4595       0.7192        0.9689
1.3783
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4445       0.7222        0.9758
1.3838
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4530       0.7122        0.9957
1.3894
     27        0.4180       0.8512        0.4261  1.4327
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7152       0.8089        0.5354
1.3532
      2        0.8355       0.7486        0.7536
1.3837
     28        0.4130       0.8400        0.4505  1.3586
      2        0.8350       0.7550        0.7487
1.3523
      2        0.8303       0.7559        0.7460
1.4097
     28        0.4098       0.8533        0.4223
1.3874
      2        0.8322       0.7494        0.7528
1.4000
      2        0.8408       0.7572        0.7529
1.3574
      2        0.8528       0.7455        0.7678
1.3486
     28        0.4158       0.8525        0.4215  1.3558
      2        0.4936       0.8191        0.5026
1.4308
      3        0.7009       0.7638        0.6753
1.5019
     29        0.4108       0.8408        0.4465  1.4155
      3        0.6990       0.7725        0.6666
1.4324
      3        0.6973       0.7667        0.6670
1.4963
     29        0.4069       0.8508        0.4209  1.4091
```

```
      3        0.7072       0.7706        0.6712
1.4131
     29        0.4131       0.8541        0.4194
1.3798
      3        0.7176       0.7650        0.6869
1.4448
      3        0.7014       0.7672        0.6777
1.4752
      3        0.4501       0.8286        0.4736
1.3106
      4        0.6392       0.7753        0.6318
1.4370
      4        0.6376       0.7883        0.6203
1.3370
     30        0.4077       0.8397        0.4498  1.4379
      4        0.6354       0.7850        0.6186
1.3541
     30        0.4048       0.8492        0.4220  1.3229
     30        0.4100       0.8548        0.4180
1.3524
      4        0.6476       0.7848        0.6254
1.4041
      4        0.6424       0.7803        0.6363
1.3412
      4        0.6565       0.7837        0.6375
1.4246
      4        0.4226       0.8387        0.4518
1.4132
      5        0.5991       0.7933        0.5895
1.3041
      5        0.6003       0.7880        0.6003
1.4001
      5        0.5964       0.7981        0.5841
1.4085
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7147       0.8098        0.5288
1.4109
      5        0.6101       0.7964        0.5950
1.3674
      5        0.6058       0.7886        0.6061
1.3993
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7170       0.8189        0.5247
1.3578
      5        0.6177       0.7931        0.6061
1.3829
```

```
      5       0.4026       0.8406       0.4457
1.3648
      6       0.5718       0.8067       0.5618
1.3052
      6       0.5716       0.7989       0.5760
1.4394
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.7291       0.8087       0.5350
1.4253
      6       0.5680       0.8069       0.5598
1.4551
      2       0.4880       0.8367       0.4601
1.3653
      6       0.5831       0.8025       0.5702
1.2975
      6       0.5799       0.7955       0.5841
1.3865
      2       0.4933       0.8320       0.4713
1.3863
      6       0.3857       0.8545       0.4090
1.3600
      6       0.5895       0.7992       0.5800
1.4112
      7       0.5509       0.8113       0.5436
1.3796
      7       0.5498       0.8031       0.5577
1.3355
      2       0.4912       0.8269       0.4767
1.4347
      7       0.5465       0.8094       0.5382
1.4135
      3       0.4455       0.8452       0.4329
1.4241
      7       0.5625       0.8089       0.5529
1.4356
      7       0.5602       0.8022       0.5677
1.4349
      7       0.3726       0.8422       0.4503  1.3992
      3       0.4478       0.8417       0.4496
1.4368
      7       0.5684       0.8086       0.5606
1.4317
      8       0.5343       0.8170       0.5275
1.4602
      8       0.5319       0.8097       0.5421
1.5224
      3       0.4476       0.8397       0.4501
```

1.5490				
8	0.5292	0.8186	0.5226	
1.4754				
8	0.5459	0.8150	0.5365	
1.4058				
4	0.4179	0.8555	0.4114	
1.5504				
8	0.5443	0.8075	0.5539	
1.4628				
8	0.3589	0.8577	0.4058	
1.4607				
4	0.4228	0.8509	0.4184	
1.4691				
8	0.5515	0.8116	0.5450	
1.4479				
9	0.5204	0.8216	0.5150	
1.4947				
9	0.5173	0.8167	0.5280	
1.4224				
9	0.5147	0.8219	0.5112	
1.2923				
4	0.4205	0.8452	0.4323	
1.4253				
9	0.5324	0.8181	0.5245	
1.5209				
9	0.5313	0.8097	0.5441	
1.5073				
9	0.3503	0.8522	0.4078	1.5331
5	0.3979	0.8570	0.3993	
1.6566				
9	0.5378	0.8217	0.5310	
1.5592				
5	0.3996	0.8591	0.4042	
1.5954				
10	0.5095	0.8256	0.5028	
1.5139				
10	0.5051	0.8209	0.5173	
1.4866				
10	0.5026	0.8255	0.4992	
1.5276				
5	0.4019	0.8442	0.4292	1.5441
10	0.5208	0.8205	0.5169	
1.4053				
6	0.3838	0.8566	0.4024	1.3731
10	0.5210	0.8109	0.5352	
1.4632				
10	0.5267	0.8208	0.5213	1.3670
6	0.3842	0.8645	0.3833	

1.3756				
10	0.3396	0.8638	0.3886	
1.4627				
11	0.4995	0.8269	0.4973	
1.3738				
11	0.4945	0.8236	0.5092	
1.3489				
11	0.4925	0.8267	0.4897	
1.3604				
6	0.3888	0.8505	0.4171	
1.3322				
11	0.5115	0.8280	0.5051	
1.3576				
11	0.5119	0.8137	0.5258	
1.3600				
7	0.3703	0.8616	0.3828	1.3380
11	0.5171	0.8220	0.5134	
1.3872				
7	0.3709	0.8616	0.3858	
1.4056				
11	0.3309	0.8620	0.3944	1.4242
12	0.4914	0.8308	0.4875	
1.3591				
12	0.4855	0.8259	0.5010	
1.3381				
12	0.4834	0.8281	0.4850	
1.3434				
7	0.3742	0.8569	0.4015	
1.3523				
12	0.5029	0.8284	0.4993	
1.3086				
12	0.5040	0.8167	0.5217	
1.3550				
8	0.3593	0.8745	0.3618	
1.3665				
12	0.5089	0.8241	0.5070	
1.3912				
8	0.3594	0.8616	0.3784	1.4124
12	0.3234	0.8631	0.3866	1.3622
13	0.4836	0.8333	0.4816	
1.3530				
13	0.4776	0.8267	0.4938	
1.3579				
13	0.4760	0.8330	0.4753	
1.3673				
8	0.3648	0.8534	0.4032	1.3444
13	0.4955	0.8328	0.4938	
1.3686				

9	0.3508	0.8662	0.3744	1.3524
13	0.4970	0.8222	0.5139	1.3875
13	0.3156	0.8644	0.3829	1.3463
9	0.3490	0.8672	0.3761	1.3932
13	0.5017	0.8295	0.4976	1.4221
14	0.4770	0.8334	0.4766	1.3287
14	0.4705	0.8291	0.4885	1.3367
14	0.4684	0.8367	0.4693	1.3626
9	0.3561	0.8570	0.3962	1.3664
14	0.4888	0.8347	0.4860	1.2954
14	0.4912	0.8222	0.5083	1.3684
10	0.3398	0.8688	0.3706	1.3951
14	0.3077	0.8642	0.3860	1.3884
14	0.4954	0.8319	0.4922	1.3876
10	0.3382	0.8625	0.3814	1.3940
15	0.4709	0.8339	0.4709	1.4049
15	0.4640	0.8287	0.4842	1.5330
15	0.4623	0.8347	0.4661	1.4165
10	0.3474	0.8570	0.3945	1.4348
15	0.4831	0.8363	0.4810	1.4519
15	0.3011	0.8697	0.3781	1.3760
11	0.3319	0.8694	0.3683	1.4416
15	0.4859	0.8256	0.5042	1.5535
15	0.4895	0.8358	0.4901	1.4803
11	0.3324	0.8722	0.3506	1.6056
16	0.4654	0.8359	0.4679	1.3985
16	0.4576	0.8317	0.4789	1.4038
16	0.4567	0.8386	0.4585	1.4561
11	0.3383	0.8481	0.4102	1.4651

16	0.4776	0.8386	0.4770	1.4195
16	0.2936	0.8387	0.4539	1.3221
16	0.4807	0.8245	0.5000	1.3800
12	0.3218	0.8683	0.3687	1.4113
16	0.4847	0.8331	0.4831	1.3750
12	0.3224	0.8689	0.3606	1.3918
17	0.4604	0.8395	0.4611	1.3933
17	0.4528	0.8342	0.4767	1.3768
17	0.4515	0.8438	0.4534	1.4344
12	0.3317	0.8639	0.3883	1.3417
17	0.4731	0.8380	0.4728	1.4673
17	0.2883	0.8583	0.3995	1.4536
13	0.3155	0.8698	0.3592	1.4003
17	0.4800	0.8367	0.4810	1.3861
17	0.4765	0.8255	0.4979	1.4781
13	0.3157	0.8758	0.3443	1.4614
18	0.4559	0.8428	0.4558	1.3883
18	0.4479	0.8319	0.4725	1.3293
18	0.4462	0.8452	0.4505	1.4227
13	0.3245	0.8611	0.3905	1.4226
18	0.4687	0.8413	0.4691	1.4408
18	0.2823	0.8722	0.3670	1.5291
18	0.4760	0.8372	0.4753	1.5735
14	0.3088	0.8730	0.3554	1.5684
18	0.4721	0.8275	0.4920	1.5553
19	0.4517	0.8434	0.4530	1.5464
14	0.3076	0.8706	0.3552	1.5894
19	0.4434	0.8353	0.4660	1.5700
14	0.3177	0.8645	0.3836	1.4744
19	0.4419	0.8425	0.4471	1.5657

19	0.4650	0.8414	0.4665	1.3282
19	0.2763	0.8672	0.3750	1.4033
19	0.4687	0.8291	0.4902	1.4027
19	0.4721	0.8375	0.4715	1.4507
15	0.3008	0.8773	0.3469	1.5072
20	0.4475	0.8419	0.4488	1.4435
15	0.3021	0.8728	0.3505	1.5153
20	0.4389	0.8358	0.4645	1.4624
15	0.3122	0.8714	0.3687	1.3924
20	0.4373	0.8447	0.4437	1.4358
20	0.4611	0.8423	0.4639	1.4148
20	0.2696	0.8630	0.3859	1.4083
20	0.4651	0.8300	0.4863	1.5600
20	0.4684	0.8395	0.4690	1.5569
16	0.2934	0.8734	0.3672	1.4988
21	0.4435	0.8447	0.4456	1.5096
16	0.2964	0.8769	0.3436	1.5206
21	0.4351	0.8378	0.4601	1.5909
16	0.3070	0.8692	0.3673	1.5135
21	0.4332	0.8455	0.4428	1.4287
21	0.4576	0.8441	0.4609	1.4032
21	0.2652	0.8738	0.3644	1.3668
21	0.4622	0.8295	0.4847	1.3642
17	0.2891	0.8780	0.3464	1.3653
21	0.4653	0.8395	0.4660	1.3823
22	0.4403	0.8434	0.4453	1.3644
17	0.2900	0.8833	0.3320	1.3631
22	0.4313	0.8386	0.4572	1.3985
22	0.4297	0.8470	0.4385	1.3614

17	0.2998	0.8620	0.3848	1.4828
22	0.4544	0.8453	0.4583	
1.4079				
22	0.2608	0.8708	0.3674	1.3903
22	0.4591	0.8311	0.4815	
1.4020				
22	0.4620	0.8422	0.4632	
1.4051				
23	0.4369	0.8477	0.4404	
1.4712				
18	0.2822	0.8748	0.3470	1.5701
18	0.2838	0.8755	0.3528	1.4068
23	0.4276	0.8402	0.4538	
1.4682				
23	0.4260	0.8489	0.4351	
1.4244				
18	0.2951	0.8680	0.3706	1.4132
23	0.4515	0.8441	0.4562	1.4319
23	0.2547	0.8748	0.3653	1.5453
23	0.4563	0.8323	0.4793	
1.4535				
23	0.4594	0.8444	0.4613	
1.4467				
24	0.4333	0.8462	0.4379	1.4628
19	0.2785	0.8795	0.3364	1.5173
19	0.2764	0.8767	0.3409	1.5772
24	0.4242	0.8397	0.4533	1.4035
24	0.4229	0.8481	0.4352	1.5110
19	0.2901	0.8639	0.3727	1.5406
24	0.4486	0.8459	0.4550	
1.4711				
24	0.2491	0.8783	0.3575	
1.3911				
24	0.4564	0.8442	0.4585	1.4010
24	0.4535	0.8325	0.4790	
1.5012				
20	0.2741	0.8798	0.3396	1.4612
25	0.4302	0.8477	0.4386	1.6105
25	0.4206	0.8395	0.4487	1.4935
20	0.2708	0.8778	0.3440	1.5116
25	0.4198	0.8495	0.4289	
1.4932				
20	0.2860	0.8548	0.3926	1.4914
25	0.4463	0.8472	0.4509	
1.4278				
25	0.2462	0.8623	0.3995	1.3288
25	0.4542	0.8431	0.4569	1.3941
25	0.4512	0.8339	0.4748	

1.3402					
	21	0.2679	0.8806	0.3322	1.3873
	26	0.4270	0.8469	0.4339	1.3909
	21	0.2639	0.8806	0.3362	
1.4233					
	26	0.4179	0.8419	0.4490	1.4463
	26	0.4167	0.8514	0.4281	
1.3867					
	21	0.2815	0.8672	0.3716	1.3597
	26	0.4440	0.8452	0.4503	1.3952
	26	0.2405	0.8761	0.3653	1.4464
	26	0.4519	0.8441	0.4549	1.4737
	26	0.4491	0.8300	0.4749	1.4988
	22	0.2625	0.8831	0.3266	1.4368
	27	0.4243	0.8498	0.4314	
1.4112					
	22	0.2599	0.8708	0.3603	1.3939
	27	0.4149	0.8427	0.4446	
1.4350					
	27	0.4135	0.8527	0.4254	
1.3857					
	22	0.2757	0.8711	0.3641	1.3677
	27	0.4417	0.8459	0.4494	1.4246
	27	0.2372	0.8764	0.3635	1.4531
	27	0.4466	0.8344	0.4724	
1.3948					
	27	0.4493	0.8489	0.4532	
1.4605					
	23	0.2578	0.8856	0.3242	
1.4278					
	28	0.4217	0.8495	0.4300	1.4431
	23	0.2545	0.8764	0.3411	1.4484
	28	0.4118	0.8433	0.4429	
1.4375					
	28	0.4110	0.8530	0.4226	
1.4995					
	23	0.2732	0.8661	0.3697	1.4784
	28	0.4396	0.8459	0.4461	1.3682
	28	0.2313	0.8683	0.3916	1.4545
	28	0.4447	0.8345	0.4708	
1.3413					
	28	0.4472	0.8466	0.4531	1.4077
	24	0.2516	0.8806	0.3284	1.3159
	29	0.4189	0.8514	0.4278	
1.4254					
	29	0.4092	0.8400	0.4432	1.3868
	24	0.2506	0.8772	0.3430	1.4510
	24	0.2688	0.8717	0.3634	

```
1.3584
     29        0.4082       0.8522        0.4226  1.4488
     29        0.4374       0.8486        0.4444
1.3833
     29        0.2287       0.8711        0.3785  1.4099
     29        0.4430       0.8333        0.4710  1.3639
     29        0.4454       0.8477        0.4493  1.4053
     25        0.2474       0.8781        0.3325  1.3617
     30        0.4165       0.8523        0.4277
1.3715
     30        0.4068       0.8464        0.4374
1.3609
     25        0.2449       0.8802        0.3303  1.3993
     25        0.2644       0.8758        0.3569
1.3902
     30        0.4058       0.8523        0.4198  1.3833
     30        0.4353       0.8481        0.4423  1.3790
     30        0.2246       0.8708        0.3770  1.4610
     30        0.4435       0.8486        0.4474  1.3353
     30        0.4409       0.8364        0.4675
1.4649
     26        0.2442       0.8719        0.3458  1.4117
     26        0.2407       0.8833        0.3350  1.4539
     26        0.2622       0.8753        0.3555  1.4506
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7225       0.8219        0.5153
1.4155
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7200       0.8194        0.5149
1.5410
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7257       0.8025        0.5533
1.4885
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7252       0.8161        0.5295
1.6438
     27        0.2390       0.8811        0.3304  1.5113
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7334       0.7992        0.5569
1.5197
     27        0.2572       0.8658        0.3729  1.5463
     27        0.2366       0.8817        0.3373  1.5952
  epoch    train_loss    valid_acc    valid_loss     dur
```

```
-------  ------------  -----------  ------------  ------
      1        0.6016       0.8205        0.4784
1.5010
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6091       0.8389        0.4574
1.5257
      2        0.4940       0.8366        0.4662
1.3514
      2        0.4974       0.8361        0.4605
1.4334
      2        0.5114       0.8173        0.5164
1.4063
      2        0.5071       0.8227        0.5024
1.4941
     28        0.2352       0.8842        0.3286  1.4648
      2        0.5161       0.8303        0.4854
1.4058
     28        0.2539       0.8753        0.3586  1.4219
     28        0.2321       0.8788        0.3465  1.5042
      2        0.4592       0.8406        0.4410
1.4131
      2        0.4577       0.8453        0.4170
1.4192
      3        0.4515       0.8486        0.4322
1.4943
      3        0.4491       0.8394        0.4455
1.4629
      3        0.4732       0.8270        0.4849
1.3826
      3        0.4718       0.8375        0.4599
1.4111
     29        0.2303       0.8803        0.3306  1.4799
      3        0.4805       0.8384        0.4617
1.4307
     29        0.2284       0.8820        0.3359  1.3437
     29        0.2506       0.8734        0.3568  1.4488
      3        0.4161       0.8144        0.5027  1.4426
      3        0.4095       0.8427        0.4207  1.4392
      4        0.4253       0.8547        0.4154
1.4861
      4        0.4234       0.8519        0.4111
1.4619
      4        0.4532       0.8397        0.4606
1.5005
      4        0.4524       0.8444        0.4433
1.5221
     30        0.2272       0.8864        0.3210
```

```
1.4439
     30       0.2239      0.8834       0.3405  1.3919
      4       0.4606      0.8403       0.4554
1.5202
     30       0.2479      0.8727       0.3592  1.4218
      4       0.3838      0.8534       0.4086
1.4644
      4       0.3802      0.8623       0.3904
1.4847
      5       0.4060      0.8488       0.4229  1.4098
      5       0.4024      0.8506       0.4188  1.3746
      5       0.4418      0.8372       0.4661  1.3600
      5       0.4383      0.8508       0.4273
1.3619
      5       0.4482      0.8456       0.4381
1.3639
      5       0.3638      0.8572       0.4147  1.3288
      5       0.3602      0.8416       0.4371  1.3777
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6076      0.8453       0.4320
1.3928
      6       0.3909      0.8577       0.4011
1.3969
      6       0.3886      0.8569       0.4035
1.3881
      6       0.4336      0.8450       0.4489
1.3356
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6107      0.8003       0.5342
1.3225
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6040      0.7784       0.5727
1.4359
      6       0.4285      0.8469       0.4334  1.3068
      6       0.4383      0.8456       0.4389  1.3185
      6       0.3461      0.8555       0.4149  1.3597
      6       0.3479      0.8666       0.3751
1.3665
      2       0.4648      0.8541       0.4093
1.3663
      7       0.3766      0.8592       0.3878
1.3309
      7       0.3778      0.8650       0.3808
1.3816
      7       0.4247      0.8480       0.4493  1.3748
```

2 0.4696 0.8367 0.4650
1.3141
2 0.4628 0.8228 0.4871
1.4193
7 0.4213 0.8531 0.4220
1.3634
7 0.4309 0.8533 0.4247
1.3749
7 0.3328 0.8616 0.4053
1.3944
7 0.3297 0.8712 0.3751
1.3991
3 0.4235 0.8462 0.4364 1.4096
8 0.3646 0.8552 0.4067 1.4372
8 0.4190 0.8358 0.4660 1.4018
8 0.3662 0.8655 0.3717
1.4983
3 0.4293 0.8383 0.4306
1.4562
3 0.4236 0.8287 0.4873 1.4182
8 0.4176 0.8550 0.4196
1.4369
8 0.4259 0.8488 0.4310 1.4763
8 0.3222 0.8594 0.4091 1.4230
8 0.3166 0.8666 0.3781 1.4098
4 0.3934 0.8506 0.4150 1.5714
9 0.4144 0.8409 0.4450 1.4286
9 0.3548 0.8619 0.3845
1.4824
9 0.3575 0.8614 0.3834 1.4717
4 0.4023 0.8361 0.4329 1.5918
4 0.4024 0.8520 0.4235
1.6186
9 0.4118 0.8530 0.4256 1.4519
9 0.4215 0.8548 0.4166
1.4251
9 0.3034 0.8747 0.3668
1.3764
9 0.3075 0.8570 0.4258 1.5210
5 0.3723 0.8639 0.3705
1.3285
10 0.4123 0.8423 0.4461 1.3410
10 0.3472 0.8688 0.3741 1.4005
10 0.3468 0.8647 0.3756
1.4562
5 0.3902 0.8086 0.5804 1.5411
5 0.3862 0.8530 0.4042
1.4341

10	0.4088	0.8525	0.4243	1.4380
10	0.4167	0.8514	0.4298	1.4909
10	0.3011	0.8588	0.4256	1.4342
10	0.2956	0.8720	0.3704	1.5383
6	0.3522	0.8645	0.3775	1.4639
11	0.4091	0.8488	0.4354	1.5135
11	0.3384	0.8664	0.3779	1.3893
11	0.3399	0.8664	0.3773	1.3945
6	0.3741	0.8511	0.4061	1.4155
6	0.3720	0.8564	0.4093	1.4247
11	0.4045	0.8536	0.4209	1.4008
11	0.4135	0.8481	0.4329	1.3927
11	0.2872	0.8662	0.3801	1.4084
11	0.2915	0.8644	0.4030	1.5669
12	0.3316	0.8652	0.3762	1.4082
7	0.3369	0.8736	0.3566	1.5301
12	0.4071	0.8495	0.4368	1.4971
12	0.3329	0.8691	0.3630	1.4845
7	0.3703	0.8642	0.3730	1.5451
7	0.3658	0.8545	0.3889	1.5602
12	0.4024	0.8523	0.4176	1.6393
12	0.4105	0.8544	0.4182	1.4824
12	0.2768	0.8770	0.3668	1.5679
12	0.2815	0.8542	0.4575	1.5442
8	0.3241	0.8777	0.3400	1.4242
13	0.3255	0.8597	0.3725	1.4010
13	0.4038	0.8461	0.4342	1.4909
13	0.3240	0.8608	0.3906	1.5678
8	0.3604	0.8567	0.3993	1.3838
8	0.3604	0.8603	0.3868	1.4533
13	0.4005	0.8531	0.4172	1.4168
13	0.4079	0.8612	0.4052	1.3596
13	0.2679	0.8656	0.4118	1.3998
9	0.3114	0.8597	0.4092	1.3832
13	0.2725	0.8666	0.4115	1.4757
14	0.3196	0.8772	0.3505	1.4682
14	0.4025	0.8472	0.4335	1.3983

14	0.3180	0.8664	0.3784	1.3981
9	0.3606	0.8294	0.4731	1.3600
9	0.3551	0.8547	0.3989	1.4279
14	0.4003	0.8555	0.4099	
1.3891				
14	0.4057	0.8570	0.4132	1.3552
14	0.2621	0.8756	0.3740	1.3883
14	0.2666	0.8523	0.4779	1.3148
10	0.3048	0.8733	0.3661	1.3767
15	0.3138	0.8706	0.3608	1.4086
15	0.3129	0.8761	0.3585	
1.4373				
15	0.4016	0.8531	0.4287	
1.4604				
10	0.3499	0.8684	0.3717	
1.4578				
10	0.3503	0.7852	0.6193	1.3793
15	0.3982	0.8506	0.4225	1.3942
15	0.4040	0.8419	0.4335	1.3759
15	0.2555	0.8817	0.3665	
1.4144				
15	0.2592	0.8534	0.4495	1.4595
11	0.2911	0.8841	0.3449	1.4450
16	0.3085	0.8731	0.3558	1.4242
16	0.3078	0.8722	0.3627	1.4094
16	0.3988	0.8502	0.4342	1.4143
11	0.3460	0.8644	0.3818	1.3602
11	0.3450	0.8555	0.4168	1.4049
16	0.3967	0.8636	0.4064	
1.4455				
16	0.4011	0.8523	0.4146	1.3774
16	0.2472	0.8778	0.3700	1.3381
12	0.2848	0.8711	0.3890	1.3431
16	0.2507	0.8767	0.4098	1.3993
17	0.3026	0.8720	0.3559	1.3630
17	0.3987	0.8505	0.4253	1.3819
17	0.3022	0.8759	0.3590	1.4591
12	0.3419	0.8603	0.3943	1.4165
12	0.3429	0.8430	0.4378	1.3702
17	0.3955	0.8584	0.4193	1.4513
17	0.4013	0.8631	0.4031	1.3518
17	0.2491	0.8800	0.3999	1.4521
13	0.2763	0.8822	0.3449	1.4277
17	0.2484	0.8652	0.4552	1.3873
18	0.2986	0.8773	0.3463	
1.4786				
18	0.2967	0.8786	0.3561	
1.3726				

18	0.3986	0.8408	0.4407	1.5833
13	0.3408	0.8578	0.4145	1.3889
13	0.3432	0.8583	0.3902	1.5372
18	0.3935	0.8572	0.4135	1.3640
18	0.3991	0.8550	0.4136	1.3982
18	0.2394	0.8762	0.4017	1.4135
18	0.2406	0.8706	0.4057	1.4361
14	0.2670	0.8603	0.4153	1.4491
19	0.2922	0.8770	0.3407	1.3844
19	0.2913	0.8725	0.3666	1.4586
19	0.3968	0.8481	0.4340	1.3978
14	0.3365	0.8605	0.4021	1.3771
14	0.3382	0.8450	0.4235	1.4042
19	0.3933	0.8497	0.4157	1.4135
19	0.3971	0.8564	0.4183	1.4896
19	0.2361	0.8711	0.3963	1.4093
15	0.2583	0.8744	0.3580	1.3925
19	0.2379	0.8594	0.4691	1.4388
20	0.2876	0.8664	0.3708	1.3809
20	0.2877	0.8733	0.3599	1.4075
20	0.3958	0.8438	0.4367	1.3622
15	0.3330	0.8542	0.4202	1.3244
15	0.3377	0.8597	0.3894	1.4028
20	0.3918	0.8586	0.4050	1.4181
20	0.3974	0.8597	0.4013	1.3456
20	0.2274	0.8789	0.3901	1.4098
16	0.2515	0.8722	0.3710	1.3543
20	0.2320	0.8691	0.4222	1.4274
21	0.2837	0.8742	0.3570	1.4324
21	0.2821	0.8786	0.3436	1.3989
21	0.3952	0.8405	0.4448	1.4342
16	0.3372	0.8544	0.4060	1.3957
16	0.3333	0.8716	0.3635	1.3811
21	0.3912	0.8525	0.4114	1.3830
21	0.3954	0.8642	0.3997	1.3923
21	0.2206	0.8811	0.3826	1.3642
17	0.2438	0.8773	0.3581	1.3765
21	0.2245	0.8689	0.4312	1.3501
22	0.2804	0.8716	0.3515	1.3974
22	0.2782	0.8805	0.3437	1.3595
22	0.3938	0.8492	0.4378	1.3162
17	0.3331	0.8473	0.4226	1.3818
17	0.3339	0.8662	0.3811	1.3647
22	0.3908	0.8662	0.3961	1.3682
22	0.3952	0.8661	0.4017	1.3084

22	0.2187	0.8769	0.4104	1.3971
18	0.2412	0.8750	0.3619	1.3930
22	0.2257	0.7980	0.7500	1.3726
23	0.2765	0.8772	0.3391	1.4228
23	0.2736	0.8727	0.3580	1.3951
23	0.3930	0.8472	0.4397	1.4225
18	0.3309	0.8606	0.3850	1.5284
18	0.3335	0.8712	0.3628	1.5037
23	0.3896	0.8583	0.4046	1.5141
23	0.3939	0.8586	0.4084	1.4640
23	0.2133	0.8708	0.4313	1.3984
19	0.2351	0.8742	0.3677	1.4721
23	0.2192	0.8561	0.4980	1.4525
24	0.2714	0.8842	0.3310	1.5371
24	0.2714	0.8725	0.3472	1.6005
24	0.3915	0.8444	0.4352	1.5672
19	0.3256	0.8645	0.3822	1.4304
19	0.3317	0.8516	0.4110	1.4786
24	0.3881	0.8552	0.4068	1.3919
24	0.3936	0.8597	0.4044	1.4954
24	0.2118	0.8772	0.4129	1.4571
20	0.2302	0.8470	0.4868	1.4254
24	0.2168	0.8667	0.4551	1.4166
25	0.2680	0.8759	0.3449	1.3484
25	0.2648	0.8806	0.3371	1.4093
25	0.3922	0.8516	0.4289	1.4412
20	0.3312	0.8577	0.4036	1.4258
20	0.3253	0.8691	0.3568	1.4586
25	0.3886	0.8608	0.4004	1.4119
25	0.3930	0.8605	0.4052	1.4884
25	0.2072	0.8661	0.4373	1.3892
21	0.2261	0.8739	0.3812	1.3671
25	0.2114	0.8608	0.4566	1.3863
26	0.2623	0.8742	0.3417	1.3884
26	0.2637	0.8784	0.3549	1.3938
26	0.3913	0.8498	0.4284	1.4273
21	0.3281	0.8527	0.4229	1.4459
21	0.3272	0.8630	0.3839	1.4266
26	0.3872	0.8659	0.3949	1.4188
26	0.3920	0.8653	0.3970	1.3642
26	0.2041	0.8748	0.4026	1.3766
26	0.2106	0.8664	0.4378	1.3671
22	0.2231	0.8739	0.3887	1.4514
27	0.2587	0.8788	0.3409	1.4098
27	0.2592	0.8819	0.3352	

```
1.4525
     27        0.3900       0.8511        0.4220  1.4568
     22        0.3222       0.8600        0.3960  1.4275
     27        0.3877       0.8614        0.3990  1.3755
     22        0.3260       0.8641        0.3971  1.5211
     27        0.3920       0.8627        0.3973  1.4030
     27        0.1983       0.8753        0.4064  1.4104
     27        0.2059       0.8616        0.4989  1.3375
     23        0.2162       0.8723        0.3873  1.3763
     28        0.2549       0.8825        0.3316  1.3467
     28        0.2554       0.8800        0.3375  1.3933
     28        0.3904       0.8531        0.4214  1.3789
     23        0.3278       0.8645        0.3820  1.3624
     28        0.3861       0.8612        0.4002  1.3849
     23        0.3254       0.8444        0.4452  1.3267
     28        0.3910       0.8545        0.4095  1.3359
     28        0.1962       0.8748        0.4363  1.3656
     28        0.1972       0.8695        0.4633  1.3817
     24        0.2095       0.8730        0.3907  1.3429
     29        0.2537       0.8712        0.3630  1.3879
     29        0.2529       0.8803        0.3448  1.4869
     24        0.3271       0.8694        0.3880  1.4004
     29        0.3905       0.8498        0.4271  1.4418
     24        0.3296       0.8727        0.3653  1.3859
     29        0.3859       0.8594        0.4128  1.4971
     29        0.3915       0.8652        0.3948  1.3938
     29        0.1940       0.8766        0.4226  1.4276
     29        0.2057       0.8680        0.5021  1.4314
     25        0.2047       0.8803        0.3879  1.4020
     30        0.2508       0.8814        0.3302  1.5178
     30        0.2496       0.8783        0.3575  1.4874
     25        0.3214       0.8647        0.4002  1.4019
     30        0.3883       0.8500        0.4229  1.4577
     25        0.3246       0.8778        0.3657  1.4592
     30        0.3859       0.8614        0.4024  1.4388
     30        0.3907       0.8602        0.4018  1.4189
     26        0.2012       0.8692        0.3996  1.3986
     30        0.1927       0.8730        0.4031  1.5353
     30        0.1981       0.8633        0.4666  1.5028
     26        0.3188       0.8350        0.4520  1.3686
     26        0.3261       0.8719        0.3738  1.3883
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6504       0.7192        0.7573
1.3487
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6153       0.8308        0.4650
```

```
1.4187
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6480       0.7978        0.5518
1.4155
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6456       0.7944        0.5491
1.3776
     27        0.2012       0.8873        0.3851  1.4183
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.3951       0.7214        0.9338
1.8305
     27        0.3233       0.8634        0.3998  1.4132
     27        0.3247       0.8486        0.4047  1.3544
      2        0.5526       0.7741        0.6078
1.3945
      2        0.4739       0.8356        0.4524
1.3675
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4109       0.7153        0.9585
1.8283
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4626       0.7245        0.9702
1.8013
      2        0.5525       0.7891        0.5589  1.4251
      2        0.5591       0.7909        0.5645  1.3794
     28        0.1969       0.8825        0.3959  1.4234
     28        0.3243       0.8508        0.4174  1.4195
     28        0.3217       0.8583        0.3985  1.3519
      2        0.8112       0.7533        0.7379
1.8032
      3        0.5471       0.8128        0.5082
1.3833
      3        0.4318       0.8405        0.4250
1.3886
      3        0.5523       0.7617        0.6290  1.3944
      3        0.5417       0.8320        0.4826
1.4783
      2        0.8279       0.7573        0.7436
1.7774
     29        0.1975       0.8756        0.4303  1.4235
      2        0.8286       0.7578        0.7436
1.9422
     29        0.3180       0.8536        0.4285  1.3239
```

```
     29        0.3238       0.8617        0.3898  1.4500
      4        0.5445       0.8125        0.5248  1.4117
      4        0.4106       0.8384        0.4540  1.4266
      3        0.6870       0.7725        0.6626
1.8276
      4        0.5472       0.8161        0.5245  1.4074
      4        0.5484       0.7678        0.6080  1.5978
     30        0.1902       0.8725        0.4190  1.5676
      3        0.6932       0.7750        0.6601
2.0395
     30        0.3217       0.8316        0.5051  1.6164
      3        0.6959       0.7742        0.6603
2.2016
     30        0.3204       0.8577        0.4031  1.7124
      5        0.5429       0.7555        0.6256  1.4987
      5        0.3919       0.8544        0.4054
1.5704
      5        0.5390       0.8120        0.5310  1.5111
      5        0.5523       0.8220        0.4787  1.4207
      4        0.6276       0.7823        0.6191
1.8178
      6        0.5421       0.8067        0.5217  1.3757
      6        0.3833       0.8316        0.4389  1.3871
      4        0.6316       0.7887        0.6123
1.8771
      4        0.6339       0.7905        0.6120
1.8397
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4263       0.7212        0.9610
1.7899
      6        0.5389       0.8156        0.5098  1.3977
      6        0.5465       0.8053        0.5558  1.4079
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4178       0.7236        0.9551
1.8455
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        1.4559       0.7133        0.9783
1.7572
      5        0.5893       0.7920        0.5888
1.7742
      7        0.5391       0.7898        0.5811  1.4105
      7        0.3720       0.8612        0.3894
1.4109
      7        0.5409       0.8137        0.5008  1.4252
      7        0.5482       0.7831        0.5591  1.3641
```

5	0.5941	0.8020	0.5800	
1.8128				
5	0.5944	0.7967	0.5816	
1.7878				
2	0.8226	0.7494	0.7448	
1.7916				
2	0.8232	0.7580	0.7383	
1.8152				
2	0.8358	0.7530	0.7527	
1.8172				
8	0.5413	0.8058	0.5313	1.3842
6	0.5618	0.8013	0.5655	
1.7573				
8	0.3659	0.8662	0.3564	
1.3991				
8	0.5366	0.7956	0.5684	1.4235
8	0.5457	0.7873	0.5591	1.4044
6	0.5665	0.8077	0.5547	
1.7105				
6	0.5660	0.8055	0.5554	
1.7969				
3	0.6896	0.7678	0.6672	
1.7927				
9	0.5366	0.8233	0.4881	
1.3948				
9	0.3603	0.8580	0.3872	1.4059
3	0.6920	0.7761	0.6574	
1.8317				
3	0.6997	0.7686	0.6699	
1.8328				
7	0.5405	0.8080	0.5478	
1.7975				
9	0.5404	0.7877	0.5489	1.3925
9	0.5455	0.7442	0.6760	1.4347
7	0.5456	0.8159	0.5343	
1.7730				
10	0.5360	0.7963	0.5504	1.4071
7	0.5443	0.8145	0.5342	
1.7842				
4	0.6291	0.7794	0.6260	
1.8402				
10	0.3551	0.8634	0.3773	1.4529
10	0.5445	0.8037	0.5428	1.4122
4	0.6308	0.7900	0.6104	
1.7818				
10	0.5476	0.7892	0.5721	1.5126
4	0.6382	0.7841	0.6200	
1.7616				

```
       8          0.5241        0.8123         0.5358
1.7842
      11          0.5417        0.7931         0.5836   1.3925
       8          0.5282        0.8177         0.5204
1.7457
      11          0.3465        0.8473         0.4195   1.3967
       8          0.5265        0.8202         0.5185
1.7475
      11          0.5302        0.8164         0.5032   1.3638
       5          0.5911        0.7916         0.5957
1.7898
      11          0.5454        0.7936         0.5399   1.4581
       5          0.5920        0.8000         0.5782
1.7842
       5          0.5988        0.7928         0.5877
1.7953
       9          0.5103        0.8164         0.5235
1.8142
      12          0.5399        0.7994         0.5573   1.3750
      12          0.3465        0.8538         0.4019   1.4338
      12          0.5364        0.7812         0.5976   1.4307
       9          0.5148        0.8213         0.5091
1.7767
      12          0.5450        0.7780         0.6017   1.3794
       9          0.5118        0.8214         0.5098
1.7652
       6          0.5641        0.7992         0.5709
1.8269
       6          0.5641        0.8111         0.5526
1.7539
      13          0.5415        0.7963         0.5499   1.3550
       6          0.5710        0.8037         0.5611
1.8058
      10          0.4988        0.8183         0.5143
1.8156
      13          0.3449        0.8677         0.3523
1.4385
      13          0.5408        0.8113         0.5519   1.3971
      13          0.5434        0.8128         0.5158   1.3637
      10          0.5034        0.8263         0.4959
1.7673
      10          0.5000        0.8283         0.4949
1.7578
      14          0.5420        0.7931         0.5559   1.3565
       7          0.5436        0.8034         0.5547
1.7790
      14          0.3401        0.8527         0.3929   1.4117
       7          0.5425        0.8163         0.5332
```

1.8135				
7	0.5492	0.8133	0.5412	
1.7823				
14	0.5388	0.8161	0.5002	1.4592
14	0.5430	0.8033	0.5396	1.3925
11	0.4891	0.8219	0.5063	
1.7977				
15	0.5367	0.8120	0.5242	1.4434
11	0.4933	0.8292	0.4871	
1.8245				
11	0.4897	0.8308	0.4851	
1.8914				
15	0.3406	0.8534	0.4004	1.4721
8	0.5268	0.8108	0.5399	
1.9843				
15	0.5386	0.8123	0.5171	1.5706
8	0.5260	0.8191	0.5188	
1.8764				
15	0.5392	0.7727	0.5993	1.6066
8	0.5320	0.8184	0.5236	
1.8215				
12	0.4802	0.8231	0.4986	
2.0689				
16	0.5405	0.7814	0.5795	1.5768
16	0.3436	0.8645	0.3617	1.4682
12	0.4851	0.8337	0.4774	
1.9789				
12	0.4805	0.8303	0.4799	1.8856
16	0.5324	0.8153	0.5344	1.5175
16	0.5468	0.8187	0.4925	1.4957
9	0.5134	0.8111	0.5284	
1.8491				
9	0.5120	0.8245	0.5053	
1.8537				
9	0.5179	0.8203	0.5129	
1.9978				
17	0.5373	0.7956	0.5379	1.4851
13	0.4727	0.8298	0.4908	
1.8915				
17	0.3350	0.8558	0.3895	1.4129
17	0.5397	0.7998	0.5441	1.4951
13	0.4767	0.8344	0.4731	
1.8593				
17	0.5455	0.8164	0.5171	1.4861
13	0.4724	0.8337	0.4717	
1.8084				
10	0.5019	0.8163	0.5196	
1.9577				

18	0.5474	0.8122	0.5081	1.4828
10	0.4997	0.8242	0.4988	1.8382
18	0.3320	0.8348	0.4425	1.4803
10	0.5064	0.8258	0.5004	
1.8941				
14	0.4658	0.8303	0.4858	
1.8021				
18	0.5412	0.7180	0.7773	1.3543
18	0.5421	0.7831	0.5712	1.3941
14	0.4704	0.8367	0.4647	
1.8091				
14	0.4654	0.8352	0.4645	
1.7816				
19	0.5364	0.7805	0.5867	1.4111
19	0.3349	0.8567	0.3927	1.4113
11	0.4922	0.8217	0.5089	
1.8544				
11	0.4904	0.8313	0.4878	
1.7838				
19	0.5366	0.8167	0.5205	1.3760
11	0.4964	0.8294	0.4896	
1.7623				
19	0.5428	0.7081	0.7320	1.3765
15	0.4599	0.8297	0.4818	1.7667
20	0.5396	0.7886	0.5623	1.3582
15	0.4640	0.8414	0.4599	
1.8306				
15	0.4584	0.8377	0.4606	
1.7705				
20	0.3310	0.8752	0.3590	1.4026
20	0.5365	0.8202	0.5210	1.4145
20	0.5415	0.7697	0.6200	1.3914
12	0.4833	0.8220	0.5036	
1.8122				
12	0.4818	0.8347	0.4778	
1.8902				
12	0.4874	0.8330	0.4822	
1.8664				
16	0.4541	0.8322	0.4764	
1.7807				
21	0.5398	0.7694	0.5874	1.3654
21	0.3309	0.8536	0.4056	1.3934
16	0.4530	0.8391	0.4564	
1.6749				
16	0.4581	0.8411	0.4541	1.7815
21	0.5341	0.7964	0.5400	1.3987
21	0.5460	0.8300	0.4703	1.3735
13	0.4758	0.8225	0.4970	

1.8074				
13	0.4740	0.8353	0.4756	
1.8470				
22	0.5386	0.7216	0.7642	1.4222
13	0.4797	0.8350	0.4745	
1.8801				
17	0.4490	0.8339	0.4729	
1.8808				
22	0.3313	0.8478	0.4056	1.4858
22	0.5403	0.8069	0.5246	1.3962
22	0.5418	0.7995	0.5590	1.3868
17	0.4471	0.8395	0.4540	
1.8422				
17	0.4526	0.8423	0.4509	
1.9194				
23	0.5397	0.8050	0.5457	1.5074
14	0.4686	0.8261	0.4908	
1.9041				
14	0.4672	0.8372	0.4697	
1.8531				
23	0.3278	0.8564	0.3893	1.4834
14	0.4727	0.8397	0.4676	
1.8291				
23	0.5386	0.7745	0.5922	1.6187
18	0.4443	0.8336	0.4700	1.9055
23	0.5422	0.8014	0.5384	1.6223
18	0.4421	0.8459	0.4465	
1.8203				
24	0.5390	0.8034	0.5413	1.4480
18	0.4479	0.8436	0.4463	
1.8932				
24	0.3297	0.8695	0.3465	1.5042
15	0.4627	0.8294	0.4854	
1.8297				
15	0.4613	0.8403	0.4619	
1.8405				
24	0.5369	0.8097	0.5250	1.4286
24	0.5409	0.8206	0.4874	1.3738
15	0.4663	0.8389	0.4631	1.8374
19	0.4399	0.8350	0.4687	
1.8760				
25	0.5385	0.7850	0.5975	1.4308
19	0.4377	0.8434	0.4434	1.7366
19	0.4431	0.8448	0.4429	
1.7300				
25	0.3259	0.8531	0.4086	1.3941
25	0.5383	0.8203	0.4892	1.3759
25	0.5451	0.7847	0.5770	1.4331

16	0.4549	0.8419	0.4573	
1.8196				
16	0.4569	0.8305	0.4802	
1.9764				
16	0.4604	0.8411	0.4590	
1.8943				
26	0.5366	0.7864	0.5893	1.5862
20	0.4358	0.8384	0.4611	
1.8640				
26	0.3276	0.8742	0.3498	1.4416
20	0.4334	0.8452	0.4390	1.9542
26	0.5416	0.8031	0.5237	1.5264
26	0.5423	0.7939	0.5564	1.5404
20	0.4386	0.8475	0.4398	
1.9916				
17	0.4501	0.8453	0.4527	
2.0572				
17	0.4515	0.8327	0.4755	
2.2309				
27	0.5409	0.7502	0.6934	1.7448
27	0.3256	0.8659	0.3813	1.7262
21	0.4317	0.8373	0.4590	2.0533
17	0.4550	0.8434	0.4535	
2.2561				
27	0.5352	0.8089	0.5388	1.5451
27	0.5404	0.8073	0.5305	1.7027
21	0.4291	0.8464	0.4360	
2.1682				
21	0.4349	0.8453	0.4364	2.0540
28	0.5339	0.7623	0.6175	1.4218
28	0.3214	0.8716	0.3531	1.4262
18	0.4453	0.8436	0.4504	1.9826
28	0.5353	0.7922	0.5531	1.4509
18	0.4469	0.8350	0.4724	
1.9063				
28	0.5439	0.8202	0.5053	1.4677
18	0.4501	0.8436	0.4513	
1.8855				
22	0.4278	0.8392	0.4554	
1.9286				
22	0.4248	0.8492	0.4348	
1.8203				
29	0.5415	0.8164	0.5116	1.4418
22	0.4306	0.8461	0.4332	1.8701
29	0.3211	0.8619	0.3733	1.4019
29	0.5404	0.7919	0.5530	1.4584
29	0.5436	0.8075	0.5232	1.3923
19	0.4405	0.8458	0.4468	

```
1.8294
     19       0.4420       0.8352        0.4679
1.9116
     19       0.4452       0.8438        0.4479
1.8589
     23       0.4245       0.8387        0.4564  1.9165
     30       0.5372       0.7616        0.6090  1.4956
     30       0.3286       0.8602        0.3675  1.4955
     30       0.5384       0.7980        0.5422  1.3861
     23       0.4211       0.8491        0.4324  2.0005
     23       0.4268       0.8512        0.4276
1.9547
     30       0.5429       0.8159        0.5262  1.4653
     20       0.4361       0.8466        0.4418
1.9203
     20       0.4376       0.8356        0.4687  1.8989
     20       0.4410       0.8466        0.4431
1.8800
     24       0.4210       0.8397        0.4503
1.8833
     24       0.4172       0.8514        0.4278
1.9155
     24       0.4232       0.8483        0.4278  1.9274
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       1.4057       0.7288        0.9476
1.9177
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       1.4368       0.7189        0.9674
1.8789
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       1.4313       0.7223        0.9705
1.9056
     21       0.4327       0.8489        0.4394
1.9427
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6975       0.8094        0.5293
1.9485
     21       0.4336       0.8380        0.4621
1.9775
     21       0.4371       0.8494        0.4369
1.8989
     25       0.4177       0.8391        0.4526  1.9215
     25       0.4142       0.8503        0.4253  1.9364
     25       0.4196       0.8519        0.4246
```

```
1.8961
     2    0.8342    0.7578    0.7480
1.9321
     2    0.8206    0.7509    0.7482
2.0256
     2    0.8334    0.7512    0.7555
2.0991
     2    0.4857    0.8305    0.4777
1.9936
    22    0.4287    0.8475    0.4364  2.0242
    22    0.4298    0.8391    0.4591
1.8660
    22    0.4330    0.8500    0.4350
2.0460
    26    0.4149    0.8383    0.4503  1.9868
    26    0.4109    0.8500    0.4231  1.9664
    26    0.4168    0.8522    0.4218
1.9523
     3    0.6964    0.7655    0.6769
1.9906
     3    0.7021    0.7769    0.6664
2.1026
     3    0.7030    0.7708    0.6739
1.8942
     3    0.4431    0.8395    0.4492
1.9674
    23    0.4252    0.8478    0.4365  2.0543
    23    0.4260    0.8380    0.4583  2.0155
    27    0.4118    0.8438    0.4456
1.9023
    23    0.4291    0.8492    0.4318  2.0870
    27    0.4078    0.8516    0.4212
1.9184
    27    0.4136    0.8520    0.4200  1.9730
     4    0.6396    0.7780    0.6336
1.9317
     4    0.6420    0.7897    0.6204
1.9190
     4    0.6443    0.7863    0.6274
1.9388
     4    0.4142    0.8464    0.4292
1.9454
    24    0.4229    0.8397    0.4553
1.9701
    24    0.4219    0.8475    0.4322  1.9985
    28    0.4090    0.8442    0.4420
1.8464
    24    0.4257    0.8519    0.4292
```

1.9831					
	28	0.4052	0.8519	0.4179	
1.9826					
	28	0.4105	0.8512	0.4159	2.0302
	5	0.6036	0.7889	0.6040	
2.0540					
	5	0.6050	0.7966	0.5897	
2.2680					
	5	0.6072	0.7941	0.5954	
2.0950					
	5	0.3939	0.8461	0.4219	2.1292
	25	0.4189	0.8492	0.4306	
2.3063					
	25	0.4195	0.8398	0.4519	
2.3504					
	29	0.4064	0.8439	0.4411	2.2993
	25	0.4223	0.8519	0.4251	2.1509
	29	0.4019	0.8541	0.4174	
2.0825					
	29	0.4076	0.8531	0.4180	2.1457
	6	0.5776	0.7994	0.5820	
2.0597					
	6	0.5782	0.8091	0.5653	
2.0028					
	6	0.5811	0.8028	0.5721	
2.0366					
	6	0.3767	0.8517	0.4141	
1.9389					
	26	0.4153	0.8506	0.4272	
1.9326					
	26	0.4162	0.8422	0.4490	
1.9854					
	30	0.4036	0.8472	0.4398	
1.9296					
	26	0.4189	0.8508	0.4240	1.9603
	30	0.3992	0.8545	0.4133	
2.0438					
	30	0.4045	0.8544	0.4132	
1.9344					
	7	0.5577	0.8002	0.5678	
1.9766					
	7	0.5579	0.8114	0.5483	
1.8674					
	7	0.5606	0.8105	0.5567	
1.8851					
	7	0.3643	0.8578	0.4031	
2.1817					
	27	0.4125	0.8488	0.4255	2.0514

```
     27        0.4138       0.8439        0.4458
2.1582
     27        0.4159       0.8533        0.4213
2.1726
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7027       0.8170        0.5139
2.1170
      8        0.5420       0.8047        0.5540
2.1279
      8        0.5420       0.8145        0.5347
2.1833
      8        0.5446       0.8098        0.5434  2.0645
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7025       0.8108        0.5347
2.1231
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7156       0.8041        0.5419
2.1754
      8        0.3515       0.8572        0.3943  1.9306
     28        0.4099       0.8512        0.4251
1.8859
     28        0.4106       0.8444        0.4457
2.1129
     28        0.4129       0.8525        0.4206  1.9395
      2        0.4859       0.8392        0.4559
1.9346
      9        0.5290       0.8202        0.5257
1.8592
      9        0.5293       0.8105        0.5408
1.9939
      9        0.5317       0.8214        0.5257
1.9194
      2        0.4888       0.8355        0.4629
1.9638
      2        0.4886       0.8320        0.4719
1.8743
      9        0.3402       0.8603        0.3967  1.9554
     29        0.4069       0.8498        0.4215  1.8954
     29        0.4080       0.8436        0.4448  1.9482
     29        0.4101       0.8545        0.4191
1.9483
      3        0.4411       0.8473        0.4314
1.8362
     10        0.5182       0.8181        0.5306
1.9046
```

```
     10       0.5183       0.8253        0.5121
1.9499
     10       0.5205       0.8203        0.5175  1.9046
      3       0.4444       0.8467        0.4294
1.9218
      3       0.4433       0.8409        0.4483
1.8545
     10       0.3307       0.8578        0.3949  1.9131
     30       0.4041       0.8522        0.4184
1.9567
     30       0.4053       0.8423        0.4429  1.9371
     30       0.4072       0.8572        0.4137
1.9135
      4       0.4123       0.8577        0.4011
1.9679
     11       0.5094       0.8170        0.5234  1.8175
     11       0.5118       0.8280        0.5079
1.8126
     11       0.5092       0.8280        0.5043
1.8739
      4       0.4159       0.8559        0.4062
1.8914
      4       0.4161       0.8408        0.4466  1.8698
     11       0.3212       0.8566        0.4043  1.9230
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.7117       0.8119        0.5184
1.9448
      5       0.3912       0.8564        0.3990  1.8900
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.7110       0.8102        0.5280
1.8827
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.7235       0.8064        0.5437
1.9534
     12       0.5011       0.8214        0.5167
1.9391
     12       0.5013       0.8280        0.4996  1.8883
     12       0.5038       0.8289        0.5001
1.9433
      5       0.3945       0.8541        0.3975  1.9370
      5       0.3958       0.8417        0.4368
1.9297
     12       0.3118       0.8572        0.4032  2.0159
      2       0.4881       0.8333        0.4705
2.1429
```

6	0.3742	0.8580	0.3931	1.9847
2	0.4920	0.8423	0.4458	2.1156
2	0.5090	0.8108	0.5190	2.0317
13	0.4944	0.8323	0.4904	2.1567
13	0.4971	0.8303	0.4943	2.1609
13	0.4943	0.8206	0.5128	2.2320
6	0.3789	0.8611	0.3818	2.1543
6	0.3799	0.8555	0.4067	2.1912
13	0.3040	0.8684	0.3867	2.0002
7	0.3614	0.8661	0.3717	2.0059
3	0.4442	0.8411	0.4426	2.1340
3	0.4686	0.8269	0.4812	1.9425
3	0.4477	0.8316	0.4552	2.0783
14	0.4877	0.8347	0.4852	1.9107
14	0.4882	0.8234	0.5052	1.8945
14	0.4907	0.8331	0.4909	1.9856
7	0.3654	0.8653	0.3819	1.9018
7	0.3679	0.8620	0.3955	1.9111
14	0.2964	0.8706	0.3701	1.8792
8	0.3478	0.8673	0.3733	1.8933
4	0.4161	0.8492	0.4235	1.9485
4	0.4492	0.8242	0.4856	1.9041
4	0.4203	0.8572	0.4154	1.9257
15	0.4824	0.8372	0.4827	1.9180
15	0.4830	0.8202	0.5021	1.9110
15	0.4858	0.8355	0.4836	1.9334
8	0.3514	0.8728	0.3633	1.8985

8	0.3562	0.8589	0.3978	1.8617
15	0.2896	0.8752	0.3580	1.9015
9	0.3373	0.8705	0.3627	1.9056
5	0.3948	0.8606	0.3952	1.9146
5	0.4353	0.8356	0.4656	1.9335
16	0.4775	0.8373	0.4763	1.8821
16	0.4779	0.8228	0.4986	1.8772
5	0.4005	0.8586	0.3941	1.9216
16	0.4807	0.8373	0.4796	1.9030
9	0.3407	0.8714	0.3645	1.9009
9	0.3463	0.8527	0.4026	1.9020
16	0.2823	0.8739	0.3645	1.8849
10	0.3282	0.8709	0.3584	1.9291
6	0.3775	0.8530	0.4067	1.9968
6	0.4283	0.8419	0.4470	1.9052
17	0.4725	0.8398	0.4727	1.8625
17	0.4734	0.8272	0.4948	1.8935
6	0.3859	0.8514	0.4043	1.9032
17	0.4763	0.8372	0.4773	1.8817
10	0.3375	0.8616	0.3896	1.9020
10	0.3312	0.8731	0.3580	1.9278
17	0.2739	0.8731	0.3572	1.9148
11	0.3172	0.8742	0.3545	1.8964
7	0.3660	0.8672	0.3752	1.9479
7	0.4208	0.8430	0.4454	1.8952
18	0.4695	0.8287	0.4897	1.8523
18	0.4684	0.8387	0.4693	1.9851
7	0.3726	0.8623	0.3764	1.9365
18	0.4720	0.8394	0.4723	1.9568
11	0.3288	0.8647	0.3825	

1.9082				
11	0.3236	0.8756	0.3550	
1.9210				
18	0.2684	0.8633	0.3775	1.9206
12	0.3098	0.8619	0.3743	1.9235
8	0.3554	0.8669	0.3776	1.9394
8	0.4149	0.8419	0.4463	1.9139
19	0.4647	0.8394	0.4669	1.9189
19	0.4660	0.8309	0.4853	
2.0271				
8	0.3614	0.8658	0.3722	
1.9547				
19	0.4685	0.8397	0.4689	
1.9390				
12	0.3218	0.8683	0.3793	
1.8908				
12	0.3143	0.8731	0.3566	1.9164
19	0.2613	0.8717	0.3678	1.9100
13	0.3017	0.8803	0.3390	
2.1365				
9	0.4116	0.8431	0.4436	
2.2446				
9	0.3449	0.8689	0.3640	
2.2981				
13	0.3141	0.8689	0.3730	
1.9221				
20	0.4613	0.8422	0.4629	
2.2997				
9	0.3511	0.8634	0.3861	2.2725
20	0.4621	0.8289	0.4843	2.3554
20	0.4650	0.8364	0.4683	2.3300
13	0.3041	0.8747	0.3559	2.3038
20	0.2567	0.8730	0.3655	2.3102
14	0.2936	0.8780	0.3452	2.3222
14	0.3074	0.8572	0.3988	1.8933
10	0.3348	0.8711	0.3691	2.1718
10	0.4083	0.8434	0.4389	
2.2738				
21	0.4579	0.8436	0.4602	
2.2076				
10	0.3411	0.8722	0.3565	
2.2116				
21	0.4590	0.8273	0.4846	2.3902
21	0.4618	0.8405	0.4632	
2.3776				
14	0.2979	0.8791	0.3389	
2.3907				
21	0.2494	0.8689	0.3805	2.2905

15	0.3022	0.8681	0.3829	1.8134
15	0.2865	0.8742	0.3517	2.3431
11	0.3272	0.8748	0.3556	2.2674
11	0.4048	0.8489	0.4318	2.2929
22	0.4550	0.8414	0.4595	2.2660
11	0.3325	0.8708	0.3577	2.2243
22	0.4587	0.8422	0.4605	2.1424
22	0.4562	0.8316	0.4784	2.3101
15	0.2889	0.8809	0.3334	2.1341
22	0.2439	0.8744	0.3594	2.1804
16	0.2956	0.8658	0.3731	1.7324
16	0.2802	0.8703	0.3562	2.1424
12	0.3193	0.8725	0.3585	2.1380
23	0.4522	0.8461	0.4553	2.1796
12	0.4022	0.8503	0.4282	2.3506
12	0.3252	0.8664	0.3709	2.2868
23	0.4557	0.8428	0.4580	2.0837
23	0.4533	0.8323	0.4771	2.2594
16	0.2821	0.8780	0.3451	2.2459
23	0.2388	0.8783	0.3517	2.1802
17	0.2897	0.8720	0.3675	1.7771
17	0.2719	0.8806	0.3382	2.1912
13	0.3118	0.8730	0.3467	2.1921
24	0.4494	0.8445	0.4551	2.1673
18	0.2845	0.8689	0.3697	1.7582
24	0.4532	0.8456	0.4554	2.1638
13	0.4002	0.8472	0.4277	2.2993
13	0.3176	0.8759	0.3432	2.2617
24	0.4505	0.8336	0.4751	2.1401
24	0.2337	0.8811	0.3429	2.1262
17	0.2756	0.8855	0.3286	2.2266

18	0.2658	0.8788	0.3381	1.9923
14	0.3049	0.8772	0.3501	1.9480
19	0.2787	0.8762	0.3561	
1.7592				
25	0.4467	0.8477	0.4521	
2.0093				
25	0.4508	0.8439	0.4536	2.0800
14	0.3100	0.8691	0.3668	2.0288
14	0.3974	0.8530	0.4241	
2.0479				
25	0.4481	0.8361	0.4725	
1.9591				
25	0.2289	0.8777	0.3526	1.9813
18	0.2694	0.8852	0.3340	2.0753
20	0.2737	0.8664	0.3829	1.7413
19	0.2609	0.8691	0.3566	2.0858
15	0.2973	0.8722	0.3515	2.0244
26	0.4446	0.8445	0.4514	2.0880
15	0.3961	0.8478	0.4246	2.0152
15	0.3032	0.8733	0.3418	2.0294
26	0.4483	0.8455	0.4524	2.0612
26	0.4458	0.8334	0.4725	1.9659
26	0.2228	0.8762	0.3595	2.0482
19	0.2638	0.8839	0.3354	2.0902
21	0.2688	0.8770	0.3522	
1.7251				
20	0.2538	0.8711	0.3567	2.0456
16	0.2920	0.8784	0.3450	
2.0553				
27	0.4417	0.8466	0.4490	2.1748
27	0.4439	0.8342	0.4691	1.9767
16	0.3941	0.8498	0.4298	2.0568
16	0.2972	0.8791	0.3371	
2.0980				
27	0.4464	0.8456	0.4494	2.1643
27	0.2198	0.8759	0.3520	2.0424
20	0.2573	0.8861	0.3221	
2.0817				
22	0.2637	0.8662	0.3747	1.7233
21	0.2488	0.8800	0.3397	1.9830
17	0.2859	0.8794	0.3416	
2.0119				
28	0.4418	0.8366	0.4670	
2.0039				
28	0.4401	0.8486	0.4462	
2.0396				
17	0.3930	0.8506	0.4304	2.0321
28	0.4439	0.8470	0.4475	

```
1.9308
     17        0.2912       0.8778       0.3412  2.0616
     28        0.2142       0.8762       0.3517  2.0411
     23        0.2587       0.8664       0.3821  1.7676
     21        0.2523       0.8800       0.3315  2.0456
     18        0.2807       0.8767       0.3452  2.1885
     22        0.2424       0.8758       0.3508  2.4268
     29        0.4398       0.8372       0.4650
2.0860
     29        0.4424       0.8477       0.4472
2.1028
     29        0.4382       0.8478       0.4449  2.1703
     18        0.3916       0.8442       0.4325  2.1727
     24        0.2550       0.8706       0.3618  1.7755
     18        0.2865       0.8789       0.3340  2.1594
     29        0.2089       0.8823       0.3447  2.0673
     22        0.2461       0.8848       0.3296  2.0920
     19        0.2756       0.8822       0.3371
1.9501
     23        0.2377       0.8794       0.3489  1.9801
     25        0.2506       0.8784       0.3538  1.7738
     30        0.4403       0.8483       0.4448
1.8955
     30        0.4377       0.8378       0.4642
1.9891
     30        0.4361       0.8483       0.4437  1.9510
     19        0.3917       0.8481       0.4296  1.9414
     19        0.2813       0.8781       0.3421  1.9511
     30        0.2048       0.8819       0.3510  1.9511
     23        0.2409       0.8856       0.3234  1.9618
     20        0.2700       0.8842       0.3298
1.8964
     24        0.2330       0.8852       0.3328
1.9230
     26        0.2459       0.8733       0.3515  1.7152
     20        0.2754       0.8750       0.3414  1.9080
     20        0.3905       0.8484       0.4229  1.9872
     24        0.2354       0.8781       0.3427  2.0160
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7259       0.8106        0.5439
1.9759
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.7092       0.8178        0.5291
1.9851
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
```

```
      1      0.6016       0.8136        0.4926
2.0887
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1      0.6005       0.8367        0.4401
1.9990
     27      0.2426       0.8786        0.3464
1.7335
     21      0.2645       0.8836        0.3300  2.0818
     25      0.2258       0.8778        0.3428  2.0619
     21      0.2699       0.8755        0.3422  2.1408
     21      0.3900       0.8491        0.4227  2.1457
     25      0.2292       0.8845        0.3270  2.0491
      2      0.5077       0.8203        0.4921
2.0081
      2      0.5123       0.8298        0.4868
2.0555
      2      0.4534       0.8348        0.4865
2.0126
      2      0.4483       0.8331        0.4551  2.0039
     28      0.2383       0.8711        0.3646  1.7011
     22      0.2602       0.8833        0.3292  2.0487
     26      0.2225       0.8853        0.3271
2.0947
     22      0.2662       0.8853        0.3287
1.9938
     22      0.3884       0.8475        0.4262  2.0525
     26      0.2251       0.8842        0.3255  2.0055
      3      0.4734       0.8367        0.4665
2.0569
      3      0.4763       0.8389        0.4651
2.1208
     29      0.2339       0.8781        0.3461  1.7369
      3      0.4064       0.8216        0.5029  2.0519
      3      0.4066       0.8558        0.4061
2.0517
     23      0.2557       0.8797        0.3450  2.0708
     27      0.2159       0.8862        0.3256
2.0870
     23      0.2606       0.8850        0.3209  1.9986
     23      0.3896       0.8541        0.4172  2.0210
     27      0.2200       0.8673        0.3622  1.9832
     30      0.2316       0.8778        0.3494  1.7558
      4      0.4518       0.8391        0.4680  1.9609
      4      0.3760       0.8584        0.3903
1.8540
      4      0.3799       0.8581        0.4036
1.8924
```

```
      4        0.4582        0.8425         0.4490
2.0066
     24        0.2524        0.8803         0.3373  1.8911
     28        0.2137        0.8831         0.3380  2.0229
     24        0.2559        0.8869         0.3214  2.0007
     24        0.3876        0.8498         0.4240  1.9738
     28        0.2152        0.8811         0.3237  1.9773
      5        0.4397        0.8391         0.4507  2.0627
      5        0.3591        0.8661         0.3946
1.9125
      5        0.3518        0.8648         0.3801
1.9781
      5        0.4458        0.8509         0.4335
1.9829
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.5969        0.8319         0.4624
1.6811
     25        0.2476        0.8803         0.3385  2.0589
     29        0.2089        0.8855         0.3253  2.2129
     25        0.2529        0.8823         0.3349  2.0989
     25        0.3853        0.8466         0.4278  2.3012
      6        0.3381        0.8514         0.4239  1.9660
      2        0.4544        0.8483         0.4067
1.8266
      6        0.4303        0.8480         0.4388
2.1054
     29        0.2103        0.8883         0.3119
2.3378
      6        0.4352        0.8438         0.4559  2.1276
      6        0.3361        0.8689         0.3584
2.2689
     26        0.2435        0.8869         0.3217
2.3242
     30        0.2046        0.8833         0.3444  2.0297
     26        0.2496        0.8856         0.3143  2.1251
      3        0.4116        0.8447         0.4175  1.7329
      7        0.3262        0.8605         0.4007  1.9850
     26        0.3869        0.8564         0.4203  2.0480
      7        0.4207        0.8484         0.4331
1.9806
     30        0.2059        0.8888         0.3206  2.0308
      7        0.4294        0.8569         0.4200
2.0041
      7        0.3168        0.8681         0.3711  2.0753
     27        0.2393        0.8884         0.3162
1.9416
      4        0.3804        0.8645         0.3802
```

```
1.6987
     27       0.2448       0.8775       0.3302  1.9003
      8       0.3073       0.8598       0.4097  1.9496
     27       0.3855       0.8558       0.4148  2.0482
  epoch   train_loss    valid_acc   valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6003       0.8163       0.5166
1.9415
      8       0.4163       0.8519       0.4256
1.9454
      8       0.4227       0.8552       0.4191  1.9884
      8       0.3026       0.8627       0.3931  1.8855
  epoch   train_loss    valid_acc   valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6019       0.8434       0.4479
2.0208
      5       0.3629       0.8677       0.3524
1.7041
     28       0.2360       0.8822       0.3416  2.0557
     28       0.2410       0.8866       0.3262  2.0830
      9       0.2911       0.8705       0.4043  2.0961
     28       0.3842       0.8578       0.4198  2.1532
      2       0.4608       0.8244       0.4767
2.1827
      9       0.4123       0.8519       0.4243  2.1512
      9       0.2918       0.8528       0.4260  2.0180
      9       0.4184       0.8531       0.4248  2.1877
      6       0.3404       0.8598       0.3918  1.7308
      2       0.4552       0.8544       0.4194
2.0853
     29       0.2331       0.8881       0.3285  2.1356
     29       0.2368       0.8862       0.3162  2.0322
     10       0.2867       0.8728       0.3932
1.9665
     29       0.3843       0.8531       0.4268  2.0297
      3       0.4229       0.8456       0.4388
2.0479
     10       0.4069       0.8516       0.4245  2.1070
     10       0.2846       0.8556       0.4404  2.0304
     10       0.4136       0.8506       0.4314  2.0050
      7       0.3234       0.8459       0.4183  1.7406
      3       0.4224       0.8627       0.3763
1.9538
     30       0.2287       0.8822       0.3316  1.9247
     30       0.2328       0.8877       0.3146  1.9763
     11       0.2767       0.8695       0.3930  1.9272
     30       0.3835       0.8525       0.4188  1.9722
      4       0.3957       0.8475       0.4366
```

```
2.0071
      8       0.3104       0.8722        0.3636  1.7026
     11       0.4050       0.8550        0.4149
2.0125
     11       0.2723       0.8731        0.3783  2.0614
     11       0.4093       0.8486        0.4244  1.9334
      4       0.3931       0.8538        0.3883  2.1417
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6068       0.8063        0.5120
1.9908
      9       0.2989       0.8775        0.3527  1.7087
     12       0.2611       0.8697        0.4011  2.1403
      5       0.3809       0.8491        0.4214
2.0003
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6436       0.7939        0.5624
1.9794
     12       0.4018       0.8536        0.4259  2.0681
     12       0.2632       0.8712        0.3744  2.0540
     12       0.4084       0.8444        0.4263  2.0888
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1       0.6364       0.8114        0.5111
2.0260
      5       0.3812       0.8592        0.4007  2.0184
     10       0.2865       0.8702        0.3607  1.7231
      2       0.4631       0.8386        0.4369
1.9251
     13       0.2587       0.8747        0.3931  1.9751
      6       0.3678       0.8148        0.5223  2.0749
      2       0.5494       0.6398        1.5004  2.0683
     13       0.3991       0.8542        0.4190  2.0907
     13       0.2571       0.8795        0.3603  2.1122
     13       0.4052       0.8595        0.4058
2.0811
      6       0.3692       0.8612        0.3902  2.0955
      2       0.5456       0.8219        0.4962
2.2281
     11       0.2783       0.8812        0.3402
1.7817
      3       0.4307       0.8452        0.4095
2.3222
     14       0.2472       0.8736        0.4033  2.1717
      7       0.3645       0.8520        0.4076
2.2097
      3       0.5515       0.7812        0.6116  2.2321
```

14	0.3974	0.8491	0.4178	2.2149
14	0.2470	0.8755	0.3754	2.1952
14	0.4036	0.8570	0.4097	2.2545
7	0.3623	0.7934	0.5448	2.1143
12	0.2638	0.8828	0.3520	1.7042
3	0.5414	0.7913	0.5604	2.1570
4	0.4016	0.8477	0.4355	2.0463
15	0.2422	0.8642	0.4178	1.9540
8	0.3562	0.8500	0.4220	1.9438
15	0.3954	0.8608	0.4018	
2.0034				
4	0.5406	0.8031	0.5545	
2.0411				
15	0.2389	0.8717	0.3846	2.0033
15	0.4005	0.8559	0.4137	1.9517
13	0.2576	0.8805	0.3644	1.7365
8	0.3546	0.8612	0.3704	2.0091
4	0.5421	0.8325	0.4746	2.0782
16	0.2323	0.8711	0.4249	1.9463
5	0.3865	0.8658	0.3811	
2.0145				
9	0.3498	0.8419	0.4627	2.1067
14	0.2471	0.8673	0.3811	1.7408
5	0.5389	0.7947	0.5256	2.0166
16	0.3944	0.8581	0.4026	2.0554
16	0.2311	0.8830	0.3592	2.0214
16	0.4002	0.8627	0.4026	
2.0185				
5	0.5367	0.8039	0.5274	1.9956
9	0.3487	0.7817	0.5987	2.0366
17	0.2236	0.8586	0.4691	2.2331
6	0.3764	0.8677	0.3749	
2.2256				
15	0.2415	0.8753	0.3784	1.7421
17	0.3927	0.8464	0.4296	2.2395
10	0.3387	0.8606	0.3946	
2.3982				
6	0.5454	0.8028	0.5433	2.3121
17	0.2230	0.8847	0.3546	
2.2731				
17	0.3986	0.8620	0.4015	2.3907
6	0.5388	0.8092	0.5184	2.2144
10	0.3390	0.8552	0.4156	2.2872
16	0.2345	0.8678	0.3816	1.7865
7	0.3641	0.8658	0.3670	2.1320
18	0.2218	0.8734	0.4044	2.1637
18	0.2172	0.8764	0.3981	2.1747
18	0.3924	0.8562	0.4150	2.2303

11	0.3411	0.8600	0.4029	2.2462
7	0.5373	0.8094	0.5333	2.2220
18	0.3967	0.8598	0.4006	2.1026
7	0.5364	0.7595	0.5903	2.1214
11	0.3387	0.8492	0.3991	2.1375
17	0.2260	0.8762	0.3847	1.7128
8	0.3586	0.8250	0.4702	2.0271
19	0.2145	0.8742	0.4186	2.1150
19	0.3891	0.8567	0.4116	1.9215
8	0.5416	0.7778	0.6103	1.9535
19	0.2104	0.8742	0.4004	2.0077
12	0.3326	0.8392	0.4906	1.9874
19	0.3958	0.8588	0.4073	1.9792
18	0.2157	0.8734	0.3740	1.6953
8	0.5387	0.7959	0.5473	1.9663
12	0.3372	0.8617	0.3832	1.9354
20	0.2061	0.8738	0.4052	1.9503
9	0.3530	0.8506	0.4054	1.9526
20	0.3893	0.8506	0.4195	1.8962
13	0.3361	0.8591	0.4111	1.9029
20	0.2038	0.8819	0.3631	2.0213
20	0.3941	0.8630	0.3996	1.9838
9	0.5351	0.8314	0.4792	1.9432
19	0.2141	0.8669	0.4352	1.7474
13	0.3325	0.8686	0.3747	1.9249
9	0.5378	0.8078	0.5397	1.9275
21	0.2045	0.8770	0.4148	2.0717
10	0.3458	0.8561	0.3963	2.0735
21	0.3885	0.8406	0.4351	2.0094
14	0.3301	0.8528	0.4212	2.0326
21	0.2036	0.8703	0.4100	2.0122
20	0.2083	0.8756	0.3890	1.8654
10	0.5405	0.8064	0.5140	2.0420
21	0.3923	0.8572	0.4093	2.0605
14	0.3300	0.8281	0.4730	2.0300
10	0.5356	0.6636	0.8248	2.1157
22	0.2008	0.8603	0.4581	2.1896
11	0.3471	0.8667	0.3729	2.1558
21	0.2044	0.8897	0.3528	1.7121
22	0.3886	0.8600	0.4091	2.1863
11	0.5411	0.7733	0.6162	2.0439
22	0.1966	0.8739	0.3968	2.1681
15	0.3266	0.8605	0.4098	2.2685
22	0.3921	0.8636	0.4022	2.1299
11	0.5403	0.7694	0.6336	2.0526
15	0.3290	0.8616	0.3801	2.2849

22	0.2003	0.8650	0.4144	1.7359
23	0.1942	0.8681	0.4559	2.1556
12	0.3376	0.8387	0.4612	2.2304
23	0.3871	0.8609	0.4062	2.1717
16	0.3279	0.8355	0.4533	2.0382
12	0.5386	0.8064	0.5205	2.0991
23	0.1906	0.8939	0.3541	2.1276
23	0.3915	0.8645	0.3938	2.1537
12	0.5319	0.7984	0.5228	2.0898
16	0.3232	0.8625	0.3884	2.0367
23	0.1959	0.8508	0.5232	1.7323
24	0.1851	0.8688	0.4528	2.0684
13	0.3360	0.8575	0.3846	2.1710
17	0.3210	0.8573	0.4055	2.0596
24	0.3855	0.8592	0.4077	2.2397
13	0.5373	0.8017	0.5396	2.1994
24	0.1857	0.8830	0.3757	2.1976
24	0.3900	0.8659	0.3936	2.2583
17	0.3203	0.8389	0.4545	2.0676
13	0.5384	0.8277	0.4764	2.1249
24	0.1877	0.8812	0.3866	1.7564
25	0.1826	0.8802	0.4203	2.0644
14	0.3302	0.8670	0.3724	2.0552
18	0.3215	0.8586	0.3907	2.1014
25	0.3847	0.8558	0.4056	2.0866
14	0.5374	0.7989	0.5450	2.0613
25	0.3909	0.8630	0.3973	1.9885
25	0.1796	0.8839	0.3765	1.7095
25	0.1768	0.8786	0.3950	2.1733
18	0.3228	0.8808	0.3365	1.9082
14	0.5370	0.8223	0.4798	1.9669
26	0.1813	0.8723	0.4616	1.9068
15	0.3318	0.8662	0.3637	1.9437
19	0.3179	0.8342	0.4592	1.8933
26	0.1793	0.8720	0.4429	1.7452
26	0.3842	0.8553	0.4086	1.9591
15	0.5357	0.7719	0.5914	2.0009
26	0.3890	0.8592	0.4034	1.9618
26	0.1765	0.8784	0.4240	1.9862
15	0.5369	0.8117	0.5088	1.9560
19	0.3155	0.8708	0.3736	2.0962
27	0.1704	0.8664	0.5046	1.9254
16	0.3282	0.8697	0.3608	1.9223
27	0.1713	0.8773	0.4038	1.7388

```
     20        0.3184       0.8542        0.4146  2.0733
     27        0.3839       0.8583        0.3995  2.0403
     16        0.5396       0.8145        0.5045  2.0943
     27        0.3881       0.8600        0.4008  2.0041
     27        0.1748       0.8722        0.4469  2.0120
     20        0.3135       0.8605        0.3852  1.9462
     16        0.5339       0.8105        0.5158  2.0690
     28        0.1723       0.8761        0.4547  1.8873
     28        0.1720       0.8830        0.3880  1.7003
     17        0.3248       0.8636        0.3804  1.9004
     21        0.3163       0.8466        0.4277  1.9577
     28        0.3831       0.8602        0.3999  1.9189
     17        0.5376       0.7605        0.6492  1.9232
     28        0.3899       0.8614        0.4025  1.9161
     28        0.1707       0.8747        0.4255  1.9852
     17        0.5374       0.8125        0.5149  1.9289
     21        0.3178       0.8572        0.4447  2.0189
     29        0.1687       0.8770        0.4227  1.7189
     29        0.1722       0.8734        0.4772  2.0246
     18        0.3271       0.8530        0.4152  1.9886
     22        0.3138       0.8442        0.4191  2.0990
     29        0.3829       0.8609        0.4010  1.9972
     29        0.3886       0.8536        0.4136  2.0593
     18        0.5334       0.8095        0.5143  2.1315
     29        0.1646       0.8708        0.4529  2.0771
     18        0.5376       0.7375        0.6796  2.0516
     22        0.3188       0.8647        0.3760  2.0891
     30        0.1643       0.8702        0.4275  1.7724
     30        0.1689       0.8748        0.4497  2.0287
     19        0.3201       0.8733        0.3494
1.9730
     30        0.3813       0.8577        0.4047  1.8955
     23        0.3184       0.8494        0.4283  1.9270
     30        0.3887       0.8594        0.4086  1.9064
     19        0.5411       0.7825        0.5931  1.8979
     30        0.1649       0.8784        0.3945  1.8467
     19        0.5361       0.8013        0.5224  1.8486
     23        0.3140       0.8569        0.4058  1.8611
  epoch    train_loss    valid_acc    valid_loss     dur
-------  ------------  -----------  ------------  ------
      1        0.6478       0.7950        0.5484
1.6580
     20        0.3199       0.8783        0.3324
1.7340
     24        0.3144       0.8609        0.3960  1.6157
     20        0.5379       0.8189        0.5028  1.5617
     20        0.5392       0.7903        0.5490  1.4239
     24        0.3129       0.8675        0.3656  1.5062
```

21	0.3205	0.8781	0.3489	1.5365
2	0.5639	0.8170	0.4852	
1.5931				
25	0.3104	0.8436	0.4556	1.3715
21	0.5380	0.7620	0.6213	1.4773
21	0.5399	0.7978	0.5338	1.5581
25	0.3098	0.8681	0.3742	1.3575
22	0.3209	0.8583	0.3960	1.3740
3	0.5541	0.8292	0.4737	
1.5850				
26	0.3126	0.8550	0.4085	1.5062
22	0.5361	0.7622	0.6267	1.3269
22	0.5353	0.8223	0.4964	1.5125
26	0.3121	0.8520	0.4056	1.5356
23	0.3186	0.8689	0.3480	1.4926
27	0.3135	0.8555	0.4226	1.3607
4	0.5438	0.7922	0.5305	1.5825
23	0.5393	0.8170	0.5125	1.4430
27	0.3119	0.8744	0.3533	1.4528
23	0.5333	0.8128	0.5091	1.5297
24	0.3173	0.8492	0.4024	1.4802
28	0.3088	0.8395	0.4512	1.4522
24	0.5402	0.7564	0.6121	1.4146
5	0.5405	0.8006	0.5510	1.5673
28	0.3129	0.8742	0.3565	1.4359
24	0.5365	0.8086	0.4997	1.4483
25	0.3181	0.8728	0.3572	1.4553
29	0.3100	0.8652	0.3878	1.4287
25	0.5329	0.8098	0.5194	1.4096
29	0.3085	0.8450	0.4227	1.3764
6	0.5419	0.8066	0.5211	1.5788
25	0.5342	0.7998	0.5398	1.4700
26	0.3151	0.8548	0.4001	1.4591
30	0.3091	0.8681	0.3764	1.5496
26	0.5457	0.7948	0.5415	1.4693
30	0.3082	0.8703	0.3787	1.3280
7	0.5473	0.8233	0.4856	1.5690
26	0.5365	0.8084	0.5148	1.5345
27	0.3123	0.8559	0.3921	1.5015
27	0.5380	0.8064	0.5373	1.5573
8	0.5457	0.8159	0.4983	1.5707
27	0.5348	0.8236	0.4860	1.5706
28	0.3178	0.8653	0.3631	1.5540
28	0.5381	0.7762	0.6063	1.4384
28	0.5309	0.7887	0.5883	1.4441
9	0.5422	0.8177	0.5007	1.5286
29	0.3134	0.8428	0.4277	1.5186
29	0.5320	0.8097	0.5383	1.5601

29	0.5444	0.7583	0.6762	1.5841
10	0.5436	0.7792	0.5750	1.5826
30	0.3122	0.8625	0.3650	1.5796
30	0.5389	0.8164	0.4972	1.5690
30	0.5399	0.8013	0.5503	1.5184
11	0.5435	0.8241	0.4852	1.5168
12	0.5425	0.8383	0.4573	1.2396
13	0.5397	0.8225	0.4904	1.2359
14	0.5412	0.8109	0.5093	1.2442
15	0.5443	0.8141	0.5088	1.2350
16	0.5414	0.8164	0.5066	1.2309
17	0.5473	0.8097	0.5089	1.2318
18	0.5397	0.7806	0.5891	1.2277
19	0.5418	0.7744	0.6019	1.2340
20	0.5420	0.8080	0.5108	1.2451
21	0.5441	0.8089	0.5420	1.2746
22	0.5488	0.7775	0.6435	1.2476
23	0.5438	0.8273	0.4894	1.2463
24	0.5486	0.8031	0.5280	1.2265
25	0.5422	0.8180	0.5155	1.2281
26	0.5425	0.8330	0.4520	1.2470
27	0.5430	0.7963	0.5545	1.2368
28	0.5452	0.7711	0.5810	1.2453
29	0.5379	0.7522	0.6820	1.2385
30	0.5415	0.8180	0.5067	1.2260

epoch	train_loss	valid_acc	valid_loss	dur
1	0.6412	0.8183	0.5092	4.8581
2	0.4574	0.8408	0.4464	4.3079
3	0.4179	0.8527	0.4154	4.1294
4	0.3937	0.8588	0.3962	5.0327
5	0.3759	0.8627	0.3885	4.6556
6	0.3609	0.8692	0.3748	5.3332
7	0.3485	0.8725	0.3650	4.1908
8	0.3385	0.8759	0.3582	5.0216
9	0.3275	0.8705	0.3609	4.7417
10	0.3194	0.8683	0.3648	5.2591
11	0.3122	0.8686	0.3652	5.7519
12	0.3042	0.8734	0.3562	5.1243
13	0.2978	0.8818	0.3383	

```
5.0761
     14        0.2923       0.8796        0.3389  5.2797
     15        0.2856       0.8818        0.3377  4.7672
     16        0.2799       0.8854        0.3226
4.9526
     17        0.2747       0.8791        0.3337  4.7476
     18        0.2704       0.8880        0.3210
4.8573
     19        0.2661       0.8893        0.3174
4.5520
     20        0.2612       0.8870        0.3193  4.3004
     21        0.2578       0.8789        0.3316  4.5095
     22        0.2532       0.8875        0.3183  4.3813
     23        0.2499       0.8854        0.3235  4.8678
     24        0.2459       0.8783        0.3394  4.5546
     25        0.2425       0.8900        0.3169
5.3659
     26        0.2402       0.8830        0.3383  5.3259
     27        0.2361       0.8854        0.3282  5.2189
     28        0.2335       0.8888        0.3182  6.6584
     29        0.2306       0.8894        0.3112  6.2885
     30        0.2276       0.8866        0.3192  6.3759
```

[144]:
```
# summarize results
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

```
Best: 0.881729 using {'module__n_hidden': 160, 'optimizer__lr': 0.01,
'optimizer__weight_decay': 0.001}
```

[149]:
```
# print(grid_result.cv_results_)
```

[160]:
```
def train_valid_wrapper(batchsize = 32, n_hidden = 48, lr = 1e-2, reg_val =␣
 ↪1e-4, device = "cpu"):
    # lr: the learning rate in TF is part of the optimizer.  Default is 1e-2

    # dataloader
    trainloader = torch.utils.data.DataLoader(trainset, batch_size=batchsize,␣
 ↪shuffle=True)
    valloader = torch.utils.data.DataLoader(valset, batch_size=batchsize,␣
 ↪shuffle=False)
    # model
    model = MLP(n_hidden)
    # loss function and optimier
    criterion = nn.CrossEntropyLoss() # includes softmax (for numerical␣
 ↪stability)
    optimizer = optim.SGD(model.parameters(), lr=lr, weight_decay=reg_val)
```

```
    # Run training and validation loop
    # Save the best model based on validation accuracy
    n_epochs = 30
    best_acc = -1
    train_loss_history = []; train_acc_history = []
    val_loss_history = []; val_acc_history = []
    for epoch in tqdm(range(n_epochs), unit="epoch"): # Iterate over epochs
        # print(f"Epoch {epoch+1} of {n_epochs}")
        train_loss, train_acc  = train(model, trainloader, criterion,␣
↪optimizer, device) # Train
        val_loss, val_acc = validate(model, valloader, criterion, device) #␣
↪Validate
        train_loss_history.append(train_loss); train_acc_history.
↪append(train_acc)
        val_loss_history.append(val_loss); val_acc_history.append(val_acc)
        if val_acc > best_acc: # Save best model
            best_acc = val_acc
            torch.save(model, "best_model.pt")
            # torch.save(model.state_dict(), "best_model.pt") # saving model␣
↪parameters ("state_dict") saves memory and is faster than saving the entire␣
↪model
    return train_loss_history, train_acc_history, val_loss_history,␣
↪val_acc_history
```

[181]:
```
def plot_learningcurve(hidden):
    batchsize = 64


    # plot train and val learning curve
    hidden_nodes = [40, 80, 160]
    lr_list = [0.001, 0.01, 0.1]
    reg_list = [1e-04, 1e-03, 1e-02]

#     train_loss_history = []
#     train_acc_history = []
#     val_loss_history = []
#     val_acc_history = []

#     for hidden in hidden_nodes:
    print(f"Hidden nodes: {hidden}")
    plt.title(f"Loss and Accuracy (Hidden Nodes: {hidden}")

    for lr in lr_list:
        print(f"lr : {lr}")
        for reg_val in reg_list:
            print(f"reg_val : {reg_val}")
            # Create a model
```

```
            model = MLP(hidden)
#                 lr = params['optimizer__lr']
#                 reg_val = params['optimizer__weight_decay']
            criterion = nn.CrossEntropyLoss() # includes softmax (for numerical␣
↪stability)
            optimizer = optim.SGD(model.parameters(), lr=lr,␣
↪weight_decay=reg_val)
            device = torch.device("cpu")
            model.to(device) # Move model to device

            train_loss_history = []
            train_acc_history = []
            val_loss_history = []
            val_acc_history = []

            trainloader = torch.utils.data.DataLoader(trainset,␣
↪batch_size=batchsize, shuffle=True)
            valloader = torch.utils.data.DataLoader(valset,␣
↪batch_size=batchsize, shuffle=False)

            epoch = 0
            for epoch in range(30):
                train_loss, train_acc  = train(model, trainloader, criterion,␣
↪optimizer, device) # Train
                val_loss, val_acc = validate(model, valloader, criterion,␣
↪device) # Validate

                train_loss_history.append(train_loss)
                train_acc_history.append(train_acc)
                val_loss_history.append(val_loss)
                val_acc_history.append(val_acc)

#                 train_loss_history, train_acc_history, val_loss_history,␣
↪val_acc_history = train_valid_wrapper(batchsize = 32, n_hidden = hidden, lr␣
↪= lr, reg_val = reg_val, device = "cpu")

            plt.plot(train_loss_history, label=f'Train loss:lr={lr},␣
↪reg={reg_val}')
            plt.plot(val_loss_history, label=f'Val loss:lr={lr}, reg={reg_val}')
            plt.plot(train_acc_history, label=f'Train acc:lr={lr},␣
↪reg={reg_val}')
            plt.plot(val_acc_history, label=f'Val acc:lr={lr}, reg={reg_val}')
            plt.xlabel('Epoch')
            plt.ylabel('Loss / Accuracy')
            plt.legend()
```

```
    plt.show()
```

[182]:
```
plot_learningcurve(40)
```

```
Hidden nodes: 40
lr : 0.001
reg_val : 0.0001
```

```
Train: 100%|          | 750/750 [00:07<00:00, 106.12batch/s, loss=1.47, acc=49.2]
Eval: 100%|           | 188/188 [00:01<00:00, 126.19batch/s, loss=1.37, acc=66.1]
Train: 100%|          | 750/750 [00:07<00:00, 100.45batch/s, loss=1.08, acc=69.2]
Eval: 100%|           | 188/188 [00:01<00:00, 119.27batch/s, loss=0.993, acc=70.6]
Train: 100%|          | 750/750 [00:06<00:00, 110.24batch/s, loss=0.88, acc=72.3]
Eval: 100%|           | 188/188 [00:01<00:00, 118.55batch/s, loss=0.878, acc=72.8]
Train: 100%|          | 750/750 [00:07<00:00, 104.67batch/s, loss=0.754, acc=74]
Eval: 100%|           | 188/188 [00:01<00:00, 119.05batch/s, loss=0.83, acc=74.2]
Train: 100%|          | 750/750 [00:07<00:00, 103.58batch/s, loss=0.68, acc=75.3]
Eval: 100%|           | 188/188 [00:01<00:00, 120.18batch/s, loss=0.795, acc=75.3]
Train: 100%|          | 750/750 [00:07<00:00, 106.58batch/s, loss=0.756, acc=76.4]
Eval: 100%|           | 188/188 [00:01<00:00, 120.19batch/s, loss=0.783, acc=76.3]
Train: 100%|          | 750/750 [00:06<00:00, 108.50batch/s, loss=0.609, acc=77.3]
Eval: 100%|           | 188/188 [00:01<00:00, 121.56batch/s, loss=0.752, acc=77]
Train: 100%|          | 750/750 [00:06<00:00, 108.97batch/s, loss=0.674, acc=78]
Eval: 100%|           | 188/188 [00:01<00:00, 122.89batch/s, loss=0.73, acc=77.7]
Train: 100%|          | 750/750 [00:07<00:00, 105.72batch/s, loss=0.573, acc=78.8]
Eval: 100%|           | 188/188 [00:01<00:00, 120.07batch/s, loss=0.711, acc=78.3]
Train: 100%|          | 750/750 [00:06<00:00, 109.66batch/s, loss=0.733, acc=79.3]
Eval: 100%|           | 188/188 [00:01<00:00, 119.17batch/s, loss=0.694, acc=78.9]
Train: 100%|          | 750/750 [00:06<00:00, 107.75batch/s, loss=0.606, acc=79.8]
Eval: 100%|           | 188/188 [00:01<00:00, 119.73batch/s, loss=0.685, acc=79.4]
Train: 100%|          | 750/750 [00:06<00:00, 108.89batch/s, loss=0.639, acc=80.2]
Eval: 100%|           | 188/188 [00:01<00:00, 120.26batch/s, loss=0.672, acc=79.7]
Train: 100%|          | 750/750 [00:07<00:00, 105.67batch/s, loss=0.525, acc=80.6]
Eval: 100%|           | 188/188 [00:01<00:00, 119.02batch/s, loss=0.657, acc=80.2]
Train: 100%|          | 750/750 [00:06<00:00, 108.68batch/s, loss=0.418, acc=80.9]
Eval: 100%|           | 188/188 [00:01<00:00, 119.73batch/s, loss=0.64, acc=80.3]
Train: 100%|          | 750/750 [00:07<00:00, 104.13batch/s, loss=0.416, acc=81.2]
Eval: 100%|           | 188/188 [00:01<00:00, 119.76batch/s, loss=0.639, acc=80.7]
Train: 100%|          | 750/750 [00:07<00:00, 105.23batch/s, loss=0.546, acc=81.6]
Eval: 100%|           | 188/188 [00:01<00:00, 117.45batch/s, loss=0.647, acc=80.8]
Train: 100%|          | 750/750 [00:07<00:00, 106.00batch/s, loss=0.528, acc=81.8]
Eval: 100%|           | 188/188 [00:01<00:00, 120.04batch/s, loss=0.624, acc=81.2]
Train: 100%|          | 750/750 [00:07<00:00, 105.03batch/s, loss=0.631, acc=82]
Eval: 100%|           | 188/188 [00:01<00:00, 117.24batch/s, loss=0.628, acc=81.4]
Train: 100%|          | 750/750 [00:07<00:00, 106.71batch/s, loss=0.442, acc=82.2]
Eval: 100%|           | 188/188 [00:01<00:00, 120.68batch/s, loss=0.602, acc=81.5]
Train: 100%|          | 750/750 [00:07<00:00, 104.85batch/s, loss=0.464, acc=82.4]
```

```
Eval: 100%|      | 188/188 [00:01<00:00, 120.16batch/s, loss=0.601, acc=81.7]
Train: 100%|     | 750/750 [00:07<00:00, 106.73batch/s, loss=0.437, acc=82.6]
Eval: 100%|      | 188/188 [00:01<00:00, 119.13batch/s, loss=0.603, acc=81.7]
Train: 100%|     | 750/750 [00:07<00:00, 104.72batch/s, loss=0.428, acc=82.8]
Eval: 100%|       | 188/188 [00:01<00:00, 119.14batch/s, loss=0.598, acc=82]
Train: 100%|     | 750/750 [00:07<00:00, 103.43batch/s, loss=0.445, acc=82.9]
Eval: 100%|      | 188/188 [00:01<00:00, 118.52batch/s, loss=0.592, acc=82.2]
Train: 100%|       | 750/750 [00:07<00:00, 104.17batch/s, loss=0.46, acc=83]
Eval: 100%|      | 188/188 [00:01<00:00, 120.22batch/s, loss=0.573, acc=82.3]
Train: 100%|     | 750/750 [00:07<00:00, 102.92batch/s, loss=0.538, acc=83.1]
Eval: 100%|      | 188/188 [00:01<00:00, 119.11batch/s, loss=0.577, acc=82.5]
Train: 100%|     | 750/750 [00:07<00:00, 106.23batch/s, loss=0.443, acc=83.2]
Eval: 100%|      | 188/188 [00:01<00:00, 120.02batch/s, loss=0.558, acc=82.6]
Train: 100%|     | 750/750 [00:06<00:00, 107.42batch/s, loss=0.514, acc=83.4]
Eval: 100%|       | 188/188 [00:01<00:00, 119.68batch/s, loss=0.56, acc=82.7]
Train: 100%|     | 750/750 [00:07<00:00, 105.11batch/s, loss=0.588, acc=83.4]
Eval: 100%|      | 188/188 [00:01<00:00, 119.61batch/s, loss=0.543, acc=82.6]
Train: 100%|      | 750/750 [00:07<00:00, 97.92batch/s, loss=0.513, acc=83.5]
Eval: 100%|      | 188/188 [00:01<00:00, 115.77batch/s, loss=0.546, acc=82.9]
Train: 100%|     | 750/750 [00:07<00:00, 106.29batch/s, loss=0.605, acc=83.6]
Eval: 100%|       | 188/188 [00:01<00:00, 121.23batch/s, loss=0.552, acc=83]
```

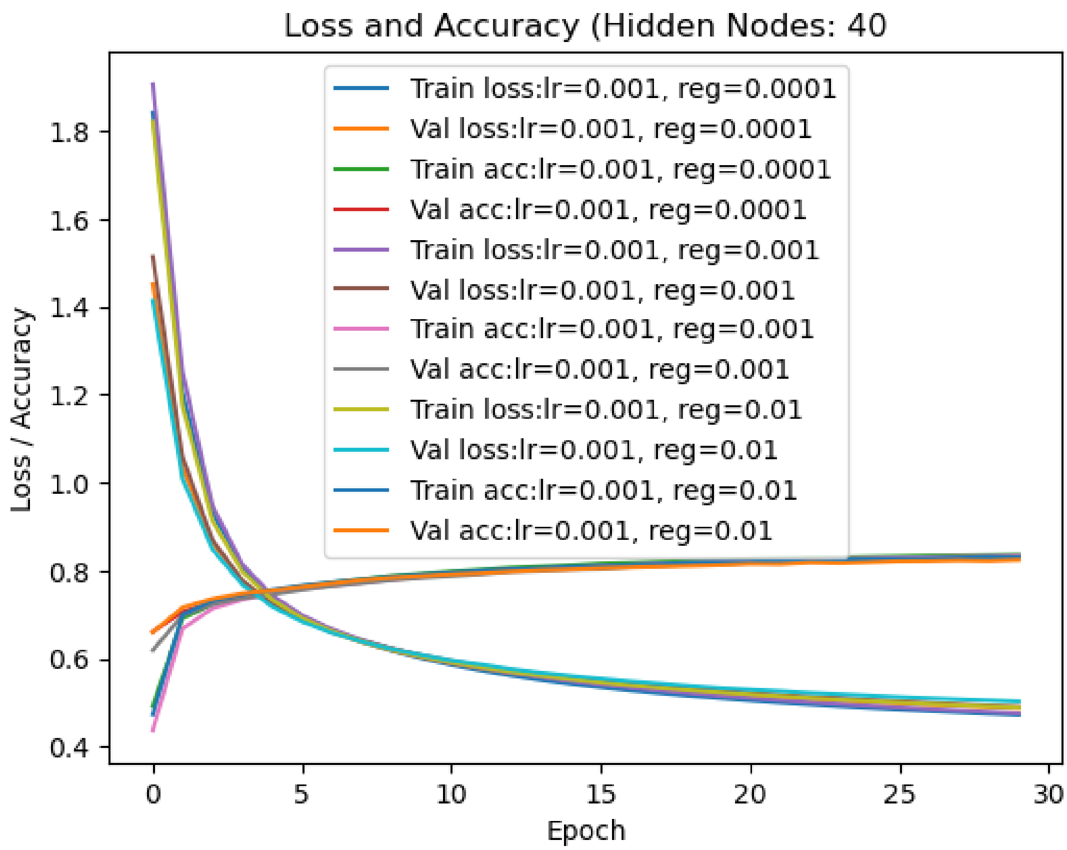
reg_val : 0.001

```
Train: 100%|      | 750/750 [00:06<00:00, 107.40batch/s, loss=1.5, acc=43.7]
Eval: 100%|       | 188/188 [00:01<00:00, 124.68batch/s, loss=1.41, acc=61.9]
Train: 100%|      | 750/750 [00:07<00:00, 106.81batch/s, loss=1.07, acc=66.8]
Eval: 100%|      | 188/188 [00:01<00:00, 125.25batch/s, loss=0.992, acc=69.8]
Train: 100%|     | 750/750 [00:06<00:00, 107.32batch/s, loss=0.954, acc=71.4]
Eval: 100%|       | 188/188 [00:01<00:00, 123.39batch/s, loss=0.85, acc=72.4]
Train: 100%|      | 750/750 [00:07<00:00, 97.93batch/s, loss=0.666, acc=73.4]
Eval: 100%|      | 188/188 [00:01<00:00, 126.55batch/s, loss=0.793, acc=73.7]
Train: 100%|     | 750/750 [00:06<00:00, 109.99batch/s, loss=0.794, acc=74.8]
Eval: 100%|      | 188/188 [00:01<00:00, 126.22batch/s, loss=0.762, acc=74.6]
Train: 100%|     | 750/750 [00:06<00:00, 110.33batch/s, loss=0.563, acc=75.8]
Eval: 100%|      | 188/188 [00:01<00:00, 117.02batch/s, loss=0.749, acc=75.6]
Train: 100%|     | 750/750 [00:06<00:00, 110.87batch/s, loss=0.642, acc=76.6]
Eval: 100%|      | 188/188 [00:01<00:00, 123.82batch/s, loss=0.731, acc=76.5]
Train: 100%|     | 750/750 [00:06<00:00, 110.83batch/s, loss=0.709, acc=77.3]
Eval: 100%|      | 188/188 [00:01<00:00, 125.24batch/s, loss=0.712, acc=77.2]
Train: 100%|      | 750/750 [00:06<00:00, 109.16batch/s, loss=0.605, acc=78]
Eval: 100%|      | 188/188 [00:01<00:00, 120.22batch/s, loss=0.696, acc=77.9]
Train: 100%|     | 750/750 [00:06<00:00, 112.67batch/s, loss=0.626, acc=78.6]
Eval: 100%|      | 188/188 [00:01<00:00, 120.08batch/s, loss=0.692, acc=78.3]
Train: 100%|     | 750/750 [00:06<00:00, 109.41batch/s, loss=0.736, acc=79.1]
Eval: 100%|      | 188/188 [00:01<00:00, 120.13batch/s, loss=0.677, acc=78.7]
Train: 100%|     | 750/750 [00:06<00:00, 111.74batch/s, loss=0.484, acc=79.6]
Eval: 100%|      | 188/188 [00:01<00:00, 119.36batch/s, loss=0.657, acc=79.2]
Train: 100%|       | 750/750 [00:06<00:00, 110.44batch/s, loss=0.53, acc=80]
```

```
Eval: 100%|     | 188/188 [00:01<00:00, 119.99batch/s, loss=0.655, acc=79.8]
Train: 100%|    | 750/750 [00:07<00:00, 104.81batch/s, loss=0.676, acc=80.3]
Eval: 100%|     | 188/188 [00:01<00:00, 113.39batch/s, loss=0.641, acc=79.9]
Train: 100%|    | 750/750 [00:07<00:00, 101.87batch/s, loss=0.585, acc=80.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.66batch/s, loss=0.619, acc=80.3]
Train: 100%|     | 750/750 [00:06<00:00, 107.95batch/s, loss=0.485, acc=81]
Eval: 100%|     | 188/188 [00:01<00:00, 120.70batch/s, loss=0.607, acc=80.4]
Train: 100%|    | 750/750 [00:06<00:00, 110.39batch/s, loss=0.539, acc=81.3]
Eval: 100%|     | 188/188 [00:01<00:00, 119.46batch/s, loss=0.617, acc=80.8]
Train: 100%|    | 750/750 [00:07<00:00, 104.94batch/s, loss=0.603, acc=81.6]
Eval: 100%|      | 188/188 [00:01<00:00, 119.51batch/s, loss=0.602, acc=81]
Train: 100%|    | 750/750 [00:06<00:00, 107.82batch/s, loss=0.567, acc=81.8]
Eval: 100%|     | 188/188 [00:01<00:00, 119.82batch/s, loss=0.611, acc=81.4]
Train: 100%|     | 750/750 [00:07<00:00, 106.60batch/s, loss=0.781, acc=82]
Eval: 100%|     | 188/188 [00:01<00:00, 120.21batch/s, loss=0.597, acc=81.4]
Train: 100%|    | 750/750 [00:07<00:00, 106.91batch/s, loss=0.453, acc=82.2]
Eval: 100%|     | 188/188 [00:01<00:00, 120.60batch/s, loss=0.596, acc=81.8]
Train: 100%|    | 750/750 [00:07<00:00, 107.06batch/s, loss=0.405, acc=82.3]
Eval: 100%|     | 188/188 [00:01<00:00, 119.40batch/s, loss=0.592, acc=81.9]
Train: 100%|    | 750/750 [00:06<00:00, 107.20batch/s, loss=0.432, acc=82.5]
Eval: 100%|     | 188/188 [00:01<00:00, 119.81batch/s, loss=0.583, acc=82.1]
Train: 100%|    | 750/750 [00:06<00:00, 107.28batch/s, loss=0.632, acc=82.7]
Eval: 100%|     | 188/188 [00:01<00:00, 119.90batch/s, loss=0.578, acc=82.1]
Train: 100%|     | 750/750 [00:06<00:00, 107.53batch/s, loss=0.59, acc=82.8]
Eval: 100%|     | 188/188 [00:01<00:00, 120.21batch/s, loss=0.587, acc=82.3]
Train: 100%|    | 750/750 [00:06<00:00, 107.45batch/s, loss=0.608, acc=82.9]
Eval: 100%|     | 188/188 [00:01<00:00, 120.13batch/s, loss=0.581, acc=82.4]
Train: 100%|      | 750/750 [00:09<00:00, 79.71batch/s, loss=0.547, acc=83]
Eval: 100%|      | 188/188 [00:01<00:00, 119.42batch/s, loss=0.57, acc=82.5]
Train: 100%|    | 750/750 [00:06<00:00, 110.67batch/s, loss=0.719, acc=83.2]
Eval: 100%|     | 188/188 [00:01<00:00, 120.18batch/s, loss=0.569, acc=82.7]
Train: 100%|    | 750/750 [00:06<00:00, 107.64batch/s, loss=0.587, acc=83.2]
Eval: 100%|     | 188/188 [00:01<00:00, 120.70batch/s, loss=0.563, acc=82.8]
Train: 100%|    | 750/750 [00:06<00:00, 108.13batch/s, loss=0.513, acc=83.3]
Eval: 100%|      | 188/188 [00:01<00:00, 120.93batch/s, loss=0.541, acc=83]

reg_val : 0.01

Train: 100%|     | 750/750 [00:07<00:00, 106.74batch/s, loss=1.25, acc=47.3]
Eval: 100%|      | 188/188 [00:01<00:00, 120.79batch/s, loss=1.42, acc=66]
Train: 100%|      | 750/750 [00:06<00:00, 110.16batch/s, loss=1.04, acc=70]
Eval: 100%|     | 188/188 [00:01<00:00, 118.96batch/s, loss=1.04, acc=71.6]
Train: 100%|    | 750/750 [00:06<00:00, 107.28batch/s, loss=0.724, acc=73.1]
Eval: 100%|     | 188/188 [00:01<00:00, 120.19batch/s, loss=0.908, acc=73.6]
Train: 100%|    | 750/750 [00:07<00:00, 102.51batch/s, loss=0.759, acc=74.5]
Eval: 100%|     | 188/188 [00:01<00:00, 120.14batch/s, loss=0.842, acc=74.8]
Train: 100%|    | 750/750 [00:06<00:00, 110.42batch/s, loss=0.598, acc=75.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.23batch/s, loss=0.808, acc=75.5]
Train: 100%|    | 750/750 [00:06<00:00, 112.37batch/s, loss=0.686, acc=76.6]
```

```
Eval: 100%|     | 188/188 [00:01<00:00, 115.97batch/s, loss=0.775, acc=76.3]
Train: 100%|     | 750/750 [00:06<00:00, 108.51batch/s, loss=0.534, acc=77.3]
Eval: 100%|     | 188/188 [00:01<00:00, 120.10batch/s, loss=0.754, acc=77.1]
Train: 100%|      | 750/750 [00:06<00:00, 111.35batch/s, loss=0.635, acc=78]
Eval: 100%|      | 188/188 [00:01<00:00, 120.34batch/s, loss=0.74, acc=77.9]
Train: 100%|     | 750/750 [00:06<00:00, 107.95batch/s, loss=0.639, acc=78.6]
Eval: 100%|      | 188/188 [00:01<00:00, 122.36batch/s, loss=0.73, acc=78.4]
Train: 100%|     | 750/750 [00:06<00:00, 109.31batch/s, loss=0.564, acc=79.1]
Eval: 100%|     | 188/188 [00:01<00:00, 120.46batch/s, loss=0.705, acc=78.8]
Train: 100%|      | 750/750 [00:08<00:00, 85.80batch/s, loss=0.486, acc=79.6]
Eval: 100%|     | 188/188 [00:01<00:00, 109.79batch/s, loss=0.693, acc=79.2]
Train: 100%|       | 750/750 [00:08<00:00, 83.91batch/s, loss=0.581, acc=80]
Eval: 100%|     | 188/188 [00:01<00:00, 111.40batch/s, loss=0.691, acc=79.4]
Train: 100%|      | 750/750 [00:08<00:00, 84.68batch/s, loss=0.698, acc=80.3]
Eval: 100%|     | 188/188 [00:01<00:00, 101.73batch/s, loss=0.665, acc=79.8]
Train: 100%|      | 750/750 [00:09<00:00, 80.37batch/s, loss=0.493, acc=80.6]
Eval: 100%|      | 188/188 [00:01<00:00, 102.64batch/s, loss=0.65, acc=80.1]
Train: 100%|      | 750/750 [00:09<00:00, 81.58batch/s, loss=0.592, acc=80.9]
Eval: 100%|     | 188/188 [00:01<00:00, 113.70batch/s, loss=0.646, acc=80.3]
Train: 100%|      | 750/750 [00:09<00:00, 75.37batch/s, loss=0.512, acc=81.1]
Eval: 100%|     | 188/188 [00:01<00:00, 112.97batch/s, loss=0.638, acc=80.5]
Train: 100%|      | 750/750 [00:08<00:00, 87.89batch/s, loss=0.482, acc=81.3]
Eval: 100%|     | 188/188 [00:01<00:00, 112.82batch/s, loss=0.627, acc=80.7]
Train: 100%|      | 750/750 [00:10<00:00, 72.00batch/s, loss=0.635, acc=81.6]
Eval: 100%|     | 188/188 [00:01<00:00, 120.61batch/s, loss=0.624, acc=80.9]
Train: 100%|     | 750/750 [00:07<00:00, 106.61batch/s, loss=0.613, acc=81.8]
Eval: 100%|      | 188/188 [00:01<00:00, 112.53batch/s, loss=0.61, acc=81.1]
Train: 100%|      | 750/750 [00:07<00:00, 106.58batch/s, loss=0.359, acc=82]
Eval: 100%|     | 188/188 [00:01<00:00, 119.85batch/s, loss=0.609, acc=81.3]
Train: 100%|      | 750/750 [00:06<00:00, 107.17batch/s, loss=0.526, acc=82]
Eval: 100%|     | 188/188 [00:01<00:00, 112.84batch/s, loss=0.606, acc=81.5]
Train: 100%|     | 750/750 [00:06<00:00, 107.61batch/s, loss=0.447, acc=82.3]
Eval: 100%|     | 188/188 [00:01<00:00, 118.85batch/s, loss=0.594, acc=81.4]
Train: 100%|      | 750/750 [00:07<00:00, 99.48batch/s, loss=0.395, acc=82.4]
Eval: 100%|     | 188/188 [00:01<00:00, 114.14batch/s, loss=0.589, acc=81.8]
Train: 100%|     | 750/750 [00:07<00:00, 106.09batch/s, loss=0.429, acc=82.5]
Eval: 100%|     | 188/188 [00:01<00:00, 119.93batch/s, loss=0.588, acc=81.7]
Train: 100%|     | 750/750 [00:07<00:00, 106.71batch/s, loss=0.473, acc=82.6]
Eval: 100%|     | 188/188 [00:01<00:00, 120.31batch/s, loss=0.579, acc=81.9]
Train: 100%|     | 750/750 [00:07<00:00, 106.95batch/s, loss=0.455, acc=82.7]
Eval: 100%|      | 188/188 [00:01<00:00, 119.85batch/s, loss=0.57, acc=82.1]
Train: 100%|     | 750/750 [00:07<00:00, 106.45batch/s, loss=0.748, acc=82.8]
Eval: 100%|     | 188/188 [00:01<00:00, 119.40batch/s, loss=0.581, acc=82.2]
Train: 100%|     | 750/750 [00:06<00:00, 109.32batch/s, loss=0.423, acc=82.9]
Eval: 100%|     | 188/188 [00:01<00:00, 120.35batch/s, loss=0.563, acc=82.3]
Train: 100%|     | 750/750 [00:07<00:00, 106.76batch/s, loss=0.362, acc=83.1]
Eval: 100%|     | 188/188 [00:01<00:00, 120.24batch/s, loss=0.572, acc=82.2]
Train: 100%|     | 750/750 [00:06<00:00, 110.01batch/s, loss=0.621, acc=83.1]
```

```
Eval: 100%|      | 188/188 [00:01<00:00, 104.41batch/s, loss=0.56, acc=82.4]
```




```
lr : 0.01
reg_val : 0.0001
```

```
Train: 100%|     | 750/750 [00:08<00:00, 89.98batch/s, loss=0.426, acc=71.9]
Eval: 100%|     | 188/188 [00:01<00:00, 124.14batch/s, loss=0.757, acc=77.9]
Train: 100%|     | 750/750 [00:06<00:00, 113.10batch/s, loss=0.479, acc=80.5]
Eval: 100%|     | 188/188 [00:01<00:00, 126.32batch/s, loss=0.581, acc=81.3]
Train: 100%|     | 750/750 [00:06<00:00, 116.64batch/s, loss=0.543, acc=82.6]
Eval: 100%|     | 188/188 [00:01<00:00, 130.26batch/s, loss=0.556, acc=82.8]
Train: 100%|     | 750/750 [00:06<00:00, 113.97batch/s, loss=0.321, acc=83.5]
Eval: 100%|     | 188/188 [00:01<00:00, 130.06batch/s, loss=0.529, acc=83.2]
Train: 100%|     | 750/750 [00:06<00:00, 122.01batch/s, loss=0.537, acc=84.2]
Eval: 100%|     | 188/188 [00:01<00:00, 140.95batch/s, loss=0.518, acc=83.7]
Train: 100%|     | 750/750 [00:06<00:00, 123.11batch/s, loss=0.453, acc=84.7]
Eval: 100%|     | 188/188 [00:01<00:00, 131.46batch/s, loss=0.465, acc=84.3]
Train: 100%|      | 750/750 [15:50<00:00,  1.27s/batch, loss=0.318, acc=85]
Eval: 100%|      | 188/188 [00:02<00:00, 67.12batch/s, loss=0.457, acc=84.7]
Train: 100%|      | 750/750 [00:11<00:00, 67.52batch/s, loss=0.429, acc=85.5]
Eval: 100%|      | 188/188 [00:02<00:00, 76.33batch/s, loss=0.47, acc=84.8]
```

```
Train: 100%|     | 750/750 [00:10<00:00, 72.22batch/s, loss=0.355, acc=85.6]
Eval: 100%|      | 188/188 [00:02<00:00, 80.82batch/s, loss=0.426, acc=85.2]
Train: 100%|      | 750/750 [00:13<00:00, 56.76batch/s, loss=0.373, acc=86]
Eval: 100%|      | 188/188 [00:02<00:00, 74.40batch/s, loss=0.508, acc=84.8]
Train: 100%|      | 750/750 [00:11<00:00, 62.55batch/s, loss=0.368, acc=86]
Eval: 100%|      | 188/188 [00:02<00:00, 73.16batch/s, loss=0.476, acc=85.3]
Train: 100%|     | 750/750 [00:10<00:00, 68.88batch/s, loss=0.446, acc=86.4]
Eval: 100%|      | 188/188 [10:46<00:00,  3.44s/batch, loss=0.498, acc=85.4]
Train: 100%|      | 750/750 [00:09<00:00, 77.31batch/s, loss=0.43, acc=86.6]
Eval: 100%|       | 188/188 [00:01<00:00, 97.68batch/s, loss=0.49, acc=85.6]
Train: 100%|     | 750/750 [00:07<00:00, 100.66batch/s, loss=0.544, acc=86.7]
Eval: 100%|      | 188/188 [00:01<00:00, 120.12batch/s, loss=0.48, acc=85.8]
Train: 100%|     | 750/750 [00:07<00:00, 106.50batch/s, loss=0.333, acc=86.8]
Eval: 100%|     | 188/188 [00:01<00:00, 119.64batch/s, loss=0.496, acc=85.9]
Train: 100%|      | 750/750 [00:07<00:00, 105.04batch/s, loss=0.318, acc=87]
Eval: 100%|     | 188/188 [00:01<00:00, 118.67batch/s, loss=0.452, acc=86.2]
Train: 100%|     | 750/750 [00:06<00:00, 107.74batch/s, loss=0.393, acc=87.2]
Eval: 100%|     | 188/188 [00:01<00:00, 119.00batch/s, loss=0.486, acc=85.9]
Train: 100%|     | 750/750 [00:06<00:00, 110.13batch/s, loss=0.235, acc=87.3]
Eval: 100%|     | 188/188 [00:01<00:00, 124.88batch/s, loss=0.427, acc=86.1]
Train: 100%|     | 750/750 [00:07<00:00, 105.36batch/s, loss=0.267, acc=87.4]
Eval: 100%|     | 188/188 [00:01<00:00, 120.19batch/s, loss=0.446, acc=86.3]
Train: 100%|     | 750/750 [00:06<00:00, 108.64batch/s, loss=0.399, acc=87.6]
Eval: 100%|     | 188/188 [00:01<00:00, 119.37batch/s, loss=0.456, acc=86.3]
Train: 100%|     | 750/750 [00:07<00:00, 105.73batch/s, loss=0.248, acc=87.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.53batch/s, loss=0.448, acc=86.5]
Train: 100%|     | 750/750 [00:07<00:00, 106.16batch/s, loss=0.421, acc=87.9]
Eval: 100%|     | 188/188 [00:01<00:00, 120.48batch/s, loss=0.447, acc=86.4]
Train: 100%|     | 750/750 [00:07<00:00, 106.64batch/s, loss=0.244, acc=87.9]
Eval: 100%|     | 188/188 [00:01<00:00, 121.36batch/s, loss=0.448, acc=86.6]
Train: 100%|     | 750/750 [00:07<00:00, 103.28batch/s, loss=0.304, acc=88.1]
Eval: 100%|     | 188/188 [00:01<00:00, 122.18batch/s, loss=0.424, acc=86.8]
Train: 100%|     | 750/750 [00:07<00:00, 104.86batch/s, loss=0.268, acc=88.1]
Eval: 100%|     | 188/188 [00:01<00:00, 121.74batch/s, loss=0.476, acc=86.8]
Train: 100%|     | 750/750 [00:07<00:00, 103.36batch/s, loss=0.289, acc=88.3]
Eval: 100%|       | 188/188 [00:01<00:00, 120.53batch/s, loss=0.442, acc=87]
Train: 100%|     | 750/750 [00:07<00:00, 101.81batch/s, loss=0.408, acc=88.4]
Eval: 100%|     | 188/188 [00:01<00:00, 119.10batch/s, loss=0.514, acc=86.1]
Train: 100%|     | 750/750 [00:07<00:00, 106.78batch/s, loss=0.272, acc=88.4]
Eval: 100%|     | 188/188 [00:01<00:00, 119.83batch/s, loss=0.474, acc=86.9]
Train: 100%|     | 750/750 [00:06<00:00, 107.58batch/s, loss=0.461, acc=88.5]
Eval: 100%|       | 188/188 [00:01<00:00, 119.46batch/s, loss=0.5, acc=86.6]
Train: 100%|     | 750/750 [00:06<00:00, 108.17batch/s, loss=0.292, acc=88.5]
Eval: 100%|       | 188/188 [00:01<00:00, 119.61batch/s, loss=0.467, acc=87]
```

```
reg_val : 0.001
```

```
Train: 100%|     | 750/750 [00:07<00:00, 102.63batch/s, loss=0.486, acc=70.8]
Eval: 100%|       | 188/188 [00:01<00:00, 118.26batch/s, loss=0.709, acc=79]
```

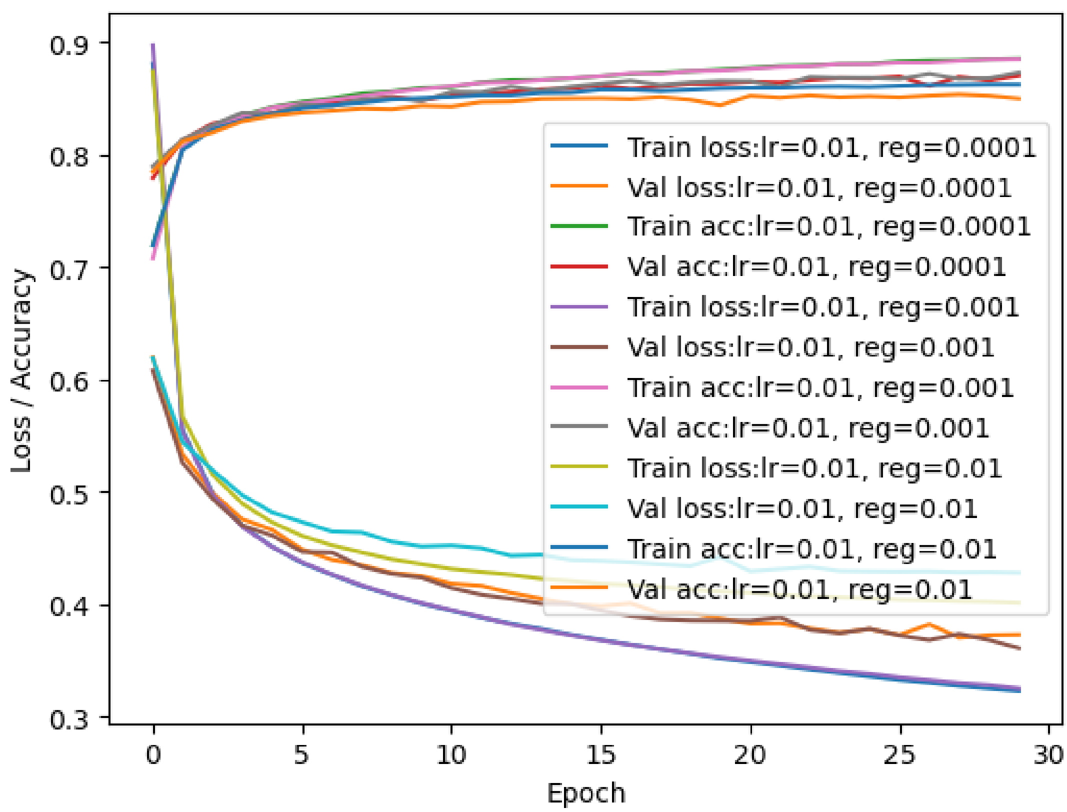
```
Train: 100%|     | 750/750 [00:07<00:00, 101.74batch/s, loss=0.407, acc=80.6]
Eval: 100%|      | 188/188 [00:01<00:00, 119.25batch/s, loss=0.594, acc=81.4]
Train: 100%|     | 750/750 [00:07<00:00, 103.65batch/s, loss=0.504, acc=82.5]
Eval: 100%|      | 188/188 [00:01<00:00, 116.99batch/s, loss=0.505, acc=82.6]
Train: 100%|     | 750/750 [00:07<00:00, 102.92batch/s, loss=0.486, acc=83.4]
Eval: 100%|      | 188/188 [00:01<00:00, 120.24batch/s, loss=0.535, acc=83.7]
Train: 100%|     | 750/750 [00:06<00:00, 108.34batch/s, loss=0.425, acc=84.2]
Eval: 100%|      | 188/188 [00:01<00:00, 118.49batch/s, loss=0.547, acc=83.6]
Train: 100%|     | 750/750 [00:07<00:00, 100.98batch/s, loss=0.407, acc=84.6]
Eval: 100%|      | 188/188 [00:01<00:00, 118.97batch/s, loss=0.491, acc=84.5]
Train: 100%|     | 750/750 [00:07<00:00, 105.65batch/s, loss=0.415, acc=84.9]
Eval: 100%|      | 188/188 [00:01<00:00, 117.52batch/s, loss=0.517, acc=84.3]
Train: 100%|     | 750/750 [00:06<00:00, 110.60batch/s, loss=0.373, acc=85.2]
Eval: 100%|      | 188/188 [00:01<00:00, 118.67batch/s, loss=0.421, acc=84.9]
Train: 100%|     | 750/750 [00:06<00:00, 113.43batch/s, loss=0.476, acc=85.6]
Eval: 100%|      | 188/188 [00:01<00:00, 120.14batch/s, loss=0.441, acc=85.1]
Train: 100%|     | 750/750 [00:06<00:00, 110.42batch/s, loss=0.656, acc=85.8]
Eval: 100%|      | 188/188 [00:01<00:00, 120.03batch/s, loss=0.519, acc=84.8]
Train: 100%|     | 750/750 [00:07<00:00, 104.85batch/s, loss=0.393, acc=86.1]
Eval: 100%|      | 188/188 [00:01<00:00, 119.40batch/s, loss=0.473, acc=85.6]
Train: 100%|      | 750/750 [00:07<00:00, 106.57batch/s, loss=0.36, acc=86.4]
Eval: 100%|      | 188/188 [00:01<00:00, 118.36batch/s, loss=0.499, acc=85.6]
Train: 100%|     | 750/750 [00:07<00:00, 105.75batch/s, loss=0.396, acc=86.4]
Eval: 100%|      | 188/188 [00:01<00:00, 116.56batch/s, loss=0.458, acc=86.1]
Train: 100%|     | 750/750 [00:07<00:00, 103.29batch/s, loss=0.366, acc=86.6]
Eval: 100%|      | 188/188 [00:01<00:00, 120.23batch/s, loss=0.476, acc=85.7]
Train: 100%|     | 750/750 [00:07<00:00, 103.75batch/s, loss=0.339, acc=86.8]
Eval: 100%|      | 188/188 [00:01<00:00, 117.19batch/s, loss=0.444, acc=86.1]
Train: 100%|       | 750/750 [00:07<00:00, 106.68batch/s, loss=0.401, acc=87]
Eval: 100%|       | 188/188 [00:01<00:00, 119.38batch/s, loss=0.46, acc=86.3]
Train: 100%|     | 750/750 [00:07<00:00, 100.79batch/s, loss=0.321, acc=87.2]
Eval: 100%|      | 188/188 [00:01<00:00, 121.22batch/s, loss=0.422, acc=86.6]
Train: 100%|     | 750/750 [00:07<00:00, 101.82batch/s, loss=0.366, acc=87.2]
Eval: 100%|      | 188/188 [00:01<00:00, 121.53batch/s, loss=0.475, acc=86.2]
Train: 100%|      | 750/750 [00:07<00:00, 99.13batch/s, loss=0.594, acc=87.5]
Eval: 100%|      | 188/188 [00:01<00:00, 125.44batch/s, loss=0.494, acc=86.4]
Train: 100%|     | 750/750 [00:07<00:00, 103.06batch/s, loss=0.352, acc=87.5]
Eval: 100%|      | 188/188 [00:01<00:00, 118.19batch/s, loss=0.389, acc=86.6]
Train: 100%|     | 750/750 [00:06<00:00, 107.43batch/s, loss=0.333, acc=87.7]
Eval: 100%|      | 188/188 [00:01<00:00, 118.45batch/s, loss=0.445, acc=86.5]
Train: 100%|     | 750/750 [00:06<00:00, 108.88batch/s, loss=0.279, acc=87.8]
Eval: 100%|      | 188/188 [00:01<00:00, 115.58batch/s, loss=0.364, acc=86.1]
Train: 100%|     | 750/750 [00:07<00:00, 105.63batch/s, loss=0.343, acc=87.9]
Eval: 100%|       | 188/188 [00:01<00:00, 117.80batch/s, loss=0.45, acc=86.9]
Train: 100%|     | 750/750 [00:06<00:00, 108.53batch/s, loss=0.401, acc=88.1]
Eval: 100%|      | 188/188 [00:01<00:00, 116.85batch/s, loss=0.458, acc=86.8]
Train: 100%|     | 750/750 [00:07<00:00, 103.30batch/s, loss=0.487, acc=88.1]
Eval: 100%|      | 188/188 [00:01<00:00, 120.54batch/s, loss=0.495, acc=86.9]
```

```
Train: 100%|    | 750/750 [00:06<00:00, 108.73batch/s, loss=0.482, acc=88.2]
Eval: 100%|     | 188/188 [00:01<00:00, 125.59batch/s, loss=0.473, acc=86.7]
Train: 100%|    | 750/750 [00:07<00:00, 103.31batch/s, loss=0.279, acc=88.2]
Eval: 100%|     | 188/188 [00:01<00:00, 120.43batch/s, loss=0.417, acc=87.2]
Train: 100%|     | 750/750 [00:07<00:00, 99.47batch/s, loss=0.38, acc=88.3]
Eval: 100%|     | 188/188 [00:01<00:00, 111.99batch/s, loss=0.476, acc=86.8]
Train: 100%|    | 750/750 [00:07<00:00, 105.84batch/s, loss=0.367, acc=88.5]
Eval: 100%|     | 188/188 [00:01<00:00, 115.19batch/s, loss=0.463, acc=86.8]
Train: 100%|    | 750/750 [00:07<00:00, 101.82batch/s, loss=0.467, acc=88.5]
Eval: 100%|     | 188/188 [00:01<00:00, 122.10batch/s, loss=0.429, acc=87.3]
```

```
reg_val : 0.01
```

```
Train: 100%|    | 750/750 [00:07<00:00, 101.65batch/s, loss=0.663, acc=71.9]
Eval: 100%|     | 188/188 [00:01<00:00, 118.07batch/s, loss=0.732, acc=78.5]
Train: 100%|    | 750/750 [00:07<00:00, 103.07batch/s, loss=0.548, acc=80.4]
Eval: 100%|      | 188/188 [00:01<00:00, 117.66batch/s, loss=0.61, acc=81.2]
Train: 100%|    | 750/750 [00:07<00:00, 103.37batch/s, loss=0.662, acc=82.2]
Eval: 100%|      | 188/188 [00:01<00:00, 119.47batch/s, loss=0.571, acc=82]
Train: 100%|    | 750/750 [00:07<00:00, 107.03batch/s, loss=0.488, acc=83.1]
Eval: 100%|       | 188/188 [00:01<00:00, 122.06batch/s, loss=0.56, acc=83]
Train: 100%|     | 750/750 [00:07<00:00, 99.11batch/s, loss=0.529, acc=83.6]
Eval: 100%|     | 188/188 [00:01<00:00, 120.58batch/s, loss=0.495, acc=83.5]
Train: 100%|    | 750/750 [00:07<00:00, 102.17batch/s, loss=0.566, acc=84.1]
Eval: 100%|     | 188/188 [00:01<00:00, 120.85batch/s, loss=0.464, acc=83.7]
Train: 100%|    | 750/750 [00:07<00:00, 105.49batch/s, loss=0.408, acc=84.4]
Eval: 100%|     | 188/188 [00:01<00:00, 116.01batch/s, loss=0.513, acc=83.9]
Train: 100%|     | 750/750 [00:07<00:00, 96.36batch/s, loss=0.349, acc=84.6]
Eval: 100%|     | 188/188 [00:01<00:00, 119.31batch/s, loss=0.465, acc=84.1]
Train: 100%|    | 750/750 [00:07<00:00, 101.31batch/s, loss=0.526, acc=84.9]
Eval: 100%|      | 188/188 [00:01<00:00, 118.62batch/s, loss=0.49, acc=84.1]
Train: 100%|      | 750/750 [00:07<00:00, 96.28batch/s, loss=0.383, acc=85]
Eval: 100%|     | 188/188 [00:01<00:00, 114.58batch/s, loss=0.494, acc=84.4]
Train: 100%|     | 750/750 [00:08<00:00, 91.87batch/s, loss=0.415, acc=85.1]
Eval: 100%|     | 188/188 [00:01<00:00, 111.43batch/s, loss=0.516, acc=84.3]
Train: 100%|     | 750/750 [00:07<00:00, 94.23batch/s, loss=0.586, acc=85.3]
Eval: 100%|     | 188/188 [00:01<00:00, 115.90batch/s, loss=0.518, acc=84.7]
Train: 100%|     | 750/750 [00:08<00:00, 89.03batch/s, loss=0.372, acc=85.3]
Eval: 100%|     | 188/188 [00:01<00:00, 119.14batch/s, loss=0.507, acc=84.7]
Train: 100%|     | 750/750 [00:08<00:00, 90.52batch/s, loss=0.313, acc=85.5]
Eval: 100%|      | 188/188 [00:01<00:00, 117.13batch/s, loss=0.459, acc=85]
Train: 100%|     | 750/750 [00:07<00:00, 93.96batch/s, loss=0.443, acc=85.5]
Eval: 100%|      | 188/188 [00:01<00:00, 116.38batch/s, loss=0.493, acc=85]
Train: 100%|     | 750/750 [00:11<00:00, 67.87batch/s, loss=0.412, acc=85.8]
Eval: 100%|      | 188/188 [00:01<00:00, 119.91batch/s, loss=0.495, acc=85]
Train: 100%|     | 750/750 [00:07<00:00, 95.08batch/s, loss=0.303, acc=85.8]
Eval: 100%|      | 188/188 [00:01<00:00, 108.72batch/s, loss=0.517, acc=85]
Train: 100%|     | 750/750 [00:07<00:00, 96.23batch/s, loss=0.367, acc=85.7]
Eval: 100%|     | 188/188 [00:01<00:00, 110.98batch/s, loss=0.488, acc=85.1]
```

```
Train: 100%|     | 750/750 [00:07<00:00, 95.13batch/s, loss=0.485, acc=85.8]
Eval: 100%|     | 188/188 [00:01<00:00, 119.67batch/s, loss=0.479, acc=84.9]
Train: 100%|    | 750/750 [00:07<00:00, 100.34batch/s, loss=0.369, acc=85.9]
Eval: 100%|     | 188/188 [00:01<00:00, 120.32batch/s, loss=0.556, acc=84.4]
Train: 100%|      | 750/750 [00:07<00:00, 96.98batch/s, loss=0.452, acc=86]
Eval: 100%|     | 188/188 [00:01<00:00, 115.79batch/s, loss=0.478, acc=85.2]
Train: 100%|      | 750/750 [00:08<00:00, 83.51batch/s, loss=0.546, acc=86]
Eval: 100%|     | 188/188 [00:01<00:00, 115.54batch/s, loss=0.497, acc=85.1]
Train: 100%|      | 750/750 [00:08<00:00, 91.35batch/s, loss=0.481, acc=86]
Eval: 100%|      | 188/188 [00:01<00:00, 109.89batch/s, loss=0.49, acc=85.3]
Train: 100%|     | 750/750 [00:08<00:00, 90.25batch/s, loss=0.402, acc=86.1]
Eval: 100%|     | 188/188 [00:01<00:00, 118.01batch/s, loss=0.516, acc=85.1]
Train: 100%|      | 750/750 [00:07<00:00, 97.77batch/s, loss=0.421, acc=86]
Eval: 100%|     | 188/188 [00:01<00:00, 100.60batch/s, loss=0.493, acc=85.2]
Train: 100%|     | 750/750 [00:08<00:00, 92.75batch/s, loss=0.333, acc=86.1]
Eval: 100%|     | 188/188 [00:01<00:00, 114.42batch/s, loss=0.489, acc=85.1]
Train: 100%|     | 750/750 [00:08<00:00, 92.93batch/s, loss=0.391, acc=86.2]
Eval: 100%|     | 188/188 [00:01<00:00, 115.99batch/s, loss=0.485, acc=85.3]
Train: 100%|    | 750/750 [00:06<00:00, 109.31batch/s, loss=0.329, acc=86.2]
Eval: 100%|     | 188/188 [00:01<00:00, 117.61batch/s, loss=0.509, acc=85.4]
Train: 100%|    | 750/750 [00:07<00:00, 105.16batch/s, loss=0.291, acc=86.3]
Eval: 100%|     | 188/188 [00:01<00:00, 116.37batch/s, loss=0.504, acc=85.2]
Train: 100%|     | 750/750 [00:07<00:00, 96.28batch/s, loss=0.438, acc=86.3]
Eval: 100%|      | 188/188 [00:01<00:00, 117.02batch/s, loss=0.493, acc=85]
```

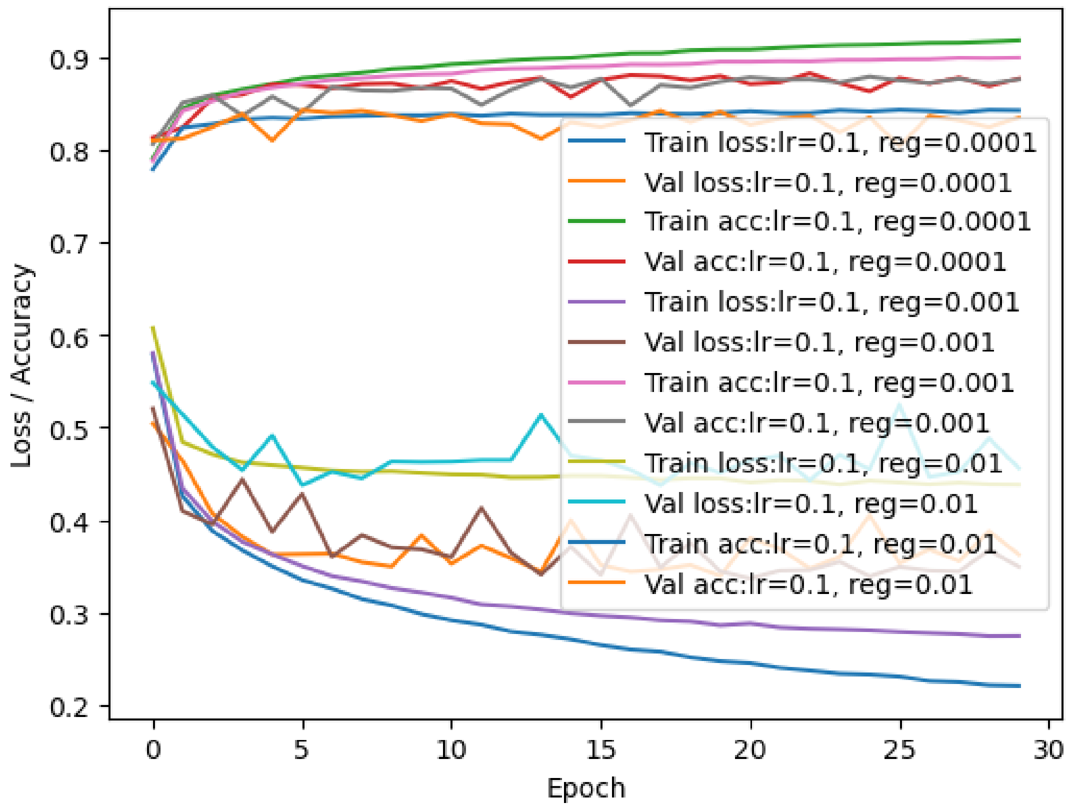


```
lr : 0.1
reg_val : 0.0001

Train: 100%|     | 750/750 [00:06<00:00, 108.74batch/s, loss=0.44, acc=79.1]
Eval: 100%|     | 188/188 [00:01<00:00, 120.01batch/s, loss=0.735, acc=81.3]
Train: 100%|     | 750/750 [00:08<00:00, 91.98batch/s, loss=0.585, acc=84.5]
Eval: 100%|     | 188/188 [00:01<00:00, 119.20batch/s, loss=0.641, acc=82.4]
Train: 100%|     | 750/750 [00:07<00:00, 99.08batch/s, loss=0.397, acc=85.9]
Eval: 100%|     | 188/188 [00:01<00:00, 121.03batch/s, loss=0.642, acc=85.5]
Train: 100%|     | 750/750 [00:07<00:00, 100.13batch/s, loss=0.378, acc=86.5]
Eval: 100%|     | 188/188 [00:01<00:00, 118.83batch/s, loss=0.538, acc=85.9]
Train: 100%|     | 750/750 [00:07<00:00, 102.32batch/s, loss=0.151, acc=87.1]
Eval: 100%|     | 188/188 [00:01<00:00, 117.46batch/s, loss=0.467, acc=87.1]
Train: 100%|      | 750/750 [00:08<00:00, 86.38batch/s, loss=0.23, acc=87.7]
Eval: 100%|      | 188/188 [00:01<00:00, 104.89batch/s, loss=0.471, acc=87]
Train: 100%|      | 750/750 [00:08<00:00, 92.67batch/s, loss=0.386, acc=88]
Eval: 100%|     | 188/188 [00:01<00:00, 115.07batch/s, loss=0.503, acc=86.7]
Train: 100%|     | 750/750 [00:07<00:00, 100.55batch/s, loss=0.224, acc=88.3]
Eval: 100%|     | 188/188 [00:01<00:00, 121.26batch/s, loss=0.478, acc=87.1]
Train: 100%|     | 750/750 [00:06<00:00, 109.89batch/s, loss=0.222, acc=88.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.13batch/s, loss=0.541, acc=87.2]
Train: 100%|     | 750/750 [00:06<00:00, 109.80batch/s, loss=0.39, acc=88.9]
Eval: 100%|     | 188/188 [00:01<00:00, 120.52batch/s, loss=0.524, acc=86.6]
Train: 100%|     | 750/750 [00:07<00:00, 106.81batch/s, loss=0.177, acc=89.2]
Eval: 100%|     | 188/188 [00:01<00:00, 120.34batch/s, loss=0.481, acc=87.4]
Train: 100%|     | 750/750 [00:06<00:00, 109.08batch/s, loss=0.458, acc=89.4]
Eval: 100%|     | 188/188 [00:01<00:00, 117.21batch/s, loss=0.429, acc=86.6]
Train: 100%|     | 750/750 [00:06<00:00, 109.47batch/s, loss=0.242, acc=89.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.25batch/s, loss=0.552, acc=87.3]
Train: 100%|     | 750/750 [00:06<00:00, 112.35batch/s, loss=0.251, acc=89.8]
Eval: 100%|     | 188/188 [00:01<00:00, 118.87batch/s, loss=0.432, acc=87.8]
Train: 100%|     | 750/750 [00:06<00:00, 110.98batch/s, loss=0.437, acc=89.9]
Eval: 100%|     | 188/188 [00:01<00:00, 120.26batch/s, loss=0.545, acc=85.7]
Train: 100%|     | 750/750 [00:06<00:00, 107.27batch/s, loss=0.365, acc=90.2]
Eval: 100%|     | 188/188 [00:01<00:00, 108.82batch/s, loss=0.592, acc=87.5]
Train: 100%|     | 750/750 [00:08<00:00, 86.15batch/s, loss=0.184, acc=90.4]
Eval: 100%|     | 188/188 [00:01<00:00, 119.17batch/s, loss=0.612, acc=88.1]
Train: 100%|     | 750/750 [00:07<00:00, 99.61batch/s, loss=0.197, acc=90.4]
Eval: 100%|     | 188/188 [00:01<00:00, 117.69batch/s, loss=0.508, acc=87.9]
Train: 100%|     | 750/750 [00:07<00:00, 96.44batch/s, loss=0.253, acc=90.7]
Eval: 100%|     | 188/188 [00:01<00:00, 119.33batch/s, loss=0.566, acc=87.5]
Train: 100%|     | 750/750 [00:06<00:00, 108.14batch/s, loss=0.162, acc=90.8]
Eval: 100%|      | 188/188 [00:01<00:00, 120.81batch/s, loss=0.424, acc=88]
Train: 100%|     | 750/750 [00:08<00:00, 88.14batch/s, loss=0.258, acc=90.8]
Eval: 100%|     | 188/188 [00:01<00:00, 108.48batch/s, loss=0.49, acc=87.1]
Train: 100%|      | 750/750 [00:08<00:00, 90.15batch/s, loss=0.145, acc=91]
```

```
Eval: 100%|     | 188/188 [00:01<00:00, 117.56batch/s, loss=0.587, acc=87.3]
Train: 100%|     | 750/750 [00:07<00:00, 96.90batch/s, loss=0.175, acc=91.2]
Eval: 100%|     | 188/188 [00:01<00:00, 116.62batch/s, loss=0.554, acc=88.3]
Train: 100%|     | 750/750 [00:08<00:00, 85.77batch/s, loss=0.264, acc=91.3]
Eval: 100%|     | 188/188 [00:01<00:00, 109.61batch/s, loss=0.443, acc=87.2]
Train: 100%|     | 750/750 [00:08<00:00, 90.16batch/s, loss=0.136, acc=91.3]
Eval: 100%|     | 188/188 [00:01<00:00, 110.85batch/s, loss=0.613, acc=86.3]
Train: 100%|     | 750/750 [00:09<00:00, 80.92batch/s, loss=0.199, acc=91.4]
Eval: 100%|      | 188/188 [00:01<00:00, 118.24batch/s, loss=0.54, acc=87.8]
Train: 100%|      | 750/750 [00:07<00:00, 94.42batch/s, loss=0.22, acc=91.5]
Eval: 100%|     | 188/188 [00:01<00:00, 117.88batch/s, loss=0.686, acc=87.2]
Train: 100%|     | 750/750 [00:08<00:00, 88.37batch/s, loss=0.241, acc=91.6]
Eval: 100%|     | 188/188 [00:01<00:00, 119.13batch/s, loss=0.682, acc=87.8]
Train: 100%|     | 750/750 [00:08<00:00, 92.13batch/s, loss=0.282, acc=91.7]
Eval: 100%|     | 188/188 [00:01<00:00, 115.97batch/s, loss=0.633, acc=86.9]
Train: 100%|     | 750/750 [00:08<00:00, 90.11batch/s, loss=0.174, acc=91.8]
Eval: 100%|     | 188/188 [00:01<00:00, 118.94batch/s, loss=0.497, acc=87.7]

reg_val : 0.001

Train: 100%|    | 750/750 [00:07<00:00, 104.90batch/s, loss=0.563, acc=78.8]
Eval: 100%|     | 188/188 [00:01<00:00, 116.06batch/s, loss=0.683, acc=80.6]
Train: 100%|     | 750/750 [00:07<00:00, 96.63batch/s, loss=0.314, acc=84.2]
Eval: 100%|     | 188/188 [00:01<00:00, 109.09batch/s, loss=0.437, acc=85.1]
Train: 100%|     | 750/750 [00:09<00:00, 80.44batch/s, loss=0.322, acc=85.3]
Eval: 100%|     | 188/188 [00:01<00:00, 115.16batch/s, loss=0.451, acc=85.9]
Train: 100%|       | 750/750 [00:08<00:00, 86.91batch/s, loss=0.4, acc=86.2]
Eval: 100%|      | 188/188 [00:01<00:00, 104.02batch/s, loss=0.64, acc=83.7]
Train: 100%|     | 750/750 [00:08<00:00, 85.71batch/s, loss=0.293, acc=86.7]
Eval: 100%|     | 188/188 [00:01<00:00, 110.67batch/s, loss=0.493, acc=85.7]
Train: 100%|     | 750/750 [00:09<00:00, 78.49batch/s, loss=0.572, acc=87.1]
Eval: 100%|      | 188/188 [00:01<00:00, 118.93batch/s, loss=0.442, acc=84]
Train: 100%|     | 750/750 [00:07<00:00, 95.22batch/s, loss=0.416, acc=87.5]
Eval: 100%|     | 188/188 [00:01<00:00, 117.23batch/s, loss=0.407, acc=86.8]
Train: 100%|    | 750/750 [00:07<00:00, 105.81batch/s, loss=0.294, acc=87.8]
Eval: 100%|     | 188/188 [00:01<00:00, 117.44batch/s, loss=0.366, acc=86.4]
Train: 100%|      | 750/750 [00:08<00:00, 87.12batch/s, loss=0.404, acc=88]
Eval: 100%|      | 188/188 [00:02<00:00, 80.68batch/s, loss=0.381, acc=86.4]
Train: 100%|     | 750/750 [00:09<00:00, 82.54batch/s, loss=0.356, acc=88.1]
Eval: 100%|     | 188/188 [00:01<00:00, 108.61batch/s, loss=0.378, acc=86.7]
Train: 100%|     | 750/750 [00:08<00:00, 87.79batch/s, loss=0.355, acc=88.2]
Eval: 100%|     | 188/188 [00:01<00:00, 102.83batch/s, loss=0.441, acc=86.6]
Train: 100%|     | 750/750 [00:08<00:00, 87.74batch/s, loss=0.575, acc=88.6]
Eval: 100%|     | 188/188 [00:01<00:00, 105.95batch/s, loss=0.617, acc=84.9]
Train: 100%|     | 750/750 [00:08<00:00, 89.85batch/s, loss=0.318, acc=88.8]
Eval: 100%|     | 188/188 [00:01<00:00, 106.18batch/s, loss=0.426, acc=86.5]
Train: 100%|     | 750/750 [00:09<00:00, 83.14batch/s, loss=0.311, acc=88.8]
Eval: 100%|     | 188/188 [00:01<00:00, 107.84batch/s, loss=0.411, acc=87.6]
Train: 100%|       | 750/750 [00:08<00:00, 87.48batch/s, loss=0.24, acc=89]
```

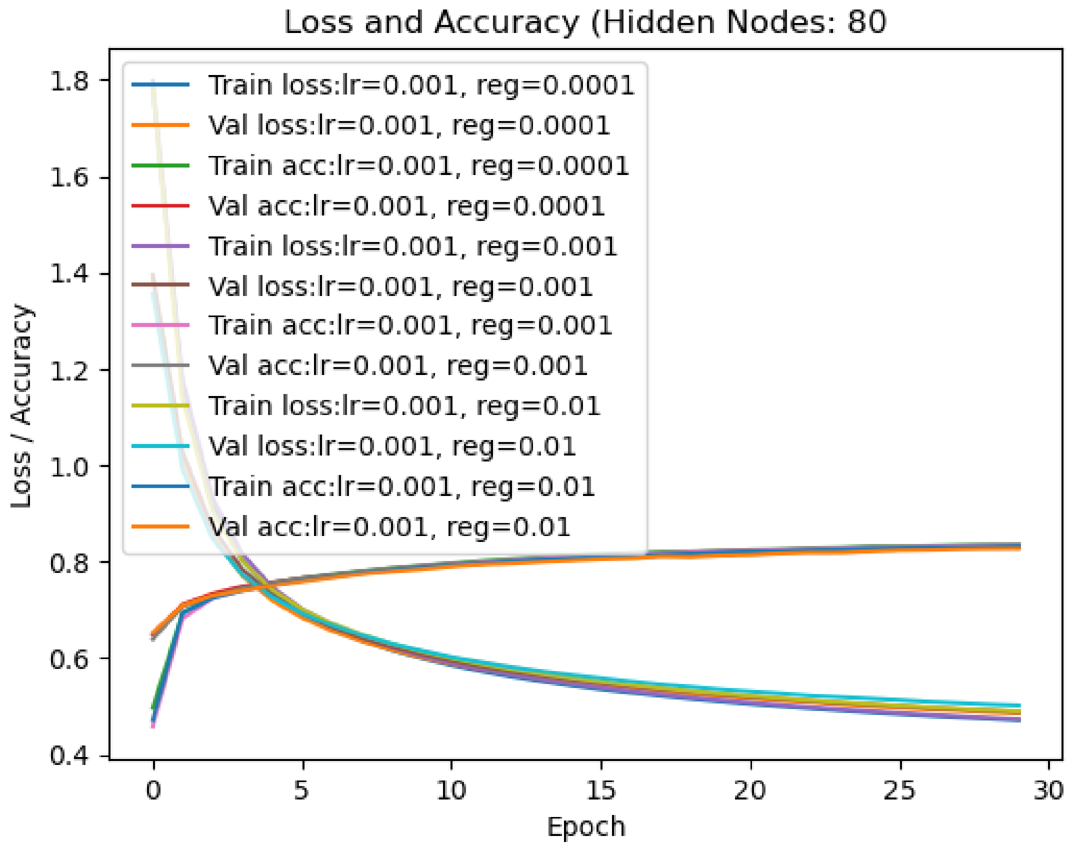
```
Eval: 100%|      | 188/188 [00:01<00:00, 97.87batch/s, loss=0.42, acc=86.7]
Train: 100%|      | 750/750 [00:08<00:00, 83.97batch/s, loss=0.53, acc=89]
Eval: 100%|     | 188/188 [00:01<00:00, 109.78batch/s, loss=0.407, acc=87.7]
Train: 100%|     | 750/750 [00:08<00:00, 89.96batch/s, loss=0.422, acc=89.2]
Eval: 100%|     | 188/188 [00:01<00:00, 110.05batch/s, loss=0.494, acc=84.8]
Train: 100%|     | 750/750 [00:09<00:00, 78.53batch/s, loss=0.388, acc=89.2]
Eval: 100%|      | 188/188 [00:01<00:00, 117.47batch/s, loss=0.378, acc=87]
Train: 100%|     | 750/750 [00:07<00:00, 97.51batch/s, loss=0.365, acc=89.3]
Eval: 100%|     | 188/188 [00:01<00:00, 112.86batch/s, loss=0.431, acc=86.7]
Train: 100%|     | 750/750 [00:08<00:00, 83.57batch/s, loss=0.198, acc=89.5]
Eval: 100%|     | 188/188 [00:01<00:00, 106.18batch/s, loss=0.548, acc=87.4]
Train: 100%|     | 750/750 [00:08<00:00, 93.42batch/s, loss=0.181, acc=89.5]
Eval: 100%|     | 188/188 [00:01<00:00, 112.77batch/s, loss=0.454, acc=87.8]
Train: 100%|     | 750/750 [00:08<00:00, 91.05batch/s, loss=0.252, acc=89.6]
Eval: 100%|     | 188/188 [00:01<00:00, 104.07batch/s, loss=0.363, acc=87.6]
Train: 100%|     | 750/750 [00:09<00:00, 80.99batch/s, loss=0.299, acc=89.5]
Eval: 100%|     | 188/188 [00:01<00:00, 104.25batch/s, loss=0.408, acc=87.6]
Train: 100%|     | 750/750 [00:08<00:00, 89.87batch/s, loss=0.363, acc=89.7]
Eval: 100%|     | 188/188 [00:01<00:00, 112.77batch/s, loss=0.473, acc=87.3]
Train: 100%|     | 750/750 [00:07<00:00, 97.35batch/s, loss=0.339, acc=89.7]
Eval: 100%|     | 188/188 [00:01<00:00, 115.75batch/s, loss=0.416, acc=87.9]
Train: 100%|     | 750/750 [00:07<00:00, 94.72batch/s, loss=0.187, acc=89.8]
Eval: 100%|     | 188/188 [00:01<00:00, 104.82batch/s, loss=0.506, acc=87.5]
Train: 100%|     | 750/750 [00:07<00:00, 99.02batch/s, loss=0.244, acc=89.8]
Eval: 100%|     | 188/188 [00:01<00:00, 117.82batch/s, loss=0.481, acc=87.2]
Train: 100%|     | 750/750 [00:07<00:00, 98.18batch/s, loss=0.294, acc=89.9]
Eval: 100%|     | 188/188 [00:01<00:00, 115.25batch/s, loss=0.438, acc=87.7]
Train: 100%|      | 750/750 [00:07<00:00, 99.58batch/s, loss=0.46, acc=89.9]
Eval: 100%|     | 188/188 [00:01<00:00, 116.90batch/s, loss=0.393, acc=87.1]
Train: 100%|      | 750/750 [00:07<00:00, 96.33batch/s, loss=0.177, acc=90]
Eval: 100%|     | 188/188 [00:01<00:00, 119.13batch/s, loss=0.343, acc=87.6]

reg_val : 0.01

Train: 100%|    | 750/750 [00:07<00:00, 104.18batch/s, loss=0.367, acc=77.9]
Eval: 100%|      | 188/188 [00:01<00:00, 119.98batch/s, loss=0.526, acc=81]
Train: 100%|    | 750/750 [00:06<00:00, 107.18batch/s, loss=0.469, acc=82.4]
Eval: 100%|     | 188/188 [00:01<00:00, 125.97batch/s, loss=0.444, acc=81.2]
Train: 100%|     | 750/750 [00:07<00:00, 94.33batch/s, loss=0.408, acc=82.8]
Eval: 100%|     | 188/188 [00:01<00:00, 109.38batch/s, loss=0.635, acc=82.5]
Train: 100%|     | 750/750 [00:08<00:00, 87.52batch/s, loss=0.421, acc=83.3]
Eval: 100%|     | 188/188 [00:01<00:00, 117.17batch/s, loss=0.463, acc=83.9]
Train: 100%|    | 750/750 [00:07<00:00, 102.66batch/s, loss=0.431, acc=83.5]
Eval: 100%|      | 188/188 [00:01<00:00, 112.71batch/s, loss=0.529, acc=81]
Train: 100%|    | 750/750 [00:07<00:00, 100.78batch/s, loss=0.291, acc=83.4]
Eval: 100%|     | 188/188 [00:01<00:00, 115.96batch/s, loss=0.542, acc=84.3]
Train: 100%|    | 750/750 [00:07<00:00, 105.37batch/s, loss=0.424, acc=83.6]
Eval: 100%|      | 188/188 [00:01<00:00, 120.38batch/s, loss=0.521, acc=84]
Train: 100%|     | 750/750 [00:07<00:00, 93.83batch/s, loss=0.462, acc=83.7]
```

```
Eval: 100%|      | 188/188 [00:01<00:00, 98.08batch/s, loss=0.585, acc=84.2]
Train: 100%|      | 750/750 [00:08<00:00, 93.11batch/s, loss=0.51, acc=83.8]
Eval: 100%|     | 188/188 [00:01<00:00, 117.59batch/s, loss=0.429, acc=83.7]
Train: 100%|     | 750/750 [00:08<00:00, 93.23batch/s, loss=0.446, acc=83.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.17batch/s, loss=0.552, acc=83.1]
Train: 100%|     | 750/750 [00:06<00:00, 110.20batch/s, loss=0.321, acc=83.9]
Eval: 100%|     | 188/188 [00:01<00:00, 114.57batch/s, loss=0.459, acc=83.8]
Train: 100%|     | 750/750 [00:07<00:00, 103.23batch/s, loss=0.453, acc=83.7]
Eval: 100%|     | 188/188 [00:01<00:00, 120.43batch/s, loss=0.434, acc=82.8]
Train: 100%|     | 750/750 [00:06<00:00, 115.61batch/s, loss=0.286, acc=83.9]
Eval: 100%|     | 188/188 [00:01<00:00, 119.84batch/s, loss=0.537, acc=82.7]
Train: 100%|     | 750/750 [00:06<00:00, 110.79batch/s, loss=0.542, acc=83.8]
Eval: 100%|     | 188/188 [00:01<00:00, 119.11batch/s, loss=0.721, acc=81.2]
Train: 100%|     | 750/750 [00:06<00:00, 109.62batch/s, loss=0.592, acc=83.8]
Eval: 100%|      | 188/188 [00:01<00:00, 116.40batch/s, loss=0.48, acc=83]
Train: 100%|     | 750/750 [00:06<00:00, 108.98batch/s, loss=0.441, acc=83.8]
Eval: 100%|     | 188/188 [00:01<00:00, 119.71batch/s, loss=0.463, acc=82.5]
Train: 100%|      | 750/750 [00:07<00:00, 100.97batch/s, loss=0.376, acc=84]
Eval: 100%|     | 188/188 [00:01<00:00, 110.80batch/s, loss=0.589, acc=83.2]
Train: 100%|     | 750/750 [00:06<00:00, 113.94batch/s, loss=0.478, acc=83.9]
Eval: 100%|     | 188/188 [00:01<00:00, 119.68batch/s, loss=0.484, acc=84.2]
Train: 100%|     | 750/750 [00:06<00:00, 107.72batch/s, loss=0.451, acc=83.9]
Eval: 100%|      | 188/188 [00:01<00:00, 120.30batch/s, loss=0.502, acc=83]
Train: 100%|      | 750/750 [00:06<00:00, 111.20batch/s, loss=0.47, acc=84]
Eval: 100%|     | 188/188 [00:01<00:00, 110.94batch/s, loss=0.521, acc=84.1]
Train: 100%|     | 750/750 [00:06<00:00, 112.37batch/s, loss=0.527, acc=84.2]
Eval: 100%|     | 188/188 [00:01<00:00, 121.18batch/s, loss=0.653, acc=82.7]
Train: 100%|      | 750/750 [00:07<00:00, 96.99batch/s, loss=0.569, acc=84]
Eval: 100%|     | 188/188 [00:01<00:00, 113.68batch/s, loss=0.525, acc=83.4]
Train: 100%|      | 750/750 [00:08<00:00, 90.09batch/s, loss=0.243, acc=84]
Eval: 100%|     | 188/188 [00:01<00:00, 119.74batch/s, loss=0.571, acc=83.7]
Train: 100%|     | 750/750 [00:07<00:00, 94.42batch/s, loss=0.286, acc=84.3]
Eval: 100%|     | 188/188 [00:01<00:00, 115.83batch/s, loss=0.552, acc=81.9]
Train: 100%|     | 750/750 [00:07<00:00, 97.87batch/s, loss=0.315, acc=84.1]
Eval: 100%|     | 188/188 [00:01<00:00, 109.96batch/s, loss=0.419, acc=83.5]
Train: 100%|      | 750/750 [00:08<00:00, 90.55batch/s, loss=0.62, acc=84.3]
Eval: 100%|     | 188/188 [00:01<00:00, 109.78batch/s, loss=0.645, acc=80.4]
Train: 100%|     | 750/750 [00:07<00:00, 94.23batch/s, loss=0.469, acc=84.2]
Eval: 100%|     | 188/188 [00:01<00:00, 116.54batch/s, loss=0.583, acc=83.7]
Train: 100%|      | 750/750 [00:08<00:00, 93.14batch/s, loss=0.411, acc=84]
Eval: 100%|     | 188/188 [00:01<00:00, 110.08batch/s, loss=0.603, acc=83.2]
Train: 100%|     | 750/750 [00:07<00:00, 97.84batch/s, loss=0.579, acc=84.3]
Eval: 100%|     | 188/188 [00:01<00:00, 114.18batch/s, loss=0.623, acc=82.5]
Train: 100%|     | 750/750 [00:07<00:00, 97.17batch/s, loss=0.445, acc=84.2]
Eval: 100%|      | 188/188 [00:01<00:00, 119.25batch/s, loss=0.5, acc=83.5]
```

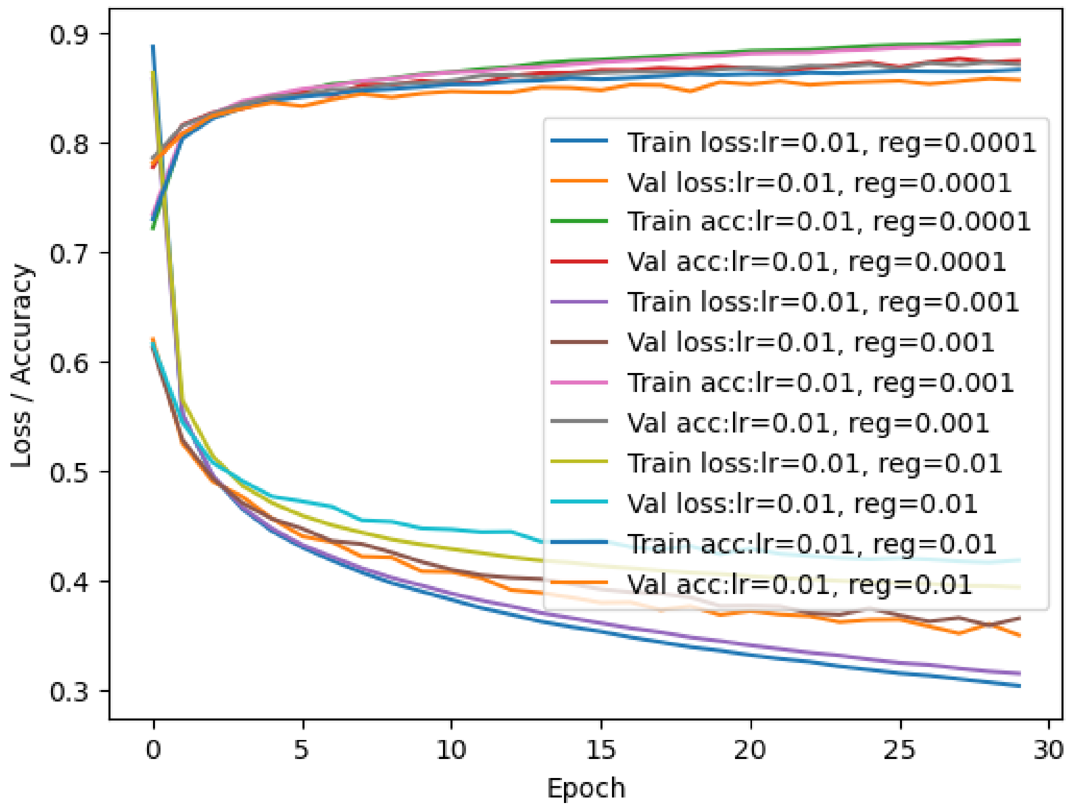


[201]:
```
plot_learningcurve(80)
```

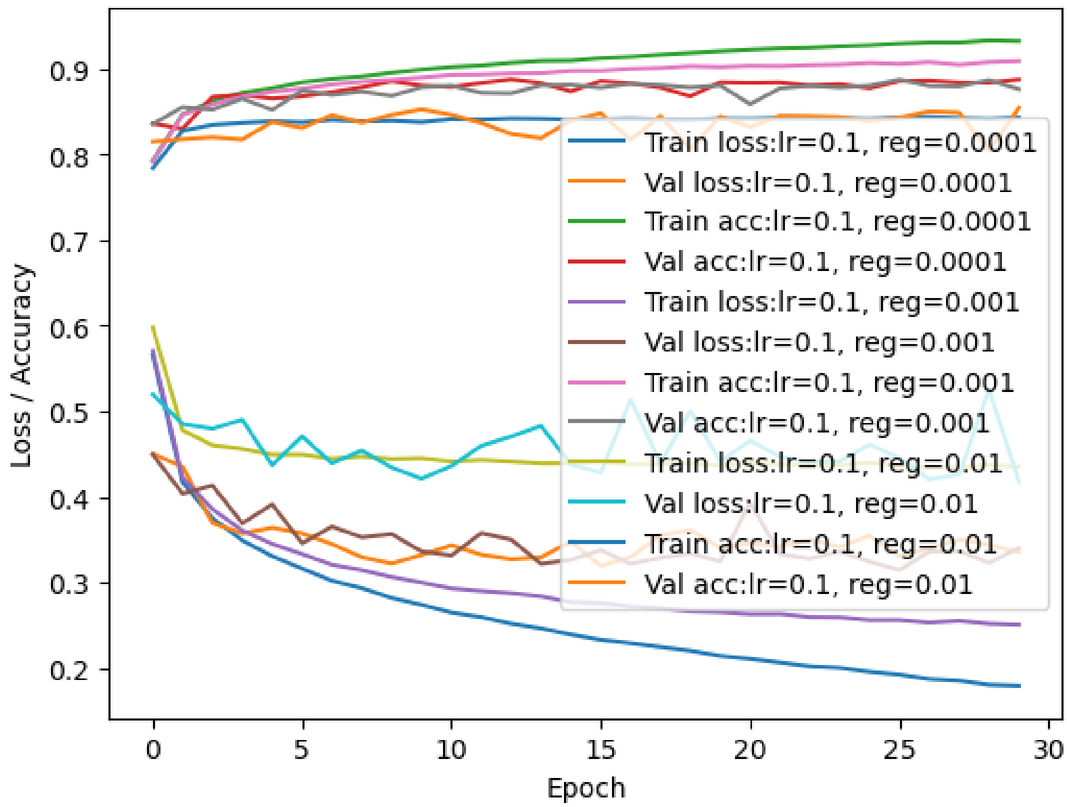
```
Hidden nodes: 80
lr : 0.001
reg_val : 0.0001
reg_val : 0.001
reg_val : 0.01
```




```
lr : 0.01
reg_val : 0.0001
reg_val : 0.001
reg_val : 0.01
```

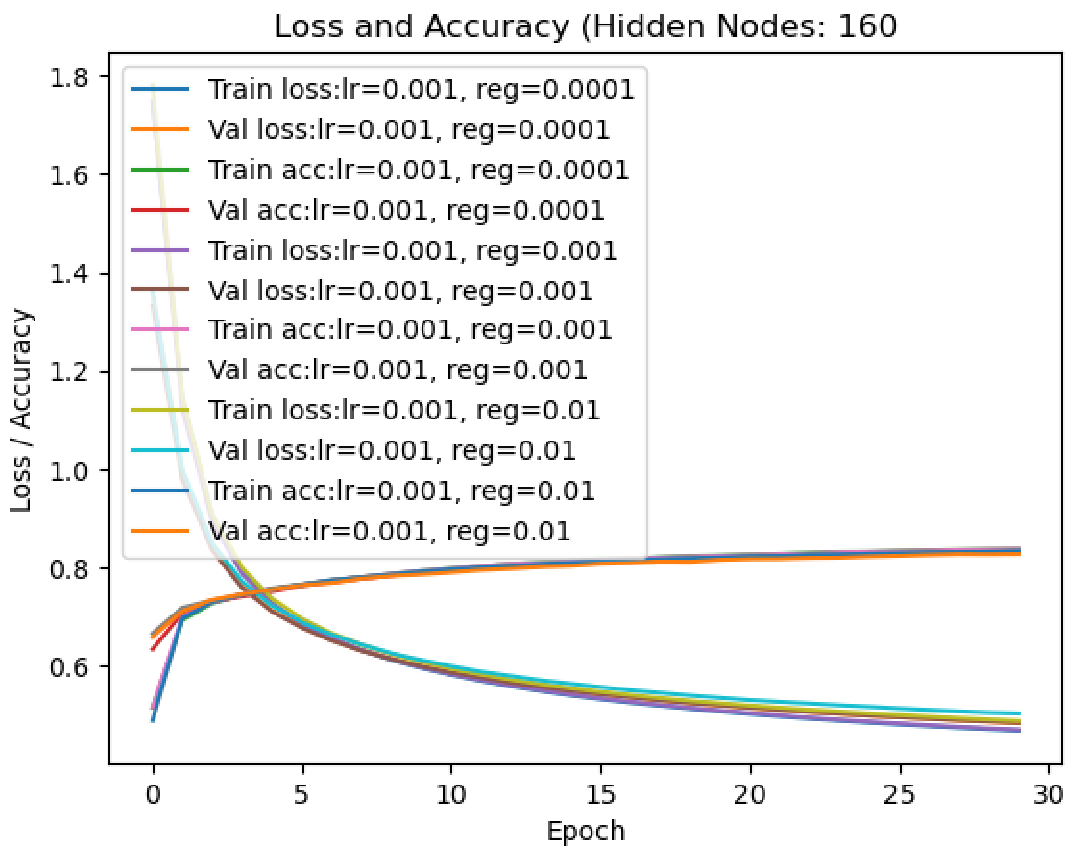



```
lr : 0.1
reg_val : 0.0001
reg_val : 0.001
reg_val : 0.01
```

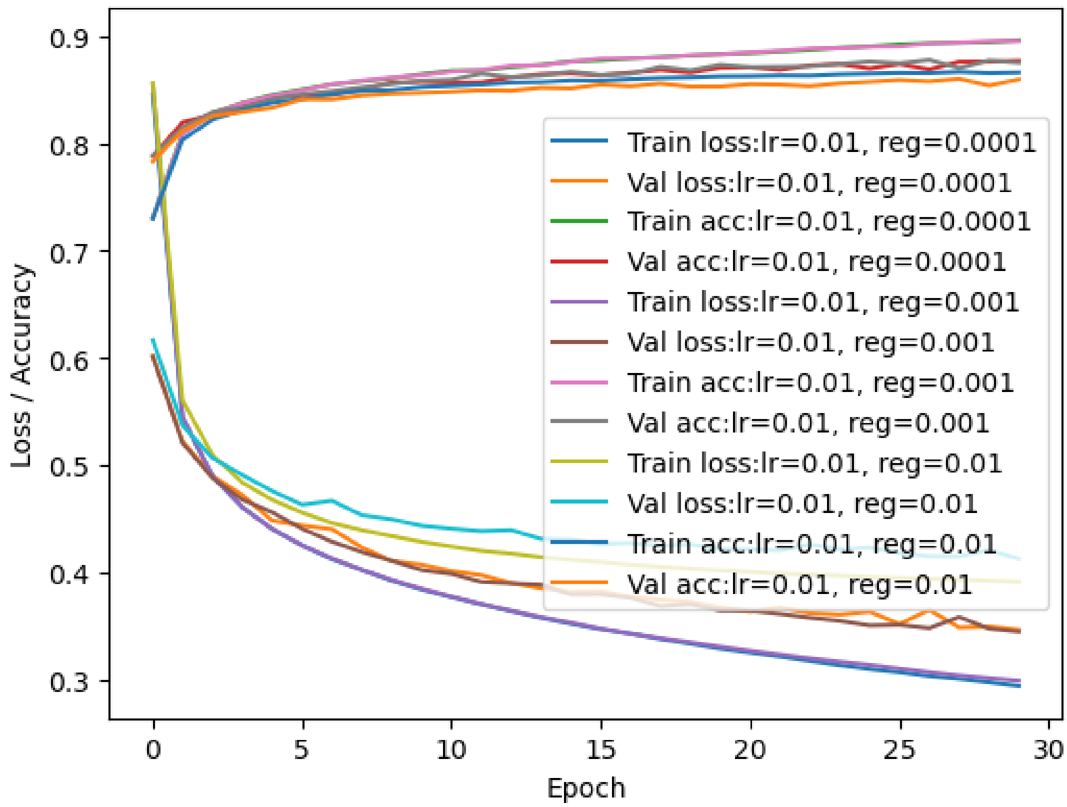


[202]:
```
plot_learningcurve(160)
```

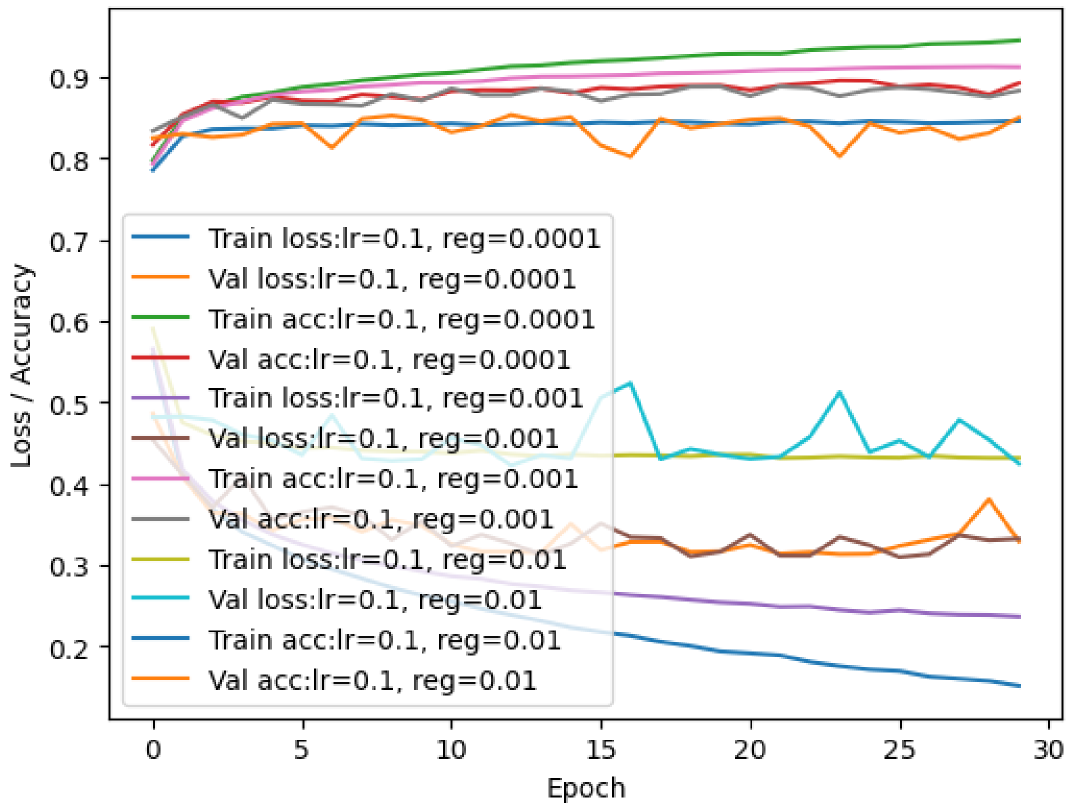
```
Hidden nodes: 160
lr : 0.001
reg_val : 0.0001
reg_val : 0.001
reg_val : 0.01
```




```
lr : 0.01
reg_val : 0.0001
reg_val : 0.001
reg_val : 0.01
```




```
lr : 0.1
reg_val : 0.0001
reg_val : 0.001
reg_val : 0.01
```



[203]:
```
def use_best_hp(hidden, lr, reg_val):
    run = 0
    val_acc_list = []
    best_acc = -1
    for run in range(5):
        # Create a model
        model = MLP(hidden)
#                   lr = params['optimizer__lr']
#                   reg_val = params['optimizer__weight_decay']
        criterion = nn.CrossEntropyLoss() # includes softmax (for numerical␣
↪stability)
        optimizer = optim.SGD(model.parameters(), lr=lr, weight_decay=reg_val)
        device = torch.device("cpu")
        model.to(device) # Move model to device

        trainloader = torch.utils.data.DataLoader(trainset, batch_size=64,␣
↪shuffle=True)
        valloader = torch.utils.data.DataLoader(valset, batch_size=batchsize,␣
↪shuffle=False)

        epoch = 0
```

```
        highest_acc = -1

        for epoch in range(100):
            train_loss, train_acc  = train(model, trainloader, criterion,␣
 ↪optimizer, device) # Train
            val_loss, val_acc = validate(model, valloader, criterion, device) #␣
 ↪Validate

            if(val_acc > highest_acc):
                highest_acc = val_acc
        val_acc_list.append(highest_acc)

        if val_acc > best_acc: # Save best model
            best_acc = val_acc
            torch.save(model.state_dict(), "best_model.pt") # saving model␣
 ↪parameters ("state_dict") saves memory and is faster than saving the entire␣
 ↪model

    print('The best accuracy (over epochs) on val for each run is',val_acc_list)
    print('The mean, max, and std deviation for these 5 values is', np.
 ↪mean(val_acc_list),max(val_acc_list),np.std(val_acc_list))

#     model.load_state_dict(torch.load("best_model.pt"))
#     testloader = torch.utils.data.DataLoader(testset, batch_size=batchsize,␣
 ↪shuffle=False)

#     test_loss, test_acc = validate(model, testloader, criterion, device)
#     print(f"Test accuracy: {test_acc:.4f}")
```

[205]:
```
use_best_hp(160, 0.01, 0.001)
```

```
The best accuracy (over epochs) on val for each run is [0.8916666666666667,
0.8916666666666667, 0.8924166666666666, 0.8924166666666666, 0.8909166666666667,
0.8909166666666667, 0.8925, 0.8925, 0.8920833333333333, 0.8920833333333333]
The mean, max, and std deviation for these 5 values is 0.8919166666666667 0.8925
0.0005797509043641774
```

[211]:
```
val_acc_list = [0.8916666666666667, 0.8924166666666666, 0.8909166666666667, 0.
 ↪8925, 0.8920833333333333]
print('The best accuracy (over epochs) on val for each run is',val_acc_list)
print('The mean, max, and std deviation for these 5 values is', np.
 ↪mean(val_acc_list),max(val_acc_list),np.std(val_acc_list))
```

```
The best accuracy (over epochs) on val for each run is [0.8916666666666667,
0.8924166666666666, 0.8909166666666667, 0.8925, 0.8920833333333333]
The mean, max, and std deviation for these 5 values is 0.8919166666666666 0.8925
0.0005797509043641774
```

[209]:
```
def test():
    model = MLP(160)
    model.load_state_dict(torch.load("best_model.pt"))
    testloader = torch.utils.data.DataLoader(testset, batch_size=batchsize,␣
 ↪shuffle=False)

    test_loss, test_acc = validate(model, testloader, criterion, device)
    print(f"Test accuracy: {test_acc:.4f}")
test()
```

```
Test accuracy: 0.8838
```

[ ]:
```

```

hw7_2

April 8, 2023

[310]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics.pairwise import rbf_kernel
from sklearn.linear_model import LinearRegression
from sklearn.metrics.pairwise import euclidean_distances
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import KFold
from sklearn.cluster import KMeans
```

[3]:
```
xdata_train = np.load('datasetA_X_train.npy')
xdata_test = np.load('datasetA_X_test.npy')
ydata_train = np.load('datasetA_y_train.npy')
ydata_test = np.load('datasetA_y_test.npy')
```

[6]:
```
print(f'xdata_train.shape:{xdata_train.shape}')
print(f'xdata_test.shape:{xdata_test.shape}')
print(f'ydata_train.shape:{xdata_train.shape}')
print(f'ydata_test.shape:{xdata_test.shape}')
```

```
xdata_train.shape:(4000, 2)
xdata_test.shape:(2000, 2)
ydata_train.shape:(4000, 2)
ydata_test.shape:(2000, 2)
```

[72]:
```
def gamma_d(M):
    return M / 200
```

[105]:
```
def Network(xdata, miu, gamma, ydata):
    layer1 = rbf_kernel(xdata, miu, gamma = gamma)
    layer2 = LinearRegression().fit(layer1, ydata)
    return layer2
```

[23]:
```
def RMSE_y(ydata):
    # Compute the mean value of y on the training set
    y_mean = np.mean(ydata)
    y_trival = np.ones(ydata.shape) * y_mean

    # Compute the root mean squared error (RMSE)
```

```
    rmse = np.sqrt(mean_squared_error(y_trival, ydata))
    return rmse
RMSE_y(ydata_train)
```

[23]: 3.2035150890062902

[19]:
```
def RMSE(xdata, ydata, reg):
    predict = reg.predict(xdata)
    MSE = mean_squared_error(ydata, predict)
    return np.sqrt(MSE)
```

0.1 Question c

Choose the basis function centers as the data points: _m = x_m , m = 1,2,...N , in which N is the number of training data points during each fold in cross validation. For this part, the only hyperparameter to choose during model selection is .

[160]:
```
def model_selection_c(xdata, ydata):
    gamma = gamma_d(3000)
    RMSE_train_list = np.zeros([6, 4])
    RMSE_val_list = np.zeros([6, 4])
    MSE_list = np.zeros([6, 4])
#     p_list = np.zeros(4)

    gamma_range = np.array([0.01, 0.1, 1, 10, 100, 1000]) * np.array(gamma)
    print(f'Gamma range is {gamma_range}')
    idx = 0 # in range of 6
    for p in gamma_range:

        # Define the cross-validation object
        cv = KFold(n_splits=4)
        for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i in␣
↪range of 4
            D_train_xdata = xdata[train_index]
            D_train_ydata = ydata[train_index]
            D_val_xdata = xdata[val_index]
            D_val_ydata = ydata[val_index]

            model = Network(D_train_xdata, D_train_xdata, p, D_train_ydata) #␣
↪3000 * 3000

            kernel = rbf_kernel(D_val_xdata, D_train_xdata, p) # 1000 * 3000
            predict = model.predict(kernel)
            MSE = mean_squared_error(D_val_ydata, predict)
            MSE_list[idx][i] = MSE
            RMSE_val_list[idx][i] = np.sqrt(MSE)
```

```python
            kernel_train = rbf_kernel(D_train_xdata, D_train_xdata, p)
            predict_train = model.predict(kernel_train)
            MSE_train = mean_squared_error(D_train_ydata, predict_train)
            RMSE_train_list[idx][i] = np.sqrt(MSE_train)

        idx += 1
    return gamma_range, RMSE_train_list, RMSE_val_list
```

[210]:
```python
gamma_range_c, RMSE_train_list_c, RMSE_val_list_c =␣
 ↪model_selection_c(xdata_train, ydata_train )
```

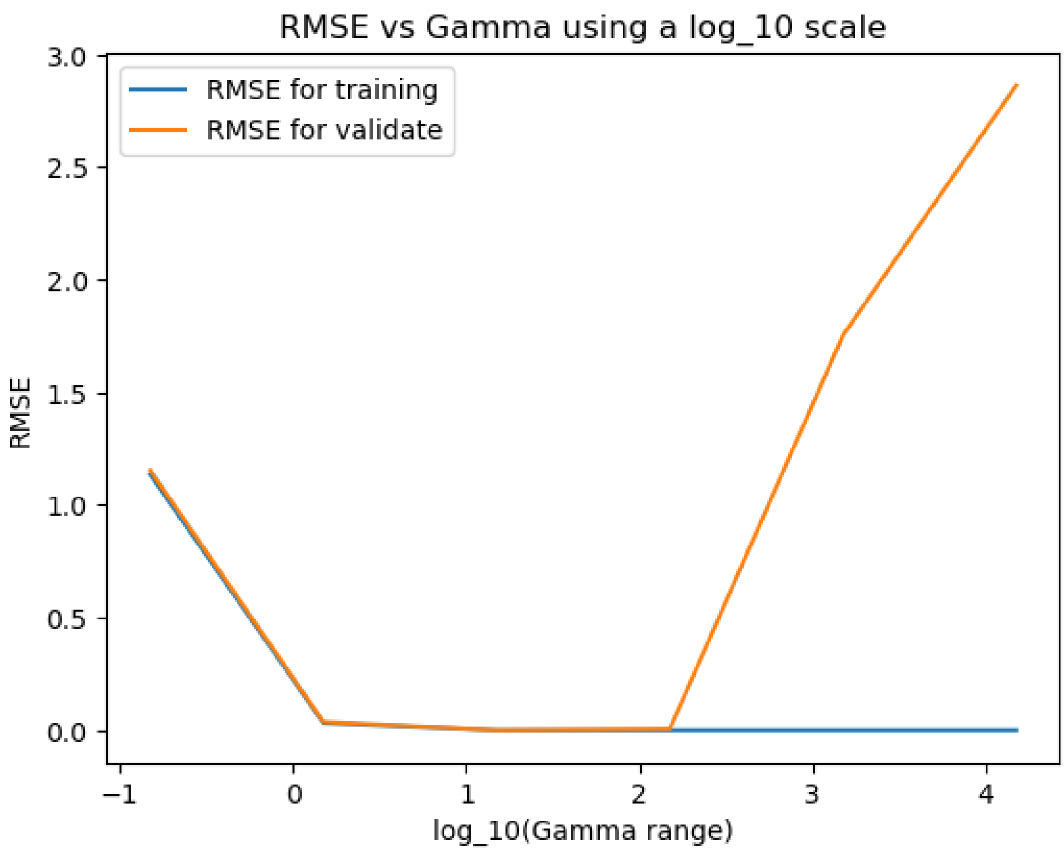
```
Gamma range is [1.5e-01 1.5e+00 1.5e+01 1.5e+02 1.5e+03 1.5e+04]
```

[211]:
```python
RMSE_mean_train_c = np.mean(RMSE_train_list_c, axis = 1)
RMSE_std_train_c = np.std(RMSE_train_list_c, axis = 1)
RMSE_mean_val_c = np.mean(RMSE_val_list_c, axis = 1)
RMSE_std_val_c = np.std(RMSE_val_list_c, axis = 1)



print(f'The mean of train RMSE is {RMSE_mean_train_c}')
print(f'The mean of validatation RMSE is {RMSE_mean_val_c}')
print(f'The std of train RMSE is {RMSE_std_train_c}')
print(f'The std of validatation RMSE is {RMSE_std_val_c}')

print(f'Therefore, the best gamma is {gamma_range_c[np.
 ↪argmin(RMSE_mean_val_c)]}')
```

```
The mean of train RMSE is [1.13475822e+00 3.11643033e-02 3.05638622e-08
3.58646810e-12
 3.23542139e-14 2.06211531e-14]
The mean of validatation RMSE is [1.15403078e+00 3.66475529e-02 7.73597943e-07
5.21484373e-03
 1.75331839e+00 2.86271151e+00]
The std of train RMSE is [7.33129502e-02 3.26677428e-03 7.91081453e-09
5.94505790e-13
 5.27441228e-15 3.43880324e-16]
The std of validatation RMSE is [7.89945707e-02 5.67047695e-03 5.96283883e-07
3.39567285e-03
 3.53033549e-01 4.21701206e-02]
Therefore, the best gamma is 15.0
```

[213]:
```python
def plot_RMSE_gamma(RMSE_train_list,RMSE_val_list, gamma_range):
    gamma_range = np.log10(gamma_range)
    plt.plot(gamma_range,RMSE_train_list, label = 'RMSE for training')
```

```
    plt.plot(gamma_range,RMSE_val_list, label = 'RMSE for validate')
    plt.xlabel('log_10(Gamma range)')
    plt.ylabel('RMSE')
    plt.title("RMSE vs Gamma using a log_10 scale")
    plt.legend()
    plt.show()


plot_RMSE_gamma(RMSE_mean_train_c,RMSE_mean_val_c, gamma_range_c )
```



0.2 Question d

Randomly choose the basis function centers, without replacement, from the training-set data. Use number of basis function centers M varying from 30 to 300 (e.g., values 30, 60, 100, 300, 600).

[221]:
```
def model_selection_d(xdata, ydata):
    M_range = [30, 60, 100, 300, 600]
    RMSE_train_list = np.zeros([len(M_range), 6])
    RMSE_val_list = np.zeros([len(M_range), 6])
    RMSE_train_std = np.zeros([len(M_range), 6])
```

```
    RMSE_val_std = np.zeros([len(M_range), 6])

    gamma_range_M = []

    for k in range(len(M_range)):

        gamma = gamma_d(M_range[k])
        gamma_range = np.array([0.01, 0.1, 1, 10, 100, 1000]) * np.array(gamma)
        print(f'Gamma range is {gamma_range}')
        gamma_range_M.append(gamma_range)
        idx = 0 # idx in range of 6
        for p in gamma_range:
            # Define the cross-validation object
            cv = KFold(n_splits=4)
            train_history = []
            val_history = []

            for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i in range of 4
                D_train_xdata = xdata[train_index]
                D_train_ydata = ydata[train_index]
                D_val_xdata = xdata[val_index]
                D_val_ydata = ydata[val_index]

                miu = D_train_xdata[np.random.choice(len(D_train_xdata), size = M_range[k], replace = False)]

                model = Network(D_train_xdata, miu, p, D_train_ydata) #

                kernel = rbf_kernel(D_val_xdata, miu, p) # 1000  3000
                predict = model.predict(kernel)
                MSE = mean_squared_error(D_val_ydata, predict)
                val_history.append(np.sqrt(MSE))

                kernel_train = rbf_kernel(D_train_xdata, miu, p)
                predict_train = model.predict(kernel_train)
                MSE_train = mean_squared_error(D_train_ydata, predict_train)
                train_history.append(np.sqrt(MSE_train))

            # Calculate the mean value of each gamma
            RMSE_train_list[k][idx] = np.mean(train_history)
            RMSE_val_list[k][idx] = np.mean(val_history)
            RMSE_train_std[k][idx] = np.std(train_history)
            RMSE_val_std[k][idx] = np.std(val_history)
            idx += 1
    return gamma_range_M, RMSE_train_list, RMSE_val_list, RMSE_train_std, RMSE_val_std
```

[227]:
```
gamma_range_d, RMSE_train_list_d, RMSE_val_list_d, RMSE_train_std_d,␣
 ↪RMSE_val_std_d = model_selection_d(xdata_train, ydata_train)
```

```
Gamma range is [1.5e-03 1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02]
Gamma range is [3.e-03 3.e-02 3.e-01 3.e+00 3.e+01 3.e+02]
Gamma range is [5.e-03 5.e-02 5.e-01 5.e+00 5.e+01 5.e+02]
Gamma range is [1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02 1.5e+03]
Gamma range is [3.e-02 3.e-01 3.e+00 3.e+01 3.e+02 3.e+03]
```

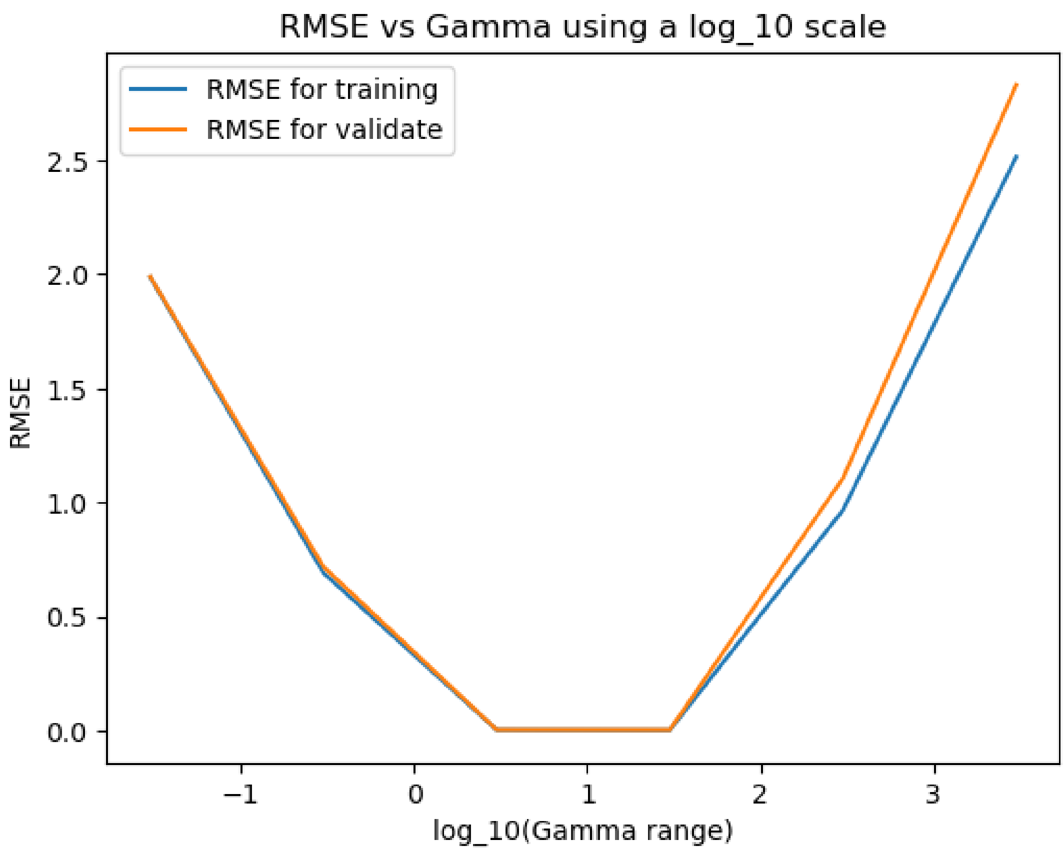
[300]:
```
# RMSE_mean_train = np.mean(RMSE_train_list)
# RMSE_std_train = np.std(RMSE_train_list)
# RMSE_mean_val = np.mean(RMSE_val_list, axis = 1)
# RMSE_std_val = np.std(RMSE_val_list, axis = 1)



print(f'The mean of train RMSE is {RMSE_train_list_d}')
print(f'The mean of validatation RMSE is {RMSE_val_list_d}')
print(f'The std of train RMSE is {RMSE_train_std_d}')
print(f'The std of validatation RMSE is {RMSE_val_std_d}')


# print(np.argmin(RMSE_val_list))
print(np.asarray(gamma_range_d))
best_idx = 4
M_range = [30, 60, 100, 300, 600]
# gamma_range_array = np.asarray(gamma_range_d).reshape(-1)
print(f'The best gamma is {gamma_range_array[np.argmin(RMSE_train_list_d)]},␣
 ↪the M = {M_range[best_idx]}')
```

```
The mean of train RMSE is [[4.45179750e+00 1.91465386e+00 1.60695655e+00
1.59816478e+00
  1.71101813e+00 2.83645023e+00]
 [2.28085926e+00 1.59204191e+00 1.15356872e+00 8.10422413e-01
  1.30890640e+00 2.72435322e+00]
 [3.33697895e+00 1.44370894e+00 5.96414971e-01 2.14450588e-01
  1.06422059e+00 2.79012141e+00]
 [3.35493428e+00 1.17856910e+00 4.10191996e-02 4.47287837e-03
  9.15911706e-01 2.70222241e+00]
 [2.30346963e+00 6.97067038e-01 2.48271892e-03 1.79378026e-03
  8.75490450e-01 2.52306720e+00]]
The mean of validatation RMSE is [[4.44080402e+00 1.95377118e+00 1.61148995e+00
1.60435959e+00
  1.71059315e+00 2.83699548e+00]
 [2.32321219e+00 1.64184933e+00 1.19648210e+00 8.37349842e-01
  1.31558598e+00 2.78517231e+00]
 [3.34377336e+00 1.45976893e+00 6.24635057e-01 2.23584693e-01
  1.08798621e+00 2.83303000e+00]
```

```
 [3.35234670e+00 1.20668983e+00 4.53877066e-02 5.75480131e-03
  1.01493664e+00 2.81418989e+00]
 [2.28566428e+00 7.30729809e-01 3.41404257e-03 2.97683835e-03
  1.04051149e+00 2.81379265e+00]]
The std of train RMSE is [[1.72824176e+00 3.90227858e-02 7.67917005e-02
4.56578670e-02
  1.29773013e-01 8.35803005e-02]
 [1.17106239e-01 6.31297122e-02 1.08666735e-01 4.46803929e-02
  2.28276814e-01 6.06963467e-02]
 [8.20273880e-01 1.33862825e-02 3.63397965e-02 9.78927784e-02
  1.66986569e-01 6.44855648e-03]
 [7.45362729e-01 6.69303116e-03 1.50332668e-03 6.51992364e-04
  1.29462761e-01 2.42484940e-02]
 [5.35818957e-01 1.61335341e-02 1.03529954e-04 2.62566146e-04
  1.04751933e-01 3.33667505e-02]]
The std of validatation RMSE is [[1.70213297e+00 6.18803269e-02 8.47780020e-02
3.27101955e-02
  1.23407094e-01 4.85061256e-02]
 [1.04444245e-01 7.89385778e-02 1.22902386e-01 6.58137238e-02
  2.27517650e-01 4.97531895e-02]
 [7.92319193e-01 5.41196716e-02 3.16149611e-02 1.03746207e-01
  1.54592931e-01 5.17729150e-02]
 [7.52317193e-01 1.01550063e-02 2.20893959e-03 1.01218304e-03
  1.07575213e-01 8.15113919e-02]
 [4.77663186e-01 1.83054271e-02 4.17104497e-04 8.32066505e-04
  1.72934319e-01 3.20417258e-02]]
[[1.5e-03 1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02]
 [3.0e-03 3.0e-02 3.0e-01 3.0e+00 3.0e+01 3.0e+02]
 [5.0e-03 5.0e-02 5.0e-01 5.0e+00 5.0e+01 5.0e+02]
 [1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02 1.5e+03]
 [3.0e-02 3.0e-01 3.0e+00 3.0e+01 3.0e+02 3.0e+03]]
The best gamma is 30.0, the M = 600
```

[224]:

```
plot_RMSE_gamma(RMSE_train_list_d[best_idx],RMSE_val_list_d[best_idx], np.
 ↪asarray(gamma_range_d)[best_idx])
```



[244]:
```
def RCC_d(xdata, ydata):
    M_range = [30, 60, 100, 300, 600]
    RMSE_train_list = np.zeros([len(M_range), 6])
    RMSE_val_list = np.zeros([len(M_range), 6])
    RMSE_train_std = np.zeros([len(M_range), 6])
    RMSE_val_std = np.zeros([len(M_range), 6])

    checkpoint = RMSE_y(ydata)

    gamma_range_M = []

    for k in range(len(M_range)):

        gamma = gamma_d(M_range[k])
        gamma_range = np.array([0.01, 0.1, 1, 10, 100, 1000]) * np.array(gamma)
        print(f'Gamma range is {gamma_range}')
        gamma_range_M.append(gamma_range)
        idx = 0 # idx in range of 6
        for p in gamma_range:
            # Define the cross-validation object
```

```
            cv = KFold(n_splits=4)
#             train_history = []
            val_history = []

            for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i␣
↪in range of 4
                D_train_xdata = xdata[train_index]
                D_train_ydata = ydata[train_index]
                D_val_xdata = xdata[val_index]
                D_val_ydata = ydata[val_index]

                miu = D_train_xdata[np.random.choice(len(D_train_xdata), size =␣
↪M_range[k], replace = False)]

                model = Network(D_train_xdata, miu, p, D_train_ydata)

                kernel = rbf_kernel(D_val_xdata, miu, p) # 1000  3000
                predict = model.predict(kernel)
                MSE = mean_squared_error(D_val_ydata, predict)
                val_history.append(np.sqrt(MSE))

#                 kernel_train = rbf_kernel(D_train_xdata, miu, p)
#                 predict_train = model.predict(kernel_train)
#                 MSE_train = mean_squared_error(D_train_ydata, predict_train)
#                 train_history.append(np.sqrt(MSE_train))

            # Calculate the mean value of each gamma
            if(np.mean(val_history) < checkpoint / 10):
                return M_range[k], p
```

[245]:
```
M_rcc_d, gamma_rcc_d = RCC_d(xdata_train, ydata_train)
```

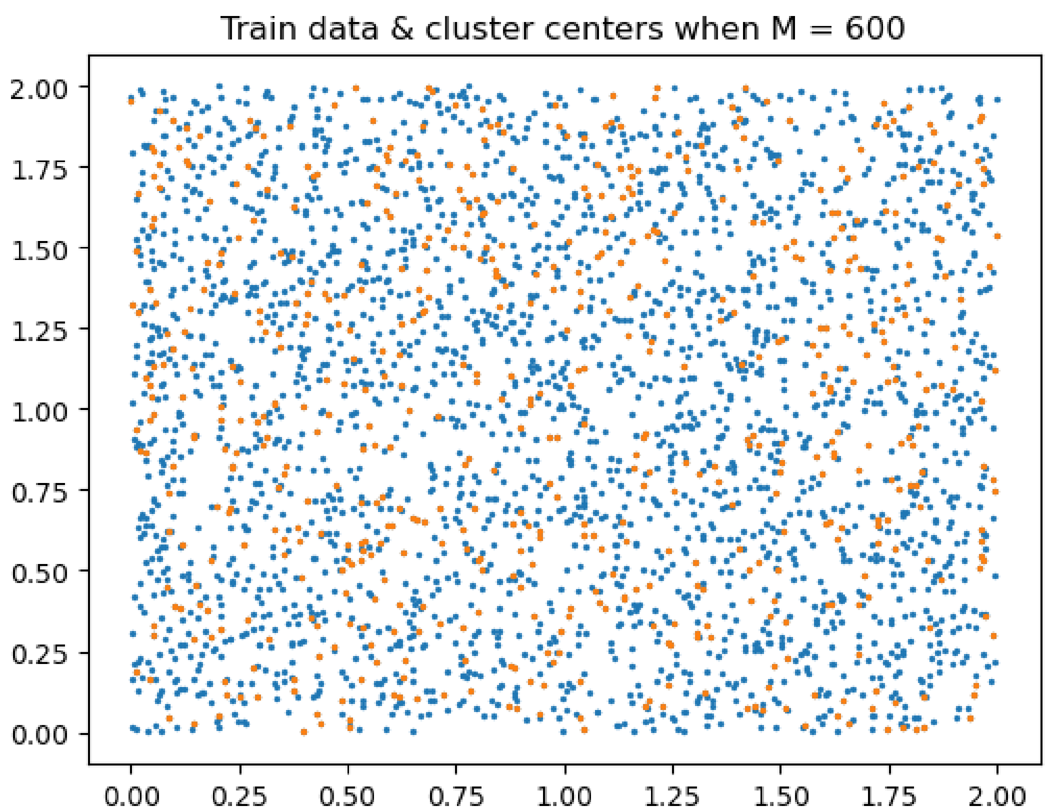
```
Gamma range is [1.5e-03 1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02]
Gamma range is [3.e-03 3.e-02 3.e-01 3.e+00 3.e+01 3.e+02]
Gamma range is [5.e-03 5.e-02 5.e-01 5.e+00 5.e+01 5.e+02]
```

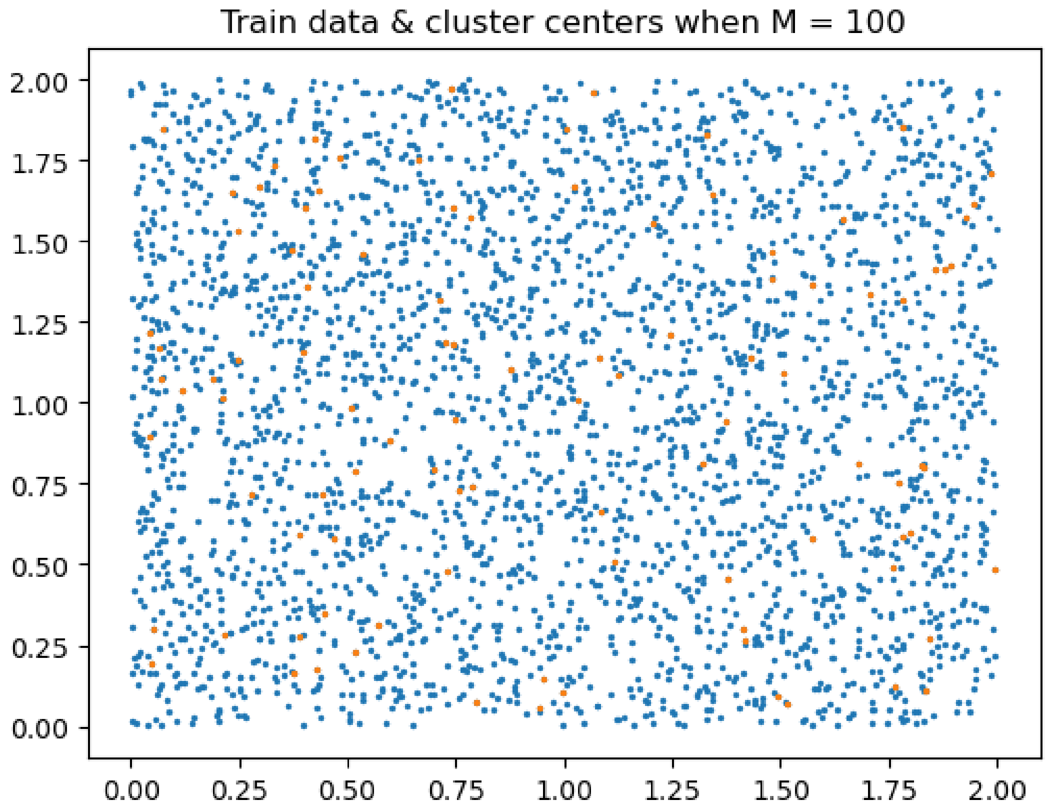
[246]:
```
print(f'The smallest M is {M_rcc_d}, the gamma = {gamma_rcc_d}')
```

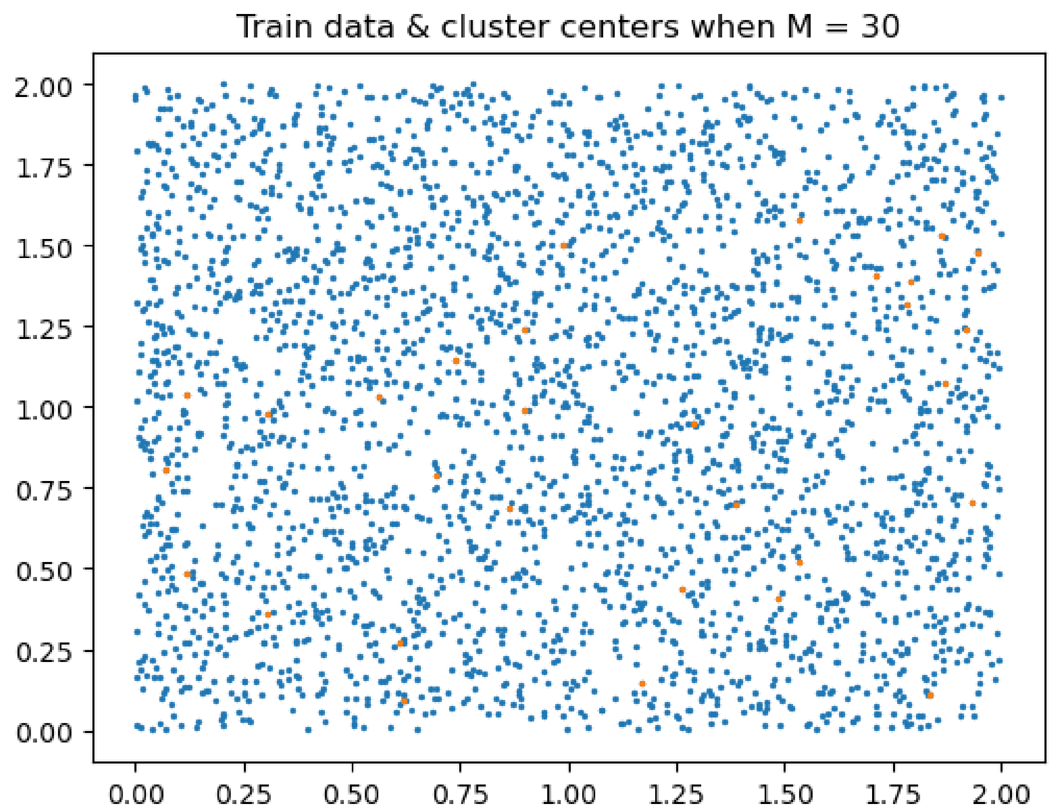
```
The smallest M is 100, the gamma = 5.0
```

[296]:
```
def plot_cluster(xdata, M, title):

    cv = KFold(n_splits=4)

    for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i in range␣
↪of 4
        D_train_xdata = xdata[train_index]
```

```
        miu = D_train_xdata[np.random.choice(len(D_train_xdata), size = M,␣
↪replace = False)]
        plt.scatter(D_train_xdata[:, 0], D_train_xdata[:, 1], s = 2, label =␣
↪'train data')
        plt.scatter(miu[:, 0], miu[:, 1], s = 2, label = 'basis function␣
↪centers')
        plt.title(title)
        plt.show()
        return

plot_cluster(xdata_train, 600, 'Train data & cluster centers when M = 600')
plot_cluster(xdata_train, 100, 'Train data & cluster centers when M = 100')
plot_cluster(xdata_train, 30, 'Train data & cluster centers when M = 30')
```

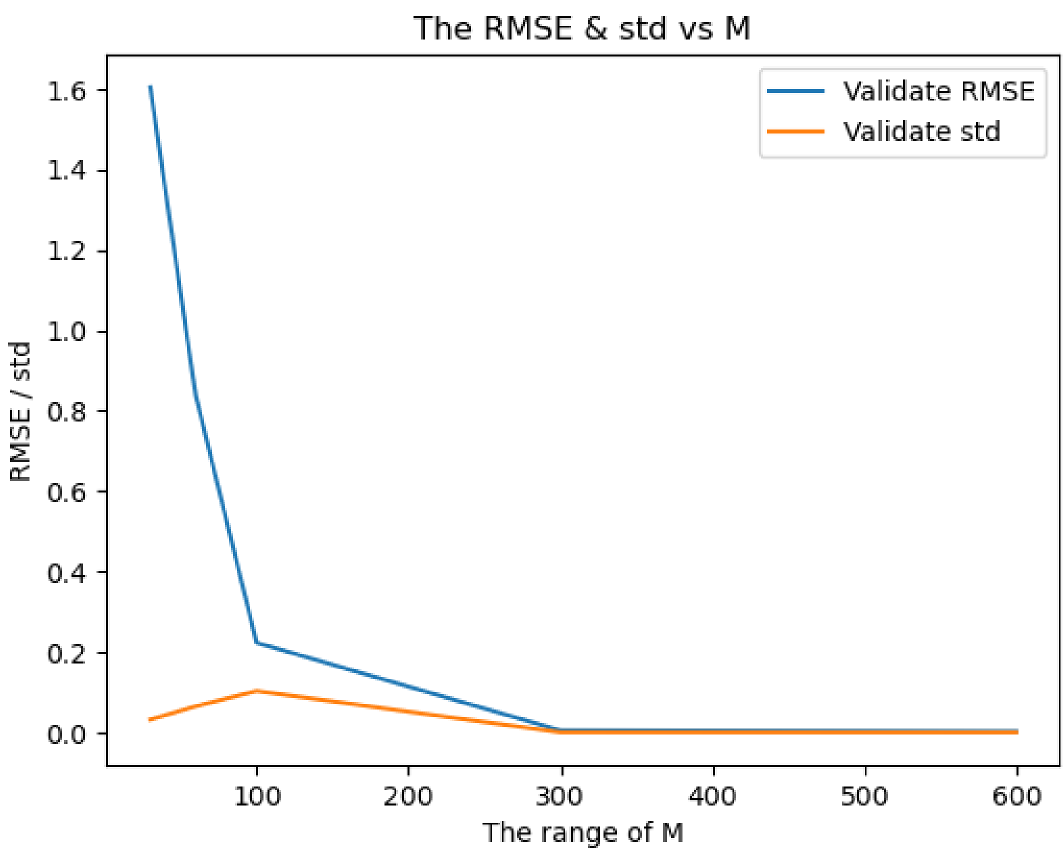



Train data & cluster centers when M = 100
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00



[309]:
```python
def plot_M_std(RMSE_val_list,RMSE_val_std):
    M_range = [30, 60, 100, 300, 600]
    mean = []
    std = []
    idx = 0
    for item in np.argmin(RMSE_val_list, axis = 1):
        mean.append(RMSE_val_list[idx][item])
        idx+=1

    idx = 0
    for item in np.argmin(RMSE_val_list, axis = 1):
        std.append(RMSE_val_std[idx][item])
        idx+=1
#     plt.plot(M_range, RMSE_val_list[:,np.argmin(RMSE_val_list, axis = 1)],␣
 ↪label = 'Validate RMSE')
#     plt.plot(M_range, RMSE_val_std[:,np.argmin(RMSE_val_list, axis = 1)],␣
 ↪label = 'Validate std')
    plt.plot(M_range, mean, label = 'Validate RMSE')
    plt.plot(M_range, std, label = 'Validate std')
    plt.legend()
    plt.xlabel("The range of M")
```

```
    plt.ylabel("RMSE / std")
    plt.title("The RMSE & std vs M")
    plt.show()


plot_M_std(RMSE_val_list_d,RMSE_val_std_d)
```



0.3 Question e

Use K-means clustering to choose basis function centers for a given K; vary K using model selection (e.g., use values 30, 60, 100, 300, 600). For each value of K, choose your initial cluster centers randomly (i.e., in sklearn's K-means).

[313]:
```
def model_selection_e(xdata, ydata):
    K_range = [30, 60, 100, 300, 600]
    RMSE_train_list = np.zeros([len(K_range), 6])
    RMSE_val_list = np.zeros([len(K_range), 6])
    RMSE_train_std = np.zeros([len(K_range), 6])
    RMSE_val_std = np.zeros([len(K_range), 6])
```

```python
    gamma_range_K = []

    for k in range(len(K_range)):

        gamma = gamma_d(K_range[k])
        gamma_range = np.array([0.01, 0.1, 1, 10, 100, 1000]) * np.array(gamma)
        print(f'Gamma range is {gamma_range}')
        gamma_range_K.append(gamma_range)

        idx = 0 # idx in range of 6
        for p in gamma_range:
            # Define the cross-validation object
            cv = KFold(n_splits=4)
            train_history = []
            val_history = []

            for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i in range of 4
                D_train_xdata = xdata[train_index]
                D_train_ydata = ydata[train_index]
                D_val_xdata = xdata[val_index]
                D_val_ydata = ydata[val_index]

                kmeans = KMeans(n_clusters=K_range[k], init="random").fit(D_train_xdata)

#                 miu = D_train_xdata[np.random.choice(len(D_train_xdata), size = K_range[k], replace = False)]
                miu = kmeans.cluster_centers_

                model = Network(D_train_xdata, miu, p, D_train_ydata)

                kernel = rbf_kernel(D_val_xdata, miu, p) # 1000  3000
                predict = model.predict(kernel)
                MSE = mean_squared_error(D_val_ydata, predict)
                val_history.append(np.sqrt(MSE))

                kernel_train = rbf_kernel(D_train_xdata, miu, p)
                predict_train = model.predict(kernel_train)
                MSE_train = mean_squared_error(D_train_ydata, predict_train)
                train_history.append(np.sqrt(MSE_train))

            # Calculate the mean value of each gamma
            RMSE_train_list[k][idx] = np.mean(train_history)
            RMSE_val_list[k][idx] = np.mean(val_history)
            RMSE_train_std[k][idx] = np.std(train_history)
            RMSE_val_std[k][idx] = np.std(val_history)
```

```
            idx += 1
    return gamma_range_K, RMSE_train_list, RMSE_val_list, RMSE_train_std,␣
↪RMSE_val_std
```

[314]:
```
gamma_range_e, RMSE_train_list_e, RMSE_val_list_e, RMSE_train_std_e,␣
↪RMSE_val_std_e = model_selection_e(xdata_train, ydata_train)
```

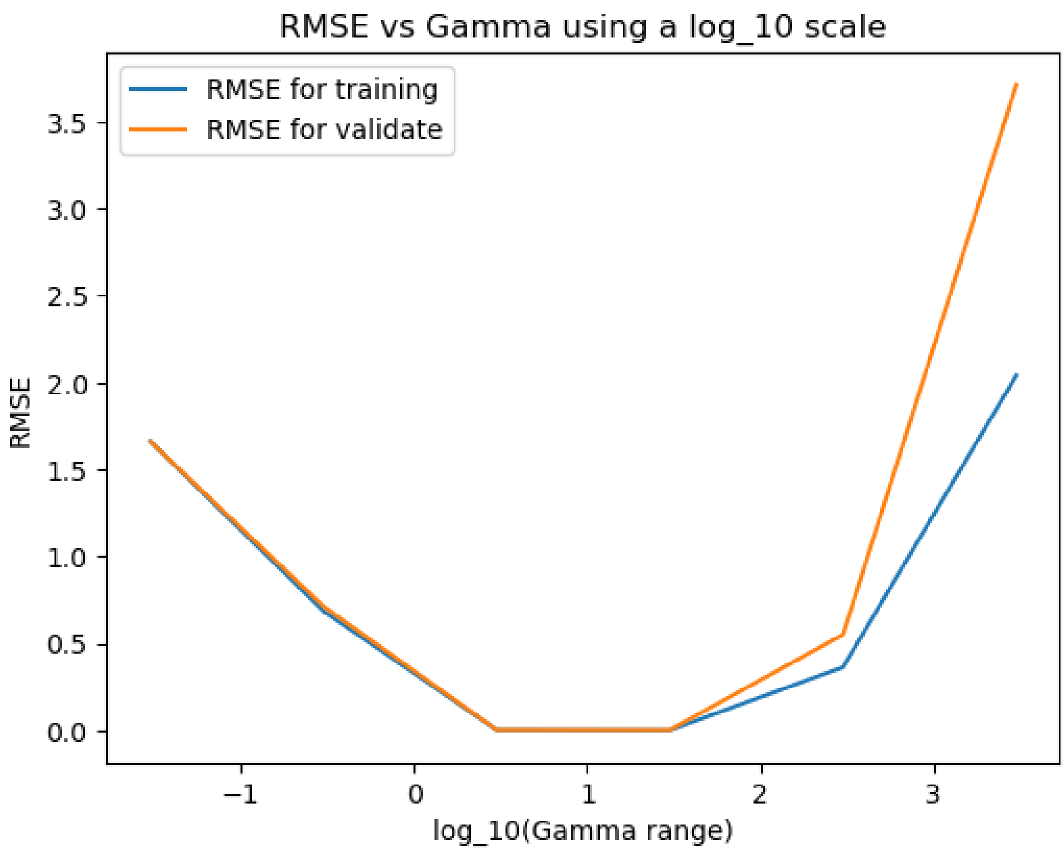
```
Gamma range is [1.5e-03 1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02]
Gamma range is [3.e-03 3.e-02 3.e-01 3.e+00 3.e+01 3.e+02]
Gamma range is [5.e-03 5.e-02 5.e-01 5.e+00 5.e+01 5.e+02]
Gamma range is [1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02 1.5e+03]
Gamma range is [3.e-02 3.e-01 3.e+00 3.e+01 3.e+02 3.e+03]
```

[318]:
```
print(f'The mean of train RMSE is {RMSE_train_list_e}')
print(f'The mean of validatation RMSE is {RMSE_val_list_e}')
print(f'The std of train RMSE is {RMSE_train_std_e}')
print(f'The std of validatation RMSE is {RMSE_val_std_e}')



print(np.argmin(RMSE_val_list_e))
print(np.asarray(gamma_range_e))
best_idx_k = 4
K_range = [30, 60, 100, 300, 600]
# gamma_range_array = np.asarray(gamma_range_d).reshape(-1)
print(f'The best gamma is {gamma_range_array[np.argmin(RMSE_train_list_e)]},␣
↪the K = {K_range[best_idx_k]}')
```

```
The mean of train RMSE is [[3.17663870e+00 2.28743449e+00 1.50812954e+00
1.62213280e+00
  1.70737407e+00 2.77255375e+00]
 [2.30468917e+00 1.75047769e+00 1.13160138e+00 8.06278503e-01
  7.71688629e-01 2.59907064e+00]
 [4.07421453e+00 1.43521740e+00 5.25773302e-01 1.61589376e-01
  4.50620011e-01 2.56324929e+00]
 [2.45798520e+00 1.21651572e+00 3.84991032e-02 3.95311604e-03
  3.59920142e-01 2.36214077e+00]
 [1.66095430e+00 6.85655228e-01 2.46499798e-03 1.22911695e-03
  3.62813619e-01 2.03885833e+00]]
The mean of validatation RMSE is [[3.16923859e+00 2.33095153e+00 1.51877461e+00
1.63575688e+00
  1.73887304e+00 2.86843057e+00]
 [2.34332994e+00 1.79674960e+00 1.16123748e+00 8.21192046e-01
  8.03159811e-01 2.68114546e+00]
 [4.08728352e+00 1.45649433e+00 5.65270843e-01 1.70310210e-01
  4.75244767e-01 2.71260080e+00]
 [2.49456692e+00 1.26578002e+00 4.32094314e-02 5.35701314e-03
  4.55467526e-01 2.86491319e+00]
```

```
 [1.66039080e+00 7.12829752e-01 3.37635740e-03 2.38184368e-03
  5.50008578e-01 3.70961585e+00]]
The std of train RMSE is [[2.43479272e-01 2.46855778e-01 1.33805483e-02
2.99488670e-03
  1.96617659e-02 3.45327146e-02]
 [1.09703059e-01 1.15534467e-01 8.74993272e-03 3.77784681e-02
  1.00069581e-01 3.70031521e-02]
 [1.34794150e+00 1.42203615e-02 2.92377730e-02 5.30098708e-03
  3.66860139e-02 1.54671647e-02]
 [6.95814156e-01 7.67864105e-02 6.21501083e-04 6.06786850e-04
  1.44977589e-02 4.53391195e-02]
 [4.80972662e-02 1.35806481e-02 6.87602201e-05 6.64829953e-05
  1.72432214e-02 5.87998623e-02]]
The std of validatation RMSE is [[2.13486901e-01 2.62599828e-01 2.38942312e-02
2.76452448e-02
  1.45549403e-02 4.35274540e-02]
 [1.05690485e-01 1.31306196e-01 1.74600032e-02 3.65774222e-02
  1.05154343e-01 6.27359836e-02]
 [1.35627427e+00 2.98839605e-02 1.04921575e-02 5.03146150e-03
  3.58754750e-02 4.33283289e-02]
 [7.10305241e-01 9.98095365e-02 1.51999187e-03 1.79993269e-03
  1.24710309e-02 1.98708682e-01]
 [6.37824637e-02 1.01025377e-02 4.06156840e-04 2.95467462e-04
  3.11428625e-02 4.36848908e-01]]
27
[[1.5e-03 1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02]
 [3.0e-03 3.0e-02 3.0e-01 3.0e+00 3.0e+01 3.0e+02]
 [5.0e-03 5.0e-02 5.0e-01 5.0e+00 5.0e+01 5.0e+02]
 [1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02 1.5e+03]
 [3.0e-02 3.0e-01 3.0e+00 3.0e+01 3.0e+02 3.0e+03]]
The best gamma is 30.0, the K = 600
```

[319]:
```python
plot_RMSE_gamma(RMSE_train_list_e[best_idx_k],RMSE_val_list_e[best_idx_k], np.
  ↪asarray(gamma_range_e)[best_idx_k])
```



[320]:
```
def RCC_e(xdata, ydata):
    K_range = [30, 60, 100, 300, 600]
#     RMSE_train_list = np.zeros([len(M_range), 6])
#     RMSE_val_list = np.zeros([len(M_range), 6])
#     RMSE_train_std = np.zeros([len(M_range), 6])
#     RMSE_val_std = np.zeros([len(M_range), 6])

    checkpoint = RMSE_y(ydata)

#     gamma_range_K = []

    for k in range(len(K_range)):

        gamma = gamma_d(M_range[k])
        gamma_range = np.array([0.01, 0.1, 1, 10, 100, 1000]) * np.array(gamma)
        print(f'Gamma range is {gamma_range}')
#         gamma_range_M.append(gamma_range)
        idx = 0 # idx in range of 6
        for p in gamma_range:
            # Define the cross-validation object
```

```
            cv = KFold(n_splits=4)
#             train_history = []
            val_history = []

            for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i␣
↪in range of 4
                D_train_xdata = xdata[train_index]
                D_train_ydata = ydata[train_index]
                D_val_xdata = xdata[val_index]
                D_val_ydata = ydata[val_index]

                kmeans = KMeans(n_clusters=K_range[k], init="random").
↪fit(D_train_xdata)

#                 miu = D_train_xdata[np.random.choice(len(D_train_xdata), size␣
↪= K_range[k], replace = False)]
                miu = kmeans.cluster_centers_
                model = Network(D_train_xdata, miu, p, D_train_ydata)

                kernel = rbf_kernel(D_val_xdata, miu, p) # 1000  3000
                predict = model.predict(kernel)
                MSE = mean_squared_error(D_val_ydata, predict)
                val_history.append(np.sqrt(MSE))

#                 kernel_train = rbf_kernel(D_train_xdata, miu, p)
#                 predict_train = model.predict(kernel_train)
#                 MSE_train = mean_squared_error(D_train_ydata, predict_train)
#                 train_history.append(np.sqrt(MSE_train))

            # Calculate the mean value of each gamma
            if(np.mean(val_history) < checkpoint / 10):
                return K_range[k], p
```

[321]:
```
K_rcc_e, gamma_rcc_e = RCC_e(xdata_train, ydata_train)
```

```
Gamma range is [1.5e-03 1.5e-02 1.5e-01 1.5e+00 1.5e+01 1.5e+02]
Gamma range is [3.e-03 3.e-02 3.e-01 3.e+00 3.e+01 3.e+02]
Gamma range is [5.e-03 5.e-02 5.e-01 5.e+00 5.e+01 5.e+02]
```

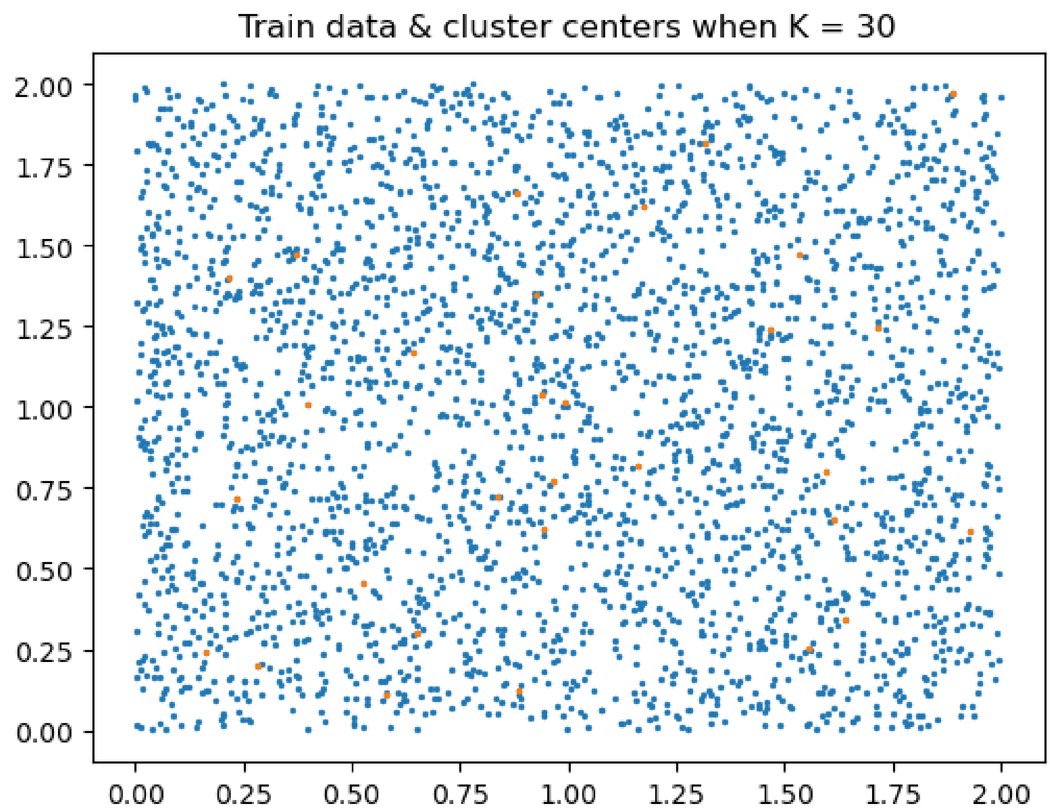
[322]:
```
print(f'The smallest K is {K_rcc_e}, the gamma = {gamma_rcc_e}')
```

```
The smallest K is 100, the gamma = 5.0
```

[323]:
```
def plot_cluster_e(xdata, K, title):

    cv = KFold(n_splits=4)
```

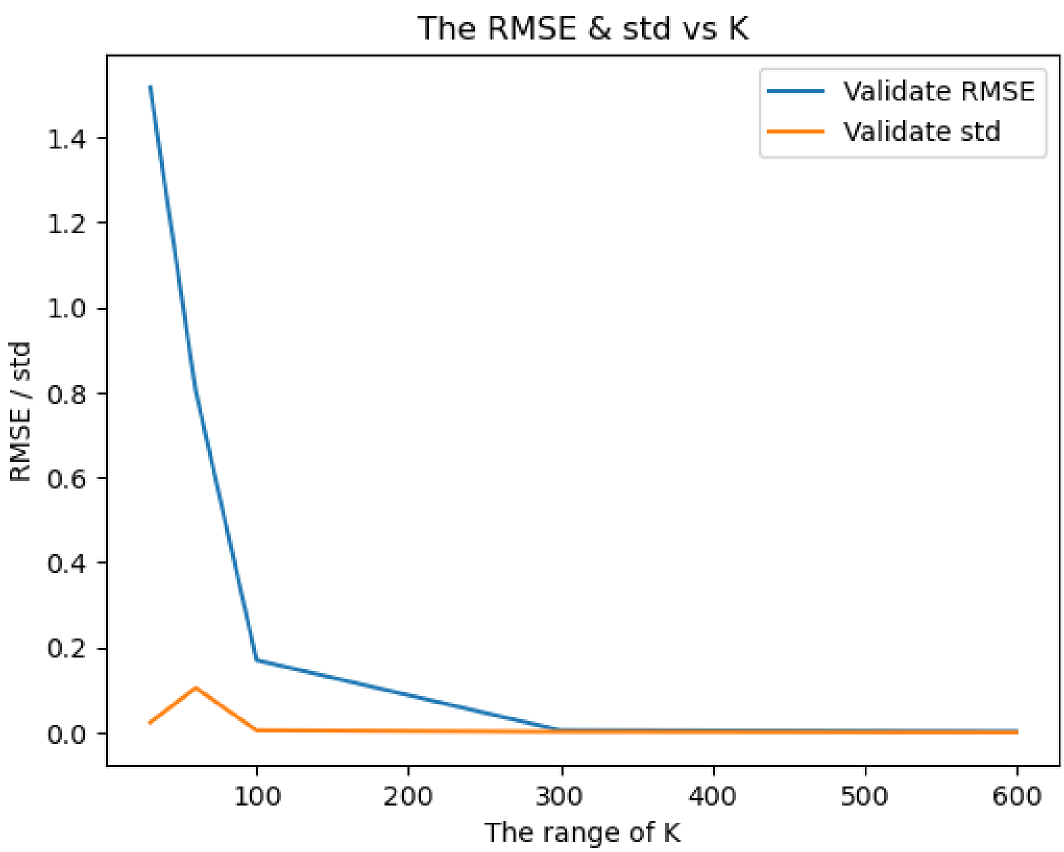
```
    for i, (train_index, val_index) in enumerate(cv.split(xdata)): # i in range
 ↪of 4
        D_train_xdata = xdata[train_index]

        kmeans = KMeans(n_clusters=K_range[k], init="random").fit(D_train_xdata)

        miu = kmeans.cluster_centers_
        plt.scatter(D_train_xdata[:, 0], D_train_xdata[:, 1], s = 2, label =
 ↪'train data')
        plt.scatter(miu[:, 0], miu[:, 1], s = 2, label = 'basis function
 ↪centers')
        plt.title(title)
        plt.show()
        return

plot_cluster(xdata_train, 600, 'Train data & cluster centers when K = 600')
plot_cluster(xdata_train, 100, 'Train data & cluster centers when K = 100')
plot_cluster(xdata_train, 30, 'Train data & cluster centers when K = 30')
```


Train data & cluster centers when K = 100
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00



[324]:
```python
def plot_K_std(RMSE_val_list,RMSE_val_std):
    K_range = [30, 60, 100, 300, 600]
    mean = []
    std = []
    idx = 0
    for item in np.argmin(RMSE_val_list, axis = 1):
        mean.append(RMSE_val_list[idx][item])
        idx+=1

    idx = 0
    for item in np.argmin(RMSE_val_list, axis = 1):
        std.append(RMSE_val_std[idx][item])
        idx+=1
#     plt.plot(M_range, RMSE_val_list[:,np.argmin(RMSE_val_list, axis = 1)],␣
 ↪label = 'Validate RMSE')
#     plt.plot(M_range, RMSE_val_std[:,np.argmin(RMSE_val_list, axis = 1)],␣
 ↪label = 'Validate std')
    plt.plot(K_range, mean, label = 'Validate RMSE')
    plt.plot(K_range, std, label = 'Validate std')
    plt.legend()
    plt.xlabel("The range of K")
```

```
    plt.ylabel("RMSE / std")
    plt.title("The RMSE & std vs K")
    plt.show()


plot_K_std(RMSE_val_list_e,RMSE_val_std_e)
```



0.4 Question g

Run the best model from each of (c), (d), and (e); and run the RCC model of (d), (e), on your test set. Report the RMSE of each (5 models total).

[384]:
```
def test(xdata_train, ydata_train, xdata_test, ydata_test, miu, gamma):

    model = Network(xdata_train, miu, gamma, ydata_train)

    kernel = rbf_kernel(xdata_test, miu, gamma)
    predict = model.predict(kernel)
    MSE = mean_squared_error(ydata_test, predict)
    return np.sqrt(MSE)
```

[385]:
```python
miu_c = xdata_train
gamma_c = 15.0
RMSE_c = test(xdata_train, ydata_train, xdata_test, ydata_test, miu_c, gamma_c)
print(f'The RMSE of best model from c is {RMSE_c}')
```

```
The RMSE of best model from c is 2.5778315496757243e-07
```

[333]:
```python
miu_d = xdata_train[np.random.choice(len(xdata_train), size = 600, replace =
 ↪False)]
gamma_d = 30.0
RMSE_d = test(xdata_train, ydata_train, xdata_test, ydata_test, miu_d, gamma_d)
print(f'The RMSE of best model from d is {RMSE_d}')
```

```
The RMSE of best model from d is 0.0021838235900409615
```

[334]:
```python
kmeans_e = KMeans(n_clusters=600, init="random").fit(xdata_train)
miu_e = kmeans_e.cluster_centers_
gamma_e = 30.0
RMSE_e = test(xdata_train, ydata_train, xdata_test, ydata_test, miu_e, gamma_e)
print(f'The RMSE of best model from e is {RMSE_e}')
```

```
The RMSE of best model from e is 0.0018462935462773137
```

[335]:
```python
miu_d_rcc = xdata_train[np.random.choice(len(xdata_train), size = 100, replace
 ↪= False)]
gamma_d_rcc = 5.0
RMSE_d_rcc = test(xdata_train, ydata_train, xdata_test, ydata_test, miu_d_rcc,
 ↪gamma_d_rcc)
print(f'The RMSE of RCC model from d is {RMSE_d_rcc}')
```

```
The RMSE of RCC model from d is 0.17901963364551002
```

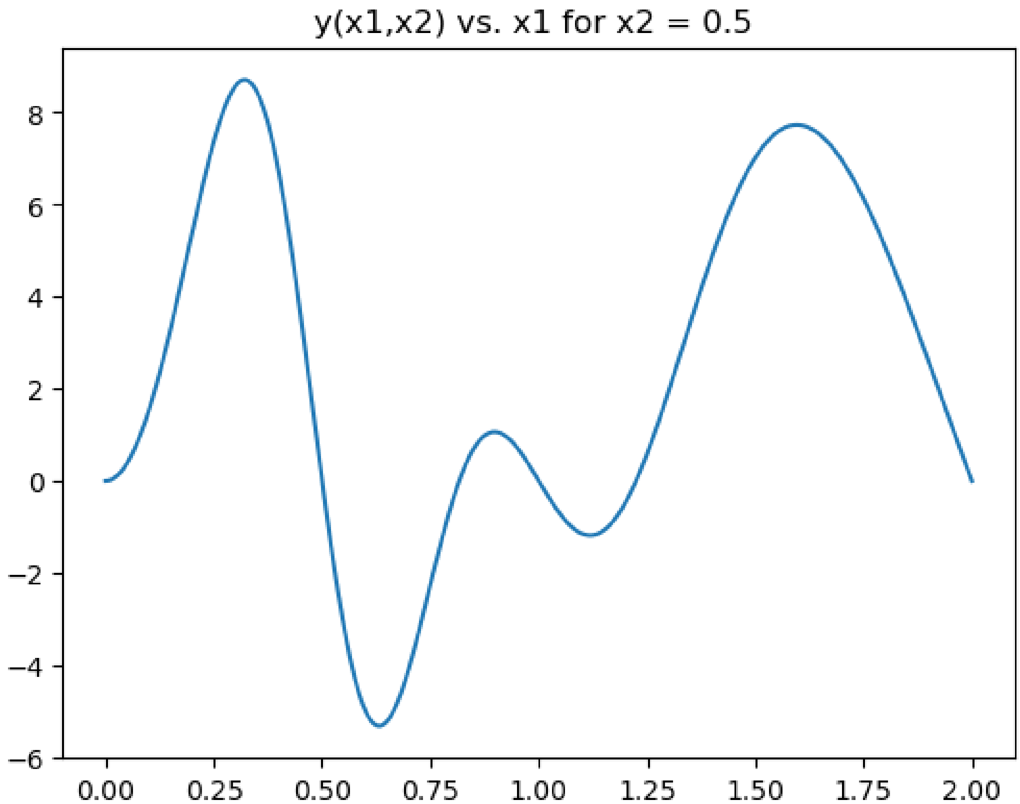
[336]:
```python
kmeans_e_rcc = KMeans(n_clusters=100, init="random").fit(xdata_train)
miu_e_rcc = kmeans_e_rcc.cluster_centers_
gamma_e_rcc = 5.0
RMSE_e_rcc = test(xdata_train, ydata_train, xdata_test, ydata_test, miu_e_rcc,
 ↪gamma_e_rcc)
print(f'The RMSE of RCC model from e is {RMSE_e_rcc}')
```

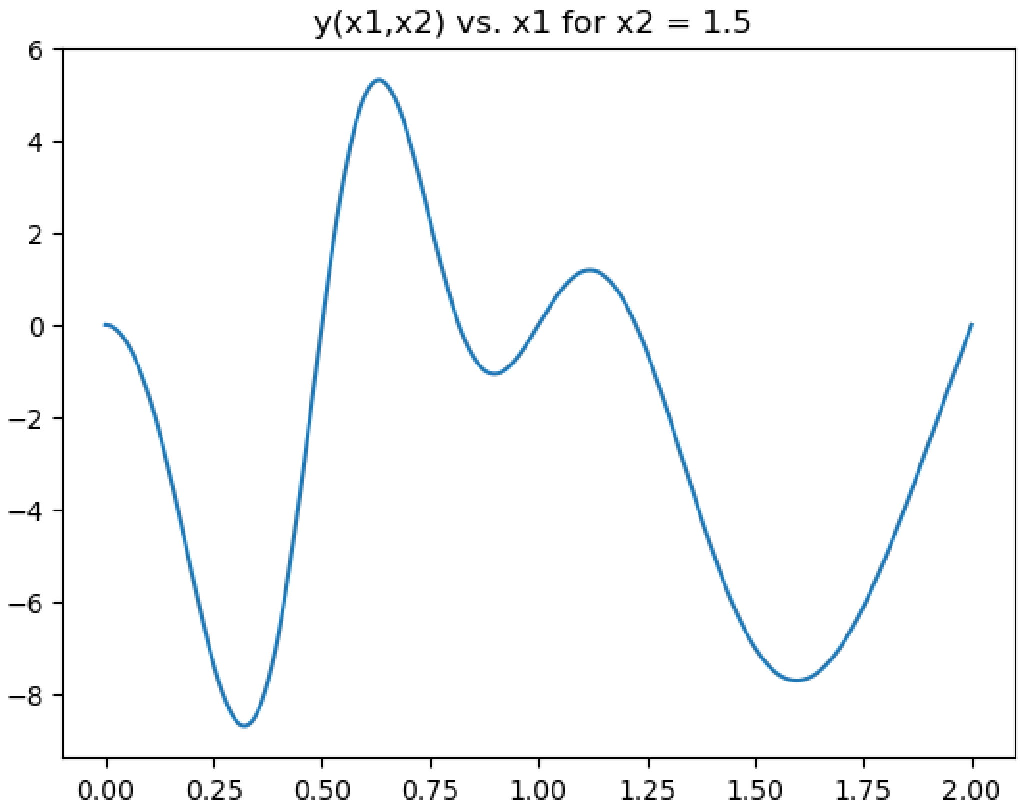
```
The RMSE of RCC model from e is 0.1661766513476608
```

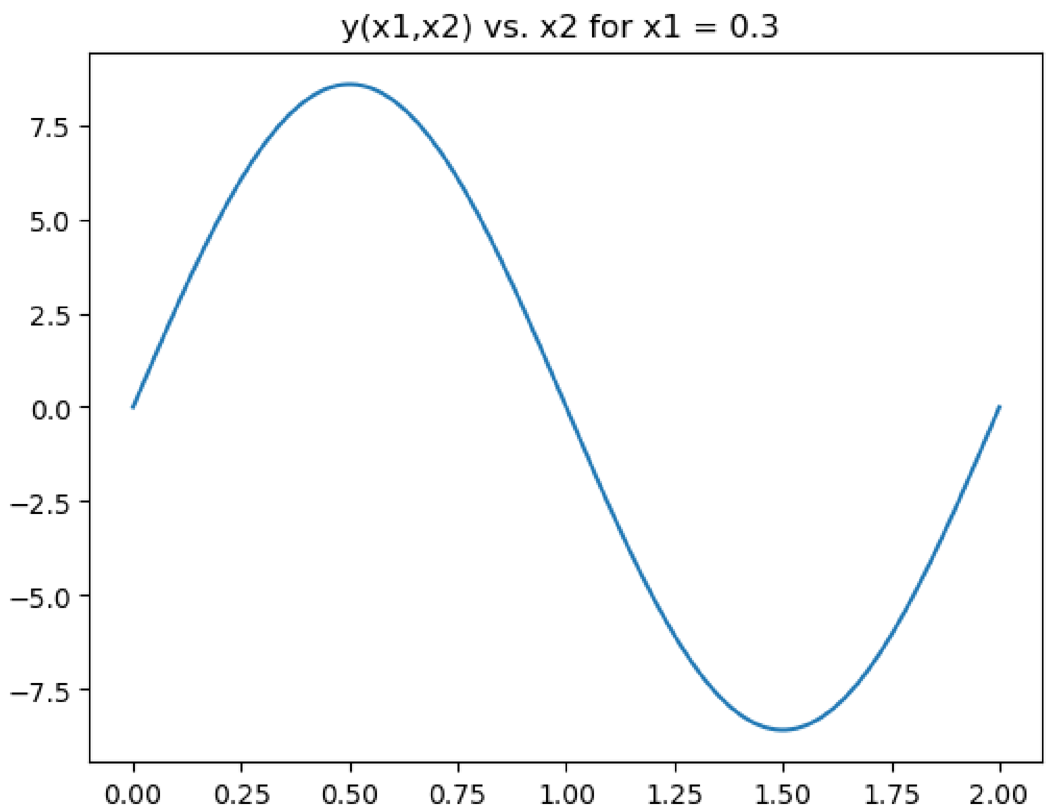
0.5 Question i

[340]:
```python
def target_function(x1, x2):
    return 10 * np.cos(np.pi / 2 * x1) * np.sin(5 * np.pi / (x1**2 + 1)) * np.
 ↪sin(np.pi * x2)
```

[375]:
```
def plot_target_function():
    x1 = np.linspace(0,2, 5000)
    x2 = 0.5

    y = target_function(x1, x2)
    plt.plot(x1, y)
    plt.title("y(x1,x2) vs. x1 for x2 = 0.5 ")
    plt.show()

    x2 = 1.5
    y = target_function(x1, x2)
    plt.plot(x1, y)
    plt.title("y(x1,x2) vs. x1 for x2 = 1.5 ")
    plt.show()

    x2 = np.linspace(0,2, 5000)
    x1 = 0.3
    y = target_function(x1, x2)
    plt.plot(x2, y)
    plt.title("y(x1,x2) vs. x2 for x1 = 0.3 ")
    plt.show()


plot_target_function()
```

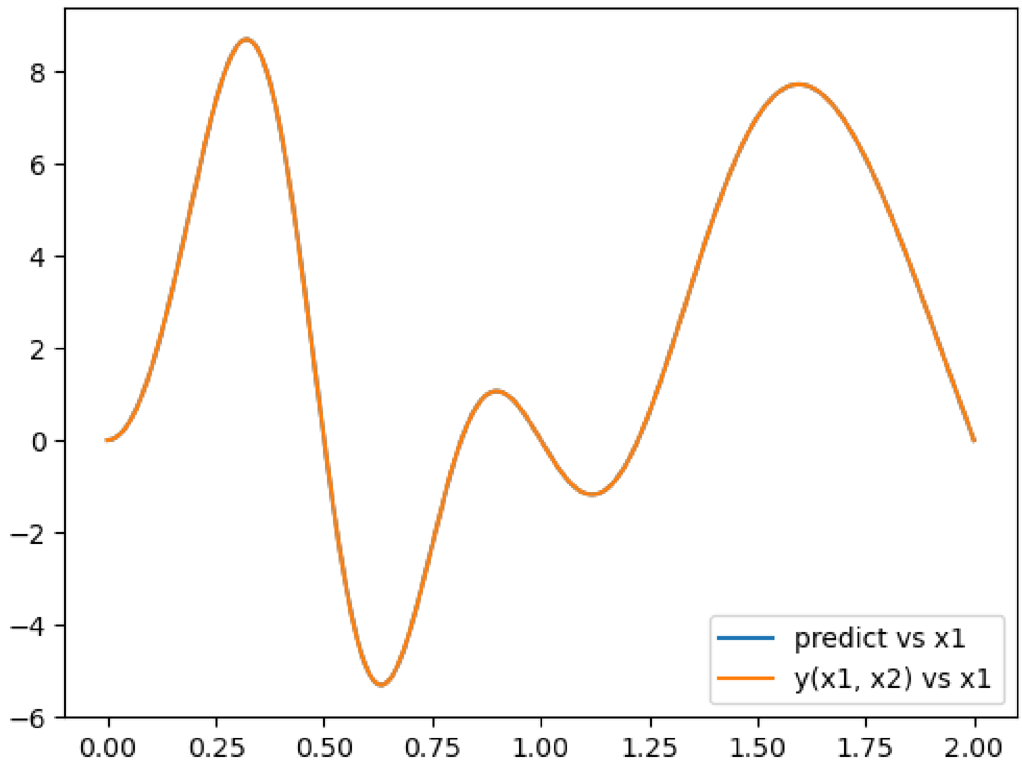

y(x1,x2) vs. x1 for x2 = 0.5
8
6
4
2
0
−2
−4
−6
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


y(x1,x2) vs. x1 for x2 = 1.5
6
4
2
0
−2
−4
−6
−8
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00



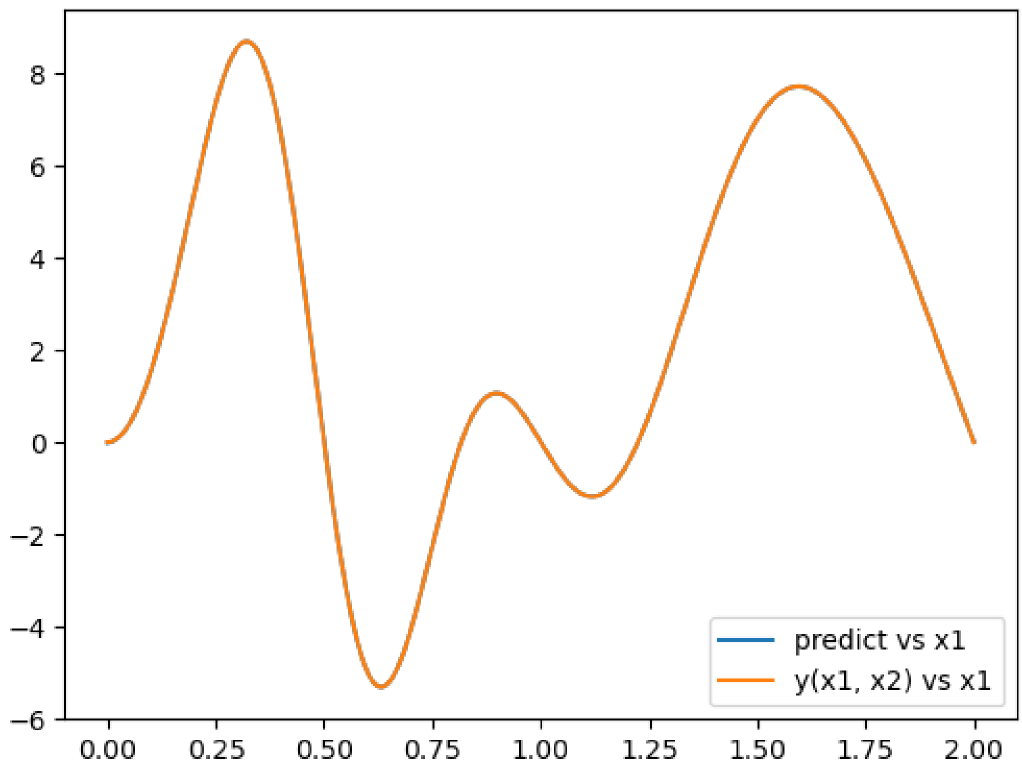
[378]:
```
def predict(xdata, ydata, miu, gamma ):
    model = Network(xdata, miu, gamma, ydata)
    x1 = np.linspace(0,2, 4000)
    x2 = 0.5
    dataset = np.column_stack((x1, np.full_like(x1, x2)))
#     print(dataset)
    kernel = rbf_kernel(dataset, miu, gamma)
    predict = model.predict(kernel)
    plt.plot(x1, predict, label = 'predict vs x1')

    y = target_function(x1, x2)
    plt.plot(x1, y, label = "y(x1, x2) vs x1")
    plt.legend()
    plt.show()
```

[379]:
```
predict(xdata_train, ydata_train, miu_c, gamma_c)
```

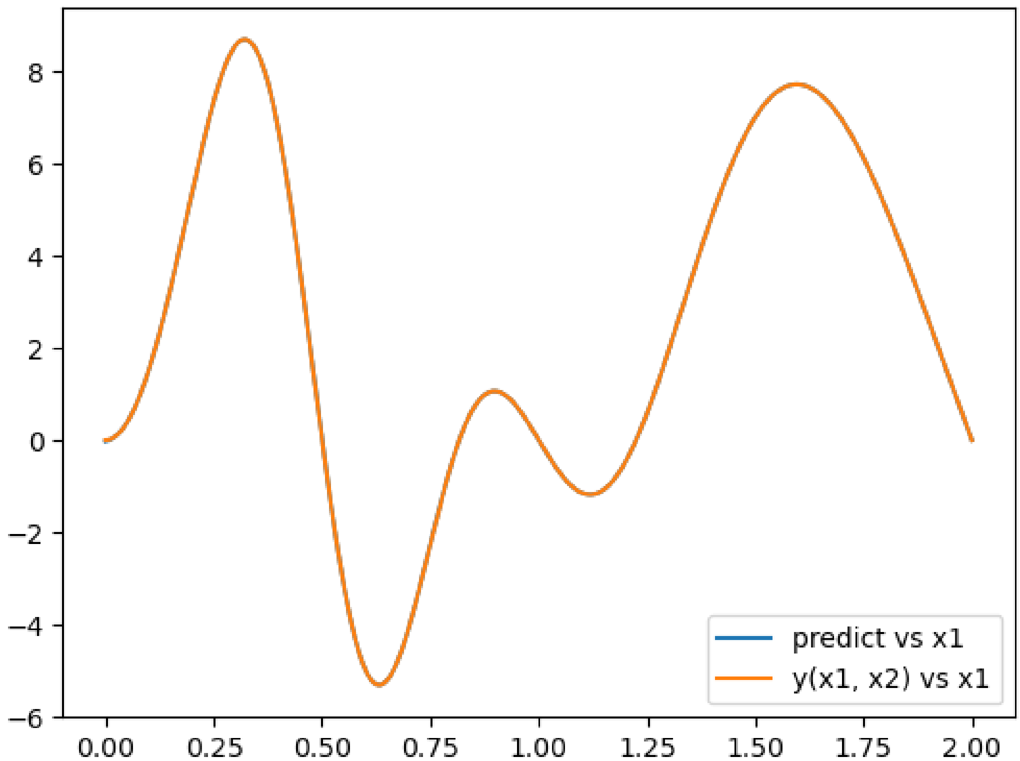


[371]:

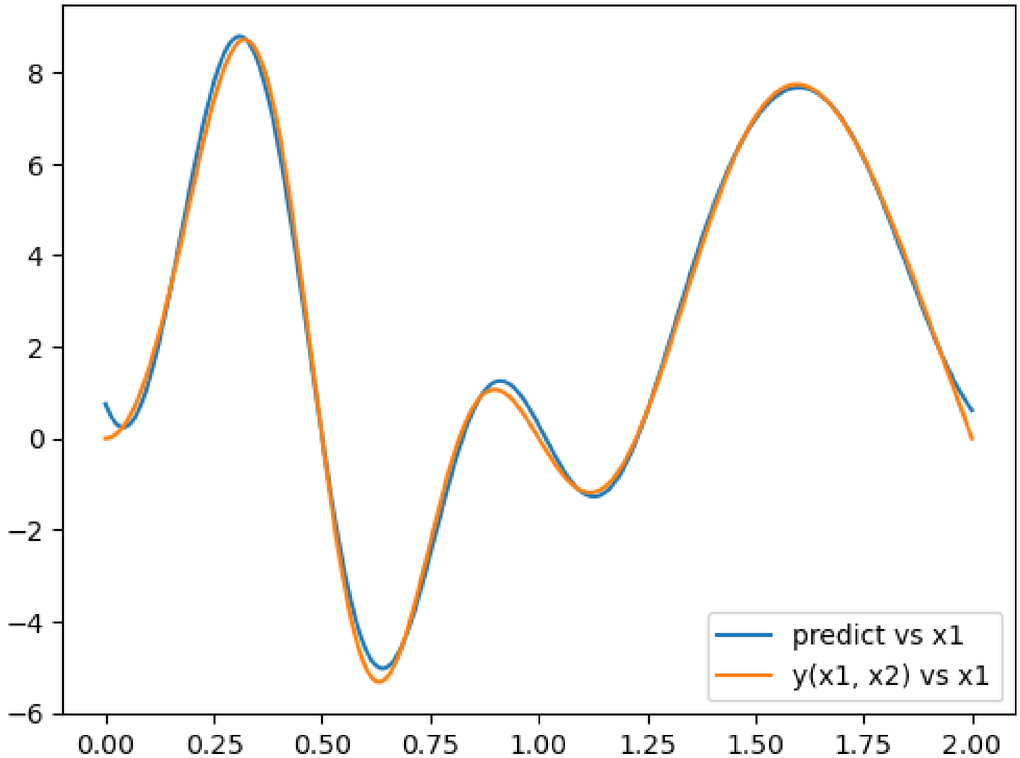
```
predict(xdata_train, ydata_train, miu_d, gamma_d)
```



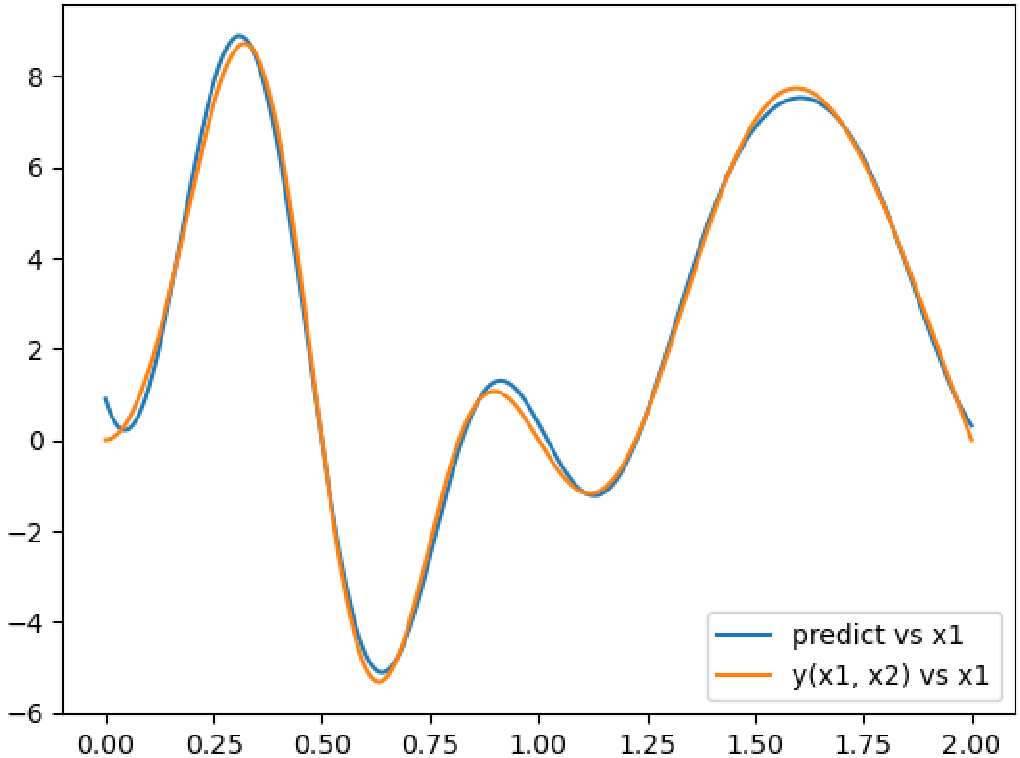
[372]:
```
predict(xdata_train, ydata_train, miu_e, gamma_e)
```




```
[373]: predict(xdata_train, ydata_train, miu_d_rcc, gamma_d_rcc)
```



[374]:
```
predict(xdata_train, ydata_train, miu_e_rcc, gamma_e_rcc)
```



8
6
4
2
0
−2
−4
−6
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
predict vs x1
y(x1, x2) vs x1


```
[ ]:
```