# HW5-Coding

January 7, 2024

## 1 Homework 5: Convolutional neural network (30 points)

In this part, you need to implement and train a convolutional neural network on the CIFAR-10 dataset with PyTorch. ### What is PyTorch?

PyTorch is a system for executing dynamic computational graphs over Tensor objects that behave similarly as numpy ndarray. It comes with a powerful automatic differentiation engine that removes the need for manual back-propagation.

#### 1.0.1 Why?

- Our code will now run on GPUs! Much faster training. When using a framework like PyTorch or TensorFlow you can harness the power of the GPU for your own custom neural network architectures without having to write CUDA code directly (which is beyond the scope of this class).
- We want you to be ready to use one of these frameworks for your project so you can experiment more efficiently than if you were writing every feature you want to use by hand.
- We want you to stand on the shoulders of giants! TensorFlow and PyTorch are both excellent frameworks that will make your lives a lot easier, and now that you understand their guts, you are free to use them :)
- We want you to be exposed to the sort of deep learning code you might run into in academia or industry. ## How can I learn PyTorch?

Justin Johnson has made an excellent tutorial for PyTorch.

You can also find the detailed API doc here. If you have other questions that are not addressed by the API docs, the PyTorch forum is a much better place to ask than StackOverflow.

Install PyTorch and Skorch.

[1]:
```
pip install -q torch skorch torchvision torchtext
```

[6]:
```
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import torchvision
import torchvision.transforms as transforms
import skorch
import sklearn
```

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

## 1.1 0. Tensor Operations (5 points)

Tensor operations are important in deep learning models. In this part, you are required to get famaliar to some common tensor operations in PyTorch.

### 1.1.1 1) Tensor squeezing, unsqueezing and viewing

Tensor squeezing, unsqueezing and viewing are important methods to change the dimension of a Tensor, and the corresponding functions are torch.squeeze, torch.unsqueeze and torch.Tensor.view. Please read the documents of the functions, and finish the following practice.

[3]:
```
# x is a tensor with size being (3, 2)
x = torch.Tensor([[1, 2],
                  [3, 4],
                  [5, 6]])
x.shape
# Add two new dimensions to x by using the function torch.unsqueeze, so that␣
 ↪the size of x becomes (3, 1, 2, 1).
x = torch.unsqueeze(torch.unsqueeze(x,1),-1)
print(x.shape)
# Remove the two dimensions justed added by using the function torch.squeeze,␣
 ↪and change the size of x back to (3, 2).
x = torch.squeeze(x)
print(x.shape)
# x is now a two-dimensional tensor, or in other words a matrix. Now use the␣
 ↪function torch.Tensor.view and change x to a one-dimensional vector with␣
 ↪size being (6).
x = x.view(6)
print(x.shape)
```

```
torch.Size([3, 1, 2, 1])
torch.Size([3, 2])
torch.Size([6])
```

### 1.1.2 2) Tensor concatenation and stack

Tensor concatenation and stack are operations to combine small tensors into big tensors. The corresponding functions are torch.cat and torch.stack. Please read the documents of the functions, and finish the following practice.

[4]:
```
# x is a tensor with size being (3, 2)
x = torch.Tensor([[1, 2], [3, 4], [5, 6]])

# y is a tensor with size being (3, 2)
```

```
y = torch.Tensor([[-1, -2], [-3, -4], [-5, -6]])

# Our goal is to generate a tensor z with size as (2, 3, 2), and z[0,:,:] = x,␣
 ↪z[1,:,:] = y.

# Use torch.stack to generate such a z
z = torch.stack((x,y))
print(z[0,:,:])
# Use torch.cat and torch.unsqueeze to generate such a z
z = torch.cat((torch.unsqueeze(x,0),torch.unsqueeze(y,0)))
print(z[1,:,:])
```

```
tensor([[1., 2.],
        [3., 4.],
        [5., 6.]])
tensor([[-1., -2.],
        [-3., -4.],
        [-5., -6.]])
```

### 1.1.3 3) Tensor expansion

Tensor expansion is to expand a tensor into a larger tensor along singleton dimensions. The corresponding functions are torch.Tensor.expand and torch.Tensor.expand_as. Please read the documents of the functions, and finish the following practice.

[5]:
```
# x is a tensor with size being (3)
x = torch.Tensor([1, 2, 3])

# Our goal is to generate a tensor z with size (2, 3), so that z[0,:,:] = x,␣
 ↪z[1,:,:] = x.

# [TO DO]
# Change the size of x into (1, 3) by using torch.unsqueeze.
x = torch.unsqueeze(x,0)
print(x.shape)

# [TO DO]
# Then expand the new tensor to the target tensor by using torch.Tensor.expand.
z = x.expand(2,3)
print(z.shape)
```

```
torch.Size([1, 3])
torch.Size([2, 3])
```

### 1.1.4 4) Tensor reduction in a given dimension

In deep learning, we often need to compute the mean/sum/max/min value in a given dimension of a tensor. Please read the document of torch.mean, torch.sum, torch.max, torch.min, torch.topk,

and finish the following practice.

[6]:
```
# x is a random tensor with size being (10, 50)
x = torch.randn(10, 50)

# Compute the mean value for each row of x.
# You need to generate a tensor x_mean of size (10), and x_mean[k, :] is the␣
 ↪mean value of the k-th row of x.
x_mean = torch.mean(x,1)
print(x_mean[3, ])

# Compute the sum value for each row of x.
# You need to generate a tensor x_sum of size (10).
x_sum = torch.sum(x,1)
print(x_sum.shape)

# Compute the max value for each row of x.
# You need to generate a tensor x_max of size (10).
x_max, s = torch.max(x,1)
print(x_max.shape)

# Compute the min value for each row of x.
# You need to generate a tensor x_min of size (10).
x_min, s = torch.min(x,1)
print(x_min.shape)

# Compute the top-5 values for each row of x.
# You need to generate a tensor x_mean of size (10, 5), and x_top[k, :] is the␣
 ↪top-5 values of each row in x.
x_mean_xtop, s = torch.topk(x,5,1)
print((x_mean_xtop.shape))
```

```
tensor(0.1522)
torch.Size([10])
torch.Size([10])
torch.Size([10])
torch.Size([10, 5])
```

### 1.2 Convolutional Neural Networks

Implement a convolutional neural network for image classification on CIFAR-10 dataset.

CIFAR-10 is an image dataset of 10 categories. Each image has a size of 32x32 pixels. The following code will download the dataset, and split it into `train` and `test`. For this question, we use the default validation split generated by Skorch.

[3]:
```
train = torchvision.datasets.CIFAR10("./data", train=True, download=True)
test = torchvision.datasets.CIFAR10("./data", train=False, download=True)
```

```
train.targets = torch.LongTensor(train.targets)
```

```
Files already downloaded and verified
Files already downloaded and verified
```

The following code visualizes some samples in the dataset. You may use it to debug your model if necessary.

[7]:
```
def plot(data, labels=None, num_sample=5):
  n = min(len(data), num_sample)
  for i in range(n):
    plt.subplot(1, n, i+1)
    plt.imshow(data[i], cmap="gray")
    plt.xticks([])
    plt.yticks([])
    if labels is not None:
      plt.title(labels[i])

train.labels = [train.classes[target] for target in train.targets]
plot(train.data, train.labels)
```


### 1.2.1 1) Basic CNN implementation

Consider a basic CNN model

- It has 3 convolutional layers, followed by a linear layer.
- Each convolutional layer has a kernel size of 3, a padding of 1.
- ReLU activation is applied on every hidden layer.

Please implement this model in the following section. The hyperparameters is then be tuned and you need to fill the results in the table.

**a) Implement convolutional layers (10 Points)** Implement the initialization function and the forward function of the CNN.

[8]:
```
class CNN(nn.Module):
  def __init__(self, channels):
    super(CNN, self).__init__()
    # implement parameter definitions here
```

```
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    self.conv1 = nn.Conv2d(3,channels,3,padding=1)
    self.conv2 = nn.Conv2d(channels,channels,3,padding=1)
    self.conv3 = nn.Conv2d(channels,channels,3,padding=1)
    self.fcl = nn.Linear(channels*32*32,10)
    # *****END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
  def forward(self, images):
    # implement the forward function here
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    images = F.relu(self.conv1(images))
    images = F.relu(self.conv2(images))
    images = F.relu(self.conv3(images))
    images = images.view(images.size(0),-1)
    images = self.fcl(images)
    # *****END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    return images
```

**b) Tune hyperparameters** Train the CNN model on CIFAR-10 dataset. We can tune the number of channels, optimizer, learning rate and the number of epochs for best validation accuracy.

[10]:
```
# implement hyperparameters, you can select and modify the hyperparameters by␣
 ↪yourself here.

optimize = [torch.optim.SGD, torch.optim.Adam]
learning_rate = [1e-5, 1e-4, 1e-3, 1e-2]
channel = [16,32,64]


train_data_normalized = torch.Tensor(train.data/255)
train_data_normalized = train_data_normalized.permute(0,3,1,2)

for l in learning_rate:
  for o in optimize:
    for c in channel:
      print(f'The channel was {c}, the learning rate was {l} and the optimizer␣
 ↪was {str(o)}')

      cnn = CNN(channels = c)

      model = skorch.NeuralNetClassifier(cnn, criterion=torch.nn.
 ↪CrossEntropyLoss,
                                         device="cpu",
                                         optimizer=o,
                                        # optimizer__momentum=0.90,
                                         lr=l,
                                         max_epochs=25,
```

```
                                    batch_size=64,
                                    callbacks=[skorch.callbacks.
↪EarlyStopping(lower_is_better=True)])
    # implement input normalization & type cast here
    model.fit(train_data_normalized, np.asarray(train.targets))
```

```
The channel was 16, the learning rate was 1e-05 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.3031       0.0922        2.3030
33.2964
      2        2.3030       0.0923        2.3030
32.4063
      3        2.3029       0.0928        2.3029
31.2097
      4        2.3029       0.0930        2.3028
32.2927
      5        2.3028       0.0932        2.3028
31.1628
      6        2.3027       0.0935        2.3027
31.4149
      7        2.3027       0.0940        2.3026
30.7325
      8        2.3026       0.0938        2.3026  30.8253
      9        2.3025       0.0942        2.3025
32.8183
     10        2.3025       0.0941        2.3024  32.3721
     11        2.3024       0.0945        2.3024
31.9081
     12        2.3024       0.0952        2.3023
31.5482
     13        2.3023       0.0959        2.3022
31.3438
     14        2.3022       0.0967        2.3022
32.1400
     15        2.3022       0.0982        2.3021
31.0039
     16        2.3021       0.0985        2.3020
31.0092
     17        2.3020       0.0996        2.3020
32.0398
     18        2.3020       0.1011        2.3019
32.9767
     19        2.3019       0.1020        2.3018
31.9393
     20        2.3018       0.1039        2.3018
```

```
31.2194
     21        2.3018       0.1051        2.3017
32.5872
     22        2.3017       0.1059        2.3016
31.4957
     23        2.3016       0.1072        2.3016
31.6174
     24        2.3016       0.1090        2.3015
32.3750
     25        2.3015       0.1118        2.3014
32.3782
The channel was 32, the learning rate was 1e-05 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.3026       0.1118        2.3025
75.5154
      2        2.3023       0.1192        2.3022
73.5765
      3        2.3021       0.1307        2.3019
73.9044
      4        2.3018       0.1455        2.3016
74.4299
      5        2.3015       0.1546        2.3014
74.1025
      6        2.3013       0.1537        2.3011  74.7158
      7        2.3010       0.1471        2.3008  75.1688
      8        2.3008       0.1398        2.3006  74.0391
      9        2.3005       0.1341        2.3004  74.8319
     10        2.3003       0.1254        2.3001  78.7981
     11        2.3001       0.1191        2.2999  74.6853
     12        2.2998       0.1144        2.2996  75.1045
     13        2.2996       0.1083        2.2994  75.2092
     14        2.2994       0.1044        2.2992  75.3185
     15        2.2992       0.1027        2.2990  73.9641
     16        2.2990       0.1009        2.2988  75.9277
     17        2.2988       0.0994        2.2985  74.4793
     18        2.2985       0.0991        2.2983  81.8071
     19        2.2983       0.0994        2.2981  74.3654
     20        2.2981       0.1005        2.2979  75.5451
     21        2.2979       0.1000        2.2977  74.9100
     22        2.2977       0.1022        2.2975  75.4648
     23        2.2975       0.1026        2.2973  74.9593
     24        2.2973       0.1048        2.2971  75.1578
     25        2.2971       0.1064        2.2968  74.4299
The channel was 64, the learning rate was 1e-05 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
```

```
-------  ------------  -----------  ------------  --------
      1        2.3036       0.1017        2.3032
198.9562
      2        2.3030       0.1063        2.3026
198.6153
      3        2.3024       0.1189        2.3021
216.3862
      4        2.3019       0.1192        2.3015
223.2036
      5        2.3014       0.1128        2.3010  216.7390
      6        2.3009       0.1063        2.3005  217.7274
      7        2.3005       0.1027        2.3001  217.4551
      8        2.3000       0.1016        2.2996  204.9226
      9        2.2996       0.1003        2.2992  188.9512
     10        2.2991       0.1001        2.2987  187.3891
     11        2.2987       0.1002        2.2983  190.4109
     12        2.2982       0.0998        2.2978  189.4197
     13        2.2978       0.0997        2.2974  188.3463
     14        2.2974       0.0996        2.2969  187.0955
     15        2.2969       0.0995        2.2965  189.2065
     16        2.2965       0.0994        2.2960  191.9360
     17        2.2961       0.0997        2.2956  189.1270
     18        2.2956       0.0997        2.2951  191.8527
     19        2.2952       0.0998        2.2946  194.3153
     20        2.2947       0.1001        2.2941  195.2716
     21        2.2942       0.1001        2.2937  195.4924
     22        2.2937       0.1000        2.2932  193.3946
     23        2.2933       0.1009        2.2927  196.2878
     24        2.2928       0.1013        2.2922  195.4867
     25        2.2923       0.1014        2.2916  190.2633
The channel was 16, the learning rate was 1e-05 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.2091       0.2735        2.0809
40.5973
      2        2.0070       0.3232        1.9368
29.1095
      3        1.9032       0.3541        1.8627
29.8156
      4        1.8488       0.3669        1.8235
30.3466
      5        1.8154       0.3802        1.7928
30.2068
      6        1.7868       0.3920        1.7661
41.0151
      7        1.7605       0.4017        1.7412
31.3739
```

```
      8        1.7365       0.4086        1.7184
31.1535
      9        1.7132       0.4165        1.6963
31.2500
     10        1.6912       0.4213        1.6774
42.8885
     11        1.6731       0.4237        1.6618
31.5856
     12        1.6574       0.4282        1.6478
31.1994
     13        1.6430       0.4318        1.6349
30.9401
     14        1.6297       0.4361        1.6229
32.1891
     15        1.6174       0.4398        1.6118
31.5050
     16        1.6059       0.4441        1.6014
30.8917
     17        1.5951       0.4468        1.5917
31.0846
     18        1.5849       0.4492        1.5825
77.5715
     19        1.5752       0.4510        1.5737
30.6983
     20        1.5659       0.4541        1.5653
31.0534
     21        1.5569       0.4564        1.5572
31.2100
     22        1.5483       0.4584        1.5494
31.0877
     23        1.5399       0.4624        1.5419
31.1569
     24        1.5318       0.4653        1.5347
31.5902
     25        1.5240       0.4679        1.5277
31.0664
The channel was 32, the learning rate was 1e-05 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.1149       0.3102        1.9411
74.5811
      2        1.8789       0.3659        1.8154
75.3951
      3        1.7784       0.3966        1.7454
75.3582
      4        1.7204       0.4144        1.6974
76.2177
```

```
      5        1.6808       0.4245        1.6626
75.5505
      6        1.6494       0.4339        1.6335
74.8786
      7        1.6202       0.4427        1.6066
75.0219
      8        1.5946       0.4501        1.5847
75.0177
      9        1.5721       0.4590        1.5651
74.8159
     10        1.5513       0.4648        1.5471
75.3309
     11        1.5320       0.4698        1.5309
106.3320
     12        1.5143       0.4753        1.5164
101.5381
     13        1.4982       0.4784        1.5034
110.2510
     14        1.4836       0.4833        1.4916
74.4855
     15        1.4703       0.4884        1.4810
74.0732
     16        1.4580       0.4913        1.4713
75.0069
     17        1.4466       0.4930        1.4623
74.2623
     18        1.4360       0.4959        1.4540
74.7864
     19        1.4259       0.4992        1.4461
74.6959
     20        1.4163       0.5011        1.4385
72.3184
     21        1.4070       0.5018        1.4311
69.3383
     22        1.3980       0.5041        1.4240
71.3612
     23        1.3893       0.5057        1.4170
75.2446
     24        1.3807       0.5084        1.4101
75.8924
     25        1.3722       0.5103        1.4032
75.8757
The channel was 64, the learning rate was 1e-05 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        2.0151       0.3576        1.8291
202.1270
```

```
      2        1.7419        0.4183        1.6770
198.7369
      3        1.6265        0.4446        1.5937
201.3592
      4        1.5591        0.4619        1.5433
202.4845
      5        1.5155        0.4742        1.5105
201.6903
      6        1.4847        0.4827        1.4870
203.2900
      7        1.4607        0.4893        1.4681
203.0224
      8        1.4403        0.4940        1.4518
200.8325
      9        1.4219        0.4975        1.4366
202.5638
     10        1.4045        0.5019        1.4221
203.0319
     11        1.3877        0.5031        1.4080
201.4224
     12        1.3717        0.5075        1.3944
203.0768
     13        1.3564        0.5113        1.3812
201.1022
     14        1.3416        0.5164        1.3682
201.2386
     15        1.3271        0.5193        1.3552
202.8182
     16        1.3126        0.5223        1.3426
202.0798
     17        1.2989        0.5263        1.3308
203.6543
     18        1.2858        0.5293        1.3196
204.5527
     19        1.2733        0.5334        1.3090
202.7144
     20        1.2615        0.5381        1.2991
202.4044
     21        1.2502        0.5427        1.2899
202.2083
     22        1.2396        0.5456        1.2816
204.1165
     23        1.2296        0.5494        1.2738
204.8423
     24        1.2201        0.5541        1.2667
204.6857
     25        1.2112        0.5566        1.2601
198.3856
```

```
The channel was 16, the learning rate was 0.0001 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.3032       0.1000        2.3029
28.7028
      2        2.3028       0.1000        2.3026  28.3354
      3        2.3025       0.1000        2.3024  28.4123
      4        2.3023       0.1000        2.3021  27.5514
      5        2.3021       0.0996        2.3019  28.1030
      6        2.3019       0.1098        2.3017
29.9903
      7        2.3017       0.1358        2.3015
28.3431
      8        2.3014       0.1344        2.3012  28.8486
      9        2.3012       0.1194        2.3010  28.3444
     10        2.3009       0.1104        2.3007  29.0628
     11        2.3006       0.1101        2.3004  29.2648
     12        2.3003       0.1119        2.3000  28.6420
     13        2.2999       0.1138        2.2996  29.5950
     14        2.2994       0.1148        2.2991  29.9916
     15        2.2990       0.1152        2.2985  31.6355
     16        2.2984       0.1147        2.2979  29.3639
     17        2.2978       0.1155        2.2973  29.2405
     18        2.2971       0.1178        2.2965  29.2563
     19        2.2963       0.1187        2.2957  28.8609
     20        2.2955       0.1214        2.2948  28.9748
     21        2.2945       0.1237        2.2937  29.7247
     22        2.2934       0.1259        2.2925  29.3646
     23        2.2921       0.1292        2.2911  29.2549
     24        2.2907       0.1312        2.2895  28.1076
     25        2.2890       0.1330        2.2876  29.5127
The channel was 32, the learning rate was 0.0001 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.3019       0.1010        2.3004
71.7964
      2        2.2990       0.0957        2.2975  71.3975
      3        2.2961       0.1035        2.2944
72.1388
      4        2.2929       0.1090        2.2909
71.7891
      5        2.2892       0.1129        2.2868
72.7883
      6        2.2847       0.1206        2.2818
71.2684
      7        2.2794       0.1338        2.2758
```

```
70.9856
      8        2.2729       0.1507        2.2685
72.8750
      9        2.2650       0.1693        2.2595
71.1035
     10        2.2553       0.1898        2.2485
73.5009
     11        2.2432       0.2095        2.2347
72.6865
     12        2.2283       0.2266        2.2178
71.6683
     13        2.2100       0.2424        2.1972
71.6845
     14        2.1879       0.2544        2.1726
72.0716
     15        2.1620       0.2632        2.1445
71.5759
     16        2.1333       0.2672        2.1144
72.5669
     17        2.1039       0.2737        2.0849
71.9720
     18        2.0761       0.2800        2.0577
71.5721
     19        2.0511       0.2833        2.0338
71.3198
     20        2.0293       0.2873        2.0132
72.4818
     21        2.0106       0.2932        1.9956
73.3503
     22        1.9946       0.2983        1.9805
71.5927
     23        1.9807       0.3028        1.9673
71.5969
     24        1.9683       0.3082        1.9554
71.0115
     25        1.9570       0.3136        1.9445
71.7138
The channel was 64, the learning rate was 0.0001 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        2.3009       0.1120        2.2982
195.1902
      2        2.2964       0.1066        2.2936  194.0137
      3        2.2916       0.1109        2.2885  194.2230
      4        2.2862       0.1200        2.2823
195.4474
      5        2.2794       0.1364        2.2745
```

```
195.2063
      6        2.2707       0.1554        2.2645
193.1715
      7        2.2595       0.1787        2.2514
194.5282
      8        2.2449       0.1982        2.2345
195.8728
      9        2.2262       0.2093        2.2133
194.8654
     10        2.2034       0.2224        2.1881
194.6705
     11        2.1772       0.2323        2.1603
195.2831
     12        2.1495       0.2421        2.1323
196.9269
     13        2.1227       0.2524        2.1065
193.8297
     14        2.0984       0.2648        2.0832
194.5961
     15        2.0761       0.2734        2.0616
194.1725
     16        2.0549       0.2848        2.0406
194.5781
     17        2.0340       0.2949        2.0197
196.1359
     18        2.0132       0.3010        1.9989
197.9209
     19        1.9929       0.3081        1.9790
195.3134
     20        1.9740       0.3152        1.9606
195.0304
     21        1.9568       0.3224        1.9441
197.6767
     22        1.9417       0.3277        1.9298
198.4896
     23        1.9287       0.3320        1.9175
196.9817
     24        1.9176       0.3345        1.9069
196.7081
     25        1.9080       0.3358        1.8977
193.1102
The channel was 16, the learning rate was 0.0001 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        1.9091       0.4104        1.6901
30.7223
      2        1.6127       0.4607        1.5194
```

```
30.9500
      3         1.5025         0.4793         1.4623
30.9233
      4         1.4517         0.4902         1.4305
30.2721
      5         1.4129         0.5029         1.3984
31.8046
      6         1.3701         0.5123         1.3735
31.1818
      7         1.3288         0.5216         1.3455
31.5237
      8         1.2937         0.5321         1.3172
31.0971
      9         1.2590         0.5437         1.2906
30.9779
     10         1.2283         0.5503         1.2708
30.2025
     11         1.2016         0.5568         1.2541
31.1930
     12         1.1771         0.5621         1.2402
31.1481
     13         1.1543         0.5667         1.2282
31.0480
     14         1.1332         0.5718         1.2180
30.6424
     15         1.1136         0.5761         1.2095
31.3306
     16         1.0954         0.5794         1.2020
31.3529
     17         1.0785         0.5811         1.1952
31.1834
     18         1.0627         0.5817         1.1891
31.1058
     19         1.0478         0.5851         1.1836
31.0362
     20         1.0336         0.5861         1.1784
30.9513
     21         1.0201         0.5883         1.1740
31.1171
     22         1.0072         0.5902         1.1700
31.0392
     23         0.9948         0.5936         1.1660
30.5617
     24         0.9829         0.5959         1.1625
30.8305
     25         0.9713         0.5972         1.1596
31.4569
The channel was 32, the learning rate was 0.0001 and the optimizer was <class
```

```
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        1.7448       0.4688        1.5068
75.5064
      2        1.4580       0.5062        1.3951
75.1589
      3        1.3332       0.5375        1.3080
75.4120
      4        1.2465       0.5582        1.2512
75.9160
      5        1.1877       0.5734        1.2123
75.0868
      6        1.1410       0.5847        1.1852
74.9629
      7        1.1002       0.5949        1.1653
75.5087
      8        1.0633       0.5996        1.1492
75.9712
      9        1.0295       0.6058        1.1342
76.0611
     10        0.9980       0.6107        1.1192
75.7848
     11        0.9679       0.6156        1.1054
75.7735
     12        0.9389       0.6199        1.0942
76.3694
     13        0.9105       0.6219        1.0854
76.6931
     14        0.8830       0.6250        1.0779
75.8649
     15        0.8561       0.6296        1.0724
76.3687
     16        0.8300       0.6347        1.0692
76.4422
     17        0.8044       0.6380        1.0671
83.2052
     18        0.7794       0.6387        1.0675  75.9804
     19        0.7552       0.6392        1.0690  75.9325
     20        0.7316       0.6395        1.0721  76.2727
     21        0.7087       0.6395        1.0767  76.2571
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 64, the learning rate was 0.0001 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        1.6846       0.4873        1.4493
199.5793
```

```
      2        1.3539       0.5523        1.2738
199.9522
      3        1.2091       0.5765        1.2135
200.8681
      4        1.1118       0.5935        1.1575
200.6454
      5        1.0238       0.6135        1.1113
201.8104
      6        0.9444       0.6264        1.0786
201.4504
      7        0.8754       0.6313        1.0595
203.6201
      8        0.8145       0.6402        1.0516
202.8307
      9        0.7593       0.6422        1.0526  200.6320
     10        0.7077       0.6450        1.0609  203.2361
     11        0.6587       0.6443        1.0765  203.4782
     12        0.6113       0.6437        1.0997  352.7898
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 16, the learning rate was 0.001 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.2964       0.1563        2.2855
62.8376
      2        2.2500       0.2369        2.1857
63.1041
      3        2.0977       0.2771        2.0253
61.0745
      4        1.9956       0.3003        1.9668
58.8770
      5        1.9464       0.3193        1.9265
59.4622
      6        1.9105       0.3325        1.8931
61.0843
      7        1.8791       0.3464        1.8611
59.7554
      8        1.8484       0.3598        1.8307
59.1445
      9        1.8205       0.3675        1.8045
59.5986
     10        1.7975       0.3725        1.7836
55.7545
     11        1.7786       0.3777        1.7670
52.4581
     12        1.7627       0.3846        1.7533
57.9407
     13        1.7489       0.3882        1.7418
```

```
59.9380
     14        1.7367       0.3925        1.7317
65.5651
     15        1.7257       0.3944        1.7224
57.1700
     16        1.7155       0.3960        1.7138
31.3681
     17        1.7055       0.3986        1.7051
35.0293
     18        1.6956       0.4018        1.6961
39.7325
     19        1.6852       0.4064        1.6861
38.0291
     20        1.6740       0.4115        1.6748
36.4835
     21        1.6617       0.4163        1.6620
41.1977
     22        1.6481       0.4227        1.6480
41.3113
     23        1.6336       0.4269        1.6332
34.2660
     24        1.6189       0.4337        1.6187
39.6896
     25        1.6043       0.4382        1.6046
39.7966
The channel was 32, the learning rate was 0.001 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        2.2918       0.1775        2.2719
100.1234
      2        2.1953       0.2770        2.0722
98.2262
      3        1.9944       0.3046        1.9449
103.1981
      4        1.9145       0.3318        1.8852
104.1640
      5        1.8644       0.3492        1.8420
104.9741
      6        1.8289       0.3612        1.8109
105.7524
      7        1.8018       0.3664        1.7870
105.2751
      8        1.7796       0.3738        1.7674
102.9188
      9        1.7605       0.3808        1.7508
100.9611
     10        1.7439       0.3855        1.7362
```

```
105.1222
     11        1.7285       0.3919        1.7221
105.9550
     12        1.7134       0.3981        1.7078
106.2921
     13        1.6978       0.4039        1.6923
105.0269
     14        1.6808       0.4117        1.6747
101.3276
     15        1.6623       0.4192        1.6553
106.9754
     16        1.6433       0.4245        1.6361
106.4313
     17        1.6254       0.4321        1.6191
106.3367
     18        1.6094       0.4386        1.6042
105.7679
     19        1.5948       0.4425        1.5906
105.5507
     20        1.5807       0.4490        1.5774
79.0545
     21        1.5669       0.4531        1.5643
110.3573
     22        1.5530       0.4584        1.5511
111.5549
     23        1.5390       0.4614        1.5379
112.9123
     24        1.5251       0.4657        1.5249
113.7903
     25        1.5112       0.4707        1.5122
111.5942
The channel was 64, the learning rate was 0.001 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        2.2575       0.2347        2.1720
284.8710
      2        2.0469       0.3039        1.9570
271.1822
      3        1.9155       0.3392        1.8792
257.0626
      4        1.8564       0.3573        1.8284
252.7677
      5        1.8129       0.3730        1.7880
253.8607
      6        1.7745       0.3881        1.7490
303.3023
      7        1.7372       0.4042        1.7110
```

```
222.1229
      8        1.7038       0.4136        1.6800
241.7212
      9        1.6768       0.4232        1.6558
202.0839
     10        1.6538       0.4310        1.6353
263.5292
     11        1.6329       0.4373        1.6168
265.5118
     12        1.6132       0.4438        1.5992
251.3128
     13        1.5942       0.4494        1.5822
226.1318
     14        1.5756       0.4549        1.5653
241.7881
     15        1.5574       0.4595        1.5487
192.4584
     16        1.5395       0.4634        1.5325
194.0290
     17        1.5223       0.4689        1.5173
198.1915
     18        1.5060       0.4732        1.5035
214.3901
     19        1.4909       0.4765        1.4911
214.4707
     20        1.4770       0.4790        1.4797
213.0608
     21        1.4642       0.4803        1.4692
212.3046
     22        1.4521       0.4821        1.4595
213.4897
     23        1.4405       0.4861        1.4501
213.9201
     24        1.4294       0.4909        1.4412
213.1770
     25        1.4184       0.4937        1.4325
214.2598
The channel was 16, the learning rate was 0.001 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        1.7436       0.4735        1.4827
33.6238
      2        1.4139       0.5314        1.3268
34.9068
      3        1.2470       0.5739        1.2012
34.2287
      4        1.1048       0.5939        1.1564
```

```
33.9101
      5        0.9872       0.6057        1.1545
33.7875
      6        0.8806       0.6068        1.1838  33.9868
      7        0.7912       0.6066        1.2303  33.8943
      8        0.7121       0.6018        1.3180  33.9959
      9        0.6346       0.5843        1.5201  34.5033
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 32, the learning rate was 0.001 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        1.6392       0.5129        1.3716
79.5216
      2        1.2768       0.5762        1.1864
80.1609
      3        1.0843       0.5977        1.1543
81.0517
      4        0.9439       0.6044        1.1950  80.9044
      5        0.8090       0.6082        1.2552  81.2396
      6        0.6837       0.5862        1.4491  80.4444
      7        0.5770       0.5707        1.6861  80.3253
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 64, the learning rate was 0.001 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        1.5455       0.5606        1.2410
215.2249
      2        1.1250       0.6300        1.0641
218.1638
      3        0.9070       0.6414        1.0592
218.0134
      4        0.7451       0.6298        1.1461  220.1412
      5        0.6064       0.6173        1.3049  217.9364
      6        0.4885       0.6053        1.4677  218.7542
      7        0.4168       0.6066        1.5865  220.4605
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 16, the learning rate was 0.01 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.0554       0.3064        1.9374
32.1266
      2        1.7919       0.3784        1.7518
31.9039
      3        1.7046       0.4043        1.6787
32.3028
```

```
      4        1.6251       0.4372        1.5879
31.8986
      5        1.5426       0.4630        1.5165
32.4555
      6        1.4749       0.4854        1.4505
32.0101
      7        1.4209       0.5007        1.3991
32.1931
      8        1.3768       0.5098        1.3647
32.3143
      9        1.3407       0.5189        1.3422
34.0823
     10        1.3103       0.5260        1.3247
32.7837
     11        1.2837       0.5310        1.3127
32.8594
     12        1.2589       0.5351        1.3050
32.8592
     13        1.2350       0.5375        1.2950
32.2958
     14        1.2113       0.5429        1.2863
32.6479
     15        1.1864       0.5448        1.2753
32.5674
     16        1.1600       0.5510        1.2620
32.5912
     17        1.1338       0.5579        1.2510
32.1261
     18        1.1076       0.5624        1.2430
33.1683
     19        1.0813       0.5708        1.2259
32.5658
     20        1.0546       0.5763        1.2112
32.9377
     21        1.0271       0.5802        1.2013
32.5382
     22        0.9991       0.5859        1.1914
32.8069
     23        0.9713       0.5882        1.1885
33.4357
     24        0.9440       0.5933        1.1867
32.8244
     25        0.9175       0.5937        1.1897  33.5969
The channel was 32, the learning rate was 0.01 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        1.9882       0.3503        1.8243
```

```
78.8034
      2        1.7197       0.4227        1.6394
79.1140
      3        1.5920       0.4604        1.5231
78.4201
      4        1.4871       0.4865        1.4378
77.9095
      5        1.4048       0.5134        1.3671
79.3930
      6        1.3368       0.5274        1.3144
78.3532
      7        1.2702       0.5384        1.2814
78.9115
      8        1.2169       0.5473        1.2562
78.8646
      9        1.1693       0.5588        1.2340
79.3294
     10        1.1236       0.5699        1.2112
79.0200
     11        1.0788       0.5806        1.1873
79.5395
     12        1.0351       0.5904        1.1709
79.4168
     13        0.9931       0.5929        1.1610
80.7001
     14        0.9530       0.6008        1.1532
79.4426
     15        0.9143       0.6039        1.1506
79.4936
     16        0.8764       0.6059        1.1516  79.8378
     17        0.8392       0.6081        1.1577  81.5592
     18        0.8021       0.6086        1.1680  79.7244
     19        0.7651       0.6106        1.1833  80.8649
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 64, the learning rate was 0.01 and the optimizer was <class
'torch.optim.sgd.SGD'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        1.9238       0.3923        1.7129
213.7789
      2        1.6223       0.4537        1.5411
216.7653
      3        1.4901       0.4887        1.4371
216.1997
      4        1.4097       0.5048        1.3796
218.8107
      5        1.3467       0.5199        1.3309
216.5749
```

```
      6        1.2840       0.5423        1.2776
214.9383
      7        1.2187       0.5589        1.2334
215.9890
      8        1.1613       0.5735        1.1945
216.7445
      9        1.1131       0.5865        1.1653
216.2992
     10        1.0708       0.5958        1.1396
217.9955
     11        1.0329       0.6027        1.1224
228.4997
     12        0.9984       0.6096        1.1077
224.0026
     13        0.9654       0.6143        1.0994
216.9344
     14        0.9331       0.6200        1.0930
236.4126
     15        0.9002       0.6204        1.0890
247.5127
     16        0.8663       0.6213        1.0879
242.1076
     17        0.8310       0.6250        1.0874
226.0298
     18        0.7947       0.6259        1.0890  216.5611
     19        0.7578       0.6282        1.0964  217.5583
     20        0.7206       0.6294        1.1091  254.2584
     21        0.6830       0.6274        1.1272  260.1418
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 16, the learning rate was 0.01 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.3163       0.1000        2.3034
50.7011
      2        2.3038       0.1000        2.3035  47.4422
      3        2.3039       0.1000        2.3035  42.5227
      4        2.3039       0.1000        2.3035  41.7330
      5        2.3039       0.1000        2.3035  38.6801
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 32, the learning rate was 0.01 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss      dur
-------  ------------  -----------  ------------  -------
      1        2.3153       0.1000        2.3035
89.8181
      2        2.3039       0.1000        2.3035  90.8093
      3        2.3039       0.1000        2.3035  123.9652
```

```
      4        2.3039       0.1000        2.3035  135.9307
      5        2.3039       0.1000        2.3035  118.6586
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 64, the learning rate was 0.01 and the optimizer was <class
'torch.optim.adam.Adam'>
  epoch    train_loss    valid_acc    valid_loss       dur
-------  ------------  -----------  ------------  --------
      1        2.7192       0.1000        2.3034
262.2795
      2        2.3038       0.1000        2.3035  242.4639
      3        2.3039       0.1000        2.3035  280.0173
      4        2.3039       0.1000        2.3035  268.4130
      5        2.3039       0.1000        2.3035  440.0961
Stopping since valid_loss has not improved in the last 5 epochs.
```

Write down **validation accuracy** of your model under different hyperparameter settings. Note the validation set is automatically split by Skorch during `model.fit()`.

| #channel for each layer  optimizer | SGD | Adam |
|---|---|---|
| (16,16,16) | 62.64 | 66.00 |
| (32,32,32) | 67.89 | 72.11 |
| (64,64,64) | 63.22 | 65.60 |

### 1.2.2 2) Full CNN implementation (10 points)

Based on the CNN in the previous question, implement a full CNN model with max pooling layer.

- Add a max pooling layer after each convolutional layer.
- Each max pooling layer has a kernel size of 2 and a stride of 2.

Please implement this model in the following section. The hyperparameters is then be tuned and fill the results in the table. You are also required to complete the questions.

**a) Implement max pooling layers** Similar to the CNN implementation in previous question, implement max pooling layers.

[9]:
```
class CNN_MaxPool(nn.Module):
  def __init__(self, channels):
    super(CNN_MaxPool, self).__init__()
    # implement parameter definitions here
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    self.conv1 = nn.Conv2d(3,512,3,padding=1)
    self.conv2 = nn.Conv2d(512,512,3,padding=1)
    self.conv3 = nn.Conv2d(512,512,3,padding=1)
    self.pool = nn.MaxPool2d(kernel_size=2, stride=2)
    self.fcl = nn.Linear(512*4*4,10)
    # *****END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****

  def forward(self, images):
```

```
    # implement the forward function here
    # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    images = self.pool(F.relu(self.conv1(images)))
    images = self.pool(F.relu(self.conv2(images)))
    images = self.pool(F.relu(self.conv3(images)))
    images = images.view(images.size(0), -1)
    images = self.fcl(images)
    # *****END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
    return images
```

**b) Tune hyperparameters** Based on the better optimizer found in the previous problem, we can tune the number of channels and learning rate for best validation accuracy.

[7]:
```
# implement hyperparameters, you can select and modify the hyperparameters by␣
 ↪yourself here.
learning_rate = [1e-5, 1e-4, 1e-3, 1e-2]
channel = [16,32,64]
# Select the better optimizer by the result shown in the previous problem, you␣
 ↪can select and modify it by yourself here.
better_optimizer = torch.optim.Adam

train_data_normalized = torch.Tensor(train.data/255)
train_data_normalized = train_data_normalized.permute(0,3,1,2)

for l in learning_rate:
    for c in channel:
      print(f'The channel was {c}, the learning rate was {l}')

      cnn = CNN_MaxPool(channels = c)

      model = skorch.NeuralNetClassifier(cnn, criterion=torch.nn.
 ↪CrossEntropyLoss,
                                         device="cpu",
                                         optimizer=better_optimizer,
                                         lr=l,
                                         max_epochs=25,
                                         batch_size=64,
                                         callbacks=[skorch.callbacks.
 ↪EarlyStopping(lower_is_better=True)])
      # implement input normalization & type cast here
      model.fit(train_data_normalized, np.asarray(train.targets))
```

```
The channel was 16, the learning rate was 1e-05
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.9620       0.4077        1.7026
2015.3576
```

```
      2        1.6225        0.4595        1.5351
2181.7414
      3        1.4975        0.4886        1.4461
2189.5115
      4        1.4178        0.5118        1.3862
2183.9949
      5        1.3609        0.5283        1.3418
2181.5302
      6        1.3178        0.5435        1.3058
2182.7197
      7        1.2821        0.5561        1.2747
2183.1216
      8        1.2505        0.5653        1.2466
2175.7946
      9        1.2216        0.5737        1.2212
2122.4000
     10        1.1948        0.5821        1.1977
2116.9122
     11        1.1697        0.5895        1.1760
2111.7452
     12        1.1462        0.5974        1.1558
2133.8165
     13        1.1239        0.6046        1.1369
2165.6519
     14        1.1029        0.6119        1.1189
2179.4310
     15        1.0829        0.6195        1.1020
2176.8308
     16        1.0637        0.6238        1.0863
2186.6272
     17        1.0455        0.6263        1.0712
2190.6645
     18        1.0280        0.6311        1.0570
2132.4965
     19        1.0113        0.6364        1.0436
2491.9667
     20        0.9951        0.6418        1.0309
2022.1192
     21        0.9796        0.6457        1.0188
1885.7328
     22        0.9648        0.6514        1.0072
1745.3710
     23        0.9505        0.6555        0.9963
1745.1618
     24        0.9367        0.6584        0.9858
1770.0926
     25        0.9235        0.6600        0.9759
2212.5278
```

```
The channel was 32, the learning rate was 1e-05
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.9730       0.4036        1.7164
1949.7775
      2        1.6381       0.4532        1.5527
1489.9995
      3        1.5127       0.4858        1.4591
1483.2545
      4        1.4310       0.5029        1.3969
1962.5577
      5        1.3724       0.5189        1.3522
1624.4438
      6        1.3286       0.5328        1.3165
1860.1606
      7        1.2928       0.5456        1.2858
2097.1347
      8        1.2615       0.5581        1.2582
1860.9127
      9        1.2330       0.5702        1.2329
1925.6908
     10        1.2065       0.5782        1.2097
1926.7113
     11        1.1819       0.5853        1.1880
1904.7657
     12        1.1586       0.5928        1.1679
2011.2623
     13        1.1366       0.6000        1.1489
1861.9987
     14        1.1157       0.6052        1.1312
1962.1067
     15        1.0958       0.6104        1.1144
2043.5940
     16        1.0768       0.6162        1.0988
2634.7108
     17        1.0587       0.6199        1.0841
2675.6508
     18        1.0413       0.6253        1.0700
2109.1479
     19        1.0246       0.6296        1.0566
2029.2728
     20        1.0086       0.6341        1.0440
2033.8819
     21        0.9933       0.6392        1.0322
2029.2635
     22        0.9785       0.6440        1.0208
2028.9265
     23        0.9644       0.6473        1.0100
```

```
2025.9630
     24        0.9507       0.6506        0.9996
2036.2984
     25        0.9375       0.6535        0.9899
2012.0991
The channel was 64, the learning rate was 1e-05
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.9789       0.4005        1.7277
1985.6530
      2        1.6450       0.4538        1.5547
1985.4058
      3        1.5141       0.4847        1.4577
1987.3407
      4        1.4299       0.5104        1.3948
1992.3526
      5        1.3713       0.5266        1.3502
1986.3687
      6        1.3274       0.5381        1.3144
2005.1232
      7        1.2912       0.5513        1.2838
2030.5202
      8        1.2592       0.5605        1.2562
2034.0302
      9        1.2302       0.5709        1.2308
2040.9535
     10        1.2033       0.5802        1.2074
3005.5707
     11        1.1782       0.5881        1.1854
3138.7492
     12        1.1547       0.5942        1.1648
2517.8228
     13        1.1324       0.6005        1.1459
1770.9409
     14        1.1113       0.6077        1.1280
1777.9515
     15        1.0913       0.6124        1.1113
1827.6349
     16        1.0723       0.6176        1.0952
1772.7285
     17        1.0541       0.6233        1.0803
1825.2955
     18        1.0368       0.6281        1.0660
1796.1912
     19        1.0201       0.6321        1.0526
1692.3421
     20        1.0042       0.6363        1.0400
1692.9635
```

```
     21        0.9889       0.6421        1.0281
1839.3732
     22        0.9742       0.6457        1.0168
2211.6594
     23        0.9601       0.6494        1.0062
1855.1671
     24        0.9465       0.6535        0.9959
1824.7757
     25        0.9334       0.6573        0.9862
1802.4270
The channel was 16, the learning rate was 0.0001
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.6355       0.5304        1.3180
2770.2951
      2        1.2318       0.6103        1.1098
2611.8159
      3        1.0593       0.6553        0.9850
1812.2924
      4        0.9358       0.6780        0.9174
1520.0275
      5        0.8430       0.6907        0.8806
2006.2764
      6        0.7693       0.7014        0.8549
1987.1612
      7        0.7071       0.7083        0.8357
2720.4578
      8        0.6510       0.7160        0.8158
2284.5833
      9        0.5984       0.7265        0.7918
1764.6349
     10        0.5477       0.7343        0.7780
1683.3413
     11        0.4995       0.7381        0.7768
1904.5913
     12        0.4555       0.7387        0.7776  1866.6369
     13        0.4127       0.7384        0.7880  1862.4910
     14        0.3709       0.7325        0.8154  1875.5805
     15        0.3301       0.7300        0.8527  2020.2487
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 32, the learning rate was 0.0001
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.6505       0.5274        1.3286
2016.2083
      2        1.2360       0.6101        1.1119
2046.6934
      3        1.0577       0.6581        0.9798
```

```
2024.6255
      4        0.9334       0.6789        0.9123
2022.7825
      5        0.8398       0.6931        0.8746
2029.5375
      6        0.7639       0.7020        0.8480
2022.9922
      7        0.6994       0.7123        0.8289
1999.5237
      8        0.6407       0.7202        0.8144
1841.0888
      9        0.5853       0.7250        0.8062
1830.9262
     10        0.5335       0.7293        0.8037
1830.7831
     11        0.4839       0.7342        0.8082  1844.9272
     12        0.4376       0.7346        0.8202  1890.5180
     13        0.3938       0.7356        0.8305  1852.2313
     14        0.3516       0.7374        0.8523  1855.6431
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 64, the learning rate was 0.0001
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.6386       0.5373        1.3020
2788.4585
      2        1.2156       0.6161        1.0923
1982.6056
      3        1.0390       0.6598        0.9749
1952.9520
      4        0.9179       0.6826        0.9064
1991.6278
      5        0.8271       0.6940        0.8686
1930.3556
      6        0.7540       0.7066        0.8407
2052.6165
      7        0.6913       0.7151        0.8192
1921.7194
      8        0.6339       0.7215        0.8040
1912.6408
      9        0.5806       0.7259        0.7941
1920.7272
     10        0.5306       0.7316        0.7892
1884.8599
     11        0.4836       0.7348        0.7915  2832.6945
     12        0.4400       0.7368        0.7914  2918.9338
     13        0.3984       0.7416        0.7927  2502.4152
     14        0.3583       0.7412        0.8081  2751.9188
Stopping since valid_loss has not improved in the last 5 epochs.
```

```
The channel was 16, the learning rate was 0.001
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.5403       0.5760        1.1992
2208.2255
      2        1.1425       0.6306        1.0490
1908.7850
      3        0.9650       0.6673        0.9531
2381.4564
      4        0.8508       0.6886        0.9051
2243.9538
      5        0.7680       0.6850        0.9369  3039.4427
      6        0.6954       0.6813        0.9643  2009.9776
      7        0.6422       0.6800        0.9796  1852.9855
      8        0.5855       0.6866        0.9889  1923.2216
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 32, the learning rate was 0.001
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.5969       0.5568        1.2330
1852.2973
      2        1.1545       0.6319        1.0502
1919.2373
      3        0.9796       0.6580        0.9890
1955.3820
      4        0.8679       0.6725        0.9481
2065.3813
      5        0.7888       0.6797        0.9225
2047.5048
      6        0.7265       0.6810        0.9570  2038.7800
      7        0.6762       0.6782        0.9940  2038.7441
      8        0.6333       0.6657        1.0982  2026.2662
      9        0.5929       0.6675        1.0986  2027.5202
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 64, the learning rate was 0.001
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        1.6344       0.5295        1.3157
1943.4180
      2        1.2120       0.6068        1.1103
1900.2877
      3        1.0708       0.6285        1.0560
1920.2538
      4        0.9739       0.6414        1.0254
2047.0268
      5        0.8986       0.6423        1.0417  2594.3312
      6        0.8384       0.6337        1.0671  2910.5896
      7        0.7984       0.6396        1.0678  2163.7094
```

```
      8        0.7519       0.6355        1.1093  1508.6315
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 16, the learning rate was 0.01
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        5.3360       0.1000        2.3033
1572.3764
      2        2.3038       0.1000        2.3034  1639.0350
      3        2.3038       0.1000        2.3035  1821.6766
      4        2.3039       0.1000        2.3035  1756.7726
      5        2.3039       0.1000        2.3035  1845.7966
Stopping since valid_loss has not improved in the last 5 epochs.
The channel was 32, the learning rate was 0.01
  epoch    train_loss    valid_acc    valid_loss        dur
-------  ------------  -----------  ------------  ---------
      1        4.6576       0.4688        1.4754
1711.5144
      2        1.4273       0.5073        1.3865
1621.4952
      3        1.3586       0.5256        1.3293
1622.3163
```

Write down the **validation accuracy** of the model under different hyperparameter settings.

| #channel for each layer | validation accuracy |
|---|---|
| (16,16,16) | 75.20 |
| (32,32,32) | 75.13 |
| (64,64,64) | 79.72 |

For the best model you have, test it on the test set.

[ ]:
```
# implement the same input normalization & type cast here
test_data_normalized = torch.Tensor(test.data/255)
test_data_normalized = test_data_normalized.permute(0,3,1,2)
test.predictions = model.predict(test_data_normalized)
sklearn.metrics.accuracy_score(test.targets, test.predictions)
```

How much **test accuracy** do you get? What can you conclude for the design of CNN structure and tuning of hyperparameters? (5 points)

**Your Answer:* I get 70.5% test accuracy. The number of channels and learning rate are the most important hyperparameters. They should not be too large or too small. Adam optimizer is better than SGD optimizer because it can converge faster. The structure of CNN is also important. The max pooling layer can improve the performance of CNN. The kernel size and stride of max pooling layer should be appropriate.***