Transfer learning Based on ResNet34 for Methylation Data

The network in this notebook is an implementation of the ResNet-50 [1] architecture on the CelebA face dataset [2] to train a gender classifier.

References

- [1] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778). (CVPR Link)
- [2] Zhang, K., Tan, L., Li, Z., & Qiao, Y. (2016). Gender and smile classification using deep convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 34-38).

The ResNet-50 architecture is similar to the ResNet-34 architecture shown below (from [1]):

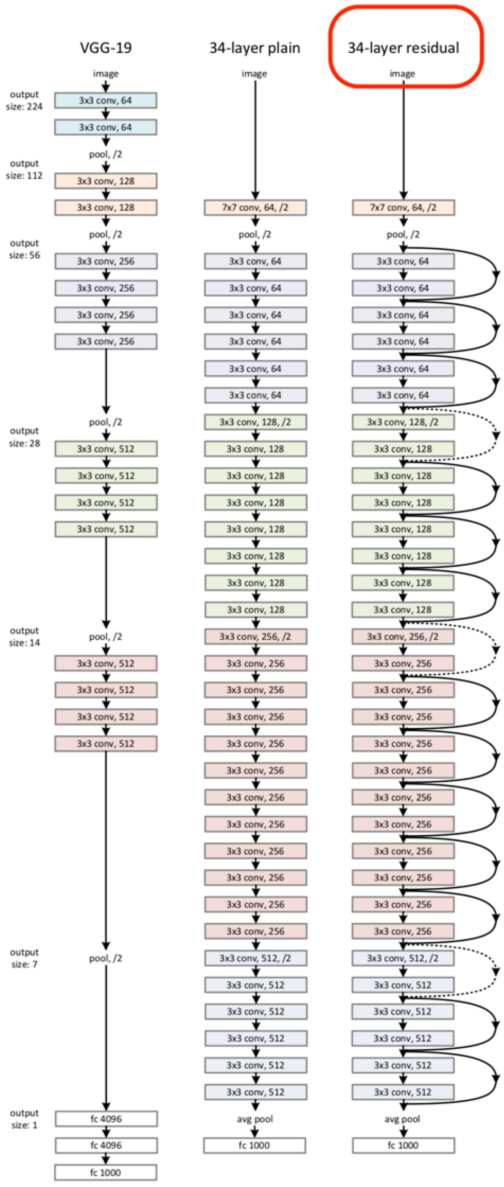


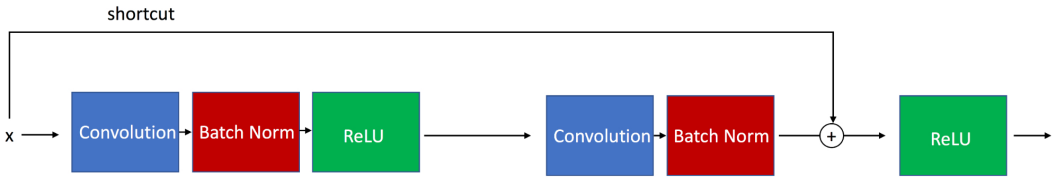
Figure 3. Example network architectures for ImageNet. **Left**: the VGG-19 model [41] (19.6 billion FLOPs) as a reference. **Middle**: a plain network with 34 parameter layers (3.6 billion FLOPs). **Right**: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. **Table 1** shows more details and other variants.

However, in ResNet-50, the skip connection uses a bottleneck (from [1]):

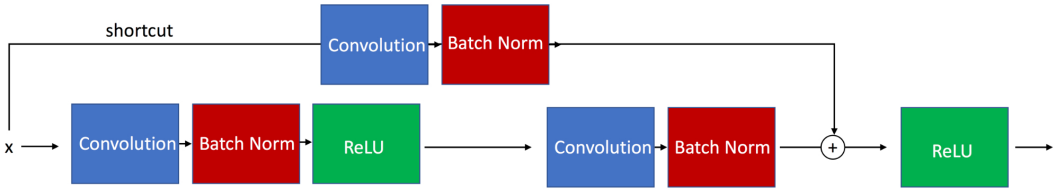
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$

conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

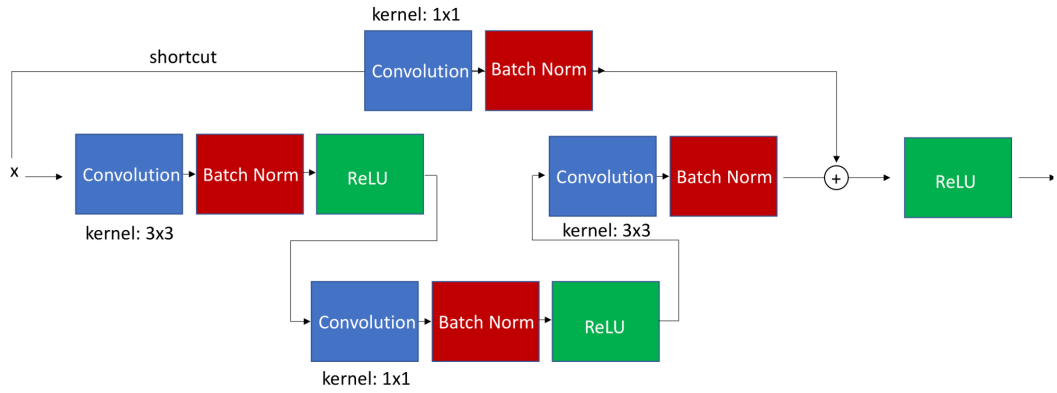
The following figure illustrates residual blocks with skip connections such that the input passed via the shortcut matches the dimensions of the main path's output, which allows the network to learn identity functions.



The ResNet-34 architecture actually uses residual blocks with skip connections such that the input passed via the shortcut matches is resized to dimensions of the main path's output. Such a residual block is illustrated below:



The ResNet-50 uses a bottleneck as shown below:



In [1]:
```
%load_ext autoreload
%autoreload 2
```

In [2]:
```
import os
from pathlib import PosixPath
import time
```

In [3]:
```
import pandas as pd
import numpy as np
```

In [365]:
```
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision
from torch.utils.data import Dataset, DataLoader
from torchvision import datasets, models, transforms
import matplotlib.pyplot as plt
import torch.nn.functional as F
from torchsummary import summary
```

Settings

In [545]:
```
## Hyper-paremeters
BATCH_SIZE = 8
NUM_EPOCHS = 30
LEARNING_RATE = 0.00001
RANDOM_SEED = 42


## Model Architecture
NUM_CLASSES = 2

DEVICE = torch.device("cuda: 0" if torch.cuda.is_available() else "cpu")
```

Dataset explore

GSE74845

Dataset Link

The development of non-invasive primary cancer preventive measures in humans require a thorough understanding of the initial cancer-driving molecular mechanisms. High grade serous extra-uterine M llerian cancers (HGSEMC; formerly classified as ovarian/tubal/peritoneal cancer) present at a very late stage and less than 40% of women survive 5 years. Although the recent TCGA initiatives revealed key molecular changes in established cancers, very little is known about the initial molecular alterations in cancer development. Analysis of normal tissue at extensively high risk prior to the development of any microscopic alterations is critical. BRCA1/2 mutation carriers have an up to 30 40 fold increased risk to develop ovarian cancer, preferentially HGSEMC. Despite a plethora of evidence linking mutations in BRCA1 or BRCA2 to cancer development the core components such as organ specificity (i.e. to breast and Fallopian Tube) are still missing. The Fallopian Tube of BRCA mutation carriers offers a unique opportunity to study carcinogenesis because these cancers originate only from the distal (i.e. fimbrial) end of the Fallopian Tube (which is in close proximity to the ovary), and not from the proximal end (which is close to the uterus). The ovary which is in extreme close proximity to the fimbriae provides an excellent soil for cancer cells which are likely shed from the fimbriae and once the cancer has been discovered the big bulk of tumour is usually present on the ovary and hence the majority of HGSEMC are also referred to as ovarian cancer . To study the earliest steps of human carcinogenesis we performed epigenome-wide DNA methylation (DNAme) analyses (using the Illumina 450k DNA methylation bead-array assay assessing DNAme at ~480 000 CpG sites) in 215 microscopic normal Fallopian Tube samples of BRCA1/2 mutation carriers (n=56) and controls (n=59) who had their tubes/ovaries removed for prophylactic or other reasons, respectively (for 52 and 49 individuals respectively we analysed both fimbrial and proximal Fallopian Tubes). In order to adjust for any epigenetic effects which are not of immediate importance to the carcinogenic process, for each volunteer we analysed both the fimbrial (at risk) and the proximal (non at risk) portion of the tubes separately.

In [546]:
```
root_dir = PosixPath(".")
dataset_path = "GSE74845_series_matrix.txt"
```

In [369]:
```
data_matrix  = pd.read_table(root_dir/dataset_path)
```

In [370]:
```
data_matrix.head()
```

Out[370]:

	ID_REF	GSM2132967	GSM2132968	GSM2132969	GSM2132970	GSM2132971	GSM2132972	GSM2132973	GSM2132974	GSM2132975	...	GSM2133183
0	cg00000029	0.413567	0.341079	0.381044	0.418526	0.308511	0.415981	0.325744	0.611465	0.426866	...	0.486653
1	cg00000108	0.943061	0.750692	0.913595	0.932710	0.914242	0.901047	0.916935	0.946930	0.935397	...	0.939712
2	cg00000109	0.799719	0.917194	0.885665	0.889474	0.866218	0.927703	0.863381	0.864142	0.873000	...	0.957135
3	cg00000165	0.212454	0.294118	0.420250	0.325410	0.282497	0.421712	0.341264	0.342029	0.280949	...	0.518261
4	cg00000236	0.893922	0.911511	0.868725	0.805311	0.890320	0.916782	0.921667	0.872765	0.911796	...	0.865964

5 rows × 217 columns

In [371]:
```
data_matrix.shape
```

Out[371]: (470426, 217)

In [372]:
```
data_matrix.drop("ID_REF", inplace=True, axis = 1)
```

In [373]:
```
sample_names = "GSM2132967 GSM2132968 GSM2132969 GSM2132970 GSM2132971 GSM2132972 GSM2132973 GSM2132974 GSM2132975
sample_title = "Patient-23-fim    Patient-29-fim    Patient-28-fim    Patient-29-prox    Patient-24-fim    Patient-3
```

In [374]:
```
target_dict = {sample: label for sample, label in zip(sample_names.split(), sample_title.split())}
```

In [375]:
```
target_dict
```

Out[375]:
```
{'GSM2132967': 'Patient-23-fim',
 'GSM2132968': 'Patient-29-fim',
 'GSM2132969': 'Patient-28-fim',
 'GSM2132970': 'Patient-29-prox',
 'GSM2132971': 'Patient-24-fim',
 'GSM2132972': 'Patient-30-fim',
 'GSM2132973': 'Patient-24-prox',
 'GSM2132974': 'Patient-30-prox',
 'GSM2132975': 'Patient-27-prox',
 'GSM2132976': 'Patient-15-fim',
 'GSM2132977': 'Patient-27-fim',
 'GSM2132978': 'Patient-15-prox',
 'GSM2132979': 'Patient-17-fim',
 'GSM2132980': 'Patient-22-fim',
 'GSM2132981': 'Patient-18-prox',
 'GSM2132982': 'Patient-25-prox',
 'GSM2132983': 'Patient-18-fim',
 'GSM2132984': 'Patient-25-fim',
 'GSM2132985': 'Patient-31-prox',
```

In [376]:
```
data_matrix.dropna(axis = 0,inplace=True)
```

In [377]:
```
data_matrix.shape
```

Out[377]: (470425, 216)

In [378]:
```
torch.from_numpy(data_matrix.iloc[:,1].values).size()
```

Out[378]: torch.Size([470425])

In [379]:
```
def transform_target(target):
    if target.split("-")[-1] == "prox":
        return 0
    else:
        return 1
```

In [380]:
```
target_dict = {target : transform_target(target_dict[target]) for target in target_dict}
```

In [381]:
```
data_matrix.iloc[:,0].values
```

Out[381]:
```
array([0.41356674, 0.94306078, 0.79971923, ..., 0.42618182, 0.65923307,
       0.92669805])
```

In [382]:
```
len(target_dict),sum(target_dict.values())
```

Out[382]: (216, 110)

In [547]:
```
class GSEDataset(Dataset):
    def __init__(self, data_matrix, target_dict):
        self.data_matrix = data_matrix
        self.target_dict = target_dict

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

    def __getitem__(self, idx):
        x = torch.from_numpy(self.data_matrix.iloc[:,idx].values).float()
        y = torch.tensor(self.target_dict[self.data_matrix.columns[idx]])
        return x,y
```

In [548]:
```
ges_dataset =  GSEDataset(data_matrix, target_dict)
```

In [549]:
```
train_size = int(0.8 * len(ges_dataset))
test_size = len(ges_dataset) - train_size

train_dataset, test_dataset = torch.utils.data.random_split(ges_dataset, [train_size, test_size])
```

In [550]:
```
train_dataloader = DataLoader(dataset = train_dataset,
                              batch_size=BATCH_SIZE,
                              shuffle=True,
                              num_workers=4)

test_dataloader = DataLoader(dataset = test_dataset,
                              batch_size=BATCH_SIZE,
                              shuffle=False,
                              num_workers=4)

dataset_loader = {"train": train_dataloader, "test": test_dataloader}
```

Transfer Model

In [551]:
```
resnet = torchvision.models.resnet18(pretrained=True)
```

In [552]:
```
for index, (name, layer) in enumerate(resnet.named_children()):
    print(index, name, " -> ", layer)
    print("--"* 40)
```

```
0 conv1  ->  Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)
--------------------------------------------------------------------------------
1 bn1  ->  BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
--------------------------------------------------------------------------------
2 relu  ->  ReLU(inplace=True)
--------------------------------------------------------------------------------
3 maxpool  ->  MaxPool2d(kernel_size=3, stride=2, padding=1, dilation=1, ceil_mode=False)
--------------------------------------------------------------------------------
4 layer1  ->  Sequential(
  (0): BasicBlock(
    (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (1): BasicBlock(
    (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
)
--------------------------------------------------------------------------------
5 layer2  ->  Sequential(
  (0): BasicBlock(
    (conv1): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (downsample): Sequential(
      (0): Conv2d(64, 128, kernel_size=(1, 1), stride=(2, 2), bias=False)
      (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
  )
  (1): BasicBlock(
    (conv1): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
)
--------------------------------------------------------------------------------
6 layer3  ->  Sequential(
  (0): BasicBlock(
    (conv1): Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (downsample): Sequential(
      (0): Conv2d(128, 256, kernel_size=(1, 1), stride=(2, 2), bias=False)
      (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
  )
  (1): BasicBlock(
    (conv1): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
)
--------------------------------------------------------------------------------
```

```
7 layer4  ->  Sequential(
  (0): BasicBlock(
    (conv1): Conv2d(256, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (downsample): Sequential(
      (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(2, 2), bias=False)
      (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
  )
  (1): BasicBlock(
    (conv1): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
)
--------------------------------------------------------------------------------
8 avgpool  ->  AdaptiveAvgPool2d(output_size=(1, 1))
--------------------------------------------------------------------------------
9 fc  ->  Linear(in_features=512, out_features=1000, bias=True)
--------------------------------------------------------------------------------
```

In [553]:
```
x,y = next(iter(dataset_loader["train"]))
```

In [554]:
```
x.size(), y.size()
```

Out[554]: (torch.Size([8, 470425]), torch.Size([8]))

In [555]:
```
def div_prime(num):
    res = []
    while num != 1:
        for i in range(2, int(num+1)):
            if num % i == 0:
                res.append(i)
                num = num // i
                break

    return res
div_prime(470425)
```

Out[555]: [5, 5, 31, 607]

Model

In [556]:
```
class GESMODEL(torch.nn.Module):
    def __init__(self, num_class = NUM_CLASSES):
        super(GESMODEL, self).__init__()


        self.pre = torch.nn.Sequential(
            torch.nn.Conv2d(in_channels=5, out_channels=3, kernel_size= 1, stride=1, padding=0, bias = False),
            torch.nn.ReLU(inplace = True),
            torch.nn.BatchNorm2d(3),
            torch.nn.AdaptiveAvgPool2d((224,224)),

        )


        self.base_model = models.resnet34(pretrained=True)
#         for param in self.base_model.parameters():
#             param.requires_grad = False

        self.base_layers = list(self.base_model.children())



        self.middle = nn.Sequential(*self.base_layers[:6])

        self.last = nn.Sequential(
            torch.nn.Conv2d(in_channels = 128, out_channels = 2, kernel_size = 1, stride = 1, padding = 0, bias = F
            torch.nn.BatchNorm2d(2),
            torch.nn.AdaptiveAvgPool2d(1),
        )


        for layer in self.pre.modules():
            if isinstance(layer, torch.nn.Conv2d):
                n = layer.kernel_size[0] * layer.kernel_size[1] * layer.out_channels
                layer.weight.data.normal_(0, (2. / n)**.5)
            elif isinstance(layer, torch.nn.BatchNorm2d):
                layer.weight.data.fill_(1)
                layer.bias.data.zero_()

        for layer in self.last.modules():
            if isinstance(layer, torch.nn.Conv2d):
                n = layer.kernel_size[0] * layer.kernel_size[1] * layer.out_channels
                layer.weight.data.normal_(0, (2. / n)**.5)
            elif isinstance(layer, torch.nn.BatchNorm2d):
                layer.weight.data.fill_(1)
                layer.bias.data.zero_()


    def forward(self, x):
        x = x.view(-1, 5,155,607)
        x = self.pre(x)
        x = self.middle(x)
#         print(x.size()) # torch.Size([2, 512, 7, 7])
        x = self.last(x)
        logits = torch.squeeze(x)
        probas = F.softmax(logits, dim = 1)
        return logits,probas
```

In [557]:
```
model = GESMODEL().to(DEVICE)
optimizer = torch.optim.Adam(model.parameters(), lr = LEARNING_RATE)
```

In [558]:
```
summary(model, (470425,))
```

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 3, 155, 607]              15
              ReLU-2          [-1, 3, 155, 607]               0
       BatchNorm2d-3          [-1, 3, 155, 607]               6
 AdaptiveAvgPool2d-4          [-1, 3, 224, 224]               0
            Conv2d-5         [-1, 64, 112, 112]           9,408
            Conv2d-6         [-1, 64, 112, 112]           9,408
       BatchNorm2d-7         [-1, 64, 112, 112]             128
       BatchNorm2d-8         [-1, 64, 112, 112]             128
              ReLU-9         [-1, 64, 112, 112]               0
             ReLU-10         [-1, 64, 112, 112]               0
        MaxPool2d-11           [-1, 64, 56, 56]               0
        MaxPool2d-12           [-1, 64, 56, 56]               0
           Conv2d-13           [-1, 64, 56, 56]          36,864
           Conv2d-14           [-1, 64, 56, 56]          36,864
      BatchNorm2d-15           [-1, 64, 56, 56]             128
      BatchNorm2d-16           [-1, 64, 56, 56]             128
```

Training

In [559]:
```
def compute_accuracy(model, data_loader, device):
    model.eval()
    correct_pred, num_examples = 0, 0
    for i, (features, targets) in enumerate(data_loader):

        features = features.to(device)
        targets = targets.to(device)

        logits, probas = model(features)
        _, predicted_labels = torch.max(probas, 1)
        num_examples += targets.size(0)
        correct_pred += (predicted_labels == targets).sum()
    return correct_pred.float()/num_examples * 100
```

In [560]:
```
def train_model(model, data_loader, optimizer, num_epochs,batch_size, device,metric_func, random_seed = 7):
    # Manual seed for deterministic data loader
    torch.manual_seed(random_seed)

    loss_list = []
    train_acc_list, valid_acc_list = [], []

    for epoch in range(num_epochs):
        start = time.time()
        # set training mode
        model.train()
        for batch_idx, (features, targets) in enumerate(data_loader["train"]):
            features = features.to(device)
            targets = targets.to(device)


            ## forward pass
            logits, probas = model(features)
            loss = F.cross_entropy(logits,targets)

            # backward pass
            # clear the gradients of all tensors being optimized
            optimizer.zero_grad()
            loss.backward()
            optimizer.step()

            ### Login
            loss_list.append(loss.item())

            print ('Epoch: {0:03d}/{1:03d} | Batch {2:03d}/{3:03d} | Loss: {4:.2f}'.format(
                epoch+1, num_epochs, batch_idx,
                     len(train_dataset)//batch_size, loss))

        end = time.time()
        with torch.set_grad_enabled(False):
            train_acc = metric_func(model, data_loader["train"], device)
            valid_acc = metric_func(model, data_loader["test"], device)

            print('Epoch: {0:03d}/{1:03d} train acc: {2:.3f} % | val acc: {3:.3f} % | time: {4:.3f} s'.format(
                  epoch+1, num_epochs, train_acc, valid_acc, end-start))



            train_acc_list.append(train_acc)
            valid_acc_list.append(valid_acc)

    return model, loss_list, train_acc_list, valid_acc_list
```

In [561]:
```
model, loss_list, train_acc_list, valid_acc_list = train_model(model,
            dataset_loader,
            optimizer,
            NUM_EPOCHS,
            device = DEVICE,
            batch_size = BATCH_SIZE,
            metric_func = compute_accuracy)
```

```
Epoch: 001/030 | Batch 000/021 | Loss: 0.70
Epoch: 001/030 | Batch 001/021 | Loss: 0.69
Epoch: 001/030 | Batch 002/021 | Loss: 0.70
Epoch: 001/030 | Batch 003/021 | Loss: 0.71
Epoch: 001/030 | Batch 004/021 | Loss: 0.69
Epoch: 001/030 | Batch 005/021 | Loss: 0.69
Epoch: 001/030 | Batch 006/021 | Loss: 0.68
Epoch: 001/030 | Batch 007/021 | Loss: 0.70
Epoch: 001/030 | Batch 008/021 | Loss: 0.70
Epoch: 001/030 | Batch 009/021 | Loss: 0.69
Epoch: 001/030 | Batch 010/021 | Loss: 0.69
Epoch: 001/030 | Batch 011/021 | Loss: 0.70
Epoch: 001/030 | Batch 012/021 | Loss: 0.69
```

```
Epoch: 001/030 | Batch 013/021 | Loss: 0.70
Epoch: 001/030 | Batch 014/021 | Loss: 0.70
Epoch: 001/030 | Batch 015/021 | Loss: 0.73
Epoch: 001/030 | Batch 016/021 | Loss: 0.69
Epoch: 001/030 | Batch 017/021 | Loss: 0.70
Epoch: 001/030 | Batch 018/021 | Loss: 0.69
Epoch: 001/030 | Batch 019/021 | Loss: 0.69
Epoch: 001/030 | Batch 020/021 | Loss: 0.69
Epoch: 001/030 | Batch 021/021 | Loss: 0.68
Epoch: 001/030 train acc: 50.581 % | val acc: 50.000 % | time: 2.603 s
Epoch: 002/030 | Batch 000/021 | Loss: 0.68
Epoch: 002/030 | Batch 001/021 | Loss: 0.69
Epoch: 002/030 | Batch 002/021 | Loss: 0.69
Epoch: 002/030 | Batch 003/021 | Loss: 0.67
Epoch: 002/030 | Batch 004/021 | Loss: 0.70
Epoch: 002/030 | Batch 005/021 | Loss: 0.70
Epoch: 002/030 | Batch 006/021 | Loss: 0.69
Epoch: 002/030 | Batch 007/021 | Loss: 0.70
Epoch: 002/030 | Batch 008/021 | Loss: 0.69
Epoch: 002/030 | Batch 009/021 | Loss: 0.67
Epoch: 002/030 | Batch 010/021 | Loss: 0.68
Epoch: 002/030 | Batch 011/021 | Loss: 0.70
Epoch: 002/030 | Batch 012/021 | Loss: 0.69
Epoch: 002/030 | Batch 013/021 | Loss: 0.70
Epoch: 002/030 | Batch 014/021 | Loss: 0.68
Epoch: 002/030 | Batch 015/021 | Loss: 0.67
Epoch: 002/030 | Batch 016/021 | Loss: 0.69
Epoch: 002/030 | Batch 017/021 | Loss: 0.69
Epoch: 002/030 | Batch 018/021 | Loss: 0.70
Epoch: 002/030 | Batch 019/021 | Loss: 0.69
Epoch: 002/030 | Batch 020/021 | Loss: 0.68
Epoch: 002/030 | Batch 021/021 | Loss: 0.70
Epoch: 002/030 train acc: 57.558 % | val acc: 72.727 % | time: 2.464 s
Epoch: 003/030 | Batch 000/021 | Loss: 0.67
Epoch: 003/030 | Batch 001/021 | Loss: 0.68
Epoch: 003/030 | Batch 002/021 | Loss: 0.68
Epoch: 003/030 | Batch 003/021 | Loss: 0.69
Epoch: 003/030 | Batch 004/021 | Loss: 0.68
Epoch: 003/030 | Batch 005/021 | Loss: 0.67
Epoch: 003/030 | Batch 006/021 | Loss: 0.70
Epoch: 003/030 | Batch 007/021 | Loss: 0.68
Epoch: 003/030 | Batch 008/021 | Loss: 0.69
Epoch: 003/030 | Batch 009/021 | Loss: 0.68
Epoch: 003/030 | Batch 010/021 | Loss: 0.68
Epoch: 003/030 | Batch 011/021 | Loss: 0.68
Epoch: 003/030 | Batch 012/021 | Loss: 0.68
Epoch: 003/030 | Batch 013/021 | Loss: 0.69
Epoch: 003/030 | Batch 014/021 | Loss: 0.70
Epoch: 003/030 | Batch 015/021 | Loss: 0.67
Epoch: 003/030 | Batch 016/021 | Loss: 0.67
Epoch: 003/030 | Batch 017/021 | Loss: 0.70
Epoch: 003/030 | Batch 018/021 | Loss: 0.69
Epoch: 003/030 | Batch 019/021 | Loss: 0.69
Epoch: 003/030 | Batch 020/021 | Loss: 0.69
Epoch: 003/030 | Batch 021/021 | Loss: 0.68
Epoch: 003/030 train acc: 70.349 % | val acc: 79.545 % | time: 2.469 s
Epoch: 004/030 | Batch 000/021 | Loss: 0.68
Epoch: 004/030 | Batch 001/021 | Loss: 0.67
Epoch: 004/030 | Batch 002/021 | Loss: 0.68
Epoch: 004/030 | Batch 003/021 | Loss: 0.67
Epoch: 004/030 | Batch 004/021 | Loss: 0.68
Epoch: 004/030 | Batch 005/021 | Loss: 0.67
Epoch: 004/030 | Batch 006/021 | Loss: 0.69
Epoch: 004/030 | Batch 007/021 | Loss: 0.68
Epoch: 004/030 | Batch 008/021 | Loss: 0.70
Epoch: 004/030 | Batch 009/021 | Loss: 0.70
Epoch: 004/030 | Batch 010/021 | Loss: 0.66
Epoch: 004/030 | Batch 011/021 | Loss: 0.67
Epoch: 004/030 | Batch 012/021 | Loss: 0.68
Epoch: 004/030 | Batch 013/021 | Loss: 0.67
Epoch: 004/030 | Batch 014/021 | Loss: 0.68
Epoch: 004/030 | Batch 015/021 | Loss: 0.69
Epoch: 004/030 | Batch 016/021 | Loss: 0.68
Epoch: 004/030 | Batch 017/021 | Loss: 0.67
Epoch: 004/030 | Batch 018/021 | Loss: 0.68
Epoch: 004/030 | Batch 019/021 | Loss: 0.67
Epoch: 004/030 | Batch 020/021 | Loss: 0.67
Epoch: 004/030 | Batch 021/021 | Loss: 0.70
Epoch: 004/030 train acc: 75.000 % | val acc: 79.545 % | time: 2.498 s
Epoch: 005/030 | Batch 000/021 | Loss: 0.67
Epoch: 005/030 | Batch 001/021 | Loss: 0.67
Epoch: 005/030 | Batch 002/021 | Loss: 0.68
Epoch: 005/030 | Batch 003/021 | Loss: 0.68
Epoch: 005/030 | Batch 004/021 | Loss: 0.67
Epoch: 005/030 | Batch 005/021 | Loss: 0.67
Epoch: 005/030 | Batch 006/021 | Loss: 0.66
Epoch: 005/030 | Batch 007/021 | Loss: 0.69
Epoch: 005/030 | Batch 008/021 | Loss: 0.67
Epoch: 005/030 | Batch 009/021 | Loss: 0.68
Epoch: 005/030 | Batch 010/021 | Loss: 0.67
Epoch: 005/030 | Batch 011/021 | Loss: 0.69
Epoch: 005/030 | Batch 012/021 | Loss: 0.68
Epoch: 005/030 | Batch 013/021 | Loss: 0.69
Epoch: 005/030 | Batch 014/021 | Loss: 0.70
Epoch: 005/030 | Batch 015/021 | Loss: 0.65
Epoch: 005/030 | Batch 016/021 | Loss: 0.67
Epoch: 005/030 | Batch 017/021 | Loss: 0.69
Epoch: 005/030 | Batch 018/021 | Loss: 0.69
Epoch: 005/030 | Batch 019/021 | Loss: 0.64
Epoch: 005/030 | Batch 020/021 | Loss: 0.66
Epoch: 005/030 | Batch 021/021 | Loss: 0.69
Epoch: 005/030 train acc: 78.488 % | val acc: 79.545 % | time: 2.478 s
Epoch: 006/030 | Batch 000/021 | Loss: 0.67
Epoch: 006/030 | Batch 001/021 | Loss: 0.67
Epoch: 006/030 | Batch 002/021 | Loss: 0.65
Epoch: 006/030 | Batch 003/021 | Loss: 0.65
Epoch: 006/030 | Batch 004/021 | Loss: 0.69
Epoch: 006/030 | Batch 005/021 | Loss: 0.68
Epoch: 006/030 | Batch 006/021 | Loss: 0.66
Epoch: 006/030 | Batch 007/021 | Loss: 0.69
Epoch: 006/030 | Batch 008/021 | Loss: 0.67
```

```
Epoch: 006/030 | Batch 009/021 | Loss: 0.66
Epoch: 006/030 | Batch 010/021 | Loss: 0.65
Epoch: 006/030 | Batch 011/021 | Loss: 0.68
Epoch: 006/030 | Batch 012/021 | Loss: 0.67
Epoch: 006/030 | Batch 013/021 | Loss: 0.67
Epoch: 006/030 | Batch 014/021 | Loss: 0.68
Epoch: 006/030 | Batch 015/021 | Loss: 0.68
Epoch: 006/030 | Batch 016/021 | Loss: 0.66
Epoch: 006/030 | Batch 017/021 | Loss: 0.67
Epoch: 006/030 | Batch 018/021 | Loss: 0.67
Epoch: 006/030 | Batch 019/021 | Loss: 0.67
Epoch: 006/030 | Batch 020/021 | Loss: 0.66
Epoch: 006/030 | Batch 021/021 | Loss: 0.64
Epoch: 006/030 train acc: 81.977 % | val acc: 81.818 % | time: 2.481 s
Epoch: 007/030 | Batch 000/021 | Loss: 0.68
Epoch: 007/030 | Batch 001/021 | Loss: 0.65
Epoch: 007/030 | Batch 002/021 | Loss: 0.69
Epoch: 007/030 | Batch 003/021 | Loss: 0.66
Epoch: 007/030 | Batch 004/021 | Loss: 0.68
Epoch: 007/030 | Batch 005/021 | Loss: 0.67
Epoch: 007/030 | Batch 006/021 | Loss: 0.65
Epoch: 007/030 | Batch 007/021 | Loss: 0.67
Epoch: 007/030 | Batch 008/021 | Loss: 0.69
Epoch: 007/030 | Batch 009/021 | Loss: 0.67
Epoch: 007/030 | Batch 010/021 | Loss: 0.65
Epoch: 007/030 | Batch 011/021 | Loss: 0.66
Epoch: 007/030 | Batch 012/021 | Loss: 0.65
Epoch: 007/030 | Batch 013/021 | Loss: 0.65
Epoch: 007/030 | Batch 014/021 | Loss: 0.62
Epoch: 007/030 | Batch 015/021 | Loss: 0.68
Epoch: 007/030 | Batch 016/021 | Loss: 0.65
Epoch: 007/030 | Batch 017/021 | Loss: 0.68
Epoch: 007/030 | Batch 018/021 | Loss: 0.68
Epoch: 007/030 | Batch 019/021 | Loss: 0.66
Epoch: 007/030 | Batch 020/021 | Loss: 0.66
Epoch: 007/030 | Batch 021/021 | Loss: 0.69
Epoch: 007/030 train acc: 84.884 % | val acc: 81.818 % | time: 2.478 s
Epoch: 008/030 | Batch 000/021 | Loss: 0.63
Epoch: 008/030 | Batch 001/021 | Loss: 0.65
Epoch: 008/030 | Batch 002/021 | Loss: 0.65
Epoch: 008/030 | Batch 003/021 | Loss: 0.64
Epoch: 008/030 | Batch 004/021 | Loss: 0.67
Epoch: 008/030 | Batch 005/021 | Loss: 0.67
Epoch: 008/030 | Batch 006/021 | Loss: 0.63
Epoch: 008/030 | Batch 007/021 | Loss: 0.65
Epoch: 008/030 | Batch 008/021 | Loss: 0.68
Epoch: 008/030 | Batch 009/021 | Loss: 0.69
Epoch: 008/030 | Batch 010/021 | Loss: 0.63
Epoch: 008/030 | Batch 011/021 | Loss: 0.64
Epoch: 008/030 | Batch 012/021 | Loss: 0.66
Epoch: 008/030 | Batch 013/021 | Loss: 0.66
Epoch: 008/030 | Batch 014/021 | Loss: 0.68
Epoch: 008/030 | Batch 015/021 | Loss: 0.65
Epoch: 008/030 | Batch 016/021 | Loss: 0.64
Epoch: 008/030 | Batch 017/021 | Loss: 0.65
Epoch: 008/030 | Batch 018/021 | Loss: 0.66
Epoch: 008/030 | Batch 019/021 | Loss: 0.67
Epoch: 008/030 | Batch 020/021 | Loss: 0.68
Epoch: 008/030 | Batch 021/021 | Loss: 0.70

Epoch: 008/030 train acc: 84.302 % | val acc: 81.818 % | time: 2.482 s
Epoch: 009/030 | Batch 000/021 | Loss: 0.66
Epoch: 009/030 | Batch 001/021 | Loss: 0.66
Epoch: 009/030 | Batch 002/021 | Loss: 0.61
Epoch: 009/030 | Batch 003/021 | Loss: 0.64
Epoch: 009/030 | Batch 004/021 | Loss: 0.67
Epoch: 009/030 | Batch 005/021 | Loss: 0.65
Epoch: 009/030 | Batch 006/021 | Loss: 0.63
```

```
Epoch: 009/030 | Batch 007/021 | Loss: 0.63
Epoch: 009/030 | Batch 008/021 | Loss: 0.63
Epoch: 009/030 | Batch 009/021 | Loss: 0.67
Epoch: 009/030 | Batch 010/021 | Loss: 0.68
Epoch: 009/030 | Batch 011/021 | Loss: 0.63
Epoch: 009/030 | Batch 012/021 | Loss: 0.64
Epoch: 009/030 | Batch 013/021 | Loss: 0.62
Epoch: 009/030 | Batch 014/021 | Loss: 0.67
Epoch: 009/030 | Batch 015/021 | Loss: 0.61
Epoch: 009/030 | Batch 016/021 | Loss: 0.63
Epoch: 009/030 | Batch 017/021 | Loss: 0.68
Epoch: 009/030 | Batch 018/021 | Loss: 0.70
Epoch: 009/030 | Batch 019/021 | Loss: 0.69
Epoch: 009/030 | Batch 020/021 | Loss: 0.64
Epoch: 009/030 | Batch 021/021 | Loss: 0.66
Epoch: 009/030 train acc: 86.628 % | val acc: 84.091 % | time: 2.481 s
Epoch: 010/030 | Batch 000/021 | Loss: 0.64
Epoch: 010/030 | Batch 001/021 | Loss: 0.67
Epoch: 010/030 | Batch 002/021 | Loss: 0.63
Epoch: 010/030 | Batch 003/021 | Loss: 0.59
Epoch: 010/030 | Batch 004/021 | Loss: 0.64
Epoch: 010/030 | Batch 005/021 | Loss: 0.64
Epoch: 010/030 | Batch 006/021 | Loss: 0.64
Epoch: 010/030 | Batch 007/021 | Loss: 0.65
Epoch: 010/030 | Batch 008/021 | Loss: 0.64
Epoch: 010/030 | Batch 009/021 | Loss: 0.63
Epoch: 010/030 | Batch 010/021 | Loss: 0.64
Epoch: 010/030 | Batch 011/021 | Loss: 0.64
Epoch: 010/030 | Batch 012/021 | Loss: 0.62
Epoch: 010/030 | Batch 013/021 | Loss: 0.61
Epoch: 010/030 | Batch 014/021 | Loss: 0.68
Epoch: 010/030 | Batch 015/021 | Loss: 0.66
Epoch: 010/030 | Batch 016/021 | Loss: 0.63
Epoch: 010/030 | Batch 017/021 | Loss: 0.67
Epoch: 010/030 | Batch 018/021 | Loss: 0.64
Epoch: 010/030 | Batch 019/021 | Loss: 0.61
Epoch: 010/030 | Batch 020/021 | Loss: 0.63
Epoch: 010/030 | Batch 021/021 | Loss: 0.65
Epoch: 010/030 train acc: 88.372 % | val acc: 86.364 % | time: 2.488 s
Epoch: 011/030 | Batch 000/021 | Loss: 0.62
Epoch: 011/030 | Batch 001/021 | Loss: 0.67
Epoch: 011/030 | Batch 002/021 | Loss: 0.68
Epoch: 011/030 | Batch 003/021 | Loss: 0.64
Epoch: 011/030 | Batch 004/021 | Loss: 0.64
Epoch: 011/030 | Batch 005/021 | Loss: 0.62
Epoch: 011/030 | Batch 006/021 | Loss: 0.61
Epoch: 011/030 | Batch 007/021 | Loss: 0.57
Epoch: 011/030 | Batch 008/021 | Loss: 0.60
Epoch: 011/030 | Batch 009/021 | Loss: 0.63
Epoch: 011/030 | Batch 010/021 | Loss: 0.66
Epoch: 011/030 | Batch 011/021 | Loss: 0.62
Epoch: 011/030 | Batch 012/021 | Loss: 0.64
Epoch: 011/030 | Batch 013/021 | Loss: 0.63
Epoch: 011/030 | Batch 014/021 | Loss: 0.62
Epoch: 011/030 | Batch 015/021 | Loss: 0.62
Epoch: 011/030 | Batch 016/021 | Loss: 0.59
Epoch: 011/030 | Batch 017/021 | Loss: 0.62
Epoch: 011/030 | Batch 018/021 | Loss: 0.65
Epoch: 011/030 | Batch 019/021 | Loss: 0.63
Epoch: 011/030 | Batch 020/021 | Loss: 0.62
Epoch: 011/030 | Batch 021/021 | Loss: 0.68
Epoch: 011/030 train acc: 90.116 % | val acc: 86.364 % | time: 2.486 s
Epoch: 012/030 | Batch 000/021 | Loss: 0.66
Epoch: 012/030 | Batch 001/021 | Loss: 0.55
Epoch: 012/030 | Batch 002/021 | Loss: 0.61
Epoch: 012/030 | Batch 003/021 | Loss: 0.58
Epoch: 012/030 | Batch 004/021 | Loss: 0.60
Epoch: 012/030 | Batch 005/021 | Loss: 0.59
Epoch: 012/030 | Batch 006/021 | Loss: 0.65
Epoch: 012/030 | Batch 007/021 | Loss: 0.66
Epoch: 012/030 | Batch 008/021 | Loss: 0.62
Epoch: 012/030 | Batch 009/021 | Loss: 0.62
Epoch: 012/030 | Batch 010/021 | Loss: 0.63
Epoch: 012/030 | Batch 011/021 | Loss: 0.66
Epoch: 012/030 | Batch 012/021 | Loss: 0.61
Epoch: 012/030 | Batch 013/021 | Loss: 0.61
Epoch: 012/030 | Batch 014/021 | Loss: 0.67
Epoch: 012/030 | Batch 015/021 | Loss: 0.60
Epoch: 012/030 | Batch 016/021 | Loss: 0.62
Epoch: 012/030 | Batch 017/021 | Loss: 0.61
Epoch: 012/030 | Batch 018/021 | Loss: 0.64
Epoch: 012/030 | Batch 019/021 | Loss: 0.64
Epoch: 012/030 | Batch 020/021 | Loss: 0.56
Epoch: 012/030 | Batch 021/021 | Loss: 0.51
Epoch: 012/030 train acc: 90.116 % | val acc: 84.091 % | time: 2.533 s
Epoch: 013/030 | Batch 000/021 | Loss: 0.61
Epoch: 013/030 | Batch 001/021 | Loss: 0.56
Epoch: 013/030 | Batch 002/021 | Loss: 0.64
Epoch: 013/030 | Batch 003/021 | Loss: 0.57
Epoch: 013/030 | Batch 004/021 | Loss: 0.63
Epoch: 013/030 | Batch 005/021 | Loss: 0.67
Epoch: 013/030 | Batch 006/021 | Loss: 0.64
Epoch: 013/030 | Batch 007/021 | Loss: 0.60
Epoch: 013/030 | Batch 008/021 | Loss: 0.58
Epoch: 013/030 | Batch 009/021 | Loss: 0.57
Epoch: 013/030 | Batch 010/021 | Loss: 0.57
Epoch: 013/030 | Batch 011/021 | Loss: 0.60
Epoch: 013/030 | Batch 012/021 | Loss: 0.59
Epoch: 013/030 | Batch 013/021 | Loss: 0.60
Epoch: 013/030 | Batch 014/021 | Loss: 0.60
Epoch: 013/030 | Batch 015/021 | Loss: 0.62
Epoch: 013/030 | Batch 016/021 | Loss: 0.60
Epoch: 013/030 | Batch 017/021 | Loss: 0.61
Epoch: 013/030 | Batch 018/021 | Loss: 0.59
Epoch: 013/030 | Batch 019/021 | Loss: 0.57
Epoch: 013/030 | Batch 020/021 | Loss: 0.59
Epoch: 013/030 | Batch 021/021 | Loss: 0.70
Epoch: 013/030 train acc: 91.279 % | val acc: 84.091 % | time: 2.491 s
Epoch: 014/030 | Batch 000/021 | Loss: 0.63
Epoch: 014/030 | Batch 001/021 | Loss: 0.59
Epoch: 014/030 | Batch 002/021 | Loss: 0.65
```

```
Epoch: 014/030 | Batch 003/021 | Loss: 0.59
Epoch: 014/030 | Batch 004/021 | Loss: 0.63
Epoch: 014/030 | Batch 005/021 | Loss: 0.57
Epoch: 014/030 | Batch 006/021 | Loss: 0.54
Epoch: 014/030 | Batch 007/021 | Loss: 0.58
Epoch: 014/030 | Batch 008/021 | Loss: 0.57
Epoch: 014/030 | Batch 009/021 | Loss: 0.54
Epoch: 014/030 | Batch 010/021 | Loss: 0.61
Epoch: 014/030 | Batch 011/021 | Loss: 0.59
Epoch: 014/030 | Batch 012/021 | Loss: 0.61
Epoch: 014/030 | Batch 013/021 | Loss: 0.62
Epoch: 014/030 | Batch 014/021 | Loss: 0.72
Epoch: 014/030 | Batch 015/021 | Loss: 0.54
Epoch: 014/030 | Batch 016/021 | Loss: 0.60
Epoch: 014/030 | Batch 017/021 | Loss: 0.58
Epoch: 014/030 | Batch 018/021 | Loss: 0.57
Epoch: 014/030 | Batch 019/021 | Loss: 0.58
Epoch: 014/030 | Batch 020/021 | Loss: 0.56
Epoch: 014/030 | Batch 021/021 | Loss: 0.65
Epoch: 014/030 train acc: 91.860 % | val acc: 84.091 % | time: 2.500 s
Epoch: 015/030 | Batch 000/021 | Loss: 0.60
Epoch: 015/030 | Batch 001/021 | Loss: 0.53
Epoch: 015/030 | Batch 002/021 | Loss: 0.53
Epoch: 015/030 | Batch 003/021 | Loss: 0.58
Epoch: 015/030 | Batch 004/021 | Loss: 0.60
Epoch: 015/030 | Batch 005/021 | Loss: 0.59
Epoch: 015/030 | Batch 006/021 | Loss: 0.58
Epoch: 015/030 | Batch 007/021 | Loss: 0.57
Epoch: 015/030 | Batch 008/021 | Loss: 0.55
Epoch: 015/030 | Batch 009/021 | Loss: 0.57
Epoch: 015/030 | Batch 010/021 | Loss: 0.56
Epoch: 015/030 | Batch 011/021 | Loss: 0.58
Epoch: 015/030 | Batch 012/021 | Loss: 0.52
Epoch: 015/030 | Batch 013/021 | Loss: 0.55
Epoch: 015/030 | Batch 014/021 | Loss: 0.52
Epoch: 015/030 | Batch 015/021 | Loss: 0.57
Epoch: 015/030 | Batch 016/021 | Loss: 0.58
Epoch: 015/030 | Batch 017/021 | Loss: 0.62
Epoch: 015/030 | Batch 018/021 | Loss: 0.61
Epoch: 015/030 | Batch 019/021 | Loss: 0.60
Epoch: 015/030 | Batch 020/021 | Loss: 0.56
Epoch: 015/030 | Batch 021/021 | Loss: 0.60
Epoch: 015/030 train acc: 93.605 % | val acc: 81.818 % | time: 2.500 s
Epoch: 016/030 | Batch 000/021 | Loss: 0.57
Epoch: 016/030 | Batch 001/021 | Loss: 0.52
Epoch: 016/030 | Batch 002/021 | Loss: 0.51
Epoch: 016/030 | Batch 003/021 | Loss: 0.59
Epoch: 016/030 | Batch 004/021 | Loss: 0.53
Epoch: 016/030 | Batch 005/021 | Loss: 0.55
Epoch: 016/030 | Batch 006/021 | Loss: 0.54
Epoch: 016/030 | Batch 007/021 | Loss: 0.54
Epoch: 016/030 | Batch 008/021 | Loss: 0.66
Epoch: 016/030 | Batch 009/021 | Loss: 0.56
Epoch: 016/030 | Batch 010/021 | Loss: 0.55
Epoch: 016/030 | Batch 011/021 | Loss: 0.53
Epoch: 016/030 | Batch 012/021 | Loss: 0.53
Epoch: 016/030 | Batch 013/021 | Loss: 0.54
Epoch: 016/030 | Batch 014/021 | Loss: 0.51
Epoch: 016/030 | Batch 015/021 | Loss: 0.50
Epoch: 016/030 | Batch 016/021 | Loss: 0.55
Epoch: 016/030 | Batch 017/021 | Loss: 0.56
Epoch: 016/030 | Batch 018/021 | Loss: 0.51
Epoch: 016/030 | Batch 019/021 | Loss: 0.51
Epoch: 016/030 | Batch 020/021 | Loss: 0.52
Epoch: 016/030 | Batch 021/021 | Loss: 0.50
Epoch: 016/030 train acc: 92.442 % | val acc: 77.273 % | time: 2.477 s
Epoch: 017/030 | Batch 000/021 | Loss: 0.58
Epoch: 017/030 | Batch 001/021 | Loss: 0.47
Epoch: 017/030 | Batch 002/021 | Loss: 0.56
Epoch: 017/030 | Batch 003/021 | Loss: 0.52
Epoch: 017/030 | Batch 004/021 | Loss: 0.54
Epoch: 017/030 | Batch 005/021 | Loss: 0.61
Epoch: 017/030 | Batch 006/021 | Loss: 0.54
Epoch: 017/030 | Batch 007/021 | Loss: 0.46
Epoch: 017/030 | Batch 008/021 | Loss: 0.47
Epoch: 017/030 | Batch 009/021 | Loss: 0.59
Epoch: 017/030 | Batch 010/021 | Loss: 0.51
Epoch: 017/030 | Batch 011/021 | Loss: 0.51
Epoch: 017/030 | Batch 012/021 | Loss: 0.45
Epoch: 017/030 | Batch 013/021 | Loss: 0.60
Epoch: 017/030 | Batch 014/021 | Loss: 0.46
Epoch: 017/030 | Batch 015/021 | Loss: 0.47
Epoch: 017/030 | Batch 016/021 | Loss: 0.47
Epoch: 017/030 | Batch 017/021 | Loss: 0.60
Epoch: 017/030 | Batch 018/021 | Loss: 0.51
Epoch: 017/030 | Batch 019/021 | Loss: 0.55
Epoch: 017/030 | Batch 020/021 | Loss: 0.48
Epoch: 017/030 | Batch 021/021 | Loss: 0.42
Epoch: 017/030 train acc: 93.605 % | val acc: 79.545 % | time: 2.484 s
Epoch: 018/030 | Batch 000/021 | Loss: 0.44
Epoch: 018/030 | Batch 001/021 | Loss: 0.46
Epoch: 018/030 | Batch 002/021 | Loss: 0.52
Epoch: 018/030 | Batch 003/021 | Loss: 0.49
Epoch: 018/030 | Batch 004/021 | Loss: 0.53
Epoch: 018/030 | Batch 005/021 | Loss: 0.60
Epoch: 018/030 | Batch 006/021 | Loss: 0.49
Epoch: 018/030 | Batch 007/021 | Loss: 0.49
Epoch: 018/030 | Batch 008/021 | Loss: 0.61
Epoch: 018/030 | Batch 009/021 | Loss: 0.54
Epoch: 018/030 | Batch 010/021 | Loss: 0.43
Epoch: 018/030 | Batch 011/021 | Loss: 0.47
Epoch: 018/030 | Batch 012/021 | Loss: 0.60
Epoch: 018/030 | Batch 013/021 | Loss: 0.38
Epoch: 018/030 | Batch 014/021 | Loss: 0.60
Epoch: 018/030 | Batch 015/021 | Loss: 0.58
Epoch: 018/030 | Batch 016/021 | Loss: 0.47
Epoch: 018/030 | Batch 017/021 | Loss: 0.55
Epoch: 018/030 | Batch 018/021 | Loss: 0.49
Epoch: 018/030 | Batch 019/021 | Loss: 0.50
Epoch: 018/030 | Batch 020/021 | Loss: 0.47
Epoch: 018/030 | Batch 021/021 | Loss: 0.54
```

```
Epoch: 018/030 train acc: 94.186 % | val acc: 84.091 % | time: 2.486 s
Epoch: 019/030 | Batch 000/021 | Loss: 0.52
Epoch: 019/030 | Batch 001/021 | Loss: 0.60
Epoch: 019/030 | Batch 002/021 | Loss: 0.46
Epoch: 019/030 | Batch 003/021 | Loss: 0.49
Epoch: 019/030 | Batch 004/021 | Loss: 0.54
Epoch: 019/030 | Batch 005/021 | Loss: 0.43
Epoch: 019/030 | Batch 006/021 | Loss: 0.44
Epoch: 019/030 | Batch 007/021 | Loss: 0.57
Epoch: 019/030 | Batch 008/021 | Loss: 0.48
Epoch: 019/030 | Batch 009/021 | Loss: 0.42
Epoch: 019/030 | Batch 010/021 | Loss: 0.49
Epoch: 019/030 | Batch 011/021 | Loss: 0.41
Epoch: 019/030 | Batch 012/021 | Loss: 0.50
Epoch: 019/030 | Batch 013/021 | Loss: 0.43
Epoch: 019/030 | Batch 014/021 | Loss: 0.44
Epoch: 019/030 | Batch 015/021 | Loss: 0.48
Epoch: 019/030 | Batch 016/021 | Loss: 0.70
Epoch: 019/030 | Batch 017/021 | Loss: 0.58
Epoch: 019/030 | Batch 018/021 | Loss: 0.55
Epoch: 019/030 | Batch 019/021 | Loss: 0.46
Epoch: 019/030 | Batch 020/021 | Loss: 0.45
Epoch: 019/030 | Batch 021/021 | Loss: 0.40
Epoch: 019/030 train acc: 94.767 % | val acc: 84.091 % | time: 2.480 s
Epoch: 020/030 | Batch 000/021 | Loss: 0.52
Epoch: 020/030 | Batch 001/021 | Loss: 0.47
Epoch: 020/030 | Batch 002/021 | Loss: 0.44
Epoch: 020/030 | Batch 003/021 | Loss: 0.42
Epoch: 020/030 | Batch 004/021 | Loss: 0.49
Epoch: 020/030 | Batch 005/021 | Loss: 0.35
Epoch: 020/030 | Batch 006/021 | Loss: 0.42
Epoch: 020/030 | Batch 007/021 | Loss: 0.45
Epoch: 020/030 | Batch 008/021 | Loss: 0.44
Epoch: 020/030 | Batch 009/021 | Loss: 0.47
Epoch: 020/030 | Batch 010/021 | Loss: 0.52
Epoch: 020/030 | Batch 011/021 | Loss: 0.45
Epoch: 020/030 | Batch 012/021 | Loss: 0.65
Epoch: 020/030 | Batch 013/021 | Loss: 0.46
Epoch: 020/030 | Batch 014/021 | Loss: 0.53
Epoch: 020/030 | Batch 015/021 | Loss: 0.31
Epoch: 020/030 | Batch 016/021 | Loss: 0.49
Epoch: 020/030 | Batch 017/021 | Loss: 0.40
Epoch: 020/030 | Batch 018/021 | Loss: 0.37
Epoch: 020/030 | Batch 019/021 | Loss: 0.46
Epoch: 020/030 | Batch 020/021 | Loss: 0.46
Epoch: 020/030 | Batch 021/021 | Loss: 0.62
Epoch: 020/030 train acc: 94.767 % | val acc: 86.364 % | time: 2.495 s
Epoch: 021/030 | Batch 000/021 | Loss: 0.37
Epoch: 021/030 | Batch 001/021 | Loss: 0.40
Epoch: 021/030 | Batch 002/021 | Loss: 0.66
Epoch: 021/030 | Batch 003/021 | Loss: 0.41
Epoch: 021/030 | Batch 004/021 | Loss: 0.36
Epoch: 021/030 | Batch 005/021 | Loss: 0.42
Epoch: 021/030 | Batch 006/021 | Loss: 0.48
Epoch: 021/030 | Batch 007/021 | Loss: 0.39
Epoch: 021/030 | Batch 008/021 | Loss: 0.45
Epoch: 021/030 | Batch 009/021 | Loss: 0.42
Epoch: 021/030 | Batch 010/021 | Loss: 0.36
Epoch: 021/030 | Batch 011/021 | Loss: 0.52
Epoch: 021/030 | Batch 012/021 | Loss: 0.41
Epoch: 021/030 | Batch 013/021 | Loss: 0.37
Epoch: 021/030 | Batch 014/021 | Loss: 0.44
Epoch: 021/030 | Batch 015/021 | Loss: 0.55
Epoch: 021/030 | Batch 016/021 | Loss: 0.42
Epoch: 021/030 | Batch 017/021 | Loss: 0.45
Epoch: 021/030 | Batch 018/021 | Loss: 0.51
Epoch: 021/030 | Batch 019/021 | Loss: 0.40
Epoch: 021/030 | Batch 020/021 | Loss: 0.43
Epoch: 021/030 | Batch 021/021 | Loss: 0.43
Epoch: 021/030 train acc: 95.349 % | val acc: 81.818 % | time: 2.485 s
Epoch: 022/030 | Batch 000/021 | Loss: 0.46
Epoch: 022/030 | Batch 001/021 | Loss: 0.32
Epoch: 022/030 | Batch 002/021 | Loss: 0.44
Epoch: 022/030 | Batch 003/021 | Loss: 0.45
Epoch: 022/030 | Batch 004/021 | Loss: 0.52
Epoch: 022/030 | Batch 005/021 | Loss: 0.37
Epoch: 022/030 | Batch 006/021 | Loss: 0.45
Epoch: 022/030 | Batch 007/021 | Loss: 0.51
Epoch: 022/030 | Batch 008/021 | Loss: 0.36
Epoch: 022/030 | Batch 009/021 | Loss: 0.54
Epoch: 022/030 | Batch 010/021 | Loss: 0.46
Epoch: 022/030 | Batch 011/021 | Loss: 0.47
Epoch: 022/030 | Batch 012/021 | Loss: 0.36
Epoch: 022/030 | Batch 013/021 | Loss: 0.33
Epoch: 022/030 | Batch 014/021 | Loss: 0.35
Epoch: 022/030 | Batch 015/021 | Loss: 0.42
Epoch: 022/030 | Batch 016/021 | Loss: 0.42
Epoch: 022/030 | Batch 017/021 | Loss: 0.36
Epoch: 022/030 | Batch 018/021 | Loss: 0.41
Epoch: 022/030 | Batch 019/021 | Loss: 0.41
Epoch: 022/030 | Batch 020/021 | Loss: 0.52
Epoch: 022/030 | Batch 021/021 | Loss: 0.35
Epoch: 022/030 train acc: 95.930 % | val acc: 79.545 % | time: 2.520 s
Epoch: 023/030 | Batch 000/021 | Loss: 0.45
Epoch: 023/030 | Batch 001/021 | Loss: 0.39
Epoch: 023/030 | Batch 002/021 | Loss: 0.32
Epoch: 023/030 | Batch 003/021 | Loss: 0.45
Epoch: 023/030 | Batch 004/021 | Loss: 0.45
Epoch: 023/030 | Batch 005/021 | Loss: 0.40
Epoch: 023/030 | Batch 006/021 | Loss: 0.39
Epoch: 023/030 | Batch 007/021 | Loss: 0.27
Epoch: 023/030 | Batch 008/021 | Loss: 0.32
Epoch: 023/030 | Batch 009/021 | Loss: 0.41
Epoch: 023/030 | Batch 010/021 | Loss: 0.48
Epoch: 023/030 | Batch 011/021 | Loss: 0.36
Epoch: 023/030 | Batch 012/021 | Loss: 0.43
Epoch: 023/030 | Batch 013/021 | Loss: 0.51
Epoch: 023/030 | Batch 014/021 | Loss: 0.38
Epoch: 023/030 | Batch 015/021 | Loss: 0.43
Epoch: 023/030 | Batch 016/021 | Loss: 0.59
Epoch: 023/030 | Batch 017/021 | Loss: 0.44
```

```
Epoch: 023/030 | Batch 018/021 | Loss: 0.42
Epoch: 023/030 | Batch 019/021 | Loss: 0.35
Epoch: 023/030 | Batch 020/021 | Loss: 0.37
Epoch: 023/030 | Batch 021/021 | Loss: 0.52
Epoch: 023/030 train acc: 95.930 % | val acc: 81.818 % | time: 2.484 s
Epoch: 024/030 | Batch 000/021 | Loss: 0.36
Epoch: 024/030 | Batch 001/021 | Loss: 0.35
Epoch: 024/030 | Batch 002/021 | Loss: 0.46
Epoch: 024/030 | Batch 003/021 | Loss: 0.41
Epoch: 024/030 | Batch 004/021 | Loss: 0.40
Epoch: 024/030 | Batch 005/021 | Loss: 0.34
Epoch: 024/030 | Batch 006/021 | Loss: 0.42
Epoch: 024/030 | Batch 007/021 | Loss: 0.42
Epoch: 024/030 | Batch 008/021 | Loss: 0.39
Epoch: 024/030 | Batch 009/021 | Loss: 0.44
Epoch: 024/030 | Batch 010/021 | Loss: 0.33
Epoch: 024/030 | Batch 011/021 | Loss: 0.42
Epoch: 024/030 | Batch 012/021 | Loss: 0.29
Epoch: 024/030 | Batch 013/021 | Loss: 0.42
Epoch: 024/030 | Batch 014/021 | Loss: 0.39
Epoch: 024/030 | Batch 015/021 | Loss: 0.41
Epoch: 024/030 | Batch 016/021 | Loss: 0.39
Epoch: 024/030 | Batch 017/021 | Loss: 0.35

Epoch: 024/030 | Batch 018/021 | Loss: 0.40
Epoch: 024/030 | Batch 019/021 | Loss: 0.62
Epoch: 024/030 | Batch 020/021 | Loss: 0.35
Epoch: 024/030 | Batch 021/021 | Loss: 0.70
Epoch: 024/030 train acc: 97.093 % | val acc: 86.364 % | time: 2.493 s
Epoch: 025/030 | Batch 000/021 | Loss: 0.47
Epoch: 025/030 | Batch 001/021 | Loss: 0.46
Epoch: 025/030 | Batch 002/021 | Loss: 0.40
Epoch: 025/030 | Batch 003/021 | Loss: 0.34
Epoch: 025/030 | Batch 004/021 | Loss: 0.36
Epoch: 025/030 | Batch 005/021 | Loss: 0.39
Epoch: 025/030 | Batch 006/021 | Loss: 0.48
Epoch: 025/030 | Batch 007/021 | Loss: 0.34
Epoch: 025/030 | Batch 008/021 | Loss: 0.49
Epoch: 025/030 | Batch 009/021 | Loss: 0.36
Epoch: 025/030 | Batch 010/021 | Loss: 0.33
Epoch: 025/030 | Batch 011/021 | Loss: 0.37
Epoch: 025/030 | Batch 012/021 | Loss: 0.43
Epoch: 025/030 | Batch 013/021 | Loss: 0.40
Epoch: 025/030 | Batch 014/021 | Loss: 0.32
Epoch: 025/030 | Batch 015/021 | Loss: 0.47
Epoch: 025/030 | Batch 016/021 | Loss: 0.38
Epoch: 025/030 | Batch 017/021 | Loss: 0.41
Epoch: 025/030 | Batch 018/021 | Loss: 0.37
Epoch: 025/030 | Batch 019/021 | Loss: 0.37
Epoch: 025/030 | Batch 020/021 | Loss: 0.38
Epoch: 025/030 | Batch 021/021 | Loss: 0.41
Epoch: 025/030 train acc: 97.674 % | val acc: 88.636 % | time: 2.498 s
Epoch: 026/030 | Batch 000/021 | Loss: 0.40
Epoch: 026/030 | Batch 001/021 | Loss: 0.39
Epoch: 026/030 | Batch 002/021 | Loss: 0.32
Epoch: 026/030 | Batch 003/021 | Loss: 0.31
Epoch: 026/030 | Batch 004/021 | Loss: 0.31
Epoch: 026/030 | Batch 005/021 | Loss: 0.39
Epoch: 026/030 | Batch 006/021 | Loss: 0.29
Epoch: 026/030 | Batch 007/021 | Loss: 0.32
Epoch: 026/030 | Batch 008/021 | Loss: 0.35
Epoch: 026/030 | Batch 009/021 | Loss: 0.40
Epoch: 026/030 | Batch 010/021 | Loss: 0.35
Epoch: 026/030 | Batch 011/021 | Loss: 0.35
Epoch: 026/030 | Batch 012/021 | Loss: 0.39
Epoch: 026/030 | Batch 013/021 | Loss: 0.40
Epoch: 026/030 | Batch 014/021 | Loss: 0.35
Epoch: 026/030 | Batch 015/021 | Loss: 0.52
Epoch: 026/030 | Batch 016/021 | Loss: 0.50
Epoch: 026/030 | Batch 017/021 | Loss: 0.45
Epoch: 026/030 | Batch 018/021 | Loss: 0.37
Epoch: 026/030 | Batch 019/021 | Loss: 0.35
Epoch: 026/030 | Batch 020/021 | Loss: 0.40
Epoch: 026/030 | Batch 021/021 | Loss: 0.43
Epoch: 026/030 train acc: 95.930 % | val acc: 84.091 % | time: 2.511 s
Epoch: 027/030 | Batch 000/021 | Loss: 0.40
Epoch: 027/030 | Batch 001/021 | Loss: 0.31
Epoch: 027/030 | Batch 002/021 | Loss: 0.35
Epoch: 027/030 | Batch 003/021 | Loss: 0.34
Epoch: 027/030 | Batch 004/021 | Loss: 0.34
Epoch: 027/030 | Batch 005/021 | Loss: 0.43
Epoch: 027/030 | Batch 006/021 | Loss: 0.45
Epoch: 027/030 | Batch 007/021 | Loss: 0.34
Epoch: 027/030 | Batch 008/021 | Loss: 0.32
Epoch: 027/030 | Batch 009/021 | Loss: 0.31
Epoch: 027/030 | Batch 010/021 | Loss: 0.33
Epoch: 027/030 | Batch 011/021 | Loss: 0.34
Epoch: 027/030 | Batch 012/021 | Loss: 0.25
Epoch: 027/030 | Batch 013/021 | Loss: 0.28
Epoch: 027/030 | Batch 014/021 | Loss: 0.27
Epoch: 027/030 | Batch 015/021 | Loss: 0.34
Epoch: 027/030 | Batch 016/021 | Loss: 0.33
Epoch: 027/030 | Batch 017/021 | Loss: 0.34
Epoch: 027/030 | Batch 018/021 | Loss: 0.30
Epoch: 027/030 | Batch 019/021 | Loss: 0.35
Epoch: 027/030 | Batch 020/021 | Loss: 0.31
Epoch: 027/030 | Batch 021/021 | Loss: 0.39
Epoch: 027/030 train acc: 97.093 % | val acc: 81.818 % | time: 2.498 s
Epoch: 028/030 | Batch 000/021 | Loss: 0.40
Epoch: 028/030 | Batch 001/021 | Loss: 0.33
Epoch: 028/030 | Batch 002/021 | Loss: 0.36
Epoch: 028/030 | Batch 003/021 | Loss: 0.28
Epoch: 028/030 | Batch 004/021 | Loss: 0.28
Epoch: 028/030 | Batch 005/021 | Loss: 0.31
Epoch: 028/030 | Batch 006/021 | Loss: 0.29
Epoch: 028/030 | Batch 007/021 | Loss: 0.28
Epoch: 028/030 | Batch 008/021 | Loss: 0.39
Epoch: 028/030 | Batch 009/021 | Loss: 0.35
Epoch: 028/030 | Batch 010/021 | Loss: 0.31
Epoch: 028/030 | Batch 011/021 | Loss: 0.41
Epoch: 028/030 | Batch 012/021 | Loss: 0.31
Epoch: 028/030 | Batch 013/021 | Loss: 0.38
```

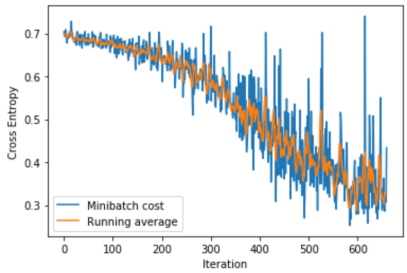
```
Epoch: 028/030 | Batch 014/021 | Loss: 0.42
Epoch: 028/030 | Batch 015/021 | Loss: 0.29
Epoch: 028/030 | Batch 016/021 | Loss: 0.39
Epoch: 028/030 | Batch 017/021 | Loss: 0.32
Epoch: 028/030 | Batch 018/021 | Loss: 0.35
Epoch: 028/030 | Batch 019/021 | Loss: 0.28
Epoch: 028/030 | Batch 020/021 | Loss: 0.34
Epoch: 028/030 | Batch 021/021 | Loss: 0.74
Epoch: 028/030 train acc: 98.256 % | val acc: 86.364 % | time: 2.508 s
Epoch: 029/030 | Batch 000/021 | Loss: 0.36
Epoch: 029/030 | Batch 001/021 | Loss: 0.29
Epoch: 029/030 | Batch 002/021 | Loss: 0.39
Epoch: 029/030 | Batch 003/021 | Loss: 0.30
Epoch: 029/030 | Batch 004/021 | Loss: 0.28
Epoch: 029/030 | Batch 005/021 | Loss: 0.26
Epoch: 029/030 | Batch 006/021 | Loss: 0.36
Epoch: 029/030 | Batch 007/021 | Loss: 0.29
Epoch: 029/030 | Batch 008/021 | Loss: 0.51
Epoch: 029/030 | Batch 009/021 | Loss: 0.46
Epoch: 029/030 | Batch 010/021 | Loss: 0.33
Epoch: 029/030 | Batch 011/021 | Loss: 0.33
Epoch: 029/030 | Batch 012/021 | Loss: 0.40
Epoch: 029/030 | Batch 013/021 | Loss: 0.29
Epoch: 029/030 | Batch 014/021 | Loss: 0.42
Epoch: 029/030 | Batch 015/021 | Loss: 0.32
Epoch: 029/030 | Batch 016/021 | Loss: 0.51
Epoch: 029/030 | Batch 017/021 | Loss: 0.37
Epoch: 029/030 | Batch 018/021 | Loss: 0.30
Epoch: 029/030 | Batch 019/021 | Loss: 0.30
Epoch: 029/030 | Batch 020/021 | Loss: 0.38
Epoch: 029/030 | Batch 021/021 | Loss: 0.38
Epoch: 029/030 train acc: 98.837 % | val acc: 88.636 % | time: 2.503 s
Epoch: 030/030 | Batch 000/021 | Loss: 0.28
Epoch: 030/030 | Batch 001/021 | Loss: 0.34
Epoch: 030/030 | Batch 002/021 | Loss: 0.27
Epoch: 030/030 | Batch 003/021 | Loss: 0.32
Epoch: 030/030 | Batch 004/021 | Loss: 0.29
Epoch: 030/030 | Batch 005/021 | Loss: 0.31
Epoch: 030/030 | Batch 006/021 | Loss: 0.38
Epoch: 030/030 | Batch 007/021 | Loss: 0.36
Epoch: 030/030 | Batch 008/021 | Loss: 0.47
Epoch: 030/030 | Batch 009/021 | Loss: 0.55
Epoch: 030/030 | Batch 010/021 | Loss: 0.31
Epoch: 030/030 | Batch 011/021 | Loss: 0.33
Epoch: 030/030 | Batch 012/021 | Loss: 0.29
Epoch: 030/030 | Batch 013/021 | Loss: 0.34
Epoch: 030/030 | Batch 014/021 | Loss: 0.30
Epoch: 030/030 | Batch 015/021 | Loss: 0.36
Epoch: 030/030 | Batch 016/021 | Loss: 0.29
Epoch: 030/030 | Batch 017/021 | Loss: 0.29
Epoch: 030/030 | Batch 018/021 | Loss: 0.32
Epoch: 030/030 | Batch 019/021 | Loss: 0.31
Epoch: 030/030 | Batch 020/021 | Loss: 0.31
Epoch: 030/030 | Batch 021/021 | Loss: 0.43
Epoch: 030/030 train acc: 99.419 % | val acc: 88.636 % | time: 2.494 s
```

Result

In [562]:

```
plt.plot(loss_list, label='Minibatch cost')
plt.plot(np.convolve(loss_list,
                     np.ones(5,)/5, mode='valid'),
         label='Running average')

plt.ylabel('Cross Entropy')
plt.xlabel('Iteration')
plt.legend()
plt.show()
```

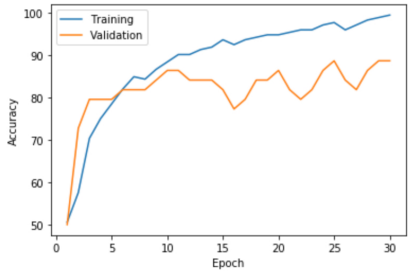


In [563]:

```
plt.plot(np.arange(1, NUM_EPOCHS+1), train_acc_list, label='Training')
plt.plot(np.arange(1, NUM_EPOCHS+1), valid_acc_list, label='Validation')

plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

```



In [564]:

```
with torch.set_grad_enabled(False):
    test_acc = compute_accuracy(model=model,
                                data_loader=dataset_loader["test"],
                                device=DEVICE)

print(f'Test ACC: {test_acc:.2f}%')
```

```
Test ACC: 88.64%
```