Deep Learning Mini-Project
Residual Network Design

Piyush Gautam *(pg2374@nyu.edu)* **Naveen Mallemala** *(nm3937@nyu.edu)*
Jayanth Guduru *(jg7162@nyu.edu)*

Abstract

In this report, we have added dropout to a PyTorch ResNet-18 model and observed the effect dropout has on the model's performance. Dropout is a machine learning technique where you remove (or "drop out") units in a neural net to simulate training large numbers of architectures simultaneously. Importantly, dropout can drastically reduce the chance of overfitting during training. An unregularized network quickly overfits on the training dataset. Training with two dropout layers with a dropout probability of 25% prevents model from overfitting. However, this brings down the training accuracy, which means a regularized network has to be trained longer. Dropout improves the model generalization. Even though the training accuracy is lower than the unregularized network, the overall validation accuracy has improved. This accounts for a lower generalization error.
GitHub: https://github.com/pg2374/Pytorch-CIFAR10-RESNET18

Introduction

The CIFAR dataset is a popular one in machine learning and computer vision. It consists of 50,000 images for training purposes with 10,000 being used as validation data which are all size 32x32 pixels arrayed across different classes such as airplanes versus cats (or vice versa). Unlike the MNSIT Dataset that has grayscale photos this collection includes RGB colors from 0-1 values each color channel measured on scale zero mean & unit variance . The normalization process ensures no bias towards any particular class; thus creating an even playing field where algorithms may compete against each other more effectively.

The deep neural networks (DNNs) are an important field of study in machine learning. These complex systems have been used to process images and text, among other things. A key feature that sets DNNs apart from traditional artificial intelligence techniques is their 'depth,' which refers to how many layers there are between any two neurons on the network.

The paths through these different processing stages will be quite lengthy because each layer adds more features while also getting harder, thus making them perfect for tasks such as image recognition or language modeling. Often, a neural network can be characterized as a linear sequence of layers with no intra-group connections. In these circumstances, the depth of a network is defined by its layer count. In recent years, the definition of "deep" has changed as it is now being measured by how many layers are in your network.

One key reason for ResNets' success is that they allow extremely deep networks with up to 1000 layers which can be used for state-of-the art performance levels when trained at high degrees off complexity and representation power. The ResNet model is a deep network with more than 100 layers. It reduced the error rate down to 3% from 7%, which was shown by vgg's predecessor. The main idea behind this type of architecture involves making use of residual connections instead; they're not designed for learning original functions but rather trying out differentiable weights on past data sets that can produce better outcomes when applied back during training sessions.

The concept of residual networks or ResNet was first introduced in [1]. It won the 2015 ILSVRC classification task and tried to solve a problem that has been prevalent until now training very deep models. The earlier primary challenge faced when training these kinds, is known as "vanishing/exploding gradients". This used normalization between different layers which enabled us with tens-of-layer depths for stochastic gradient descent (SGD) back propagation; however there arose another issue too: degradation as mentioned in [1] . This refers to saturation followed by decrease in accuracy of models with increasing depth. [1] argued that by ensuring the deeper model learns an identity mapping in a few of its layers, its performance could be brought to be as good as its shallower counterparts. But their experiments showed that used training methods failed to learn this solution yielding poorer accuracy.

Hence, it was proposed that instead of simply stacking layers upon layers, a residual mapping be explicitly ingrained in the model. This refers to saturation followed by decrease in accuracy of models with increasing depth. [1] argued that by ensuring the deeper model learns an identity mapping in a few of its layers, its performance could be brought to be as good as its shallower counterparts. But their experiments showed that the then used training methods failed to learn this solution yielding poorer accuracy. Thus, it was proposed that instead of simply stacking layers upon layers, a residual mapping be explicitly ingrained in the model. These skip connections do not introduce any extra parameters or computation complexity which not only makes it attractive in practice, but also important while comparing with plain networks. Throughout this assignment, we use ResNet-18 where “18” stands for the number of layers in the model.

Methodology

When deeper networks are able to start converging, a degradation problem has been exposed with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by over-fitting, and adding more layers to a suitably deep model leads to higher training error.[2] Hence, it was proposed that instead of simply stacking layers upon layers, a residual mapping be explicitly ingrained in the model. These skip connections do not introduce any extra parameters or computation complexity which not only makes it attractive in practice but also important while comparing with plain networks.[2] Formally, denoting the desired underlying mapping as H(x), we let the stacked nonlinear layers fit another mapping of $F(x) = H(x)$ - x. The original mapping is recast into $F(x) + x$. We hypothesize that it is easier to optimize the residual mapping than to optimize.

The original, unreferenced mapping the formulation of F(x) + x can be realized by feed-forward neural networks with “shortcut connections” (Fig. 2). Shortcut connections [2, 34, 49] are those skipping one or more layers.[2] In our case, the shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers (Fig. 2). Identity shortcut connections add neither extra parameters nor computational complexity. The entire network can still be trained end-to-end by SGD with backpropagation, and can be easily implemented using common libraries. [1]

Without Fine-Tuning:

Throughout this assignment, we use ResNet-18 where “18” stands for the number of layers in the model.[3] To create a ResNet-18 network from scratch, we train it on 50K training images without applying any augmentations using cross entropy as our loss function.[3] We optimize by stochastic gradient descent beginning with an initial learning rate of 0.05. This shows that while the model overfits to the training data, we can observe the degradation as mentioned in [1].[3] This refers to saturation followed by decrease in accuracy of models with increasing depth. [1].

The maximum accuracy achieved on the validation data for the ResNet18 model is approximately 84% when no

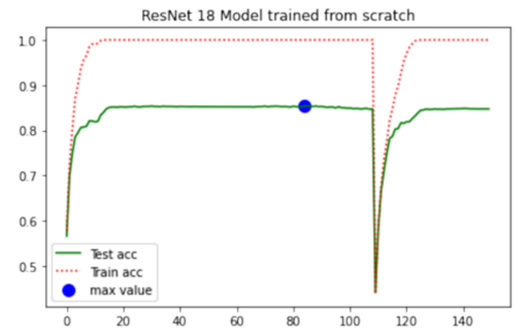


Figure 1: *ResNet Model*

finetuning is applied. Hence we decided to include Dropout applied to the ResNet18 architecture.[5] The dropout technique is a way to reduce overfitting in neural networks by preventing co-adaptation on training data. It’s an efficient method for performing model averaging so that they become more flexible when learning new patterns. The term ”dropouts” refers to both hidden units being omitted as well visible ones - this means we’re dropping items out until our network has learned what works best.[6]

With Fine-Tuning:

We are fine tuning the final linear layers that learn a mapping of 512 dimensions to 10 classes. Since ImageNet is trained on 1000 class machines, we remove an FC layer and add another with only 1 output dimension which will allow us more flexibility when it comes time for training our model later in this workbook chapter.[7] We then freeze all other Layers except topmost one - leaving its weights untouched so they can keep providing useful information about what kind of images may appear.

With Augmentation:

1. Random cropping, with size 32x32 and padding 4.
2. Random horizontal flipping with a probability of 0.5.
3. Normalize each image’s RGB channel with mean() and std().

Experiment Results

Model

We implemented the rule

$Con \rightarrow BN \rightarrow activation \rightarrow Dropout$

The first convolutional layer has 3 input channels, 64 output channels, 3x3 kernel, with stride=1 and padding=1. Followed by 8 basic blocks in 4 subgroups (i.e. 2 basic blocks in each subgroup):

1. The first sub-group contains a convolutional layer with 64 output channels, 3x3 kernel, stride=1, padding=1.

2. The second sub-group contains a convolutional layer with 128 output channels, 3x3 kernel, stride=2, padding=1.
3. The third sub-group contains a convolutional layer with 256 output channels, 3x3 kernel, stride=2, padding=1.
4. The fourth sub-group contains a convolutional layer with 512 output channels, 3x3 kernel, stride=2, padding=1.
5. The final linear layer is of 10 output classes.

For all convolutional layers, use RELU activation functions, and use batch normal layers to avoid covariant shift. Since batch-norm layers regularize the training, set the bias to 0 for all the convolutional layers.[8] Use SGD optimizers with 0.1 as the learning rate, momentum 0.9, weight decay 5e-4. The loss function is cross-entropy.[8]

First, we trained the ResNet-18 model for different batch sizes to gauge the performance of the model based upon the batch size of data loader. We achieve an accuracy of approximately 92% for batch sizes 256 and 512 for a fixed epoch of 80. Big batch size can really speed up your training, and even have better generalization performance.[8]

Figure 2: Batch size = 256

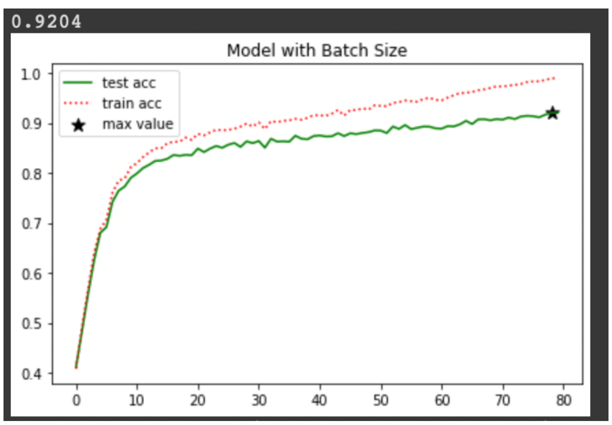
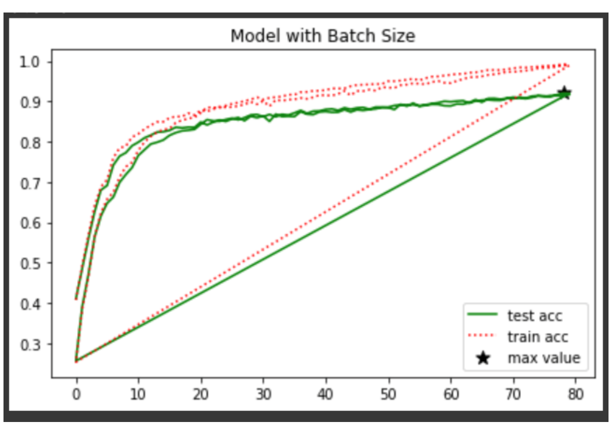


Figure 3: Batch size = 512



Secondly, we trained the ResNet-18 model for different epochs to gauge the performance of the model based upon the no of epochs the model is trained for. We achieve an accuracy of approximately 92.5% for batch size 512 and a epoch of 80 and 100. We see that the accuracy gradually increases as the number of epochs increases.[8] This is also true in the sense that the model learns and predicts better when it has been trained many times, more number of epochs.

Figure 4: Number of epochs= 80

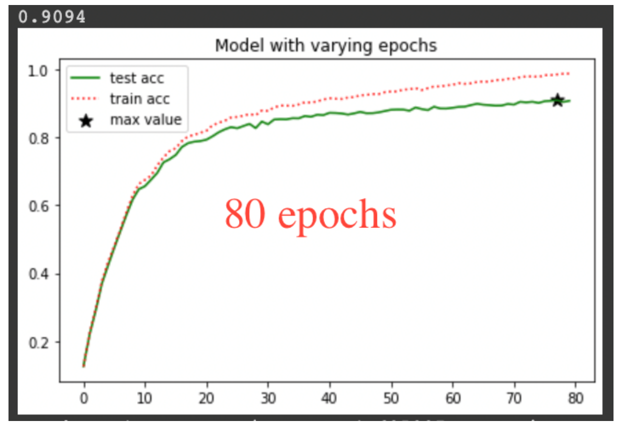
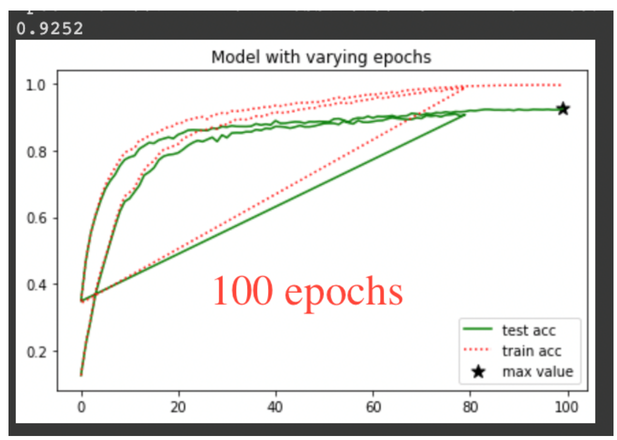
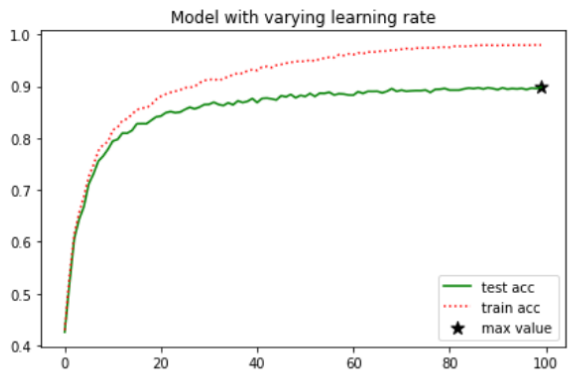


Figure 5: Number of epochs = 100



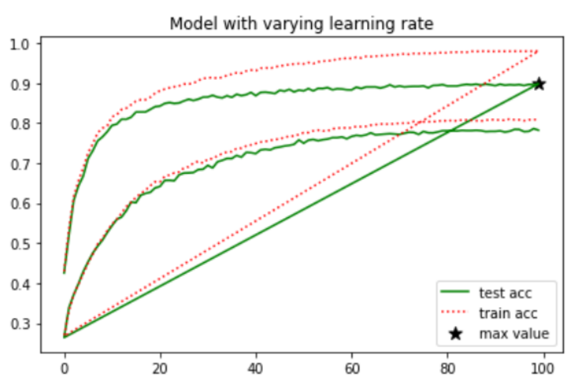
Thirdly, we trained the ResNet-18 model for different values of learning rates to gauge the performance of the model based upon the learning rate of the model. The learning rate controls how quickly the model is adapted to the problem. Smaller learning rates require more training epochs given the smaller changes made to the weights each update, whereas larger learning rates result in rapid changes and require fewer training epochs. If the learning rate is very large we will skip the optimal solution. If it is too small we will need too many iterations to converge to the best values. So using a good learning rate is crucial.

Figure 6: Learning rate = 0.01



We achieve a Validation Accuracy: 0.898200 for batch size 512 and epoch 100, with learning rate = 0.01 and Validation Accuracy: 0.782400 for learning rate 0.001 respectively.

Figure 7: Learning rate = 0.001



Conclusion

In this work, we have tried to come up with a modified residual network (ResNet) architecture using Dropout with the highest test accuracy on the CIFAR-10 image classification dataset. The Train and Validation accuracies are observed and we are able to achieve an accuracy of approx 92%. The model achieves this accuracy, with more number of epochs mostly due to the Dropout layer used in the model. This helps the model to generalise better. Specifically, The first convolutional layer has 3 input channels, 64 output channels, 3x3 kernel, with stride=1 and padding=1. Followed by 8 basic blocks in 4 subgroups (i.e. 2 basic blocks in each subgroup). The first sub-group contains a convolutional layer with 64 output channels, 3x3 kernel, stride=1, padding=1. The second sub-group contains a convolutional layer with 128 output channels, 3x3 kernel, stride=2, padding=1. The third sub-group contains a convolutional layer with 256 output channels, 3x3 kernel, stride=2, padding=1. The fourth sub-group contains a convolutional layer with 512 output channels, 3x3 kernel, stride=2, padding=1. The final linear layer is of 10 output classes.

References

1 . K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

2 . J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

3 . K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015

4 . S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015

5 . K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In ICCV, 2015.

6 . S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, 2015

7 . A. Krizhevsky. Learning multiple layers of features from tiny im- ages. Tech Report, 2009

8 . P.Sermanet,D.Eigen,X.Zhang,M.Mathieu,R.Fergus,andY.LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In ICLR, 2014.

9 . R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks, 2015.

10 . M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In Computer vision–ECCV 2014, pages 818–833. Springer, 2014.

11 . P. Goyal, P. Dollár, R.B. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, K. He, Accurate, large minibatch SGD: training imagenet in 1 hour, Tech. rep. Facebook (2017).

12 . Nesterov Y. A method of solving a convex programming problem with convergence rate o (1/k2) Soviet Mathematics Doklady, Vol. 27 (1983), pp. 372-376

13 . Liao Z., Carneiro G. On the importance of normalisation layers in deep learning with piecewise linear activation units Winter Conference on Applications of Computer Vision (WACV) (2016), pp. 1-8

14 . Fast and accurate deep network learning by exponential linear units (elus) International Conference on Learning Representations (ICLR) (2016)

15 . Xie S., Girshick R., Dollár P., Tu Z., He K. Aggregated residual transformations for deep neural networks Computer Vision and Pattern Recognition (CVPR) (2017), pp. 5987-5995

16 . Goodfellow I.J., Warde-Farley D., Mirza M., Courville A., Bengio Y. Maxout networks International Conference on Machine Learning (ICML), Vol. 28 (2013), pp. III-1319–III–1327

17 . He K., Zhang X., Ren S., Sun J. Identity mappings in deep residual networks European Conference on Computer Vision (ECCV) (2016), pp. 630-645

Mini_Project_DeepLearning

November 23, 2022

Team Members: 1. Piyush Gautam (pg2374) 2. Jayanth Guduru (jg7162) 3. Naveen Mallemala (nm3937)

GitHub : https://github.com/pg2374/Pytorch-CIFAR10-RESNET18

#Pytorch on CIFAR-10 Dataset with Batch Normalisation and Dropout on Convolution Layers output after the activation function

[2]:
```
# Install torchvision
!pip3 install torch==1.2.0+cu92 torchvision==0.4.0+cu92 -f https://download.
↪pytorch.org/whl/torch_stable.html
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Looking in links: https://download.pytorch.org/whl/torch_stable.html
Collecting torch==1.2.0+cu92
  Downloading https://download.pytorch.org/whl/cu92/torch-1.2.0%2Bcu92-cp37-cp37
m-manylinux1_x86_64.whl (663.1 MB)
     |                 | 663.1 MB 1.5 kB/s
Collecting torchvision==0.4.0+cu92
  Downloading https://download.pytorch.org/whl/cu92/torchvision-0.4.0%2Bcu92-cp3
7-cp37m-manylinux1_x86_64.whl (8.8 MB)
     |                 | 8.8 MB 89.0 MB/s
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-
packages (from torch==1.2.0+cu92) (1.21.6)
Requirement already satisfied: pillow>=4.1.1 in /usr/local/lib/python3.7/dist-
packages (from torchvision==0.4.0+cu92) (7.1.2)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages
(from torchvision==0.4.0+cu92) (1.15.0)
Installing collected packages: torch, torchvision
  Attempting uninstall: torch
    Found existing installation: torch 1.12.1+cu113
    Uninstalling torch-1.12.1+cu113:
      Successfully uninstalled torch-1.12.1+cu113
  Attempting uninstall: torchvision
    Found existing installation: torchvision 0.13.1+cu113
    Uninstalling torchvision-0.13.1+cu113:
      Successfully uninstalled torchvision-0.13.1+cu113
```

```
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
torchtext 0.13.1 requires torch==1.12.1, but you have torch 1.2.0+cu92 which is
incompatible.
torchaudio 0.12.1+cu113 requires torch==1.12.1, but you have torch 1.2.0+cu92
which is incompatible.
fastai 2.7.10 requires torch<1.14,>=1.7, but you have torch 1.2.0+cu92 which is
incompatible.
fastai 2.7.10 requires torchvision>=0.8.2, but you have torchvision 0.4.0+cu92
which is incompatible.
Successfully installed torch-1.2.0+cu92 torchvision-0.4.0+cu92
```

[3]:

```
'''ResNet in PyTorch.
Reference:
[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun
    Deep Residual Learning for Image Recognition.
[2] https://github.com/kuangliu/pytorch-cifar
[3] https://github.com/abhisikdar/RESNET18-CIFAR10
[4] https://www.srose.biz/wp-content/uploads/2020/08/Batch-Size-and-Epochs.html␣
 ↪
'''
import torch
import torch.nn as nn
import torch.nn.functional as F


class BasicBlock(nn.Module):
    expansion = 1

    def __init__(self, in_planes, planes, stride=1):
        super(BasicBlock, self).__init__()
        self.conv1 = nn.Conv2d(
            in_planes, planes, kernel_size=3, stride=stride, padding=1,␣
 ↪bias=False)
        self.bn1 = nn.BatchNorm2d(planes)
        self.conv2 = nn.Conv2d(planes, planes, kernel_size=3,
                               stride=1, padding=1, bias=False)
        self.bn2 = nn.BatchNorm2d(planes)

        self.shortcut = nn.Sequential()
```

```python
        if stride != 1 or in_planes != self.expansion*planes:
            self.shortcut = nn.Sequential(
                nn.Conv2d(in_planes, self.expansion*planes,
                          kernel_size=1, stride=stride, bias=False),
                nn.BatchNorm2d(self.expansion*planes)
            )
        # Dropout after Convolutional BasicBlock
        self.dropout = nn.Dropout(0.25)

    def forward(self, x):
        out = F.relu(self.bn1(self.conv1(x)))
        out = self.dropout(out)
        out = self.bn2(self.conv2(out))
        out += self.shortcut(x)
        out = F.relu(out)
        return out




class ResNet(nn.Module):
    def __init__(self, block, num_blocks, num_classes=10):
        super(ResNet, self).__init__()
        self.in_planes = 64

        self.conv1 = nn.Conv2d(3, 64, kernel_size=3,
                               stride=1, padding=1, bias=False)
        self.bn1 = nn.BatchNorm2d(64)
        self.layer1 = self._make_layer(block, 64, num_blocks[0], stride=1)
        self.layer2 = self._make_layer(block, 128, num_blocks[1], stride=2)
        self.layer3 = self._make_layer(block, 256, num_blocks[2], stride=2)
        self.layer4 = self._make_layer(block, 512, num_blocks[3], stride=2)
        self.linear = nn.Linear(512*block.expansion, num_classes)
        self.dropout = nn.Dropout(0.25)

    def _make_layer(self, block, planes, num_blocks, stride):
        strides = [stride] + [1]*(num_blocks-1)
        layers = []
        for stride in strides:
            layers.append(block(self.in_planes, planes, stride))
            self.in_planes = planes * block.expansion
        return nn.Sequential(*layers)

    def forward(self, x):
        out = F.relu(self.bn1(self.conv1(x)))
        out = self.dropout(out)
        out = self.layer1(out)
```

```
        out = self.layer2(out)
        out = self.layer3(out)
        out = self.layer4(out)
        out = F.avg_pool2d(out, 4)
        out = out.view(out.size(0), -1)
        out = self.linear(out)
        return out


def ResNet18():
    return ResNet(BasicBlock, [2, 2, 2, 2])


def test():
    net = ResNet18()
    y = net(torch.randn(1, 3, 32, 32))
    print(y.size())
```

[ ]:

```
# # Calculating the top 1% accuracy
# def accuracy(output: torch.Tensor, target: torch.Tensor, topk=(1,)):
#     with torch.no_grad():
#         maxk = max(topk)
#         batch_size = target.size(0)
#         _, y_pred = output.topk(k=maxk, dim=1)
#         y_pred = y_pred.t()
#         target_reshaped = target.view(1, -1).expand_as(y_pred)
#         correct = (y_pred == target_reshaped)

#         list_topk_accs = []
#         for k in topk:
#             ind_which_topk_matched_truth = correct[:k]
#             flattened_indicator_which_topk_matched_truth =␣
↪ind_which_topk_matched_truth.reshape(-1).float()
#             tot_correct_topk = flattened_indicator_which_topk_matched_truth.
↪float().sum(dim=0, keepdim=True)
#             topk_acc = tot_correct_topk / batch_size
#             list_topk_accs.append(topk_acc)
#         return list_topk_accs
```

[4]:

```
import torch
from torchvision import datasets
from torchvision import transforms
import matplotlib.pyplot as plt
import torch.nn as nn
import torch.nn.functional as F
import torchvision.models as models
import torch.backends.cudnn as cudnn
```

```python
print('Is CUDA available', torch.cuda.is_available())
print('Torch', torch.__version__, 'CUDA', torch.version.cuda)
print('Device:', torch.device('cuda:0'))
```

```
Is CUDA available True
Torch 1.2.0+cu92 CUDA 9.2.148
Device: cuda:0
```

1 Data Pre-processing

1. Random cropping, with size 32x32 and padding 4
2. Random horizontal flipping with a probability of 0.5
3. Normalize each image's RGB channel with mean() and std()

[5]:
```python
data_path='../data/'
cifar=datasets.CIFAR10(data_path, train= True, download=True, transform=transforms.ToTensor())
cifar_val=datasets.CIFAR10(data_path, train=False, download= True, transform=transforms.ToTensor())
```

```
Downloading https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz to
../data/cifar-10-python.tar.gz

170500096it [00:11, 14509573.04it/s]

Extracting ../data/cifar-10-python.tar.gz to ../data/
Files already downloaded and verified
```

[6]:
```python
cifar_stack = torch.stack([img for img, _ in cifar], dim=3)
print('Shape of the CIFAR stack is',cifar_stack.shape)
mean= cifar_stack.view(3,-1).mean(dim=1)
std= cifar_stack.view(3,-1).std(dim=1)
print('Mean of training data is', mean)
print('Standard deviation of training data is', std)
```

```
Shape of the CIFAR stack is torch.Size([3, 32, 32, 50000])
Mean of training data is tensor([0.4915, 0.4823, 0.4468])
Standard deviation of training data is tensor([0.2470, 0.2435, 0.2616])
```

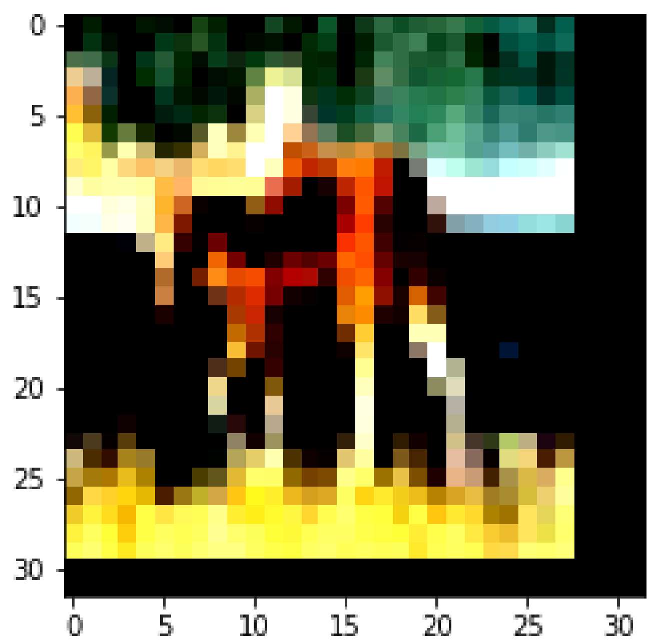
[7]:
```python
cifar_transformed = datasets.CIFAR10(data_path,train=True,download=False, transform=transforms.Compose([
                          transforms.RandomCrop(32, padding=4), transforms.RandomHorizontalFlip(p=0.5),transforms.ToTensor(),transforms.Normalize(mean,std)
]))
```

```
cifar_val_transformed = datasets.CIFAR10(data_path,train=False,download=False,␣
↪transform=transforms.Compose([
                                                                              ␣
↪                            transforms.RandomCrop(32, padding=4),
   transforms.RandomHorizontalFlip(p=0.5),transforms.ToTensor(),transforms.
↪Normalize(mean,std)
]))
```

[8]:
```
# View images

img, label = cifar_transformed[28]
plt.imshow(img.permute(1, 2, 0))
plt.show()
```

```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers).
```



#Varied the batch size for training dataset with other parameters fixed to measure the effect of batch size. Now, our Data is ready for Training.

[ ]:
```
# Parameters
batch_size=[256,512]
val_batch_size=100
num_epochs=80
# learning_rate=0.1
```

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

max_validation = 0
max_epoch=0
val_acc=[]
train_acc=[]
epochs=[]
```

[ ]:
```
for batch_no in range(len(batch_size)):
  # Train/Test Data
  train_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=batch_size[batch_no],shuffle=True,␣
↪num_workers=4)
  train_acc_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=val_batch_size,shuffle=False,␣
↪num_workers=4)
  val_loader = torch.utils.data.DataLoader(cifar_val_transformed,␣
↪batch_size=val_batch_size, shuffle=False, num_workers=4)

  # Model
  resnet18 =ResNet18()
  resnet18=resnet18.to(dev)
  loss_func= torch.nn.CrossEntropyLoss()
  optimizer = torch.optim.SGD(resnet18.parameters(), lr=0.1, momentum=0.9,␣
↪weight_decay=5e-4)
  scheduler = torch.optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=100)

  for i in range(num_epochs):
    # Training
    for imgs, labels in train_loader:
      if dev is not None:
        imgs,labels=imgs.to(dev),labels.to(dev)
      out= resnet18(imgs)
      loss=loss_func(out,labels)
      optimizer.zero_grad()
      loss.backward()
      optimizer.step()

    correct_val = 0
    total_val = 0
    correct_train_acc=0
    total_train_acc=0
    # Testing
    with torch.no_grad():
      for imgs, labels in val_loader:
        if dev is not None:
          imgs,labels=imgs.to(dev),labels.to(dev)
```

```
      outputs = resnet18(imgs)
      _, predicted = torch.max(outputs, dim=1)
      total_val += labels.shape[0]
      correct_val += int((predicted == labels).sum())
    val_acc.append(correct_val/total_val)

  # Compute Loss and Accuracy on training data
    for train_acc_imgs,train_acc_labels in train_acc_loader:
      if dev is not None:
        train_acc_imgs,train_acc_labels=train_acc_imgs.
↪to(dev),train_acc_labels.to(dev)
      train_acc_out=resnet18(train_acc_imgs)
      _, train_acc_predicted = torch.max(train_acc_out, dim=1)
      total_train_acc += train_acc_labels.shape[0]
      correct_train_acc += int((train_acc_predicted == train_acc_labels).
↪sum())
      # minibatch_acc = accuracy(train_acc, train_acc_labels,1)[0]
      # print("Top-1 training accuracy for minibatch", minibatch_acc)
    train_acc.append(correct_train_acc/total_train_acc)


    if correct_val/total_val > max_validation:
      max_validation=correct_val/total_val
      max_epoch=i
      torch.save(resnet18,'./scratch.pt' )
  epochs.append(i)



  if i%1==0:
    print("Epoch no %d:\t Train Loss: %f \t Train Accuracy: %f \t Validation␣
↪Accuracy: %f" % (i+1, float(loss), correct_train_acc / total_train_acc,␣
↪correct_val / total_val))
    # print("Train Accuracy: ", correct_train_acc / total_train_acc)
    # print("Validation Accuracy: ", correct_val / total_val)

  scheduler.step()

print(max_validation)

# Plot Train Accuracy vs Test Accuracy
plt.plot(epochs, val_acc, label="test acc", color="green", linestyle='-')
plt.plot(epochs, train_acc, label="train acc", color="red",linestyle=':')
plt.scatter([max_epoch], [max_validation],color="black", marker="*",␣
↪label="max value", s=100 )
plt.title("Model with Batch Size")
plt.legend()
```

```
plt.tight_layout()
plt.show()
```

```
Epoch no 1:      Train Loss: 2.059316    Train Accuracy: 0.407820
Validation Accuracy: 0.410500
```

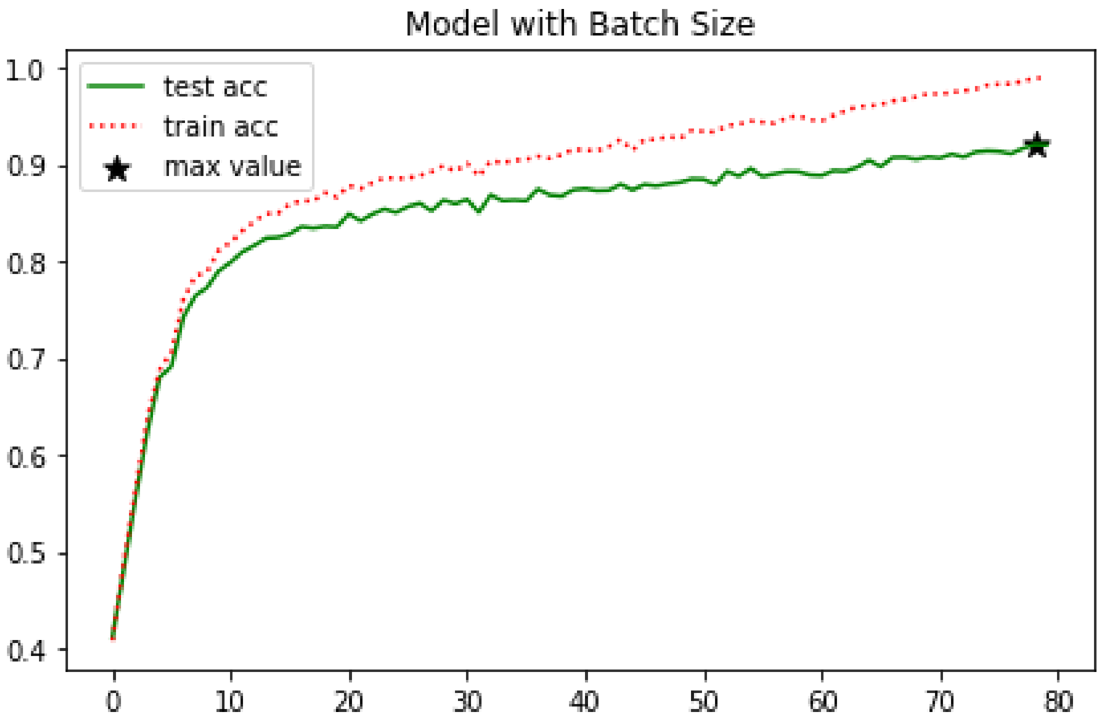
```
/usr/local/lib/python3.7/dist-packages/torch/serialization.py:256: UserWarning:
Couldn't retrieve source code for container of type ResNet. It won't be checked
for correctness upon loading.
  "type " + obj.__name__ + ". It won't be checked "
/usr/local/lib/python3.7/dist-packages/torch/serialization.py:256: UserWarning:
Couldn't retrieve source code for container of type BasicBlock. It won't be
checked for correctness upon loading.
  "type " + obj.__name__ + ". It won't be checked "
```

```
Epoch no 2:      Train Loss: 1.409685    Train Accuracy: 0.491840
Validation Accuracy: 0.483800
Epoch no 3:      Train Loss: 1.353077    Train Accuracy: 0.567820
Validation Accuracy: 0.558500
Epoch no 4:      Train Loss: 1.023971    Train Accuracy: 0.640840
Validation Accuracy: 0.626000
Epoch no 5:      Train Loss: 0.733464    Train Accuracy: 0.687800
Validation Accuracy: 0.679900
Epoch no 6:      Train Loss: 0.746233    Train Accuracy: 0.706020
Validation Accuracy: 0.691300
Epoch no 7:      Train Loss: 0.760231    Train Accuracy: 0.760340
Validation Accuracy: 0.742000
Epoch no 8:      Train Loss: 0.619271    Train Accuracy: 0.784280
Validation Accuracy: 0.764100
Epoch no 9:      Train Loss: 0.762107    Train Accuracy: 0.788940
Validation Accuracy: 0.772600
Epoch no 10:     Train Loss: 0.454060    Train Accuracy: 0.811220
Validation Accuracy: 0.789900
Epoch no 11:     Train Loss: 0.462879    Train Accuracy: 0.819220
Validation Accuracy: 0.798700
Epoch no 12:     Train Loss: 0.415925    Train Accuracy: 0.831320
Validation Accuracy: 0.809500
Epoch no 13:     Train Loss: 0.558582    Train Accuracy: 0.840680
Validation Accuracy: 0.816000
Epoch no 14:     Train Loss: 0.451855    Train Accuracy: 0.848940
Validation Accuracy: 0.823700
Epoch no 15:     Train Loss: 0.551994    Train Accuracy: 0.849280
Validation Accuracy: 0.824500
Epoch no 16:     Train Loss: 0.394413    Train Accuracy: 0.859080
Validation Accuracy: 0.828000
Epoch no 17:     Train Loss: 0.585508    Train Accuracy: 0.862120
Validation Accuracy: 0.835600
Epoch no 18:     Train Loss: 0.385920    Train Accuracy: 0.862140
```

```
Validation Accuracy: 0.834000
Epoch no 19:     Train Loss: 0.518638    Train Accuracy: 0.870620
Validation Accuracy: 0.835900
Epoch no 20:     Train Loss: 0.487737    Train Accuracy: 0.865820
Validation Accuracy: 0.835200
Epoch no 21:     Train Loss: 0.473014    Train Accuracy: 0.878020
Validation Accuracy: 0.848600
Epoch no 22:     Train Loss: 0.545335    Train Accuracy: 0.874780
Validation Accuracy: 0.841400
Epoch no 23:     Train Loss: 0.446088    Train Accuracy: 0.880780
Validation Accuracy: 0.848400
Epoch no 24:     Train Loss: 0.496873    Train Accuracy: 0.886240
Validation Accuracy: 0.853700
Epoch no 25:     Train Loss: 0.304612    Train Accuracy: 0.885200
Validation Accuracy: 0.850100
Epoch no 26:     Train Loss: 0.419857    Train Accuracy: 0.885760
Validation Accuracy: 0.856300
Epoch no 27:     Train Loss: 0.481881    Train Accuracy: 0.888340
Validation Accuracy: 0.859700
Epoch no 28:     Train Loss: 0.424157    Train Accuracy: 0.892260
Validation Accuracy: 0.851800
Epoch no 29:     Train Loss: 0.383066    Train Accuracy: 0.898640
Validation Accuracy: 0.863000
Epoch no 30:     Train Loss: 0.357070    Train Accuracy: 0.892580
Validation Accuracy: 0.859300
Epoch no 31:     Train Loss: 0.220823    Train Accuracy: 0.900920
Validation Accuracy: 0.863700
Epoch no 32:     Train Loss: 0.345880    Train Accuracy: 0.887520
Validation Accuracy: 0.850200
Epoch no 33:     Train Loss: 0.262769    Train Accuracy: 0.903620
Validation Accuracy: 0.868200
Epoch no 34:     Train Loss: 0.385984    Train Accuracy: 0.901320
Validation Accuracy: 0.862300
Epoch no 35:     Train Loss: 0.462091    Train Accuracy: 0.904080
Validation Accuracy: 0.863000
Epoch no 36:     Train Loss: 0.340670    Train Accuracy: 0.904980
Validation Accuracy: 0.862200
Epoch no 37:     Train Loss: 0.410593    Train Accuracy: 0.908240
Validation Accuracy: 0.874300
Epoch no 38:     Train Loss: 0.315806    Train Accuracy: 0.906060
Validation Accuracy: 0.868200
Epoch no 39:     Train Loss: 0.233351    Train Accuracy: 0.909880
Validation Accuracy: 0.867100
Epoch no 40:     Train Loss: 0.181170    Train Accuracy: 0.915440
Validation Accuracy: 0.873600
Epoch no 41:     Train Loss: 0.293333    Train Accuracy: 0.914700
Validation Accuracy: 0.874700
Epoch no 42:     Train Loss: 0.180180    Train Accuracy: 0.914160
```

```
Validation Accuracy: 0.872700
Epoch no 43:     Train Loss: 0.355048    Train Accuracy: 0.918340
Validation Accuracy: 0.873100
Epoch no 44:     Train Loss: 0.202712    Train Accuracy: 0.925700
Validation Accuracy: 0.879300
Epoch no 45:     Train Loss: 0.191978    Train Accuracy: 0.914500
Validation Accuracy: 0.873400
Epoch no 46:     Train Loss: 0.208726    Train Accuracy: 0.925540
Validation Accuracy: 0.879100
Epoch no 47:     Train Loss: 0.150810    Train Accuracy: 0.925380
Validation Accuracy: 0.877500
Epoch no 48:     Train Loss: 0.257669    Train Accuracy: 0.928900
Validation Accuracy: 0.879800
Epoch no 49:     Train Loss: 0.201115    Train Accuracy: 0.926400
Validation Accuracy: 0.881300
Epoch no 50:     Train Loss: 0.330637    Train Accuracy: 0.934980
Validation Accuracy: 0.884800
Epoch no 51:     Train Loss: 0.386218    Train Accuracy: 0.933960
Validation Accuracy: 0.884500
Epoch no 52:     Train Loss: 0.194330    Train Accuracy: 0.933120
Validation Accuracy: 0.879500
Epoch no 53:     Train Loss: 0.158355    Train Accuracy: 0.939980
Validation Accuracy: 0.892600
Epoch no 54:     Train Loss: 0.287424    Train Accuracy: 0.941300
Validation Accuracy: 0.887800
Epoch no 55:     Train Loss: 0.182691    Train Accuracy: 0.944640
Validation Accuracy: 0.895600
Epoch no 56:     Train Loss: 0.051513    Train Accuracy: 0.943440
Validation Accuracy: 0.887500
Epoch no 57:     Train Loss: 0.141267    Train Accuracy: 0.942240
Validation Accuracy: 0.890100
Epoch no 58:     Train Loss: 0.130443    Train Accuracy: 0.947700
Validation Accuracy: 0.892500
Epoch no 59:     Train Loss: 0.198764    Train Accuracy: 0.950200
Validation Accuracy: 0.892300
Epoch no 60:     Train Loss: 0.148120    Train Accuracy: 0.945900
Validation Accuracy: 0.889000
Epoch no 61:     Train Loss: 0.297293    Train Accuracy: 0.945000
Validation Accuracy: 0.888400
Epoch no 62:     Train Loss: 0.174329    Train Accuracy: 0.950480
Validation Accuracy: 0.893400
Epoch no 63:     Train Loss: 0.135455    Train Accuracy: 0.955100
Validation Accuracy: 0.892800
Epoch no 64:     Train Loss: 0.109519    Train Accuracy: 0.959580
Validation Accuracy: 0.897000
Epoch no 65:     Train Loss: 0.159389    Train Accuracy: 0.960060
Validation Accuracy: 0.904000
Epoch no 66:     Train Loss: 0.063829    Train Accuracy: 0.961780
```

```
Validation Accuracy: 0.897600
Epoch no 67:     Train Loss: 0.147558    Train Accuracy: 0.965300
Validation Accuracy: 0.906900
Epoch no 68:     Train Loss: 0.151854    Train Accuracy: 0.966980
Validation Accuracy: 0.907200
Epoch no 69:     Train Loss: 0.138293    Train Accuracy: 0.969600
Validation Accuracy: 0.905100
Epoch no 70:     Train Loss: 0.063318    Train Accuracy: 0.973140
Validation Accuracy: 0.907500
Epoch no 71:     Train Loss: 0.040865    Train Accuracy: 0.972500
Validation Accuracy: 0.906400
Epoch no 72:     Train Loss: 0.146871    Train Accuracy: 0.974420
Validation Accuracy: 0.910600
Epoch no 73:     Train Loss: 0.093193    Train Accuracy: 0.975900
Validation Accuracy: 0.907500
Epoch no 74:     Train Loss: 0.081559    Train Accuracy: 0.977800
Validation Accuracy: 0.912700
Epoch no 75:     Train Loss: 0.035086    Train Accuracy: 0.981780
Validation Accuracy: 0.913900
Epoch no 76:     Train Loss: 0.064822    Train Accuracy: 0.982800
Validation Accuracy: 0.913300
Epoch no 77:     Train Loss: 0.120495    Train Accuracy: 0.983160
Validation Accuracy: 0.911000
Epoch no 78:     Train Loss: 0.009831    Train Accuracy: 0.985680
Validation Accuracy: 0.916300
Epoch no 79:     Train Loss: 0.041006    Train Accuracy: 0.989080
Validation Accuracy: 0.920400
Epoch no 80:     Train Loss: 0.053192    Train Accuracy: 0.989060
Validation Accuracy: 0.919800
0.9204
```

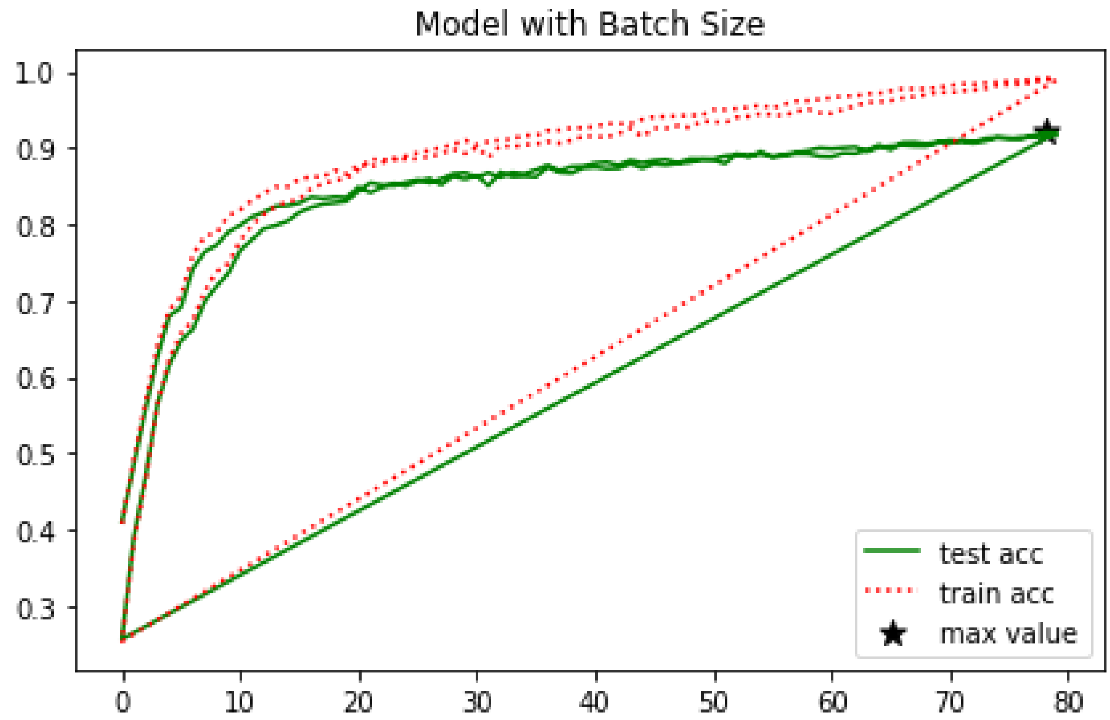



```
Epoch no 1:       Train Loss: 1.851783    Train Accuracy: 0.254240
Validation Accuracy: 0.256800
Epoch no 2:       Train Loss: 1.632281    Train Accuracy: 0.382580
Validation Accuracy: 0.389900
Epoch no 3:       Train Loss: 1.505038    Train Accuracy: 0.465100
Validation Accuracy: 0.470600
Epoch no 4:       Train Loss: 1.293925    Train Accuracy: 0.563840
Validation Accuracy: 0.563800
Epoch no 5:       Train Loss: 1.042036    Train Accuracy: 0.623260
Validation Accuracy: 0.615700
Epoch no 6:       Train Loss: 1.019472    Train Accuracy: 0.657540
Validation Accuracy: 0.647500
Epoch no 7:       Train Loss: 0.872649    Train Accuracy: 0.674360
Validation Accuracy: 0.662700
Epoch no 8:       Train Loss: 0.737331    Train Accuracy: 0.712220
Validation Accuracy: 0.699100
Epoch no 9:       Train Loss: 0.681784    Train Accuracy: 0.737900
Validation Accuracy: 0.718600
Epoch no 10:      Train Loss: 0.717634    Train Accuracy: 0.749120
Validation Accuracy: 0.735800
Epoch no 11:      Train Loss: 0.684109    Train Accuracy: 0.778700
Validation Accuracy: 0.765800
Epoch no 12:      Train Loss: 0.542677    Train Accuracy: 0.799140
Validation Accuracy: 0.780100
```

```
Epoch no 13:     Train Loss: 0.548151    Train Accuracy: 0.813120
Validation Accuracy: 0.794200
Epoch no 14:     Train Loss: 0.536025    Train Accuracy: 0.823360
Validation Accuracy: 0.798400
Epoch no 15:     Train Loss: 0.425151    Train Accuracy: 0.829720
Validation Accuracy: 0.804100
Epoch no 16:     Train Loss: 0.457473    Train Accuracy: 0.832280
Validation Accuracy: 0.815600
Epoch no 17:     Train Loss: 0.488935    Train Accuracy: 0.847120
Validation Accuracy: 0.822300
Epoch no 18:     Train Loss: 0.446378    Train Accuracy: 0.853100
Validation Accuracy: 0.826700
Epoch no 19:     Train Loss: 0.376557    Train Accuracy: 0.857160
Validation Accuracy: 0.828300
Epoch no 20:     Train Loss: 0.394112    Train Accuracy: 0.860660
Validation Accuracy: 0.830500
Epoch no 21:     Train Loss: 0.462066    Train Accuracy: 0.871400
Validation Accuracy: 0.843200
Epoch no 22:     Train Loss: 0.310641    Train Accuracy: 0.884940
Validation Accuracy: 0.854400
Epoch no 23:     Train Loss: 0.383523    Train Accuracy: 0.884580
Validation Accuracy: 0.851400
Epoch no 24:     Train Loss: 0.310772    Train Accuracy: 0.884140
Validation Accuracy: 0.850400
Epoch no 25:     Train Loss: 0.323176    Train Accuracy: 0.889220
Validation Accuracy: 0.854800
Epoch no 26:     Train Loss: 0.322214    Train Accuracy: 0.893800
Validation Accuracy: 0.858200
Epoch no 27:     Train Loss: 0.329695    Train Accuracy: 0.896760
Validation Accuracy: 0.854100
Epoch no 28:     Train Loss: 0.353648    Train Accuracy: 0.900440
Validation Accuracy: 0.859400
Epoch no 29:     Train Loss: 0.279826    Train Accuracy: 0.904440
Validation Accuracy: 0.866100
Epoch no 30:     Train Loss: 0.307256    Train Accuracy: 0.910060
Validation Accuracy: 0.867200
Epoch no 31:     Train Loss: 0.337365    Train Accuracy: 0.903880
Validation Accuracy: 0.862000
Epoch no 32:     Train Loss: 0.274735    Train Accuracy: 0.908620
Validation Accuracy: 0.867400
Epoch no 33:     Train Loss: 0.301377    Train Accuracy: 0.910040
Validation Accuracy: 0.861500
Epoch no 34:     Train Loss: 0.215414    Train Accuracy: 0.911820
Validation Accuracy: 0.867700
Epoch no 35:     Train Loss: 0.274173    Train Accuracy: 0.916420
Validation Accuracy: 0.865800
Epoch no 36:     Train Loss: 0.263953    Train Accuracy: 0.916160
Validation Accuracy: 0.871200
```

```
Epoch no 37:     Train Loss: 0.193064    Train Accuracy: 0.922720
Validation Accuracy: 0.877400
Epoch no 38:     Train Loss: 0.174487    Train Accuracy: 0.923140
Validation Accuracy: 0.870200
Epoch no 39:     Train Loss: 0.217410    Train Accuracy: 0.925820
Validation Accuracy: 0.875400
Epoch no 40:     Train Loss: 0.190403    Train Accuracy: 0.925920
Validation Accuracy: 0.876000
Epoch no 41:     Train Loss: 0.163484    Train Accuracy: 0.927860
Validation Accuracy: 0.879900
Epoch no 42:     Train Loss: 0.229255    Train Accuracy: 0.931780
Validation Accuracy: 0.881100
Epoch no 43:     Train Loss: 0.223143    Train Accuracy: 0.931500
Validation Accuracy: 0.877500
Epoch no 44:     Train Loss: 0.165186    Train Accuracy: 0.931980
Validation Accuracy: 0.883800
Epoch no 45:     Train Loss: 0.154615    Train Accuracy: 0.937320
Validation Accuracy: 0.880200
Epoch no 46:     Train Loss: 0.168452    Train Accuracy: 0.940480
Validation Accuracy: 0.883500
Epoch no 47:     Train Loss: 0.172931    Train Accuracy: 0.940940
Validation Accuracy: 0.884700
Epoch no 48:     Train Loss: 0.210732    Train Accuracy: 0.941800
Validation Accuracy: 0.880300
Epoch no 49:     Train Loss: 0.149000    Train Accuracy: 0.942740
Validation Accuracy: 0.885600
Epoch no 50:     Train Loss: 0.212602    Train Accuracy: 0.942640
Validation Accuracy: 0.885400
Epoch no 51:     Train Loss: 0.185709    Train Accuracy: 0.949760
Validation Accuracy: 0.885400
Epoch no 52:     Train Loss: 0.109900    Train Accuracy: 0.949840
Validation Accuracy: 0.889500
Epoch no 53:     Train Loss: 0.185856    Train Accuracy: 0.951320
Validation Accuracy: 0.891000
Epoch no 54:     Train Loss: 0.108121    Train Accuracy: 0.953360
Validation Accuracy: 0.890500
Epoch no 55:     Train Loss: 0.167838    Train Accuracy: 0.955880
Validation Accuracy: 0.891600
Epoch no 56:     Train Loss: 0.069487    Train Accuracy: 0.951900
Validation Accuracy: 0.889000
Epoch no 57:     Train Loss: 0.157026    Train Accuracy: 0.957160
Validation Accuracy: 0.891300
Epoch no 58:     Train Loss: 0.079631    Train Accuracy: 0.961060
Validation Accuracy: 0.894200
Epoch no 59:     Train Loss: 0.143043    Train Accuracy: 0.961660
Validation Accuracy: 0.896400
Epoch no 60:     Train Loss: 0.061202    Train Accuracy: 0.963660
Validation Accuracy: 0.900100
```

```
Epoch no 61:     Train Loss: 0.117966    Train Accuracy: 0.965920
Validation Accuracy: 0.898900
Epoch no 62:     Train Loss: 0.088418    Train Accuracy: 0.967620
Validation Accuracy: 0.900400
Epoch no 63:     Train Loss: 0.055056    Train Accuracy: 0.967940
Validation Accuracy: 0.901500
Epoch no 64:     Train Loss: 0.084932    Train Accuracy: 0.970880
Validation Accuracy: 0.895300
Epoch no 65:     Train Loss: 0.052879    Train Accuracy: 0.971620
Validation Accuracy: 0.900900
Epoch no 66:     Train Loss: 0.062437    Train Accuracy: 0.974560
Validation Accuracy: 0.902600
Epoch no 67:     Train Loss: 0.095221    Train Accuracy: 0.977400
Validation Accuracy: 0.903600
Epoch no 68:     Train Loss: 0.090090    Train Accuracy: 0.977300
Validation Accuracy: 0.902500
Epoch no 69:     Train Loss: 0.099281    Train Accuracy: 0.977440
Validation Accuracy: 0.902300
Epoch no 70:     Train Loss: 0.052526    Train Accuracy: 0.979240
Validation Accuracy: 0.906800
Epoch no 71:     Train Loss: 0.040677    Train Accuracy: 0.981600
Validation Accuracy: 0.910100
Epoch no 72:     Train Loss: 0.056239    Train Accuracy: 0.983960
Validation Accuracy: 0.908100
Epoch no 73:     Train Loss: 0.024714    Train Accuracy: 0.983580
Validation Accuracy: 0.906500
Epoch no 74:     Train Loss: 0.030588    Train Accuracy: 0.984880
Validation Accuracy: 0.909100
Epoch no 75:     Train Loss: 0.029567    Train Accuracy: 0.986100
Validation Accuracy: 0.913500
Epoch no 76:     Train Loss: 0.038966    Train Accuracy: 0.987080
Validation Accuracy: 0.911900
Epoch no 77:     Train Loss: 0.027646    Train Accuracy: 0.988380
Validation Accuracy: 0.914100
Epoch no 78:     Train Loss: 0.039755    Train Accuracy: 0.989580
Validation Accuracy: 0.913800
Epoch no 79:     Train Loss: 0.040502    Train Accuracy: 0.991360
Validation Accuracy: 0.915200
Epoch no 80:     Train Loss: 0.030072    Train Accuracy: 0.991280
Validation Accuracy: 0.916600
0.9204
```



#Varied the no of epochs with a fixed batch size to measure the effect of varying the epoch size.

[9]:
```
# Parameters
batch_size=512
val_batch_size=100
num_epochs=[80,100]
# learning_rate=2*1e-3

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

max_validation = 0
max_epoch=0
val_acc=[]
train_acc=[]
epochs=[]
```

[ ]:
```
for epoch_no in range(len(num_epochs)):
    # Train/Test Data
    train_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=batch_size,shuffle=True,␣
↪num_workers=4)
    train_acc_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=val_batch_size,shuffle=False,␣
↪num_workers=4)
```

```
Validation Accuracy: 0.897600
Epoch no 67:     Train Loss: 0.147558    Train Accuracy: 0.965300
Validation Accuracy: 0.906900
```

#Varied the no of epochs with a fixed batch size to measure the effect of varying the epoch size.

[9]:
```
# Parameters
batch_size=512
val_batch_size=100
num_epochs=[80,100]
# learning_rate=2*1e-3

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

max_validation = 0
max_epoch=0
val_acc=[]
train_acc=[]
epochs=[]
```

[10]:
```
for epoch_no in range(len(num_epochs)):
  # Train/Test Data
  train_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=batch_size,shuffle=True,␣
↪num_workers=4)
  train_acc_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=val_batch_size,shuffle=False,␣
↪num_workers=4)
  val_loader = torch.utils.data.DataLoader(cifar_val_transformed,␣
↪batch_size=val_batch_size, shuffle=False, num_workers=4)

  # Model
  resnet18 =ResNet18()
  resnet18=resnet18.to(dev)
  loss_func= torch.nn.CrossEntropyLoss()
  optimizer = torch.optim.SGD(resnet18.parameters(), lr=0.1, momentum=0.9,␣
↪weight_decay=5e-4)
  scheduler = torch.optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=100)

  for i in range(num_epochs[epoch_no]):
    # Training
    for imgs, labels in train_loader:
      if dev is not None:
        imgs,labels=imgs.to(dev),labels.to(dev)
      out= resnet18(imgs)
      loss=loss_func(out,labels)
      optimizer.zero_grad()
      loss.backward()
```

```
    optimizer.step()

  correct_val = 0
  total_val = 0
  correct_train_acc=0
  total_train_acc=0
  # Testing
  with torch.no_grad():
    for imgs, labels in val_loader:
      if dev is not None:
        imgs,labels=imgs.to(dev),labels.to(dev)
      outputs = resnet18(imgs)
      _, predicted = torch.max(outputs, dim=1)
      total_val += labels.shape[0]
      correct_val += int((predicted == labels).sum())
    val_acc.append(correct_val/total_val)

  # Compute Loss and Accuracy on training data
    for train_acc_imgs,train_acc_labels in train_acc_loader:
      if dev is not None:
        train_acc_imgs,train_acc_labels=train_acc_imgs.
↪to(dev),train_acc_labels.to(dev)
      train_acc_out=resnet18(train_acc_imgs)
      _, train_acc_predicted = torch.max(train_acc_out, dim=1)
      total_train_acc += train_acc_labels.shape[0]
      correct_train_acc += int((train_acc_predicted == train_acc_labels).
↪sum())
      # minibatch_acc = accuracy(train_acc, train_acc_labels,1)[0]
      # print("Top-1 training accuracy for minibatch", minibatch_acc)
    train_acc.append(correct_train_acc/total_train_acc)


    if correct_val/total_val > max_validation:
      max_validation=correct_val/total_val
      max_epoch=i
  epochs.append(i)



  if i%1==0:
    print("Epoch no %d:\t Train Loss: %f \t Train Accuracy: %f \t Validation␣
↪Accuracy: %f" % (i+1, float(loss), correct_train_acc / total_train_acc,␣
↪correct_val / total_val))

  scheduler.step()

print(max_validation)
```

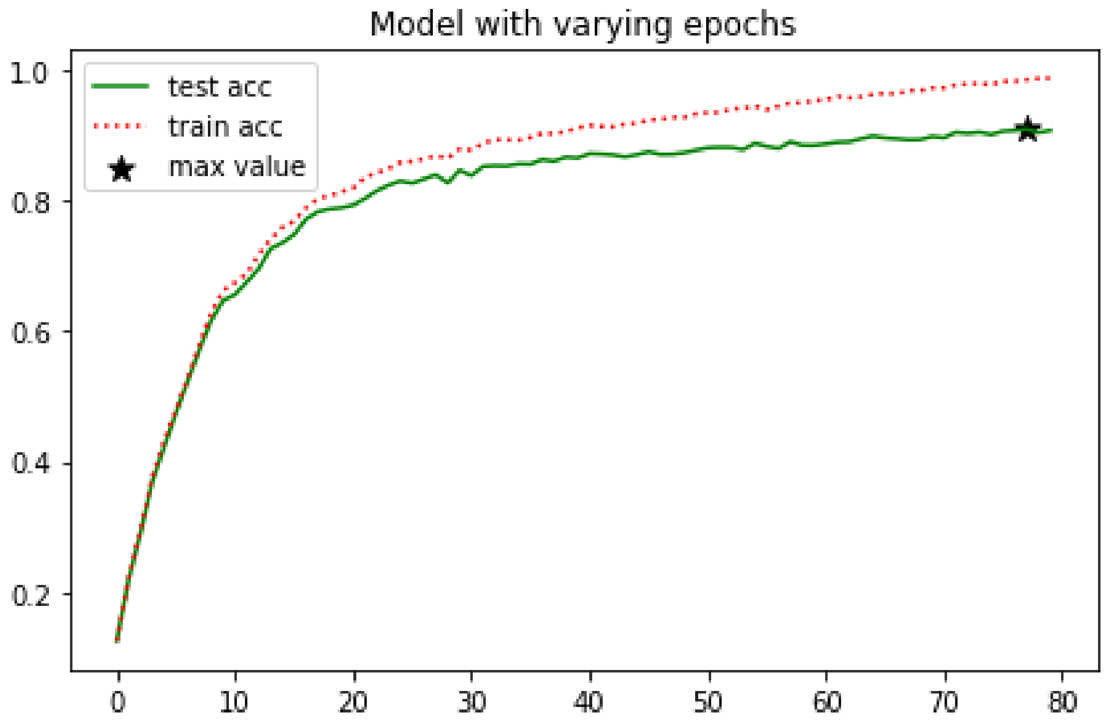
```python
# Plot Train Accuracy vs Test Accuracy
plt.plot(epochs, val_acc, label="test acc", color="green", linestyle='-')
plt.plot(epochs, train_acc, label="train acc", color="red",linestyle=':')
plt.scatter([max_epoch], [max_validation],color="black", marker="*",␣
↪label="max value", s=100 )
plt.title("Model with varying epochs")
plt.legend()
plt.tight_layout()
plt.show()
```

```
Epoch no 1:      Train Loss: 2.264586    Train Accuracy: 0.125240
Validation Accuracy: 0.126000
Epoch no 2:      Train Loss: 2.054015    Train Accuracy: 0.221880
Validation Accuracy: 0.219200
Epoch no 3:      Train Loss: 1.796045    Train Accuracy: 0.295700
Validation Accuracy: 0.290400
Epoch no 4:      Train Loss: 1.604048    Train Accuracy: 0.376880
Validation Accuracy: 0.369700
Epoch no 5:      Train Loss: 1.473720    Train Accuracy: 0.432740
Validation Accuracy: 0.422600
Epoch no 6:      Train Loss: 1.467716    Train Accuracy: 0.479700
Validation Accuracy: 0.474300
Epoch no 7:      Train Loss: 1.265979    Train Accuracy: 0.528500
Validation Accuracy: 0.523800
Epoch no 8:      Train Loss: 1.231469    Train Accuracy: 0.580820
Validation Accuracy: 0.572200
Epoch no 9:      Train Loss: 1.003173    Train Accuracy: 0.629060
Validation Accuracy: 0.617000
Epoch no 10:     Train Loss: 1.017268    Train Accuracy: 0.662840
Validation Accuracy: 0.647300
Epoch no 11:     Train Loss: 1.046619    Train Accuracy: 0.674720
Validation Accuracy: 0.656200
Epoch no 12:     Train Loss: 0.830696    Train Accuracy: 0.687700
Validation Accuracy: 0.675500
Epoch no 13:     Train Loss: 0.742466    Train Accuracy: 0.717400
Validation Accuracy: 0.695400
Epoch no 14:     Train Loss: 0.699909    Train Accuracy: 0.740940
Validation Accuracy: 0.726200
Epoch no 15:     Train Loss: 0.817326    Train Accuracy: 0.759440
Validation Accuracy: 0.735600
Epoch no 16:     Train Loss: 0.799796    Train Accuracy: 0.768220
Validation Accuracy: 0.748100
Epoch no 17:     Train Loss: 0.614111    Train Accuracy: 0.788980
Validation Accuracy: 0.771500
Epoch no 18:     Train Loss: 0.617471    Train Accuracy: 0.802680
Validation Accuracy: 0.782900
```

```
Epoch no 19:     Train Loss: 0.458486    Train Accuracy: 0.807620
Validation Accuracy: 0.787300
Epoch no 20:     Train Loss: 0.470877    Train Accuracy: 0.813560
Validation Accuracy: 0.789100
Epoch no 21:     Train Loss: 0.514263    Train Accuracy: 0.819460
Validation Accuracy: 0.793200
Epoch no 22:     Train Loss: 0.509757    Train Accuracy: 0.835860
Validation Accuracy: 0.803400
Epoch no 23:     Train Loss: 0.553705    Train Accuracy: 0.842820
Validation Accuracy: 0.815100
Epoch no 24:     Train Loss: 0.536142    Train Accuracy: 0.849060
Validation Accuracy: 0.823800
Epoch no 25:     Train Loss: 0.499699    Train Accuracy: 0.858460
Validation Accuracy: 0.830100
Epoch no 26:     Train Loss: 0.377888    Train Accuracy: 0.860060
Validation Accuracy: 0.826800
Epoch no 27:     Train Loss: 0.461240    Train Accuracy: 0.863880
Validation Accuracy: 0.833400
Epoch no 28:     Train Loss: 0.357848    Train Accuracy: 0.868680
Validation Accuracy: 0.839800
Epoch no 29:     Train Loss: 0.446539    Train Accuracy: 0.865020
Validation Accuracy: 0.827200
Epoch no 30:     Train Loss: 0.355929    Train Accuracy: 0.879080
Validation Accuracy: 0.846600
Epoch no 31:     Train Loss: 0.365238    Train Accuracy: 0.877820
Validation Accuracy: 0.838500
Epoch no 32:     Train Loss: 0.352352    Train Accuracy: 0.888060
Validation Accuracy: 0.852600
Epoch no 33:     Train Loss: 0.306451    Train Accuracy: 0.893500
Validation Accuracy: 0.853500
Epoch no 34:     Train Loss: 0.393646    Train Accuracy: 0.893180
Validation Accuracy: 0.853100
Epoch no 35:     Train Loss: 0.298468    Train Accuracy: 0.892680
Validation Accuracy: 0.856400
Epoch no 36:     Train Loss: 0.281884    Train Accuracy: 0.896980
Validation Accuracy: 0.856000
Epoch no 37:     Train Loss: 0.314198    Train Accuracy: 0.903180
Validation Accuracy: 0.862900
Epoch no 38:     Train Loss: 0.315254    Train Accuracy: 0.902560
Validation Accuracy: 0.860600
Epoch no 39:     Train Loss: 0.288702    Train Accuracy: 0.905740
Validation Accuracy: 0.866700
Epoch no 40:     Train Loss: 0.272967    Train Accuracy: 0.911540
Validation Accuracy: 0.865400
Epoch no 41:     Train Loss: 0.207101    Train Accuracy: 0.915120
Validation Accuracy: 0.872300
Epoch no 42:     Train Loss: 0.159768    Train Accuracy: 0.913640
Validation Accuracy: 0.871700
```

```
Epoch no 43:     Train Loss: 0.229951    Train Accuracy: 0.913000
Validation Accuracy: 0.870300
Epoch no 44:     Train Loss: 0.241581    Train Accuracy: 0.917380
Validation Accuracy: 0.867100
Epoch no 45:     Train Loss: 0.274180    Train Accuracy: 0.918340
Validation Accuracy: 0.870600
Epoch no 46:     Train Loss: 0.201056    Train Accuracy: 0.923840
Validation Accuracy: 0.875100
Epoch no 47:     Train Loss: 0.184158    Train Accuracy: 0.924760
Validation Accuracy: 0.871100
Epoch no 48:     Train Loss: 0.231331    Train Accuracy: 0.927560
Validation Accuracy: 0.871300
Epoch no 49:     Train Loss: 0.236435    Train Accuracy: 0.927540
Validation Accuracy: 0.873600
Epoch no 50:     Train Loss: 0.190156    Train Accuracy: 0.932980
Validation Accuracy: 0.877400
Epoch no 51:     Train Loss: 0.199045    Train Accuracy: 0.935060
Validation Accuracy: 0.881200
Epoch no 52:     Train Loss: 0.192580    Train Accuracy: 0.935000
Validation Accuracy: 0.881900
Epoch no 53:     Train Loss: 0.155599    Train Accuracy: 0.940720
Validation Accuracy: 0.881700
Epoch no 54:     Train Loss: 0.168415    Train Accuracy: 0.941440
Validation Accuracy: 0.878400
Epoch no 55:     Train Loss: 0.143875    Train Accuracy: 0.945120
Validation Accuracy: 0.888100
Epoch no 56:     Train Loss: 0.165222    Train Accuracy: 0.939360
Validation Accuracy: 0.883300
Epoch no 57:     Train Loss: 0.143930    Train Accuracy: 0.944880
Validation Accuracy: 0.880100
Epoch no 58:     Train Loss: 0.175468    Train Accuracy: 0.950040
Validation Accuracy: 0.890300
Epoch no 59:     Train Loss: 0.146276    Train Accuracy: 0.949660
Validation Accuracy: 0.885000
Epoch no 60:     Train Loss: 0.175510    Train Accuracy: 0.952920
Validation Accuracy: 0.884800
Epoch no 61:     Train Loss: 0.162454    Train Accuracy: 0.954820
Validation Accuracy: 0.887000
Epoch no 62:     Train Loss: 0.127419    Train Accuracy: 0.959660
Validation Accuracy: 0.889700
Epoch no 63:     Train Loss: 0.118050    Train Accuracy: 0.957460
Validation Accuracy: 0.890400
Epoch no 64:     Train Loss: 0.115520    Train Accuracy: 0.959100
Validation Accuracy: 0.895000
Epoch no 65:     Train Loss: 0.102905    Train Accuracy: 0.963020
Validation Accuracy: 0.899000
Epoch no 66:     Train Loss: 0.151740    Train Accuracy: 0.964620
Validation Accuracy: 0.896200
```

```
Epoch no 67:     Train Loss: 0.098996    Train Accuracy: 0.963620
Validation Accuracy: 0.894900
Epoch no 68:     Train Loss: 0.085225    Train Accuracy: 0.969020
Validation Accuracy: 0.893300
Epoch no 69:     Train Loss: 0.118625    Train Accuracy: 0.968720
Validation Accuracy: 0.893700
Epoch no 70:     Train Loss: 0.072653    Train Accuracy: 0.972800
Validation Accuracy: 0.898400
Epoch no 71:     Train Loss: 0.075032    Train Accuracy: 0.972820
Validation Accuracy: 0.896400
Epoch no 72:     Train Loss: 0.080750    Train Accuracy: 0.977380
Validation Accuracy: 0.904800
Epoch no 73:     Train Loss: 0.101631    Train Accuracy: 0.979060
Validation Accuracy: 0.903000
Epoch no 74:     Train Loss: 0.063905    Train Accuracy: 0.979500
Validation Accuracy: 0.904900
Epoch no 75:     Train Loss: 0.059534    Train Accuracy: 0.977900
Validation Accuracy: 0.901300
Epoch no 76:     Train Loss: 0.056714    Train Accuracy: 0.982960
Validation Accuracy: 0.906500
Epoch no 77:     Train Loss: 0.038449    Train Accuracy: 0.983280
Validation Accuracy: 0.907600
Epoch no 78:     Train Loss: 0.060045    Train Accuracy: 0.985360
Validation Accuracy: 0.909400
Epoch no 79:     Train Loss: 0.025628    Train Accuracy: 0.987260
Validation Accuracy: 0.904300
Epoch no 80:     Train Loss: 0.036574    Train Accuracy: 0.987840
Validation Accuracy: 0.907600
0.9094
```



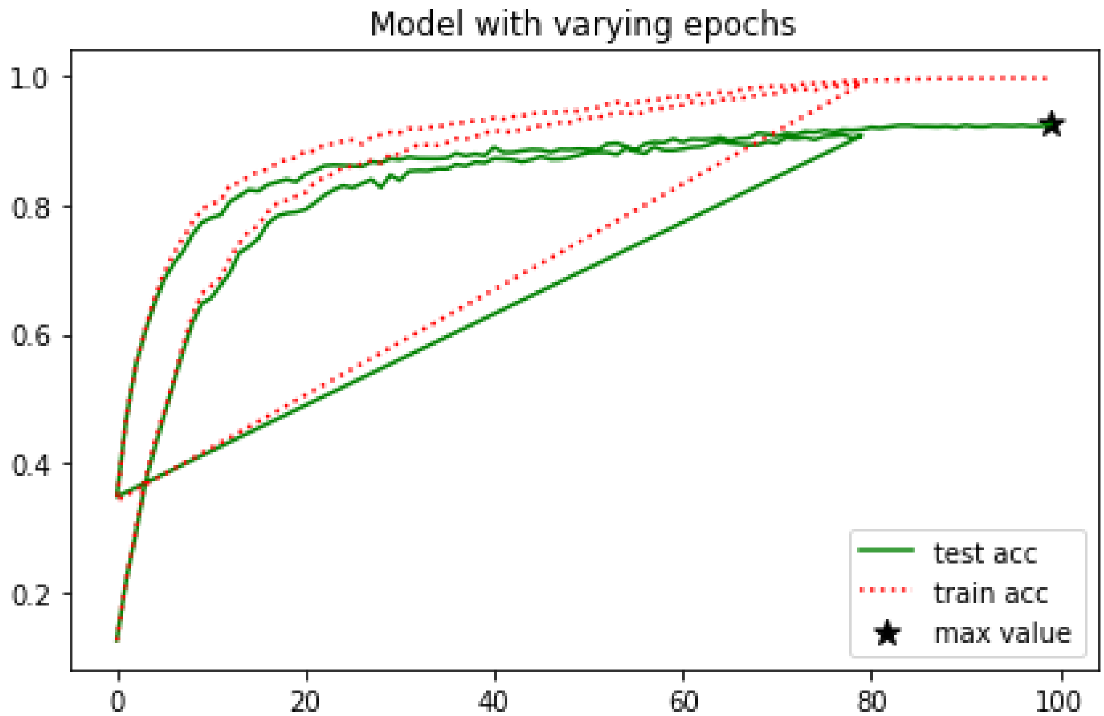

```
Epoch no 1:      Train Loss: 1.635337    Train Accuracy: 0.342100
Validation Accuracy: 0.348300
Epoch no 2:      Train Loss: 1.484460    Train Accuracy: 0.472960
Validation Accuracy: 0.469100
Epoch no 3:      Train Loss: 1.160352    Train Accuracy: 0.551080
Validation Accuracy: 0.551300
Epoch no 4:      Train Loss: 1.149050    Train Accuracy: 0.608200
Validation Accuracy: 0.604000
Epoch no 5:      Train Loss: 0.968862    Train Accuracy: 0.656500
Validation Accuracy: 0.647800
Epoch no 6:      Train Loss: 0.927355    Train Accuracy: 0.695600
Validation Accuracy: 0.684300
Epoch no 7:      Train Loss: 0.724464    Train Accuracy: 0.723760
Validation Accuracy: 0.708900
Epoch no 8:      Train Loss: 0.676686    Train Accuracy: 0.748100
Validation Accuracy: 0.727700
Epoch no 9:      Train Loss: 0.722302    Train Accuracy: 0.772800
Validation Accuracy: 0.754200
Epoch no 10:     Train Loss: 0.623231    Train Accuracy: 0.793600
Validation Accuracy: 0.772700
Epoch no 11:     Train Loss: 0.546767    Train Accuracy: 0.801700
Validation Accuracy: 0.780100
Epoch no 12:     Train Loss: 0.585612    Train Accuracy: 0.806540
Validation Accuracy: 0.784600
```

```
Epoch no 13:     Train Loss: 0.469988    Train Accuracy: 0.831300
Validation Accuracy: 0.805300
Epoch no 14:     Train Loss: 0.369418    Train Accuracy: 0.834340
Validation Accuracy: 0.814200
Epoch no 15:     Train Loss: 0.421574    Train Accuracy: 0.844760
Validation Accuracy: 0.823700
Epoch no 16:     Train Loss: 0.459799    Train Accuracy: 0.851760
Validation Accuracy: 0.821700
Epoch no 17:     Train Loss: 0.492511    Train Accuracy: 0.855200
Validation Accuracy: 0.831500
Epoch no 18:     Train Loss: 0.388689    Train Accuracy: 0.865120
Validation Accuracy: 0.835800
Epoch no 19:     Train Loss: 0.412799    Train Accuracy: 0.870820
Validation Accuracy: 0.840000
Epoch no 20:     Train Loss: 0.401323    Train Accuracy: 0.873140
Validation Accuracy: 0.836100
Epoch no 21:     Train Loss: 0.368066    Train Accuracy: 0.884180
Validation Accuracy: 0.848100
Epoch no 22:     Train Loss: 0.296090    Train Accuracy: 0.883320
Validation Accuracy: 0.853300
Epoch no 23:     Train Loss: 0.334298    Train Accuracy: 0.894140
Validation Accuracy: 0.860900
Epoch no 24:     Train Loss: 0.357113    Train Accuracy: 0.894500
Validation Accuracy: 0.862100
Epoch no 25:     Train Loss: 0.261653    Train Accuracy: 0.899340
Validation Accuracy: 0.861500
Epoch no 26:     Train Loss: 0.299314    Train Accuracy: 0.900920
Validation Accuracy: 0.860700
Epoch no 27:     Train Loss: 0.320887    Train Accuracy: 0.894360
Validation Accuracy: 0.859900
Epoch no 28:     Train Loss: 0.294592    Train Accuracy: 0.905240
Validation Accuracy: 0.867100
Epoch no 29:     Train Loss: 0.292338    Train Accuracy: 0.908960
Validation Accuracy: 0.865700
Epoch no 30:     Train Loss: 0.258491    Train Accuracy: 0.912780
Validation Accuracy: 0.872000
Epoch no 31:     Train Loss: 0.261893    Train Accuracy: 0.911880
Validation Accuracy: 0.870500
Epoch no 32:     Train Loss: 0.246609    Train Accuracy: 0.918300
Validation Accuracy: 0.874300
Epoch no 33:     Train Loss: 0.265860    Train Accuracy: 0.917520
Validation Accuracy: 0.872400
Epoch no 34:     Train Loss: 0.261719    Train Accuracy: 0.920600
Validation Accuracy: 0.872200
Epoch no 35:     Train Loss: 0.215484    Train Accuracy: 0.920540
Validation Accuracy: 0.872100
Epoch no 36:     Train Loss: 0.235680    Train Accuracy: 0.925340
Validation Accuracy: 0.873000
```

```
Epoch no 37:     Train Loss: 0.219329    Train Accuracy: 0.925960
Validation Accuracy: 0.880300
Epoch no 38:     Train Loss: 0.233058    Train Accuracy: 0.928240
Validation Accuracy: 0.876400
Epoch no 39:     Train Loss: 0.233170    Train Accuracy: 0.930940
Validation Accuracy: 0.879300
Epoch no 40:     Train Loss: 0.168152    Train Accuracy: 0.931280
Validation Accuracy: 0.876700
Epoch no 41:     Train Loss: 0.235531    Train Accuracy: 0.935000
Validation Accuracy: 0.889600
Epoch no 42:     Train Loss: 0.211478    Train Accuracy: 0.933200
Validation Accuracy: 0.881800
Epoch no 43:     Train Loss: 0.218399    Train Accuracy: 0.936780
Validation Accuracy: 0.881800
Epoch no 44:     Train Loss: 0.191687    Train Accuracy: 0.938460
Validation Accuracy: 0.884300
Epoch no 45:     Train Loss: 0.173940    Train Accuracy: 0.941100
Validation Accuracy: 0.886200
Epoch no 46:     Train Loss: 0.214177    Train Accuracy: 0.943960
Validation Accuracy: 0.885600
Epoch no 47:     Train Loss: 0.180323    Train Accuracy: 0.942260
Validation Accuracy: 0.881900
Epoch no 48:     Train Loss: 0.121708    Train Accuracy: 0.944760
Validation Accuracy: 0.886500
Epoch no 49:     Train Loss: 0.118306    Train Accuracy: 0.945560
Validation Accuracy: 0.886000
Epoch no 50:     Train Loss: 0.183837    Train Accuracy: 0.945580
Validation Accuracy: 0.887200
Epoch no 51:     Train Loss: 0.142907    Train Accuracy: 0.948980
Validation Accuracy: 0.886400
Epoch no 52:     Train Loss: 0.168027    Train Accuracy: 0.951600
Validation Accuracy: 0.885900
Epoch no 53:     Train Loss: 0.131299    Train Accuracy: 0.956180
Validation Accuracy: 0.894700
Epoch no 54:     Train Loss: 0.126270    Train Accuracy: 0.959280
Validation Accuracy: 0.895800
Epoch no 55:     Train Loss: 0.147499    Train Accuracy: 0.955260
Validation Accuracy: 0.891100
Epoch no 56:     Train Loss: 0.092188    Train Accuracy: 0.958940
Validation Accuracy: 0.885500
Epoch no 57:     Train Loss: 0.099735    Train Accuracy: 0.960360
Validation Accuracy: 0.894700
Epoch no 58:     Train Loss: 0.063666    Train Accuracy: 0.964280
Validation Accuracy: 0.897900
Epoch no 59:     Train Loss: 0.094152    Train Accuracy: 0.962920
Validation Accuracy: 0.896900
Epoch no 60:     Train Loss: 0.104387    Train Accuracy: 0.970420
Validation Accuracy: 0.902700
```

```
Epoch no 61:     Train Loss: 0.097262    Train Accuracy: 0.965720
Validation Accuracy: 0.897800
Epoch no 62:     Train Loss: 0.096673    Train Accuracy: 0.970000
Validation Accuracy: 0.902400
Epoch no 63:     Train Loss: 0.091786    Train Accuracy: 0.971140
Validation Accuracy: 0.901600
Epoch no 64:     Train Loss: 0.107318    Train Accuracy: 0.972480
Validation Accuracy: 0.902500
Epoch no 65:     Train Loss: 0.038118    Train Accuracy: 0.974780
Validation Accuracy: 0.904900
Epoch no 66:     Train Loss: 0.114781    Train Accuracy: 0.973340
Validation Accuracy: 0.900000
Epoch no 67:     Train Loss: 0.099272    Train Accuracy: 0.978120
Validation Accuracy: 0.903500
Epoch no 68:     Train Loss: 0.070103    Train Accuracy: 0.980800
Validation Accuracy: 0.904000
Epoch no 69:     Train Loss: 0.036715    Train Accuracy: 0.981540
Validation Accuracy: 0.911500
Epoch no 70:     Train Loss: 0.070550    Train Accuracy: 0.983620
Validation Accuracy: 0.909900
Epoch no 71:     Train Loss: 0.053657    Train Accuracy: 0.983760
Validation Accuracy: 0.906200
Epoch no 72:     Train Loss: 0.018012    Train Accuracy: 0.984340
Validation Accuracy: 0.911000
Epoch no 73:     Train Loss: 0.023387    Train Accuracy: 0.985340
Validation Accuracy: 0.909300
Epoch no 74:     Train Loss: 0.015473    Train Accuracy: 0.988060
Validation Accuracy: 0.913800
Epoch no 75:     Train Loss: 0.017772    Train Accuracy: 0.988300
Validation Accuracy: 0.912600
Epoch no 76:     Train Loss: 0.031606    Train Accuracy: 0.989400
Validation Accuracy: 0.915900
Epoch no 77:     Train Loss: 0.018919    Train Accuracy: 0.988940
Validation Accuracy: 0.913500
Epoch no 78:     Train Loss: 0.039256    Train Accuracy: 0.991360
Validation Accuracy: 0.916700
Epoch no 79:     Train Loss: 0.007827    Train Accuracy: 0.992280
Validation Accuracy: 0.915900
Epoch no 80:     Train Loss: 0.015992    Train Accuracy: 0.993080
Validation Accuracy: 0.917000
Epoch no 81:     Train Loss: 0.009097    Train Accuracy: 0.993480
Validation Accuracy: 0.920200
Epoch no 82:     Train Loss: 0.019572    Train Accuracy: 0.993840
Validation Accuracy: 0.919200
Epoch no 83:     Train Loss: 0.011127    Train Accuracy: 0.994280
Validation Accuracy: 0.921200
Epoch no 84:     Train Loss: 0.007986    Train Accuracy: 0.994400
Validation Accuracy: 0.923300
```

```
Epoch no 85:     Train Loss: 0.009734    Train Accuracy: 0.995120
Validation Accuracy: 0.923300
Epoch no 86:     Train Loss: 0.005988    Train Accuracy: 0.995280
Validation Accuracy: 0.921700
Epoch no 87:     Train Loss: 0.009106    Train Accuracy: 0.995800
Validation Accuracy: 0.921800
Epoch no 88:     Train Loss: 0.004447    Train Accuracy: 0.995860
Validation Accuracy: 0.921300
Epoch no 89:     Train Loss: 0.011634    Train Accuracy: 0.995920
Validation Accuracy: 0.922200
Epoch no 90:     Train Loss: 0.013455    Train Accuracy: 0.996040
Validation Accuracy: 0.920400
Epoch no 91:     Train Loss: 0.005378    Train Accuracy: 0.995760
Validation Accuracy: 0.923500
Epoch no 92:     Train Loss: 0.008141    Train Accuracy: 0.996300
Validation Accuracy: 0.921500
Epoch no 93:     Train Loss: 0.006011    Train Accuracy: 0.996500
Validation Accuracy: 0.921400
Epoch no 94:     Train Loss: 0.005351    Train Accuracy: 0.996520
Validation Accuracy: 0.923300
Epoch no 95:     Train Loss: 0.006752    Train Accuracy: 0.996820
Validation Accuracy: 0.923100
Epoch no 96:     Train Loss: 0.005068    Train Accuracy: 0.996600
Validation Accuracy: 0.922500
Epoch no 97:     Train Loss: 0.006925    Train Accuracy: 0.996640
Validation Accuracy: 0.923400
Epoch no 98:     Train Loss: 0.003495    Train Accuracy: 0.996720
Validation Accuracy: 0.922100
Epoch no 99:     Train Loss: 0.004069    Train Accuracy: 0.996120
Validation Accuracy: 0.922300
Epoch no 100:    Train Loss: 0.002434    Train Accuracy: 0.996520
Validation Accuracy: 0.925200
0.9252
```


Model with varying epochs
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
test acc
train acc
max value

```
Epoch no 19:     Train Loss: 0.458486    Train Accuracy: 0.807620
Validation Accuracy: 0.787300
Epoch no 20:     Train Loss: 0.470877    Train Accuracy: 0.813560
Validation Accuracy: 0.789100
Epoch no 21:     Train Loss: 0.514263    Train Accuracy: 0.819460
Validation Accuracy: 0.793200
Epoch no 22:     Train Loss: 0.509757    Train Accuracy: 0.835860
Validation Accuracy: 0.803400
Epoch no 23:     Train Loss: 0.553705    Train Accuracy: 0.842820
Validation Accuracy: 0.815100
Epoch no 24:     Train Loss: 0.536142    Train Accuracy: 0.849060
Validation Accuracy: 0.823800
Epoch no 25:     Train Loss: 0.499699    Train Accuracy: 0.858460
Validation Accuracy: 0.830100
Epoch no 26:     Train Loss: 0.377888    Train Accuracy: 0.860060
Validation Accuracy: 0.826800
Epoch no 27:     Train Loss: 0.461240    Train Accuracy: 0.863880
Validation Accuracy: 0.833400
Epoch no 28:     Train Loss: 0.357848    Train Accuracy: 0.868680
Validation Accuracy: 0.839800
Epoch no 29:     Train Loss: 0.446539    Train Accuracy: 0.865020
Validation Accuracy: 0.827200
Epoch no 30:     Train Loss: 0.355929    Train Accuracy: 0.879080
Validation Accuracy: 0.846600
Epoch no 31:     Train Loss: 0.365238    Train Accuracy: 0.877820
Validation Accuracy: 0.838500
```

#Varied the learning rate [0.1, 0.01, 0.001, 0.0001] on fixed batch size and fixed no of epoch to observe the effect of varying the epoch size Results to lr = 0.1 have been shown above

[10]:
```python
# Parameters
batch_size=512
val_batch_size=100
num_epochs=100
learning_rate=[0.01, 0.001, 0.0001]

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

max_validation = 0
max_epoch=0
val_acc=[]
train_acc=[]
epochs=[]
```

[11]:
```python
for lr_no in range(len(learning_rate)):
  # Train/Test Data
```

```python
train_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=batch_size,shuffle=True,␣
↪num_workers=4)
train_acc_loader=torch.utils.data.
↪DataLoader(cifar_transformed,batch_size=val_batch_size,shuffle=False,␣
↪num_workers=4)
val_loader = torch.utils.data.DataLoader(cifar_val_transformed,␣
↪batch_size=val_batch_size, shuffle=False, num_workers=4)

# Model
resnet18 =ResNet18()
resnet18=resnet18.to(dev)
loss_func= torch.nn.CrossEntropyLoss()
optimizer = torch.optim.SGD(resnet18.parameters(), lr=learning_rate[lr_no],␣
↪momentum=0.9, weight_decay=5e-4)
scheduler = torch.optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=100)

for i in range(num_epochs):
  # Training
  for imgs, labels in train_loader:
    if dev is not None:
      imgs,labels=imgs.to(dev),labels.to(dev)
    out= resnet18(imgs)
    loss=loss_func(out,labels)
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

  correct_val = 0
  total_val = 0
  correct_train_acc=0
  total_train_acc=0
  # Testing
  with torch.no_grad():
    for imgs, labels in val_loader:
      if dev is not None:
        imgs,labels=imgs.to(dev),labels.to(dev)
      outputs = resnet18(imgs)
      _, predicted = torch.max(outputs, dim=1)
      total_val += labels.shape[0]
      correct_val += int((predicted == labels).sum())
    val_acc.append(correct_val/total_val)

  # Compute Loss and Accuracy on training data
    for train_acc_imgs,train_acc_labels in train_acc_loader:
      if dev is not None:
```

```
        train_acc_imgs,train_acc_labels=train_acc_imgs.
↪to(dev),train_acc_labels.to(dev)
      train_acc_out=resnet18(train_acc_imgs)
      _, train_acc_predicted = torch.max(train_acc_out, dim=1)
      total_train_acc += train_acc_labels.shape[0]
      correct_train_acc += int((train_acc_predicted == train_acc_labels).
↪sum())
      # minibatch_acc = accuracy(train_acc, train_acc_labels,1)[0]
      # print("Top-1 training accuracy for minibatch", minibatch_acc)
    train_acc.append(correct_train_acc/total_train_acc)


    if correct_val/total_val > max_validation:
      max_validation=correct_val/total_val
      max_epoch=i
  epochs.append(i)



  if i%1==0:
    print("Epoch no %d:\t Train Loss: %f \t Train Accuracy: %f \t Validation␣
↪Accuracy: %f" % (i+1, float(loss), correct_train_acc / total_train_acc,␣
↪correct_val / total_val))

  scheduler.step()

print(max_validation)

# Plot Train Accuracy vs Test Accuracy
plt.plot(epochs, val_acc, label="test acc", color="green", linestyle='-')
plt.plot(epochs, train_acc, label="train acc", color="red",linestyle=':')
plt.scatter([max_epoch], [max_validation],color="black", marker="*",␣
↪label="max value", s=100 )
plt.title("Model with varying learning rate")
plt.legend()
plt.tight_layout()
plt.show()
```

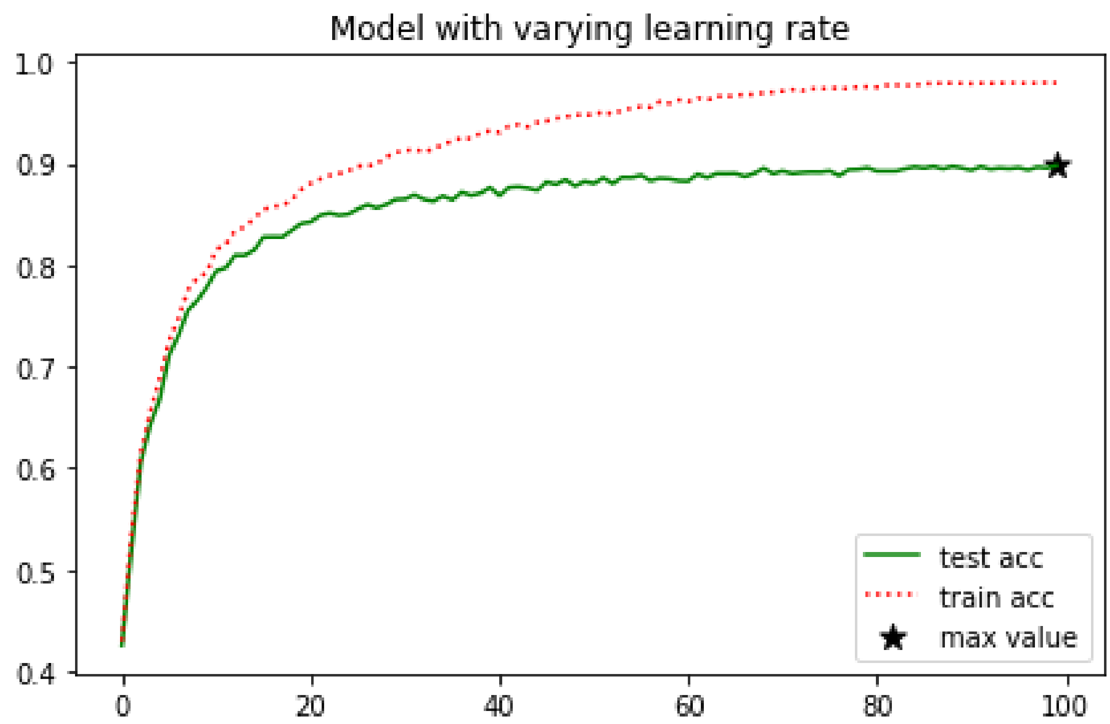
```
Epoch no 1:      Train Loss: 1.552232    Train Accuracy: 0.429620
Validation Accuracy: 0.425700
Epoch no 2:      Train Loss: 1.295115    Train Accuracy: 0.534040
Validation Accuracy: 0.519400
Epoch no 3:      Train Loss: 1.103490    Train Accuracy: 0.615320
Validation Accuracy: 0.606400
Epoch no 4:      Train Loss: 0.975402    Train Accuracy: 0.656060
Validation Accuracy: 0.643100
Epoch no 5:      Train Loss: 0.779251    Train Accuracy: 0.688060
```

```
Validation Accuracy: 0.667800
Epoch no 6:      Train Loss: 0.735042    Train Accuracy: 0.725840
Validation Accuracy: 0.711500
Epoch no 7:      Train Loss: 0.664171    Train Accuracy: 0.748440
Validation Accuracy: 0.731200
Epoch no 8:      Train Loss: 0.607026    Train Accuracy: 0.775640
Validation Accuracy: 0.755700
Epoch no 9:      Train Loss: 0.626022    Train Accuracy: 0.787420
Validation Accuracy: 0.765500
Epoch no 10:     Train Loss: 0.554748    Train Accuracy: 0.792940
Validation Accuracy: 0.778500
Epoch no 11:     Train Loss: 0.497290    Train Accuracy: 0.815620
Validation Accuracy: 0.794400
Epoch no 12:     Train Loss: 0.509008    Train Accuracy: 0.820960
Validation Accuracy: 0.797500
Epoch no 13:     Train Loss: 0.468392    Train Accuracy: 0.834020
Validation Accuracy: 0.809800
Epoch no 14:     Train Loss: 0.530023    Train Accuracy: 0.837280
Validation Accuracy: 0.809600
Epoch no 15:     Train Loss: 0.402277    Train Accuracy: 0.846140
Validation Accuracy: 0.814900
Epoch no 16:     Train Loss: 0.390247    Train Accuracy: 0.854700
Validation Accuracy: 0.827700
Epoch no 17:     Train Loss: 0.463846    Train Accuracy: 0.858240
Validation Accuracy: 0.827900
Epoch no 18:     Train Loss: 0.373162    Train Accuracy: 0.858400
Validation Accuracy: 0.827800
Epoch no 19:     Train Loss: 0.383332    Train Accuracy: 0.864660
Validation Accuracy: 0.834500
Epoch no 20:     Train Loss: 0.365766    Train Accuracy: 0.874620
Validation Accuracy: 0.841000
Epoch no 21:     Train Loss: 0.318598    Train Accuracy: 0.880720
Validation Accuracy: 0.842600
Epoch no 22:     Train Loss: 0.358109    Train Accuracy: 0.884880
Validation Accuracy: 0.849100
Epoch no 23:     Train Loss: 0.324620    Train Accuracy: 0.888320
Validation Accuracy: 0.851300
Epoch no 24:     Train Loss: 0.318866    Train Accuracy: 0.890480
Validation Accuracy: 0.849100
Epoch no 25:     Train Loss: 0.319717    Train Accuracy: 0.893000
Validation Accuracy: 0.850300
Epoch no 26:     Train Loss: 0.253647    Train Accuracy: 0.897580
Validation Accuracy: 0.855400
Epoch no 27:     Train Loss: 0.207129    Train Accuracy: 0.897660
Validation Accuracy: 0.859600
Epoch no 28:     Train Loss: 0.341369    Train Accuracy: 0.899560
Validation Accuracy: 0.856500
Epoch no 29:     Train Loss: 0.338506    Train Accuracy: 0.906480
```

```
Validation Accuracy: 0.859800
Epoch no 30:     Train Loss: 0.287129    Train Accuracy: 0.911320
Validation Accuracy: 0.864800
Epoch no 31:     Train Loss: 0.289820    Train Accuracy: 0.912620
Validation Accuracy: 0.865200
Epoch no 32:     Train Loss: 0.209114    Train Accuracy: 0.914320
Validation Accuracy: 0.869400
Epoch no 33:     Train Loss: 0.277847    Train Accuracy: 0.911380
Validation Accuracy: 0.864900
Epoch no 34:     Train Loss: 0.243478    Train Accuracy: 0.914620
Validation Accuracy: 0.863000
Epoch no 35:     Train Loss: 0.175035    Train Accuracy: 0.919080
Validation Accuracy: 0.868300
Epoch no 36:     Train Loss: 0.240257    Train Accuracy: 0.922160
Validation Accuracy: 0.864300
Epoch no 37:     Train Loss: 0.267375    Train Accuracy: 0.925660
Validation Accuracy: 0.872000
Epoch no 38:     Train Loss: 0.233675    Train Accuracy: 0.924660
Validation Accuracy: 0.868700
Epoch no 39:     Train Loss: 0.226387    Train Accuracy: 0.930120
Validation Accuracy: 0.871000
Epoch no 40:     Train Loss: 0.157203    Train Accuracy: 0.932360
Validation Accuracy: 0.876600
Epoch no 41:     Train Loss: 0.194394    Train Accuracy: 0.930740
Validation Accuracy: 0.868600
Epoch no 42:     Train Loss: 0.172515    Train Accuracy: 0.937000
Validation Accuracy: 0.876600
Epoch no 43:     Train Loss: 0.211421    Train Accuracy: 0.938500
Validation Accuracy: 0.877300
Epoch no 44:     Train Loss: 0.192515    Train Accuracy: 0.936180
Validation Accuracy: 0.875900
Epoch no 45:     Train Loss: 0.179416    Train Accuracy: 0.940700
Validation Accuracy: 0.873700
Epoch no 46:     Train Loss: 0.181160    Train Accuracy: 0.942200
Validation Accuracy: 0.882000
Epoch no 47:     Train Loss: 0.159630    Train Accuracy: 0.944640
Validation Accuracy: 0.879700
Epoch no 48:     Train Loss: 0.110579    Train Accuracy: 0.946360
Validation Accuracy: 0.884400
Epoch no 49:     Train Loss: 0.132300    Train Accuracy: 0.947800
Validation Accuracy: 0.877900
Epoch no 50:     Train Loss: 0.188296    Train Accuracy: 0.948500
Validation Accuracy: 0.882600
Epoch no 51:     Train Loss: 0.147990    Train Accuracy: 0.947800
Validation Accuracy: 0.880500
Epoch no 52:     Train Loss: 0.120545    Train Accuracy: 0.951380
Validation Accuracy: 0.886200
Epoch no 53:     Train Loss: 0.089948    Train Accuracy: 0.948460
```

```
Validation Accuracy: 0.880500
Epoch no 54:     Train Loss: 0.106824    Train Accuracy: 0.952920
Validation Accuracy: 0.886700
Epoch no 55:     Train Loss: 0.154720    Train Accuracy: 0.954400
Validation Accuracy: 0.886400
Epoch no 56:     Train Loss: 0.137852    Train Accuracy: 0.956120
Validation Accuracy: 0.888700
Epoch no 57:     Train Loss: 0.124454    Train Accuracy: 0.955420
Validation Accuracy: 0.883300
Epoch no 58:     Train Loss: 0.105909    Train Accuracy: 0.961100
Validation Accuracy: 0.885700
Epoch no 59:     Train Loss: 0.105528    Train Accuracy: 0.959540
Validation Accuracy: 0.885100
Epoch no 60:     Train Loss: 0.087450    Train Accuracy: 0.962840
Validation Accuracy: 0.883300
Epoch no 61:     Train Loss: 0.113311    Train Accuracy: 0.961180
Validation Accuracy: 0.883000
Epoch no 62:     Train Loss: 0.058651    Train Accuracy: 0.964740
Validation Accuracy: 0.889800
Epoch no 63:     Train Loss: 0.067150    Train Accuracy: 0.962760
Validation Accuracy: 0.886400
Epoch no 64:     Train Loss: 0.101704    Train Accuracy: 0.966560
Validation Accuracy: 0.890300
Epoch no 65:     Train Loss: 0.089976    Train Accuracy: 0.966020
Validation Accuracy: 0.890200
Epoch no 66:     Train Loss: 0.115720    Train Accuracy: 0.967180
Validation Accuracy: 0.890400
Epoch no 67:     Train Loss: 0.101382    Train Accuracy: 0.968400
Validation Accuracy: 0.887500
Epoch no 68:     Train Loss: 0.097978    Train Accuracy: 0.968080
Validation Accuracy: 0.890500
Epoch no 69:     Train Loss: 0.066093    Train Accuracy: 0.970060
Validation Accuracy: 0.895600
Epoch no 70:     Train Loss: 0.042100    Train Accuracy: 0.970460
Validation Accuracy: 0.890200
Epoch no 71:     Train Loss: 0.098983    Train Accuracy: 0.971560
Validation Accuracy: 0.892800
Epoch no 72:     Train Loss: 0.068785    Train Accuracy: 0.973080
Validation Accuracy: 0.890400
Epoch no 73:     Train Loss: 0.085806    Train Accuracy: 0.971800
Validation Accuracy: 0.891300
Epoch no 74:     Train Loss: 0.089749    Train Accuracy: 0.973980
Validation Accuracy: 0.891800
Epoch no 75:     Train Loss: 0.092726    Train Accuracy: 0.973720
Validation Accuracy: 0.891700
Epoch no 76:     Train Loss: 0.055105    Train Accuracy: 0.974580
Validation Accuracy: 0.892900
Epoch no 77:     Train Loss: 0.059039    Train Accuracy: 0.974360
```

```
Validation Accuracy: 0.888200
Epoch no 78:     Train Loss: 0.064944    Train Accuracy: 0.974000
Validation Accuracy: 0.894100
Epoch no 79:     Train Loss: 0.045372    Train Accuracy: 0.975840
Validation Accuracy: 0.894500
Epoch no 80:     Train Loss: 0.052372    Train Accuracy: 0.975740
Validation Accuracy: 0.896000
Epoch no 81:     Train Loss: 0.044988    Train Accuracy: 0.975420
Validation Accuracy: 0.892700
Epoch no 82:     Train Loss: 0.042488    Train Accuracy: 0.977320
Validation Accuracy: 0.892700
Epoch no 83:     Train Loss: 0.064203    Train Accuracy: 0.977440
Validation Accuracy: 0.892700
Epoch no 84:     Train Loss: 0.063173    Train Accuracy: 0.977300
Validation Accuracy: 0.895100
Epoch no 85:     Train Loss: 0.046325    Train Accuracy: 0.977060
Validation Accuracy: 0.896800
Epoch no 86:     Train Loss: 0.047078    Train Accuracy: 0.979120
Validation Accuracy: 0.895700
Epoch no 87:     Train Loss: 0.067610    Train Accuracy: 0.978740
Validation Accuracy: 0.897300
Epoch no 88:     Train Loss: 0.045319    Train Accuracy: 0.979580
Validation Accuracy: 0.894900
Epoch no 89:     Train Loss: 0.059688    Train Accuracy: 0.979380
Validation Accuracy: 0.897400
Epoch no 90:     Train Loss: 0.033511    Train Accuracy: 0.979220
Validation Accuracy: 0.895800
Epoch no 91:     Train Loss: 0.056988    Train Accuracy: 0.978940
Validation Accuracy: 0.893600
Epoch no 92:     Train Loss: 0.036510    Train Accuracy: 0.979380
Validation Accuracy: 0.897200
Epoch no 93:     Train Loss: 0.026120    Train Accuracy: 0.979200
Validation Accuracy: 0.894600
Epoch no 94:     Train Loss: 0.051515    Train Accuracy: 0.979400
Validation Accuracy: 0.895800
Epoch no 95:     Train Loss: 0.055090    Train Accuracy: 0.979800
Validation Accuracy: 0.895100
Epoch no 96:     Train Loss: 0.072777    Train Accuracy: 0.979480
Validation Accuracy: 0.896100
Epoch no 97:     Train Loss: 0.048367    Train Accuracy: 0.979320
Validation Accuracy: 0.893900
Epoch no 98:     Train Loss: 0.048502    Train Accuracy: 0.979580
Validation Accuracy: 0.896600
Epoch no 99:     Train Loss: 0.053568    Train Accuracy: 0.979960
Validation Accuracy: 0.895800
Epoch no 100:    Train Loss: 0.052389    Train Accuracy: 0.979740
Validation Accuracy: 0.898200
0.8982
```



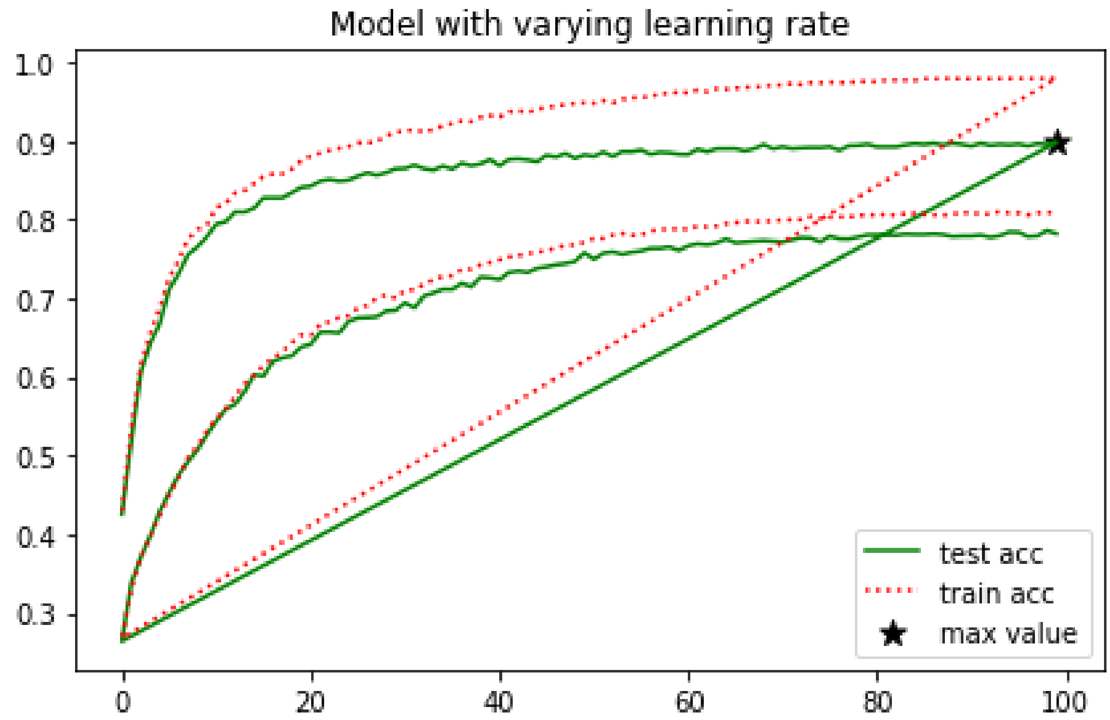

```
Epoch no 1:      Train Loss: 1.937199    Train Accuracy: 0.268000
Validation Accuracy: 0.264500
Epoch no 2:      Train Loss: 1.736936    Train Accuracy: 0.330880
Validation Accuracy: 0.338600
Epoch no 3:      Train Loss: 1.675602    Train Accuracy: 0.371100
Validation Accuracy: 0.371200
Epoch no 4:      Train Loss: 1.650199    Train Accuracy: 0.397520
Validation Accuracy: 0.398800
Epoch no 5:      Train Loss: 1.478430    Train Accuracy: 0.424040
Validation Accuracy: 0.429200
Epoch no 6:      Train Loss: 1.502993    Train Accuracy: 0.449880
Validation Accuracy: 0.453200
Epoch no 7:      Train Loss: 1.438302    Train Accuracy: 0.472440
Validation Accuracy: 0.475600
Epoch no 8:      Train Loss: 1.495572    Train Accuracy: 0.498360
Validation Accuracy: 0.492500
Epoch no 9:      Train Loss: 1.393568    Train Accuracy: 0.511940
Validation Accuracy: 0.507600
Epoch no 10:     Train Loss: 1.297100    Train Accuracy: 0.531480
Validation Accuracy: 0.526600
Epoch no 11:     Train Loss: 1.225275    Train Accuracy: 0.547840
Validation Accuracy: 0.543800
Epoch no 12:     Train Loss: 1.198554    Train Accuracy: 0.560180
Validation Accuracy: 0.559500
```

```
Epoch no 13:     Train Loss: 1.079572    Train Accuracy: 0.575520
Validation Accuracy: 0.565500
Epoch no 14:     Train Loss: 1.119933    Train Accuracy: 0.591600
Validation Accuracy: 0.581500
Epoch no 15:     Train Loss: 1.004398    Train Accuracy: 0.602640
Validation Accuracy: 0.603000
Epoch no 16:     Train Loss: 1.036238    Train Accuracy: 0.614780
Validation Accuracy: 0.601300
Epoch no 17:     Train Loss: 1.177518    Train Accuracy: 0.624260
Validation Accuracy: 0.619800
Epoch no 18:     Train Loss: 1.011388    Train Accuracy: 0.632360
Validation Accuracy: 0.624500
Epoch no 19:     Train Loss: 0.987702    Train Accuracy: 0.643840
Validation Accuracy: 0.627200
Epoch no 20:     Train Loss: 0.984136    Train Accuracy: 0.653520
Validation Accuracy: 0.638200
Epoch no 21:     Train Loss: 1.065981    Train Accuracy: 0.653020
Validation Accuracy: 0.641700
Epoch no 22:     Train Loss: 0.975319    Train Accuracy: 0.666800
Validation Accuracy: 0.656900
Epoch no 23:     Train Loss: 0.993495    Train Accuracy: 0.670500
Validation Accuracy: 0.657200
Epoch no 24:     Train Loss: 0.859980    Train Accuracy: 0.675680
Validation Accuracy: 0.656000
Epoch no 25:     Train Loss: 0.873882    Train Accuracy: 0.677820
Validation Accuracy: 0.670700
Epoch no 26:     Train Loss: 0.967795    Train Accuracy: 0.688140
Validation Accuracy: 0.675200
Epoch no 27:     Train Loss: 0.908626    Train Accuracy: 0.690960
Validation Accuracy: 0.675700
Epoch no 28:     Train Loss: 0.855202    Train Accuracy: 0.700220
Validation Accuracy: 0.676600
Epoch no 29:     Train Loss: 0.915573    Train Accuracy: 0.704560
Validation Accuracy: 0.683900
Epoch no 30:     Train Loss: 0.926570    Train Accuracy: 0.700100
Validation Accuracy: 0.684800
Epoch no 31:     Train Loss: 0.756546    Train Accuracy: 0.710340
Validation Accuracy: 0.694400
Epoch no 32:     Train Loss: 0.770505    Train Accuracy: 0.711040
Validation Accuracy: 0.688000
Epoch no 33:     Train Loss: 0.763044    Train Accuracy: 0.718140
Validation Accuracy: 0.703200
Epoch no 34:     Train Loss: 0.700670    Train Accuracy: 0.723580
Validation Accuracy: 0.707000
Epoch no 35:     Train Loss: 0.820081    Train Accuracy: 0.727900
Validation Accuracy: 0.711700
Epoch no 36:     Train Loss: 0.757850    Train Accuracy: 0.728720
Validation Accuracy: 0.711500
```

```
Epoch no 37:     Train Loss: 0.788248    Train Accuracy: 0.736780
Validation Accuracy: 0.719300
Epoch no 38:     Train Loss: 0.703931    Train Accuracy: 0.737280
Validation Accuracy: 0.714800
Epoch no 39:     Train Loss: 0.656543    Train Accuracy: 0.742740
Validation Accuracy: 0.726900
Epoch no 40:     Train Loss: 0.779114    Train Accuracy: 0.745520
Validation Accuracy: 0.726100
Epoch no 41:     Train Loss: 0.625197    Train Accuracy: 0.749220
Validation Accuracy: 0.723800
Epoch no 42:     Train Loss: 0.713501    Train Accuracy: 0.752580
Validation Accuracy: 0.732900
Epoch no 43:     Train Loss: 0.587899    Train Accuracy: 0.754240
Validation Accuracy: 0.734600
Epoch no 44:     Train Loss: 0.717642    Train Accuracy: 0.756620
Validation Accuracy: 0.733000
Epoch no 45:     Train Loss: 0.700913    Train Accuracy: 0.759140
Validation Accuracy: 0.738400
Epoch no 46:     Train Loss: 0.639544    Train Accuracy: 0.760260
Validation Accuracy: 0.740200
Epoch no 47:     Train Loss: 0.700491    Train Accuracy: 0.764460
Validation Accuracy: 0.742800
Epoch no 48:     Train Loss: 0.524590    Train Accuracy: 0.767060
Validation Accuracy: 0.744400
Epoch no 49:     Train Loss: 0.604569    Train Accuracy: 0.767840
Validation Accuracy: 0.747000
Epoch no 50:     Train Loss: 0.612466    Train Accuracy: 0.771520
Validation Accuracy: 0.758000
Epoch no 51:     Train Loss: 0.600165    Train Accuracy: 0.776000
Validation Accuracy: 0.749800
Epoch no 52:     Train Loss: 0.554372    Train Accuracy: 0.776240
Validation Accuracy: 0.757400
Epoch no 53:     Train Loss: 0.558364    Train Accuracy: 0.780620
Validation Accuracy: 0.759200
Epoch no 54:     Train Loss: 0.657679    Train Accuracy: 0.782520
Validation Accuracy: 0.756000
Epoch no 55:     Train Loss: 0.555470    Train Accuracy: 0.780420
Validation Accuracy: 0.758600
Epoch no 56:     Train Loss: 0.602070    Train Accuracy: 0.780520
Validation Accuracy: 0.760200
Epoch no 57:     Train Loss: 0.590423    Train Accuracy: 0.785740
Validation Accuracy: 0.763600
Epoch no 58:     Train Loss: 0.649964    Train Accuracy: 0.787020
Validation Accuracy: 0.763600
Epoch no 59:     Train Loss: 0.624270    Train Accuracy: 0.788600
Validation Accuracy: 0.765400
Epoch no 60:     Train Loss: 0.580016    Train Accuracy: 0.787780
Validation Accuracy: 0.760900
```

```
Epoch no 61:     Train Loss: 0.603930    Train Accuracy: 0.788300
Validation Accuracy: 0.766700
Epoch no 62:     Train Loss: 0.578228    Train Accuracy: 0.791740
Validation Accuracy: 0.769500
Epoch no 63:     Train Loss: 0.520798    Train Accuracy: 0.791000
Validation Accuracy: 0.768100
Epoch no 64:     Train Loss: 0.591339    Train Accuracy: 0.791800
Validation Accuracy: 0.770500
Epoch no 65:     Train Loss: 0.540336    Train Accuracy: 0.794820
Validation Accuracy: 0.775600
Epoch no 66:     Train Loss: 0.502203    Train Accuracy: 0.796100
Validation Accuracy: 0.770700
Epoch no 67:     Train Loss: 0.558397    Train Accuracy: 0.797840
Validation Accuracy: 0.771800
Epoch no 68:     Train Loss: 0.581024    Train Accuracy: 0.797860
Validation Accuracy: 0.772700
Epoch no 69:     Train Loss: 0.530847    Train Accuracy: 0.797220
Validation Accuracy: 0.773500
Epoch no 70:     Train Loss: 0.547827    Train Accuracy: 0.798900
Validation Accuracy: 0.772600
Epoch no 71:     Train Loss: 0.533988    Train Accuracy: 0.799500
Validation Accuracy: 0.771800
Epoch no 72:     Train Loss: 0.481491    Train Accuracy: 0.799880
Validation Accuracy: 0.774400
Epoch no 73:     Train Loss: 0.527654    Train Accuracy: 0.800100
Validation Accuracy: 0.777300
Epoch no 74:     Train Loss: 0.655238    Train Accuracy: 0.803920
Validation Accuracy: 0.776700
Epoch no 75:     Train Loss: 0.600064    Train Accuracy: 0.801300
Validation Accuracy: 0.771700
Epoch no 76:     Train Loss: 0.489581    Train Accuracy: 0.803120
Validation Accuracy: 0.779300
Epoch no 77:     Train Loss: 0.509487    Train Accuracy: 0.803460
Validation Accuracy: 0.775800
Epoch no 78:     Train Loss: 0.561105    Train Accuracy: 0.803840
Validation Accuracy: 0.775600
Epoch no 79:     Train Loss: 0.510850    Train Accuracy: 0.804000
Validation Accuracy: 0.778000
Epoch no 80:     Train Loss: 0.568577    Train Accuracy: 0.807320
Validation Accuracy: 0.780900
Epoch no 81:     Train Loss: 0.551413    Train Accuracy: 0.804380
Validation Accuracy: 0.781000
Epoch no 82:     Train Loss: 0.487197    Train Accuracy: 0.807540
Validation Accuracy: 0.780800
Epoch no 83:     Train Loss: 0.491432    Train Accuracy: 0.805020
Validation Accuracy: 0.780300
Epoch no 84:     Train Loss: 0.487874    Train Accuracy: 0.807960
Validation Accuracy: 0.781800
```

```
Epoch no 85:     Train Loss: 0.536771    Train Accuracy: 0.807720
Validation Accuracy: 0.781300
Epoch no 86:     Train Loss: 0.546496    Train Accuracy: 0.806640
Validation Accuracy: 0.781300
Epoch no 87:     Train Loss: 0.599500    Train Accuracy: 0.805940
Validation Accuracy: 0.782100
Epoch no 88:     Train Loss: 0.516174    Train Accuracy: 0.805360
Validation Accuracy: 0.781200
Epoch no 89:     Train Loss: 0.552343    Train Accuracy: 0.807160
Validation Accuracy: 0.778800
Epoch no 90:     Train Loss: 0.426796    Train Accuracy: 0.808960
Validation Accuracy: 0.779400
Epoch no 91:     Train Loss: 0.567955    Train Accuracy: 0.808420
Validation Accuracy: 0.779400
Epoch no 92:     Train Loss: 0.435370    Train Accuracy: 0.807520
Validation Accuracy: 0.780600
Epoch no 93:     Train Loss: 0.529083    Train Accuracy: 0.806980
Validation Accuracy: 0.777700
Epoch no 94:     Train Loss: 0.663087    Train Accuracy: 0.810480
Validation Accuracy: 0.779700
Epoch no 95:     Train Loss: 0.526050    Train Accuracy: 0.808380
Validation Accuracy: 0.784400
Epoch no 96:     Train Loss: 0.650508    Train Accuracy: 0.808360
Validation Accuracy: 0.784100
Epoch no 97:     Train Loss: 0.463966    Train Accuracy: 0.805980
Validation Accuracy: 0.778700
Epoch no 98:     Train Loss: 0.527538    Train Accuracy: 0.808320
Validation Accuracy: 0.779500
Epoch no 99:     Train Loss: 0.474008    Train Accuracy: 0.809100
Validation Accuracy: 0.786200
Epoch no 100:    Train Loss: 0.533869    Train Accuracy: 0.808100
Validation Accuracy: 0.782400
0.8982
```



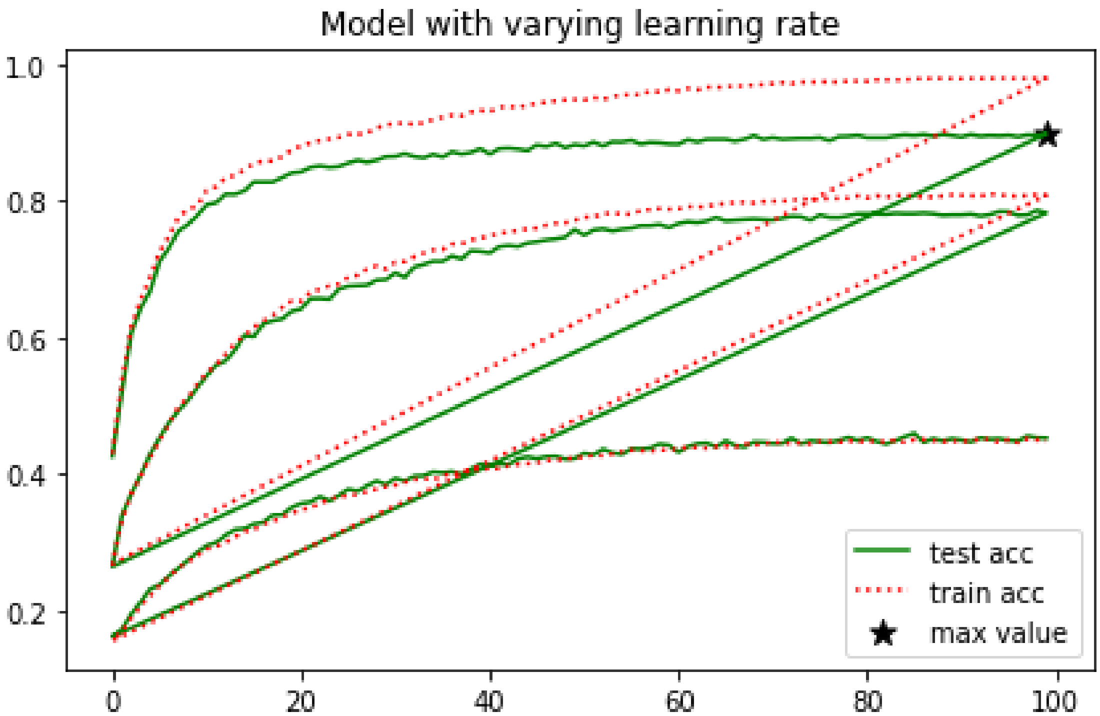

```
Epoch no 1:      Train Loss: 2.258975    Train Accuracy: 0.155800
Validation Accuracy: 0.162000
Epoch no 2:      Train Loss: 2.237513    Train Accuracy: 0.169620
Validation Accuracy: 0.172300
Epoch no 3:      Train Loss: 2.166926    Train Accuracy: 0.188860
Validation Accuracy: 0.194900
Epoch no 4:      Train Loss: 2.111222    Train Accuracy: 0.206640
Validation Accuracy: 0.212700
Epoch no 5:      Train Loss: 2.112028    Train Accuracy: 0.221640
Validation Accuracy: 0.231400
Epoch no 6:      Train Loss: 2.088784    Train Accuracy: 0.236500
Validation Accuracy: 0.237900
Epoch no 7:      Train Loss: 2.067389    Train Accuracy: 0.249100
Validation Accuracy: 0.250800
Epoch no 8:      Train Loss: 1.992879    Train Accuracy: 0.262860
Validation Accuracy: 0.261900
Epoch no 9:      Train Loss: 1.915033    Train Accuracy: 0.271160
Validation Accuracy: 0.273700
Epoch no 10:     Train Loss: 1.918703    Train Accuracy: 0.280360
Validation Accuracy: 0.285400
Epoch no 11:     Train Loss: 1.865704    Train Accuracy: 0.289960
Validation Accuracy: 0.295000
Epoch no 12:     Train Loss: 1.867377    Train Accuracy: 0.294020
Validation Accuracy: 0.299700
```

```
Epoch no 13:     Train Loss: 1.835096    Train Accuracy: 0.299740
Validation Accuracy: 0.310400
Epoch no 14:     Train Loss: 1.832796    Train Accuracy: 0.307940
Validation Accuracy: 0.318600
Epoch no 15:     Train Loss: 1.921151    Train Accuracy: 0.315600
Validation Accuracy: 0.319900
Epoch no 16:     Train Loss: 1.789480    Train Accuracy: 0.318360
Validation Accuracy: 0.326900
Epoch no 17:     Train Loss: 1.784620    Train Accuracy: 0.326620
Validation Accuracy: 0.329200
Epoch no 18:     Train Loss: 1.783508    Train Accuracy: 0.330360
Validation Accuracy: 0.339800
Epoch no 19:     Train Loss: 1.762589    Train Accuracy: 0.336840
Validation Accuracy: 0.340200
Epoch no 20:     Train Loss: 1.726450    Train Accuracy: 0.341940
Validation Accuracy: 0.351000
Epoch no 21:     Train Loss: 1.747860    Train Accuracy: 0.348240
Validation Accuracy: 0.356900
Epoch no 22:     Train Loss: 1.649477    Train Accuracy: 0.349560
Validation Accuracy: 0.359000
Epoch no 23:     Train Loss: 1.767146    Train Accuracy: 0.358260
Validation Accuracy: 0.367000
Epoch no 24:     Train Loss: 1.705548    Train Accuracy: 0.361520
Validation Accuracy: 0.360700
Epoch no 25:     Train Loss: 1.728027    Train Accuracy: 0.367920
Validation Accuracy: 0.368400
Epoch no 26:     Train Loss: 1.661944    Train Accuracy: 0.367540
Validation Accuracy: 0.377700
Epoch no 27:     Train Loss: 1.720719    Train Accuracy: 0.370560
Validation Accuracy: 0.374800
Epoch no 28:     Train Loss: 1.608156    Train Accuracy: 0.374360
Validation Accuracy: 0.382800
Epoch no 29:     Train Loss: 1.668926    Train Accuracy: 0.378580
Validation Accuracy: 0.389600
Epoch no 30:     Train Loss: 1.607167    Train Accuracy: 0.381520
Validation Accuracy: 0.384500
Epoch no 31:     Train Loss: 1.617993    Train Accuracy: 0.383800
Validation Accuracy: 0.393800
Epoch no 32:     Train Loss: 1.570849    Train Accuracy: 0.387780
Validation Accuracy: 0.389800
Epoch no 33:     Train Loss: 1.636252    Train Accuracy: 0.391560
Validation Accuracy: 0.396900
Epoch no 34:     Train Loss: 1.534938    Train Accuracy: 0.393360
Validation Accuracy: 0.397500
Epoch no 35:     Train Loss: 1.655763    Train Accuracy: 0.391100
Validation Accuracy: 0.403200
Epoch no 36:     Train Loss: 1.592922    Train Accuracy: 0.398120
Validation Accuracy: 0.401800
```

```
Epoch no 37:     Train Loss: 1.622477    Train Accuracy: 0.400620
Validation Accuracy: 0.403700
Epoch no 38:     Train Loss: 1.541352    Train Accuracy: 0.401160
Validation Accuracy: 0.399800
Epoch no 39:     Train Loss: 1.538736    Train Accuracy: 0.407820
Validation Accuracy: 0.407000
Epoch no 40:     Train Loss: 1.578691    Train Accuracy: 0.405140
Validation Accuracy: 0.414500
Epoch no 41:     Train Loss: 1.536106    Train Accuracy: 0.408520
Validation Accuracy: 0.411500
Epoch no 42:     Train Loss: 1.521565    Train Accuracy: 0.407680
Validation Accuracy: 0.414600
Epoch no 43:     Train Loss: 1.557563    Train Accuracy: 0.412360
Validation Accuracy: 0.413900
Epoch no 44:     Train Loss: 1.597900    Train Accuracy: 0.414160
Validation Accuracy: 0.422200
Epoch no 45:     Train Loss: 1.627629    Train Accuracy: 0.415360
Validation Accuracy: 0.425100
Epoch no 46:     Train Loss: 1.540853    Train Accuracy: 0.416260
Validation Accuracy: 0.422500
Epoch no 47:     Train Loss: 1.472290    Train Accuracy: 0.419700
Validation Accuracy: 0.425500
Epoch no 48:     Train Loss: 1.586229    Train Accuracy: 0.420700
Validation Accuracy: 0.425900
Epoch no 49:     Train Loss: 1.562360    Train Accuracy: 0.420860
Validation Accuracy: 0.421800
Epoch no 50:     Train Loss: 1.565719    Train Accuracy: 0.424340
Validation Accuracy: 0.422700
Epoch no 51:     Train Loss: 1.643622    Train Accuracy: 0.423980
Validation Accuracy: 0.430500
Epoch no 52:     Train Loss: 1.527971    Train Accuracy: 0.427840
Validation Accuracy: 0.428000
Epoch no 53:     Train Loss: 1.482338    Train Accuracy: 0.426640
Validation Accuracy: 0.436500
Epoch no 54:     Train Loss: 1.451283    Train Accuracy: 0.429420
Validation Accuracy: 0.431100
Epoch no 55:     Train Loss: 1.586983    Train Accuracy: 0.432880
Validation Accuracy: 0.428700
Epoch no 56:     Train Loss: 1.527161    Train Accuracy: 0.435080
Validation Accuracy: 0.436900
Epoch no 57:     Train Loss: 1.436552    Train Accuracy: 0.434640
Validation Accuracy: 0.436600
Epoch no 58:     Train Loss: 1.554333    Train Accuracy: 0.433800
Validation Accuracy: 0.438000
Epoch no 59:     Train Loss: 1.520711    Train Accuracy: 0.435100
Validation Accuracy: 0.444000
Epoch no 60:     Train Loss: 1.429845    Train Accuracy: 0.435380
Validation Accuracy: 0.442200
```

```
Epoch no 61:     Train Loss: 1.566994    Train Accuracy: 0.436820
Validation Accuracy: 0.432100
Epoch no 62:     Train Loss: 1.496023    Train Accuracy: 0.439240
Validation Accuracy: 0.440200
Epoch no 63:     Train Loss: 1.478031    Train Accuracy: 0.439000
Validation Accuracy: 0.444500
Epoch no 64:     Train Loss: 1.540565    Train Accuracy: 0.438100
Validation Accuracy: 0.445500
Epoch no 65:     Train Loss: 1.380395    Train Accuracy: 0.440160
Validation Accuracy: 0.439700
Epoch no 66:     Train Loss: 1.466800    Train Accuracy: 0.441080
Validation Accuracy: 0.448600
Epoch no 67:     Train Loss: 1.475549    Train Accuracy: 0.439620
Validation Accuracy: 0.442100
Epoch no 68:     Train Loss: 1.511357    Train Accuracy: 0.444520
Validation Accuracy: 0.439800
Epoch no 69:     Train Loss: 1.524244    Train Accuracy: 0.442240
Validation Accuracy: 0.446800
Epoch no 70:     Train Loss: 1.459030    Train Accuracy: 0.443300
Validation Accuracy: 0.446500
Epoch no 71:     Train Loss: 1.548821    Train Accuracy: 0.447620
Validation Accuracy: 0.444800
Epoch no 72:     Train Loss: 1.503010    Train Accuracy: 0.444740
Validation Accuracy: 0.445100
Epoch no 73:     Train Loss: 1.462397    Train Accuracy: 0.445140
Validation Accuracy: 0.452500
Epoch no 74:     Train Loss: 1.438167    Train Accuracy: 0.445560
Validation Accuracy: 0.444500
Epoch no 75:     Train Loss: 1.575677    Train Accuracy: 0.444020
Validation Accuracy: 0.448600
Epoch no 76:     Train Loss: 1.444219    Train Accuracy: 0.447620
Validation Accuracy: 0.447500
Epoch no 77:     Train Loss: 1.490081    Train Accuracy: 0.447140
Validation Accuracy: 0.446100
Epoch no 78:     Train Loss: 1.533145    Train Accuracy: 0.446680
Validation Accuracy: 0.449200
Epoch no 79:     Train Loss: 1.457329    Train Accuracy: 0.448540
Validation Accuracy: 0.453600
Epoch no 80:     Train Loss: 1.488118    Train Accuracy: 0.447800
Validation Accuracy: 0.449200
Epoch no 81:     Train Loss: 1.412159    Train Accuracy: 0.447180
Validation Accuracy: 0.452800
Epoch no 82:     Train Loss: 1.415582    Train Accuracy: 0.448540
Validation Accuracy: 0.447900
Epoch no 83:     Train Loss: 1.435688    Train Accuracy: 0.446620
Validation Accuracy: 0.444000
Epoch no 84:     Train Loss: 1.561076    Train Accuracy: 0.449800
Validation Accuracy: 0.448900
```

```
Epoch no 85:     Train Loss: 1.436736    Train Accuracy: 0.446600
Validation Accuracy: 0.454600
Epoch no 86:     Train Loss: 1.562685    Train Accuracy: 0.450440
Validation Accuracy: 0.459600
Epoch no 87:     Train Loss: 1.475741    Train Accuracy: 0.450300
Validation Accuracy: 0.446500
Epoch no 88:     Train Loss: 1.439545    Train Accuracy: 0.448880
Validation Accuracy: 0.451800
Epoch no 89:     Train Loss: 1.438568    Train Accuracy: 0.447380
Validation Accuracy: 0.448600
Epoch no 90:     Train Loss: 1.444297    Train Accuracy: 0.448040
Validation Accuracy: 0.451700
Epoch no 91:     Train Loss: 1.426236    Train Accuracy: 0.447600
Validation Accuracy: 0.449800
Epoch no 92:     Train Loss: 1.423243    Train Accuracy: 0.448520
Validation Accuracy: 0.447100
Epoch no 93:     Train Loss: 1.476407    Train Accuracy: 0.451200
Validation Accuracy: 0.452800
Epoch no 94:     Train Loss: 1.501691    Train Accuracy: 0.450960
Validation Accuracy: 0.451100
Epoch no 95:     Train Loss: 1.470573    Train Accuracy: 0.446220
Validation Accuracy: 0.451700
Epoch no 96:     Train Loss: 1.595160    Train Accuracy: 0.450140
Validation Accuracy: 0.449300
Epoch no 97:     Train Loss: 1.515221    Train Accuracy: 0.446140
Validation Accuracy: 0.448300
Epoch no 98:     Train Loss: 1.478431    Train Accuracy: 0.451700
Validation Accuracy: 0.455700
Epoch no 99:     Train Loss: 1.538060    Train Accuracy: 0.450140
Validation Accuracy: 0.453100
Epoch no 100:    Train Loss: 1.452517    Train Accuracy: 0.449260
Validation Accuracy: 0.452700
0.8982
```


Model with varying learning rate
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100
test acc
train acc
max value