Project Name：　Localization

序号	学号	专业班级	姓名	性别
1				

1. Project Introduction

内容包括：（这部分内容不要太长，讲清楚即可）

（1） 开发环境及系统运行要求，包括所用的开发工具、开发包、开源库、系统运行要求等；

- Linux
- Ubuntu16.04
- Python3.6.1
- CUDA 9.0
- Pytorch
- Jupyter Notebook

本次实验使用 AWS 的 p2.xlarge 服务器，在 Anaconda 中配置对应的环境。

（2） 工作分配简介，即谁要做什么事情

独立完成

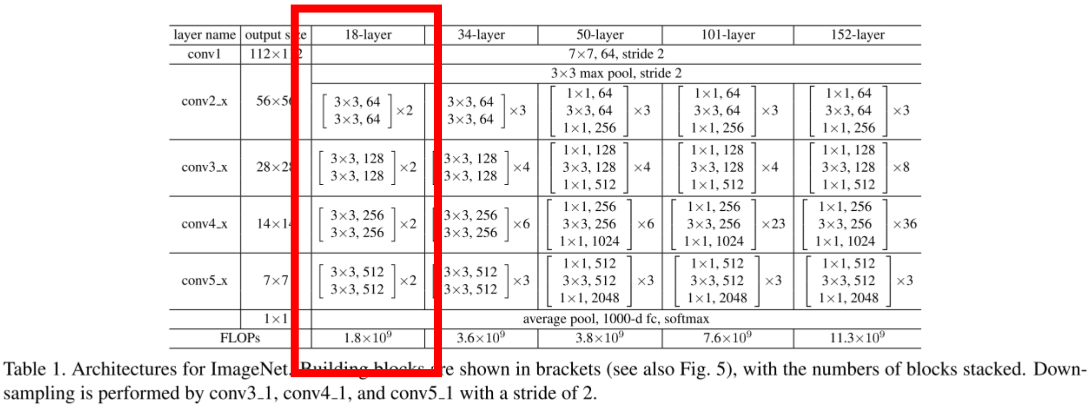
2. Technical Details

内容包括：

（1） 工程实践当中所用到的理论知识阐述

- 模型

使用在 ImageNet 上预训练好的 18 层残差网络。



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\left[\begin{array}{c}3\times3, 64\\3\times3, 64\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 64\\3\times3, 64\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 64\\3\times3, 64\\1\times1, 256\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 64\\3\times3, 64\\1\times1, 256\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 64\\3\times3, 64\\1\times1, 256\end{array}\right]\times3$
conv3_x	28×28	$\left[\begin{array}{c}3\times3, 128\\3\times3, 128\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 128\\3\times3, 128\end{array}\right]\times4$	$\left[\begin{array}{c}1\times1, 128\\3\times3, 128\\1\times1, 512\end{array}\right]\times4$	$\left[\begin{array}{c}1\times1, 128\\3\times3, 128\\1\times1, 512\end{array}\right]\times4$	$\left[\begin{array}{c}1\times1, 128\\3\times3, 128\\1\times1, 512\end{array}\right]\times8$
conv4_x	14×14	$\left[\begin{array}{c}3\times3, 256\\3\times3, 256\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 256\\3\times3, 256\end{array}\right]\times6$	$\left[\begin{array}{c}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{array}\right]\times6$	$\left[\begin{array}{c}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{array}\right]\times23$	$\left[\begin{array}{c}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{array}\right]\times36$
conv5_x	7×7	$\left[\begin{array}{c}3\times3, 512\\3\times3, 512\end{array}\right]\times2$	$\left[\begin{array}{c}3\times3, 512\\3\times3, 512\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{array}\right]\times3$	$\left[\begin{array}{c}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{array}\right]\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 1. Architectures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stacked. Downsampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

- 损失函数

$$L_{\text{loc}}(t^u, v) = \sum_{i\in\{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i),$$

in which

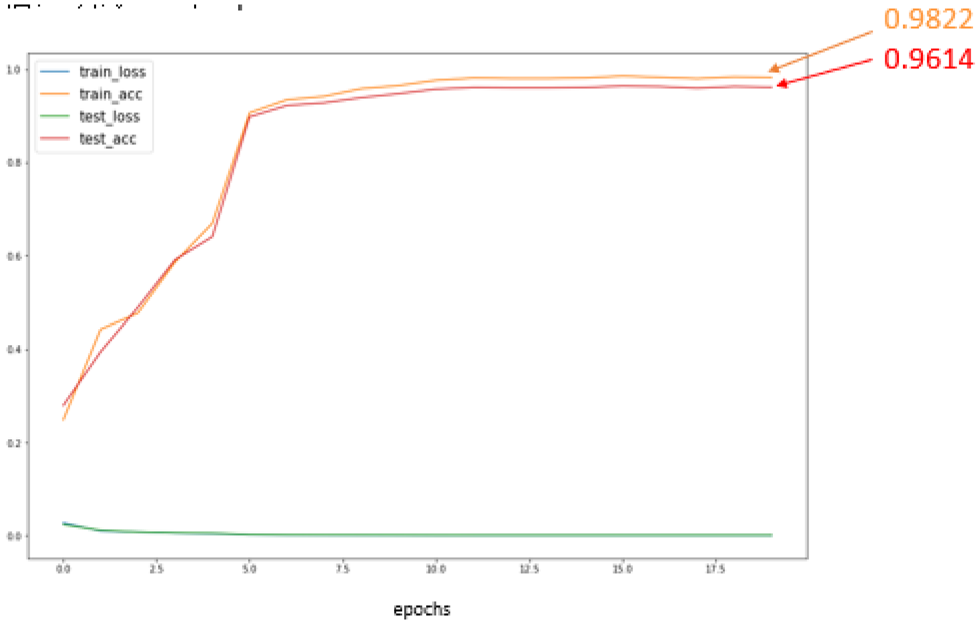
$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

- 正确率

若预测的 bounding box 与 ground truth 的 IoU 大于等于 0.75， 则认为预测正确。

- 训练

使用 Adam 算法进行优化，batch size 为 32，初始学习率设置为 1e-3，每隔 5 个 epoch 将学习率降低 10 倍，训练 20 个 epochs。



（2） 具体的算法，请用文字、示意图或者是伪代码等形式进行描述（不要

贴大段的代码）

- 准备数据：

```
dataloaders = {split: torch.utils.data.DataLoader(
                datasets[split], batch_size=32,shuffle=(split=='train'),
                num_workers=2, pin_memory=True) for split in
('train', 'test')}
```

- 构建模型

```
bar = progressbar.ProgressBar()
        for ims, boxes, im_sizes in bar(dataloaders[phase]):
            boxes = crop_boxes(boxes, im_sizes)
            boxes = box_transform(boxes, im_sizes)

            inputs = Variable(ims.cuda())
            targets = Variable(boxes.cuda())

            optimizer.zero_grad()

            # forward
            outputs = model(inputs)
            loss = criterion(outputs, targets)
            acc = compute_acc(outputs.data.cpu(), targets.data.cpu(),
im_sizes)

            nsample = inputs.size(0)
            accs.update(acc, nsample)
            losses.update(loss.data[0], nsample)

            if phase == 'train':
                loss.backward()
                optimizer.step()
```

- 计算 Acc：

```
def compute_acc(preds, targets, im_sizes, theta=0.75):
    preds = box_transform_inv(preds.clone(), im_sizes)
    preds = crop_boxes(preds, im_sizes)
    targets = box_transform_inv(targets.clone(), im_sizes)
```

```
IoU = compute_IoU(preds, targets)
corr = (IoU >= theta).sum()
return corr.item() / preds.size(0)
```

（3） 程序开发中重要的技术细节，比如用到了哪些重要的函数？这些函数来自于哪些基本库？功能是什么？自己编写了哪些重要的功能函数？等等

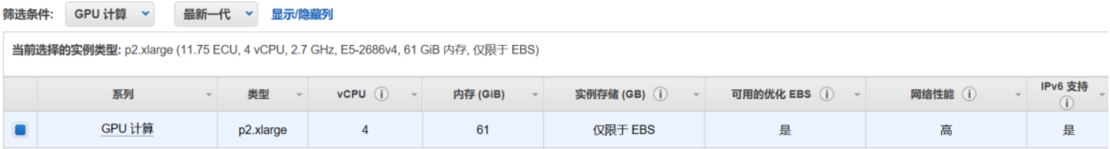
- torch 本次实验最重要的库，本次程序基于 pytorch 完成。

3. Experiment Results

用图文并茂的形式给出实验结果，如系统界面、操作说明、运行结果等，并对实验结果进行总结和说明。

1. 服务器：

本次实验在 AWS 端选择 p2.xlarge 类服务器。

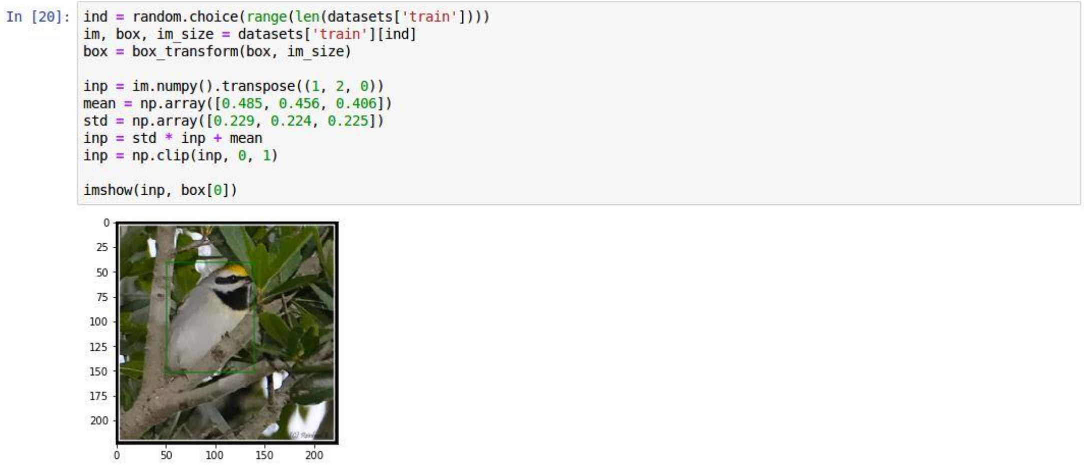


筛选条件: GPU 计算 ⌄ 最新一代 ⌄ 显示/隐藏列

当前选择的实例类型: p2.xlarge (11.75 ECU, 4 vCPU, 2.7 GHz, E5-2686v4, 61 GiB 内存, 仅限于 EBS)

	系列	类型	vCPU	内存 (GiB)	实例存储 (GB)	可用的优化 EBS	网络性能	IPv6 支持
■	GPU 计算	p2.xlarge	4	61	仅限于 EBS	是	高	是

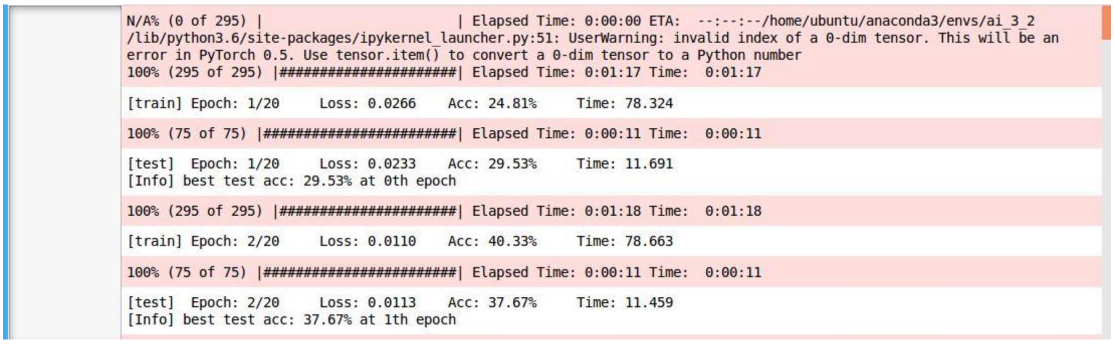
2. Localization 结果：



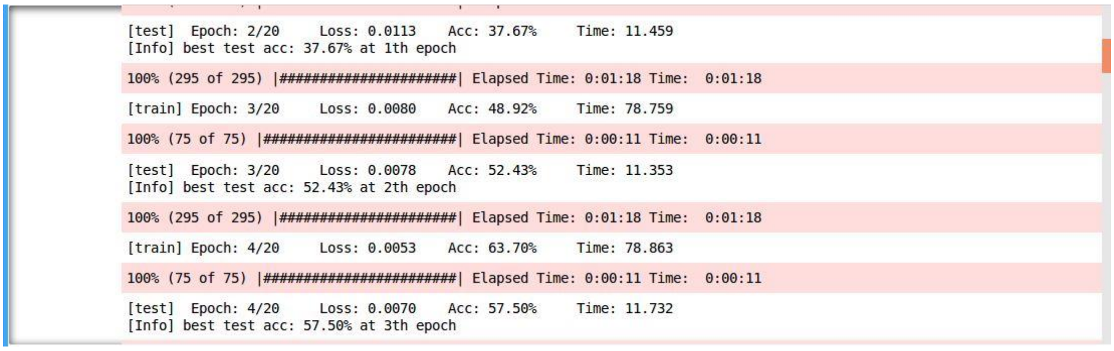
```
In [20]: ind = random.choice(range(len(datasets['train'])))
         im, box, im_size = datasets['train'][ind]
         box = box_transform(box, im_size)

         inp = im.numpy().transpose((1, 2, 0))
         mean = np.array([0.485, 0.456, 0.406])
         std = np.array([0.229, 0.224, 0.225])
         inp = std * inp + mean
         inp = np.clip(inp, 0, 1)

         imshow(inp, box[0])
```

在 Jupyter Notebook 运行源代码，显示最终的 localization 结果。此结果显示训练效果较好。

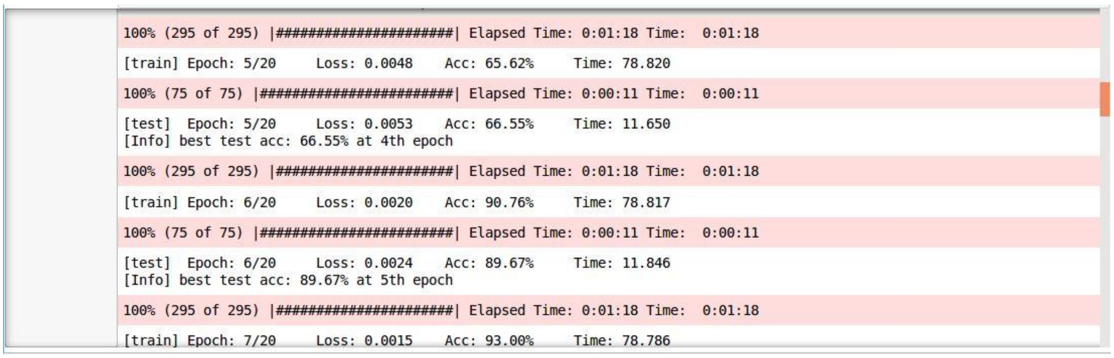
3. 损失函数：



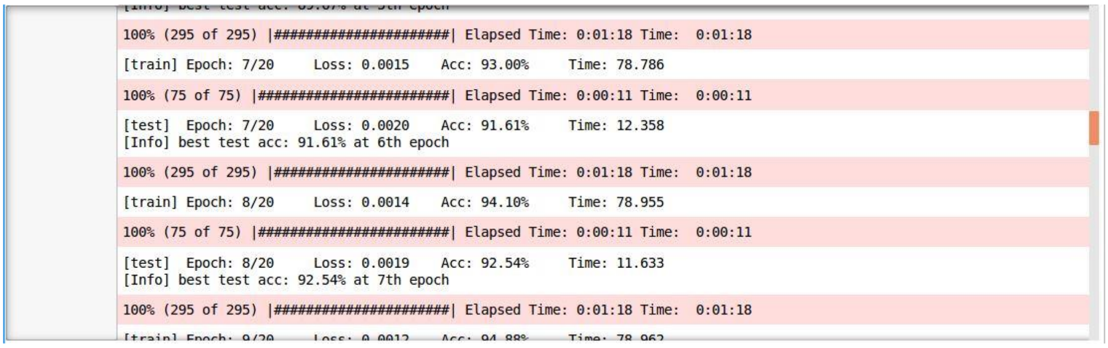
```
N/A% (0 of 295) |                        | Elapsed Time: 0:00:00 ETA:  --:--:--/home/ubuntu/anaconda3/envs/ai_3_2
/lib/python3.6/site-packages/ipykernel_launcher.py:51: UserWarning: invalid index of a 0-dim tensor. This will be an
error in PyTorch 0.5. Use tensor.item() to convert a 0-dim tensor to a Python number
100% (295 of 295) |######################| Elapsed Time: 0:01:17 Time:  0:01:17

[train] Epoch: 1/20     Loss: 0.0266    Acc: 24.81%     Time: 78.324

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 1/20     Loss: 0.0233    Acc: 29.53%     Time: 11.691
[Info] best test acc: 29.53% at 0th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 2/20     Loss: 0.0110    Acc: 40.33%     Time: 78.663

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 2/20     Loss: 0.0113    Acc: 37.67%     Time: 11.459
[Info] best test acc: 37.67% at 1th epoch
```



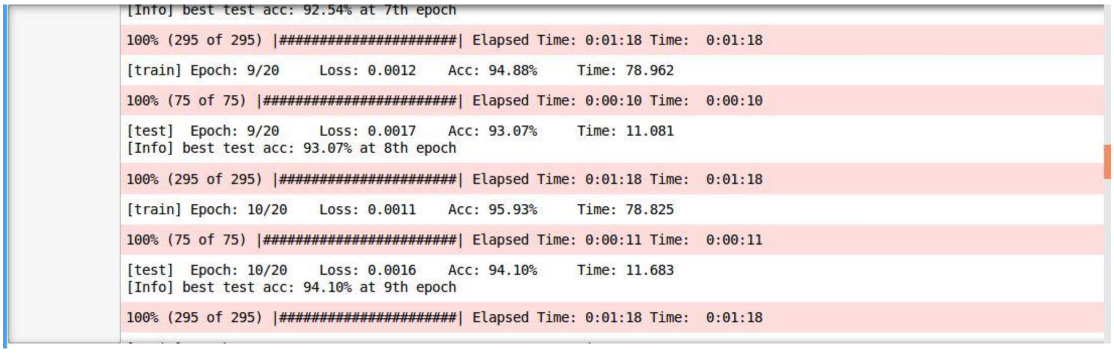
```
[test]  Epoch: 2/20     Loss: 0.0113    Acc: 37.67%     Time: 11.459
[Info] best test acc: 37.67% at 1th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 3/20     Loss: 0.0080    Acc: 48.92%     Time: 78.759

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 3/20     Loss: 0.0078    Acc: 52.43%     Time: 11.353
[Info] best test acc: 52.43% at 2th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 4/20     Loss: 0.0053    Acc: 63.70%     Time: 78.863

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 4/20     Loss: 0.0070    Acc: 57.50%     Time: 11.732
[Info] best test acc: 57.50% at 3th epoch
```



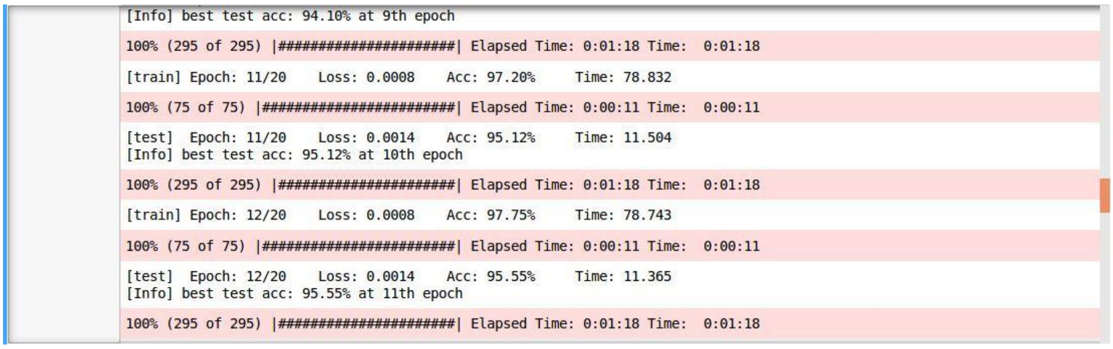
```
100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 5/20     Loss: 0.0048    Acc: 65.62%     Time: 78.820

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 5/20     Loss: 0.0053    Acc: 66.55%     Time: 11.650
[Info] best test acc: 66.55% at 4th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 6/20     Loss: 0.0020    Acc: 90.76%     Time: 78.817

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 6/20     Loss: 0.0024    Acc: 89.67%     Time: 11.846
[Info] best test acc: 89.67% at 5th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 7/20     Loss: 0.0015    Acc: 93.00%     Time: 78.786
```



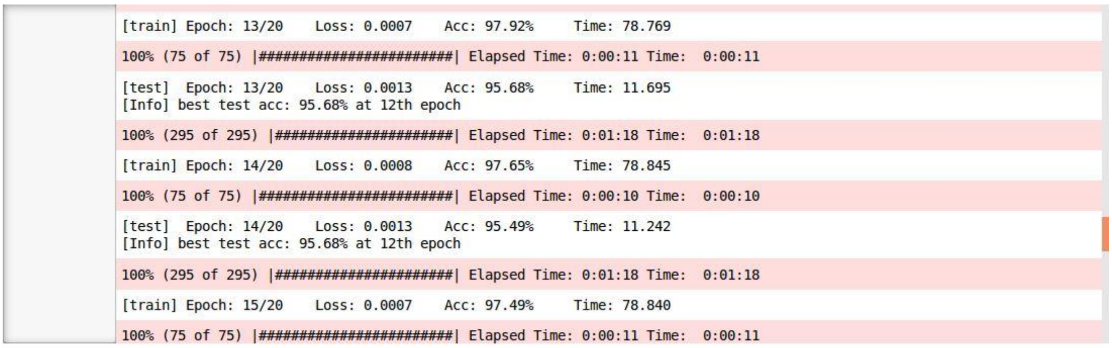
```
[Info] best test acc: 89.67% at 5th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 7/20     Loss: 0.0015    Acc: 93.00%     Time: 78.786

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 7/20     Loss: 0.0020    Acc: 91.61%     Time: 12.358
[Info] best test acc: 91.61% at 6th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 8/20     Loss: 0.0014    Acc: 94.10%     Time: 78.955

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 8/20     Loss: 0.0019    Acc: 92.54%     Time: 11.633
[Info] best test acc: 92.54% at 7th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 9/20     Loss: 0.0012    Acc: 94.88%     Time: 78.962
```



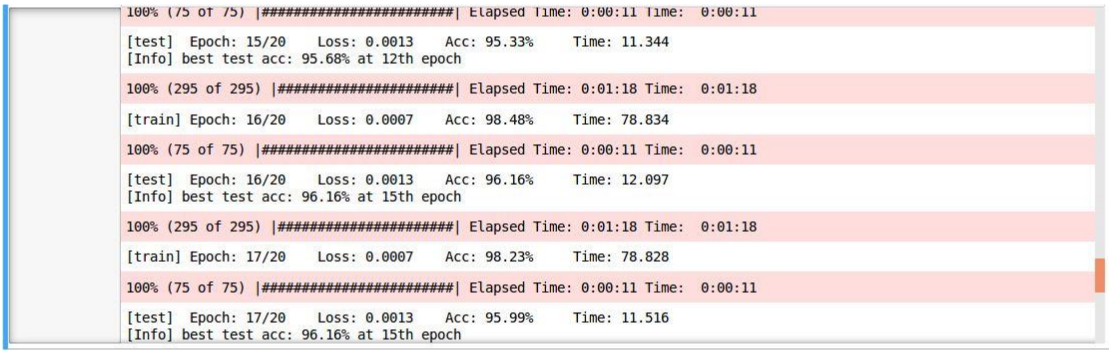
```
[Info] best test acc: 92.54% at 7th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 9/20     Loss: 0.0012    Acc: 94.88%     Time: 78.962

100% (75 of 75) |########################| Elapsed Time: 0:00:10 Time:  0:00:10

[test]  Epoch: 9/20     Loss: 0.0017    Acc: 93.07%     Time: 11.081
[Info] best test acc: 93.07% at 8th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18

[train] Epoch: 10/20    Loss: 0.0011    Acc: 95.93%     Time: 78.825

100% (75 of 75) |########################| Elapsed Time: 0:00:11 Time:  0:00:11

[test]  Epoch: 10/20    Loss: 0.0016    Acc: 94.10%     Time: 11.683
[Info] best test acc: 94.10% at 9th epoch

100% (295 of 295) |######################| Elapsed Time: 0:01:18 Time:  0:01:18
```



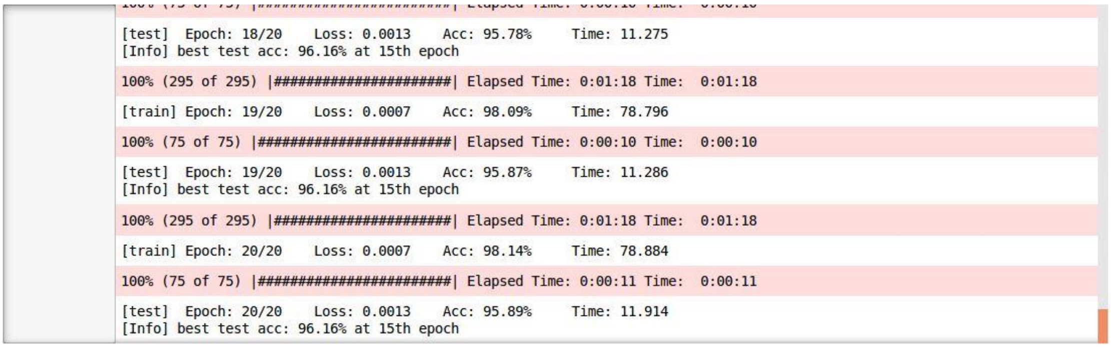
```
[Info] best test acc: 94.10% at 9th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 11/20    Loss: 0.0008    Acc: 97.20%     Time: 78.832
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 11/20    Loss: 0.0014    Acc: 95.12%     Time: 11.504
[Info] best test acc: 95.12% at 10th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 12/20    Loss: 0.0008    Acc: 97.75%     Time: 78.743
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 12/20    Loss: 0.0014    Acc: 95.55%     Time: 11.365
[Info] best test acc: 95.55% at 11th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
```



```
[train] Epoch: 13/20    Loss: 0.0007    Acc: 97.92%     Time: 78.769
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 13/20    Loss: 0.0013    Acc: 95.68%     Time: 11.695
[Info] best test acc: 95.68% at 12th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 14/20    Loss: 0.0008    Acc: 97.65%     Time: 78.845
100% (75 of 75) |#######################| Elapsed Time: 0:00:10 Time:  0:00:10
[test]  Epoch: 14/20    Loss: 0.0013    Acc: 95.49%     Time: 11.242
[Info] best test acc: 95.68% at 12th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 15/20    Loss: 0.0007    Acc: 97.49%     Time: 78.840
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
```



```
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 15/20    Loss: 0.0013    Acc: 95.33%     Time: 11.344
[Info] best test acc: 95.68% at 12th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 16/20    Loss: 0.0007    Acc: 98.48%     Time: 78.834
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 16/20    Loss: 0.0013    Acc: 96.16%     Time: 12.097
[Info] best test acc: 96.16% at 15th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 17/20    Loss: 0.0007    Acc: 98.23%     Time: 78.828
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 17/20    Loss: 0.0013    Acc: 95.99%     Time: 11.516
[Info] best test acc: 96.16% at 15th epoch
```



```
100% (75 of 75) |#######################| Elapsed Time: 0:00:10 Time:  0:00:10
[test]  Epoch: 18/20    Loss: 0.0013    Acc: 95.78%     Time: 11.275
[Info] best test acc: 96.16% at 15th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 19/20    Loss: 0.0007    Acc: 98.09%     Time: 78.796
100% (75 of 75) |#######################| Elapsed Time: 0:00:10 Time:  0:00:10
[test]  Epoch: 19/20    Loss: 0.0013    Acc: 95.87%     Time: 11.286
[Info] best test acc: 96.16% at 15th epoch
100% (295 of 295) |#####################| Elapsed Time: 0:01:18 Time:  0:01:18
[train] Epoch: 20/20    Loss: 0.0007    Acc: 98.14%     Time: 78.884
100% (75 of 75) |#######################| Elapsed Time: 0:00:11 Time:  0:00:11
[test]  Epoch: 20/20    Loss: 0.0013    Acc: 95.89%     Time: 11.914
[Info] best test acc: 96.16% at 15th epoch
```

由上述实验结果可知，总体来说随着训练次数的增加 Acc 在增加，而 Loss 在减少。此实验现象说明，实验训练效果随着训练次数的增加而提高。但随着训练次数的不断增加，将出现过拟合结果，即虽然 training 的效果提高，但是 test 的结果反而下降。

本次实验由于时间比较赶，因此没有在源码的基础上进行优化。前期较大的经历用来配置程序运行所需要的环境。虽然对本次实验的原理没有非常清晰的了解，但是在实验过程中，学会了配置 GPU 以及远程的 Jupyter Notebook 环境，最终

可以运行本次实验原程序。

References:

给出主要的参考文献，可以是论文、网站、书籍、别人的技术报告等。

[1]: https://arxiv.org/abs/1512.03385 ‘Deep Residual Learning for Image Recognition’

备注：

代码中请给出较为详细的注释，此报告中切勿粘贴大量代码，否则扣分。