1. 展示了什么样的困难和挑战？

(1) 如何灵活使用 GPU?

```
rch.Tensor(device='cuda')
import cupy as np
```

(2) CNN 的 loss 下降很慢,如果增大 lr，则会溢出?(目前没什么头绪)

(3) CNN 卷积操作速率慢

2. 取得什么结果？

(1)训练集、测试集准确率

(2)训练时的 loss

(3)混淆矩阵

3. 展示了什么特性?

(1)自动梯度求导

(2)自定义网络结构

```
class MLP:
    def __init__(self, alpha, batch_size):
        self.networks = [
            Linear(1024, 512, alpha, batch_size),
            ReLU(),
            Linear(512, 256, alpha, batch_size),
            ReLU(),
            Linear(256, 64, alpha, batch_size),
            ReLU(),
            Linear(64, 32, alpha, batch_size),
            ReLU(),
            Linear(32, 10, alpha, batch_size),
        ]
        self.loss_f = CrossEntropyLoss(batch_size)

    def forward(self, x, y):
        x = x / 255
        for n in self.networks:
            x = n(x)
        return self.loss_f(x, y)
```

(3)使用 Im2col 算法(把高维矩阵[NxCxHxW]和卷积核分别按照卷积的方式转化成二维矩阵，将不连续的内存连续，利用 cpu 读取内存时的局部性原理，优化矩阵乘法的速率)以此来优化 CNN 的卷积的速率。卷积的池化和反向传播同理。

如图：

假设输入维度为（3,3,3）kernel_size (2,2)