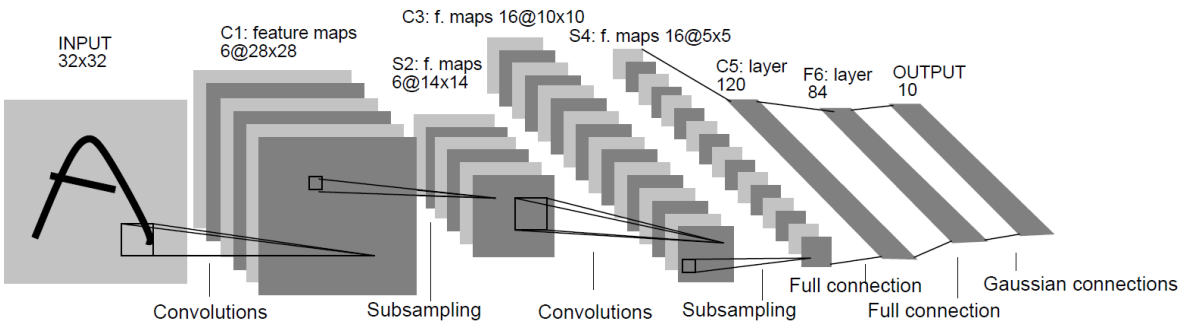
【个人信息】

学号：2011920

姓名：王禹曦

专业：计算机科学与技术

一、网络结构



- 32*32的输入
  - 卷积层*2
  - 池化层*2（最大池化）
  - 全连接层*2
- 10输出

具体网络架构代码如下：

```
class LeNet5:

    def __init__(self):
        self.conv1 = conv.conv((6, 1, 5, 5), stride=1, padding='SAME', bias=True,
requires_grad=True)
        self.pooling1 = pooling.Maxpooling(kernel_size=(2, 2), stride=2)
        self.BN1 = batch_normal.BN(6, moving_decay=0.9, is_train=True)
        self.relu1 = activate.Relu()

        self.conv2 = conv.conv((16, 6, 5, 5), stride=1, padding="VALID", bias=True,
requires_grad=True)
```

```
        self.pooling2 = pooling.Maxpooling(kernel_size=(2, 2), stride=2)
        self.BN2 = batch_normal.BN(16, moving_decay=0.9, is_train=True)
        self.relu2 = activate.Relu()

        self.conv3 = conv.conv((120, 16, 5, 5), stride=1, padding="VALID", bias=True,
requires_grad=True)

        self.fc4 = fc.fc(120*1*1, 84, bias=True, requires_grad=True)
        self.relu4 = activate.Relu()
        self.fc5 = fc.fc(84, 10, bias=True, requires_grad=True)

        self.softmax = loss.softmax()
```

二、各层细节

1.卷积层

按LeNet5卷积核实现即可，但for循环的效率很低，应使用img2col实现快速卷积

具体实现：

```
def img2col(self, x, filter_size_x, filter_size_y, stride):
    """
        :param x:输入的feature map形状: [N,C,H,W]
        :param filter_size_x:卷积核的尺寸x
        :param filter_size_y:卷积核的尺寸y
        :param stride:卷积步长
        :return:二维矩阵 形状: [(H-filter_size+1)/stride * (W-filter_size+1)/stride*N, C *
filter_size_x * filter_size_y]
        """

    N, C, H, W = x.shape
    output_H, output_W = (H-filter_size_x)//stride + 1, (W-filter_size_y)//stride + 1
    out_size = output_H * output_W
    x_cols = np.zeros((out_size*N, filter_size_x*filter_size_y*C))
    for i in range(0, H-filter_size_x+1, stride):
        i_start = i * output_W
        for j in range(0, W-filter_size_y+1, stride):
            temp = x[:,:, i:i+filter_size_x, j:j+filter_size_y].reshape(N,-1)
            x_cols[i_start+j::out_size, :] = temp
            return x_cols
```

2.池化层

- 使用最大池化
- `kernel size = (2,2)`
- `stride = 2`

```
class Maxpooling:
    def __init__(self, kernel_size=(2, 2), stride=2, ):
        """
        :param kernel_size:池化核的大小(kx,ky)
        :param stride: 步长
```

```
        这里有个默认的前提条件就是：kernel_size=stride
        """
        self.ksize = kernel_size
        self.stride = stride

    def forward(self, input):
        """
        :param input:feature map形状[N,C,H,W]
        :return:maxpooling后的结果[N,C,H/ksize,W/ksize]
        """
        N, C, H, W = input.shape
        out = input.reshape(N, C, H//self.stride, self.stride, W//self.stride, self.stride)
        out = out.max(axis=(3,5))
        self.mask = out.repeat(self.ksize[0], axis=2).repeat(self.ksize[1], axis=3) != input
        return out

    def backward(self, eta):
        """
        :param eta:上一层返回的梯度[N,O,H,W]
        :return:
        """
        result = eta.repeat(self.ksize[0], axis=2).repeat(self.ksize[1], axis=3)
        result[self.mask] = 0
        return result
```

3.全连接层

实现全连接即可。

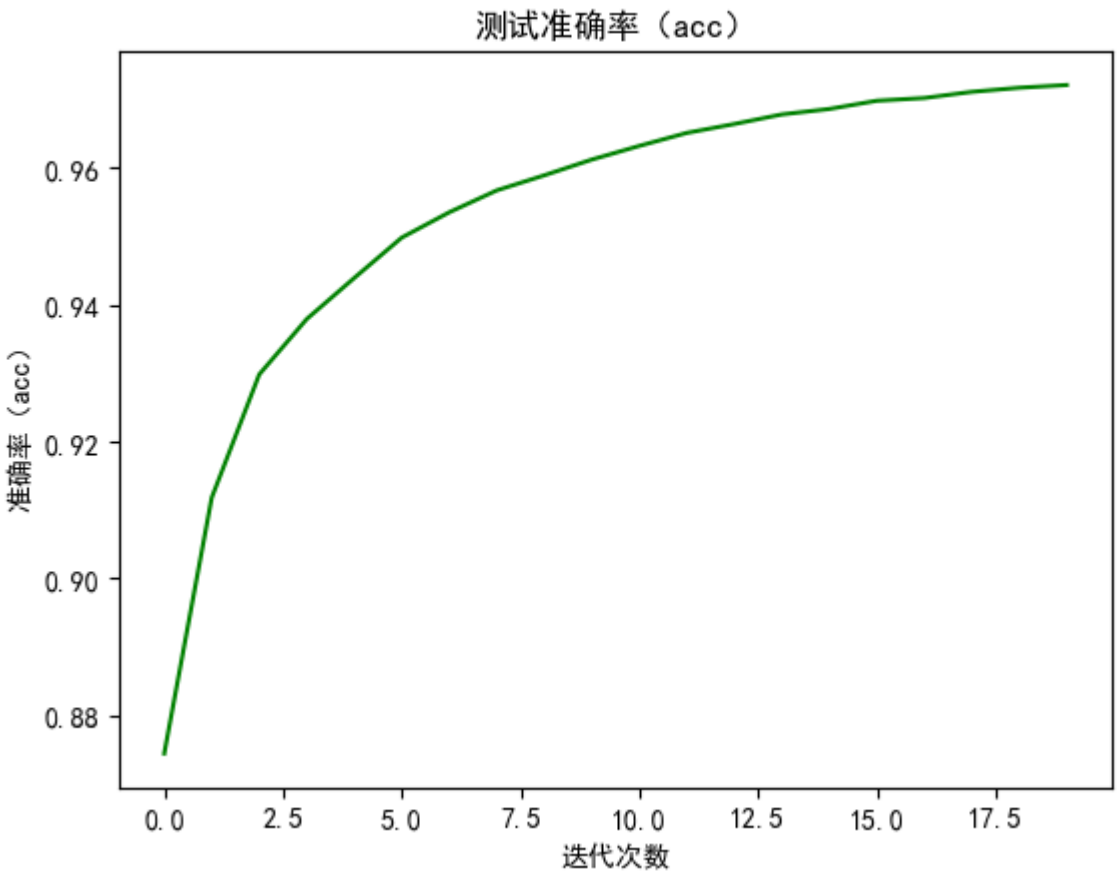
4.其他细节

- 损失函数：`softmax`损失函数
- 激活函数：`Relu()`

由于每个Epoch是按batch为单位训练的，因此额外加入BN层加快训练，防止梯度爆炸。

三、实验结果

- 环境：
  - python3.7
  - Numpy1.21.6 Matplotlib3.5.3
  - PyCharm Pro 2022.1.3
- 结果



经过20个Epoch后，准确率可达到97.1%，但训练速率很慢，每个Epoch需要5分钟以上。

- 分析

  由于按batch进行训练且数据集较大，因此训练速度较慢，但结果客观。

四、总结

通过本次实验，我学习了LeNet5的网络架构，并对卷积神经网络有了更深的理解。