## 模型介绍：

LeNet-5是Yann LeCun于1998年发布的论文《Gradient based learning applied to document-recognition》所提出的卷积神经网络。是一种用于手写体字符识别的非常高效的卷积神经网络。

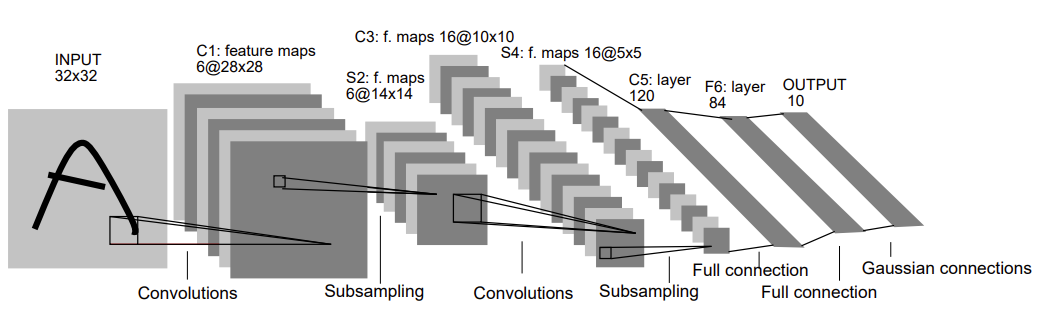


图1 LeNet-5网络结构

如图1所示，LeNet5共有七层，输入的二维图像（单通道），先经过两次卷积层到池化层，再经过全连接层，最后为输出层。每层都包含可训练参数，每个层有多个Feature Map，每个Feature Map通过一种卷积滤波器提取输入的一种特征，然后每Feature Map有多个神经元。

LeNet-5 的模型结构相对简单，它包含了两个卷积层和三个全连接层，共计7层。输入数据是一个28x28的灰度图像，输出是10个类别的概率。下面对其结构进行详细介绍：

输入层 (Input Layer)

输入层接收28x28的灰度图像。

卷积层1 (Convolutional Layer 1)

第一层卷积层包含6个卷积核，每个卷积核大小为5x5。这个卷积层的输出是6张24x24的特征图像，每个卷积核学习从原始图像中提取一种特征。

池化层1 (Pooling Layer 1)

第一层池化层使用的是平均池化，并将Sigmoid函数作为激活函数，大小为2x2。这个池化层的输出是6张12x12的特征图像，对卷积层的输出进行降维，同时增强了特征的鲁棒性。

卷积层2 (Convolutional Layer 2)

第二层卷积层包含16个卷积核，每个卷积核大小为5x5。这个卷积层的输出是16张8x8的特征图像，每个卷积核学习从前一层的特征图像中提取一种更高层次的特征。

池化层2 (Pooling Layer 2)

第二层池化层同样使用的是平均池化， Sigmoid函数作为激活函数，大小为2x2。这个池化层的输出是16张4x4的特征图像，对卷积层的输出进行降维，同时增强了特征的鲁棒性。

全连接层1 (Fully Connected Layer 1)

第一层全连接层包含120个神经元，每个神经元与前一层的所有神经元相连。这一层的输入是展开后的16张4x4的特征图像，输出是一个120维的向量。

全连接层2 (Fully Connected Layer 2)

第二层全连接层包含84个神经元，每个神经元与前一层的所有神经元相连。这一层的输入是前一层的输出120维向量，输出是一个84维的向量。

输出层 (Output Layer)

输出层是一个包含10个神经元的全连接层，每个神经元对应着一个类别。这一层的输入是前一层的输出84维向量，每个神经元的输出代表着输入图像属于对应类别的概率。

LeNet-5模型在训练过程中使用反向传播算法（backpropagation）更新网络参数，通过最小化代价函数（cost function）来提高模型的准确性。在测试时，输入一张手写数字图片，LeNet-5模型将预测其对应的数字类别。

## 代码实现：

class LeNet(nn.Module):

    def \_\_init\_\_(self):

        super(LeNet, self).\_\_init\_\_()

        self.conv1 = nn.Sequential(

            nn.Conv2d(in\_channels=1, out\_channels=6, kernel\_size=5, stride=1),

            nn.MaxPool2d(kernel\_size=2),

            nn.Sigmoid()

        )

        self.conv2 = nn.Sequential(

            nn.Conv2d(in\_channels=6, out\_channels=16, kernel\_size=5, stride=1),

            nn.MaxPool2d(kernel\_size=2),

            nn.Sigmoid()

        )

        self.fc1 = nn.Sequential(

            nn.Linear(in\_features=4 \* 4 \* 16, out\_features=120)

        )

        self.fc2 = nn.Sequential(

            nn.Linear(in\_features=120, out\_features=84)

        )

        self.fc3 = nn.Sequential(

            nn.Linear(in\_features=84, out\_features=10)

        )

    def forward(self, input):

        conv1\_output = self.conv1(input)  # [28,28,1]-->[24,24,6]-->[12,12,6]

        conv2\_output = self.conv2(conv1\_output)  # [12,12,6]-->[8,8,16]-->[4,4,16]

        conv2\_output = conv2\_output.view(-1, 4\*4\*16)  # [n,4,4,16]-->[n,4\*4\*16]

        fc1\_output = self.fc1(conv2\_output)  # [n,256]-->[n,120]

        fc2\_output=self.fc2(fc1\_output)  # [n,120]-->[n,84]

        fc3\_output = self.fc3(fc2\_output)  # [n,84]-->[n,10]

        return fc3\_output

## 模型结果：

我们使用以下配置进行训练：

损失函数：交叉熵损失函数

优化器：随机梯度下降（SGD）

学习率：0.01

批量大小：64

训练轮数：20

最终模型在测试集上的精度为：93%

最终得到训练损失曲线图2和测试集精度曲线图3。

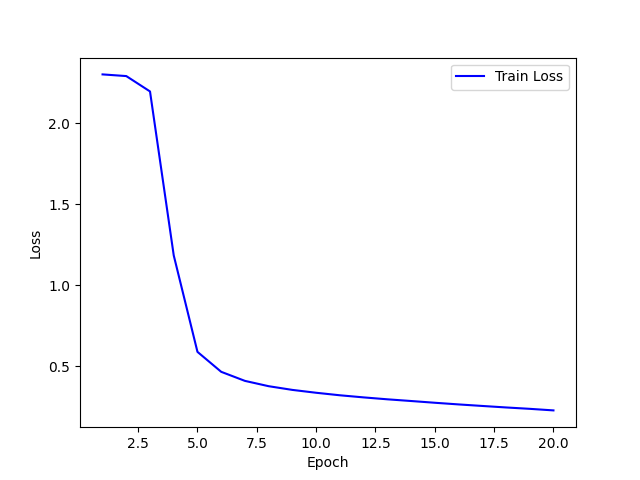


图2 损失曲线图

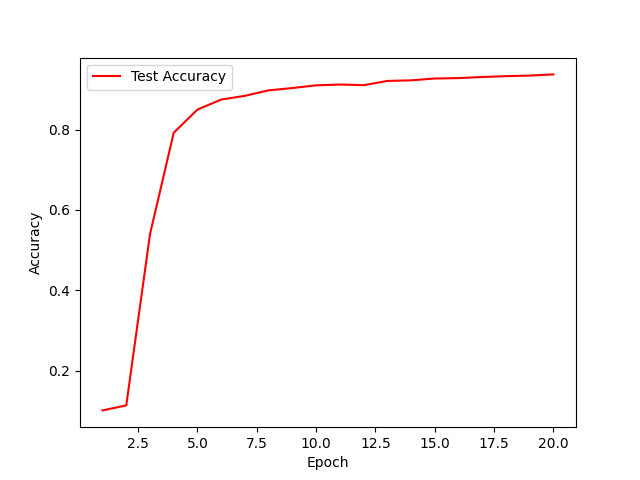


图3 测试集精度曲线图

根据实验结果，LeNet-5在MNIST数据集上的表现非常出色。

LeNet-5的设计和实现使得它在图像识别方面有很好的性能。它是一个经典的卷积神经网络模型，被广泛应用于手写数字识别、人脸识别、物体识别等领域。除了MNIST数据集外，LeNet-5还可以应用于其他数据集上，比如CIFAR-10、CIFAR-100、ImageNet等。