# 机器视觉实践报告(二)

目录

[机器视觉实践报告(二) 1](#_Toc132901010)

[一、实验目的 1](#_Toc132901011)

[二、实验原理 1](#_Toc132901012)

[2.1 卷积层 1](#_Toc132901013)

[2.2池化层 2](#_Toc132901014)

[2.3 全连接层 3](#_Toc132901015)

[2.4 激活函数 3](#_Toc132901016)

[2.5 损失函数 4](#_Toc132901017)

[2.6 优化器 5](#_Toc132901018)

[2.7 LeNet5网络 5](#_Toc132901019)

[三、数据集 6](#_Toc132901020)

[四、程序代码 7](#_Toc132901021)

[4.1 LeNet5模型构建 7](#_Toc132901022)

[4.2 读取数据集 7](#_Toc132901023)

[4.3 定义损失函数和优化器 8](#_Toc132901024)

[4.4 模型的训练 9](#_Toc132901025)

[4.5 模型的测试 10](#_Toc132901026)

[五、实验结果 10](#_Toc132901027)

[六、实验分析和总结 13](#_Toc132901028)

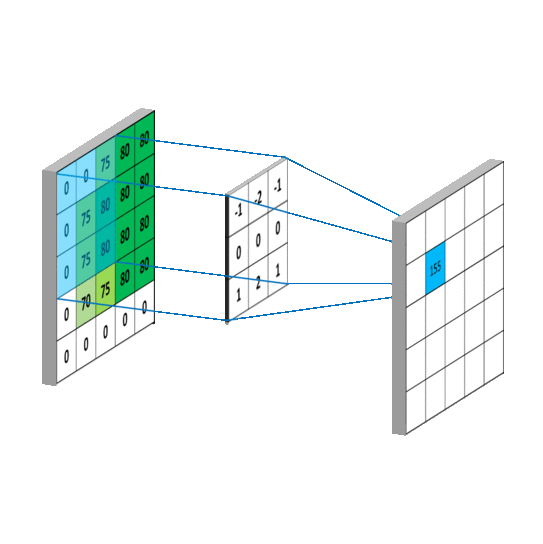
## 一、实验目的

* 熟悉卷积神经网络的基本结构，包括卷积层，池化层，激活函数及最后的全连接层等。
* 学习经典手写数字识别网络 LeNet 及其他经典特征提取网络。
* 熟悉 Pytorch深度学习框架。
* 在 MNIST数据集上完成图像分类任务，使训练集或测试集的准确率达到 98%以上。

## 二、实验原理

主要包括卷积层，池化层，全连接层，激活函数，损失函数，优化器和LeNet5网络7 个部分。

### 2.1 卷积层



在深度学习中，卷积层是一种用于处理图像、音频、视频等二维或三维数据的核心组件。卷积层的主要作用是提取数据的空间结构特征，这些特征可以被用来识别、分类或定位图像中的物体。

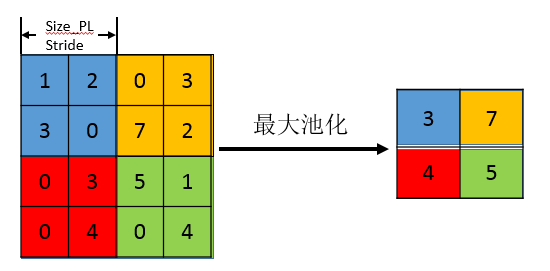
卷积层是由若干个卷积核组成的，每个卷积核都是一个二维或三维的小矩阵。卷积层将卷积核应用于输入数据的每个位置，生成一个新的特征图。卷积核可以看作是一组可学习的参数，它们被用来检测输入数据中的某种特征，如边缘、纹理、颜色等。

卷积操作的基本原理是将卷积核和输入数据做逐元素的乘法，然后将结果相加。这个过程可以看作是将卷积核对输入数据的一部分进行“扫描”，并生成一个新的特征。重复这个过程，可以生成一个特征图，其中每个像素都对应了输入数据的一个局部区域。

在卷积层中，通常会对特征图进行池化操作，以降低特征图的维度，并减少参数数量。常见的池化操作包括最大池化和平均池化，它们分别选择局部区域中的最大值和平均值作为输出。

卷积层是深度学习中的核心组件之一，它的设计和优化对于模型的性能和训练效率有着重要的影响。随着深度学习技术的不断发展，新的卷积操作和网络结构也在不断涌现，例如深度可分离卷积、可变形卷积、残差连接等，这些技术可以进一步提高卷积神经网络的性能。

### 2.2池化层



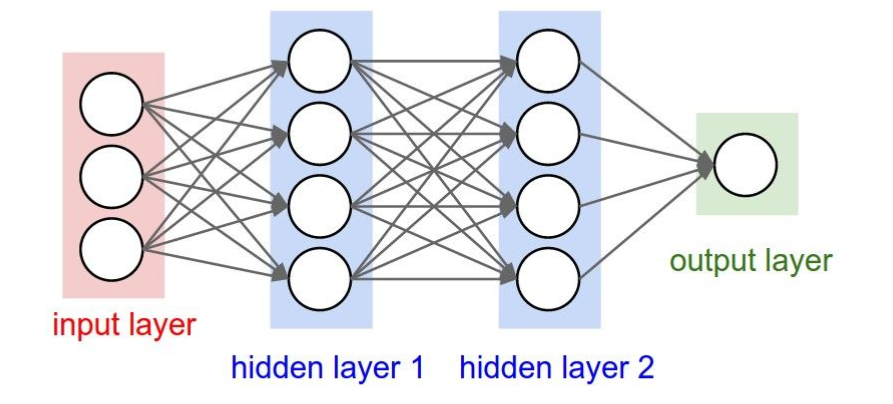
池化层是卷积神经网络中的一种重要层，用于对卷积层输出的特征图进行下采样。其主要作用是在保留特征信息的前提下，减小特征图的维度，从而减少网络参数和计算量，避免过拟合。

常见的池化操作包括最大池化和平均池化。最大池化在局部区域中选择最大值作为输出，而平均池化则选择局部区域中的平均值作为输出。在池化操作中，通常可以设置池化窗口大小、步幅、填充等参数，以控制输出的特征图的大小和形状。

池化层的优点包括：

* 减少参数数量和计算量：池化层可以将特征图的大小减小到原来的几分之一，从而大大减少了模型参数和计算量。
* 提高模型的鲁棒性：池化操作可以减小输入数据的微小变化对模型输出的影响，从而提高模型的鲁棒性和泛化能力。
* 增强特征的平移不变性：池化操作可以将输入数据的局部区域映射到一个池化输出值，从而增强了特征的平移不变性，使模型更具有泛化能力。
* 压缩数据：通过对特征图进行下采样，可以压缩输入数据的表示，使其更易于存储和处理。

### 2.3 全连接层



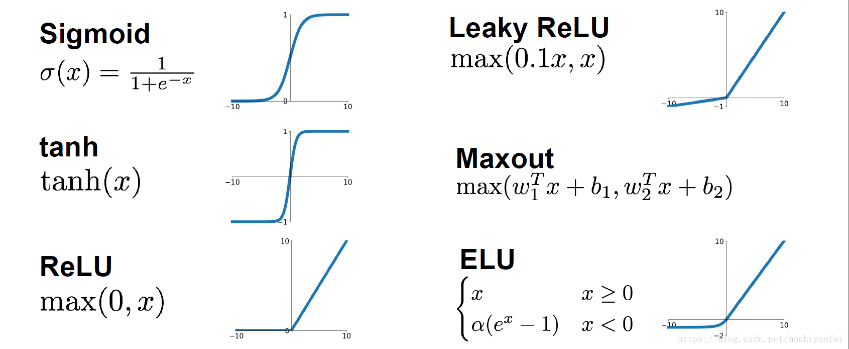
全连接层是卷积神经网络中的一种常见的层，也被称为密集连接层。全连接层将上一层的所有神经元与当前层的所有神经元相连接，形成一个全连接的权重矩阵。该权重矩阵包含了网络的所有参数，可以对输入数据进行高维度的线性变换。

在卷积神经网络中，全连接层通常作为卷积层的输出层，用于将高维的特征向量映射到目标类别的概率分布。在训练过程中，全连接层的参数通常通过反向传播算法进行更新。

全连接层的主要优点是具有较强的表达能力，能够对输入数据进行高维度的线性变换，并学习到复杂的特征表示。它可以通过大量的数据进行训练，以提高分类和回归任务的准确性。

然而，全连接层的缺点也显而易见。首先，全连接层的参数数量随着输入数据的维度增加而快速增加，会导致模型参数过多，容易过拟合。其次，全连接层忽略了输入数据的空间结构信息，因此可能丢失一些重要的局部特征。因此，在一些特定的任务中，如图像分类、目标检测、图像分割等，通常需要结合卷积层等其他层来构建更加有效的神经网络模型。

### 2.4 激活函数



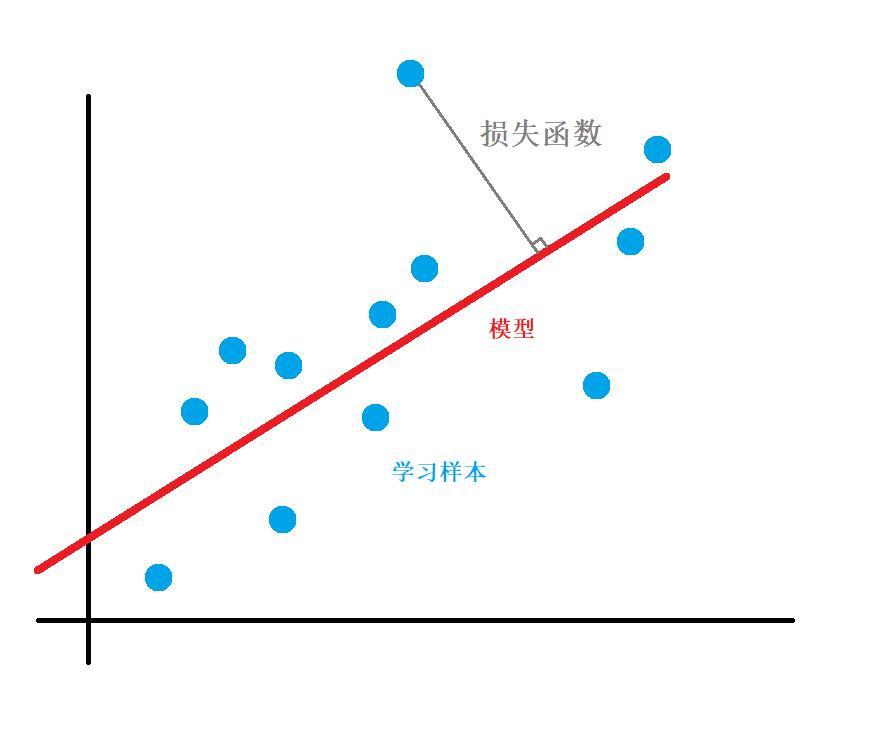
激活函数（Activation Function）是神经网络中的一种非线性函数，通常被添加到每个神经元的输出上，以增加网络的表达能力和非线性特性。激活函数将神经元的输入加权和转换成一个输出值，并确定是否应该激活神经元并向下一层传递信息。

常见的激活函数包括：

* Sigmoid函数：Sigmoid函数是一种常用的平滑激活函数，它可以将输入值映射到一个0到1之间的范围内，具有平滑的非线性特性。然而，由于其梯度在较大或较小的输入值处非常小，可能导致梯度消失问题。
* ReLU函数：ReLU函数是一种简单有效的激活函数，它可以在输入为正数时输出输入值，否则输出0。它具有快速计算速度和稀疏激活特性，但容易出现“死神经元”问题。
* LeakyReLU函数：LeakyReLU函数是ReLU函数的一种改进，它在输入为负数时输出一个小的非零值，以解决ReLU函数中的“死神经元”问题。
* Tanh函数：Tanh函数是一种常用的双曲正切激活函数，它可以将输入值映射到-1到1之间的范围内，具有平滑的非线性特性，但也存在梯度消失问题。
* Softmax函数：Softmax函数是一种常用的多元分类激活函数，它可以将输入向量映射为概率分布向量，用于多元分类问题的输出层。

选择合适的激活函数对于神经网络的性能和准确性具有重要影响。通常需要根据不同的任务和数据特点来选择合适的激活函数，或者结合多种不同的激活函数构建更加复杂的神经网络模型。

### 2.5 损失函数



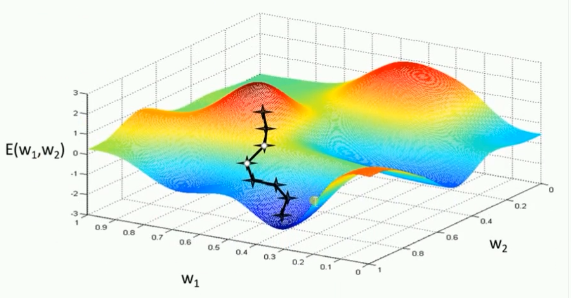
损失函数（Loss Function）是神经网络中用来衡量模型预测结果和实际标签之间的差异的一种函数。损失函数的值越小，表示模型的预测结果越接近真实值，因此神经网络的训练过程就是通过最小化损失函数来学习网络参数。

不同的任务和模型需要选择不同的损失函数。以下是几种常见的损失函数：

* 均方误差损失（Mean Squared Error）：均方误差损失是回归问题中最常用的损失函数，它用来衡量模型预测值与真实值之间的差异的平方和。
* 交叉熵损失（Cross Entropy Loss）：交叉熵损失是分类问题中常用的损失函数，它衡量了模型预测值和真实值之间的差异，越小表示模型预测的类别越接近真实标签。
* 对数似然损失（Log-Likelihood Loss）：对数似然损失也是分类问题中常用的损失函数，它与交叉熵损失非常相似，不同之处在于对数似然损失使用的是类别的对数概率，因此更加稳定。

选择合适的损失函数对于神经网络的性能和准确性具有重要影响。通常需要根据不同的任务和数据特点来选择合适的损失函数，或者结合多种不同的损失函数构建更加复杂的神经网络模型。

### 2.6 优化器

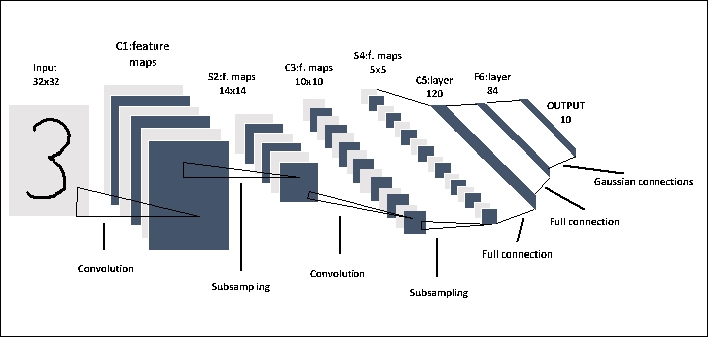


在卷积神经网络中，优化器是一种用来自适应调整神经网络参数以最小化损失函数的算法。常见的优化器包括梯度下降法、动量优化、Adam等。

* 梯度下降法：梯度下降法是最基本的优化器之一，它通过计算每个参数的梯度，并将参数沿着梯度的反方向更新，从而最小化损失函数。梯度下降法存在局部最优解的问题，因此可能需要使用一些改进的优化器。
* 动量优化：动量优化可以加速梯度下降法的收敛，同时也能够减小梯度方向的震荡，从而更稳定地更新参数。它在更新参数时，不仅使用当前的梯度，还考虑了之前的梯度信息。
* Adam：Adam是一种自适应学习率的优化器，它可以在训练过程中自动调整学习率，并同时使用梯度的一阶矩和二阶矩信息来更新参数。Adam通常能够更快地收敛，并且比其他优化器更容易使用。
* Adagrad：Adagrad是一种自适应学习率的优化器，它将学习率适应每个参数的历史梯度大小，从而使得参数更新更加准确。Adagrad在处理稀疏数据时效果比较好，但在处理稠密数据时可能会出现学习率下降过快的问题。
* RMSProp：RMSProp是一种自适应学习率的优化器，它通过对梯度的平方做指数加权平均来自适应地调整每个参数的学习率。RMSProp在处理稀疏数据时效果较好，但在处理非平稳数据时可能会出现过早收敛的问题。

需要注意的是，优化器的选择应该基于具体的任务和数据集来确定。在实际应用中，需要尝试不同的优化器，并根据验证集的结果选择最合适的优化器。同时，还可以考虑使用学习率衰减、权重衰减等技术来进一步提高模型的性能。

### 2.7 LeNet5网络



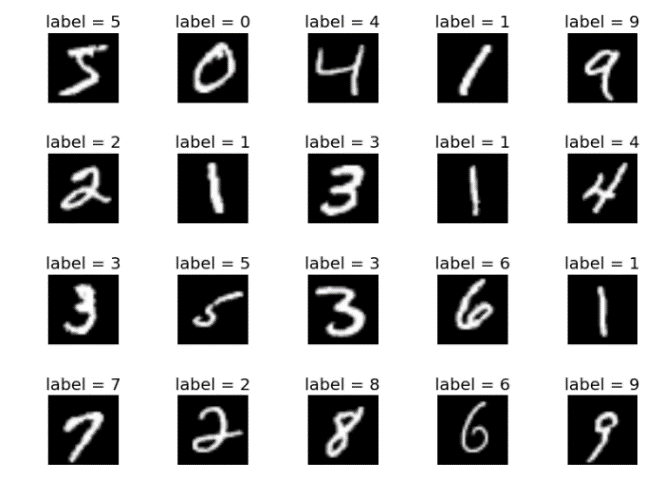
LeNet5是一种经典的卷积神经网络，是由Yann LeCun在1998年提出的，被广泛应用于手写数字识别任务中。它是深度学习领域的里程碑之一，为后来的卷积神经网络奠定了基础。

LeNet5网络包含7个层，其中包括3个卷积层、2个池化层和2个全连接层。其结构如下：

* 卷积层：输入图像经过6个卷积核（filter）的卷积，每个卷积核大小为5×5，得到6个特征图。
* 池化层：对每个特征图进行2×2的最大池化操作，将图像尺寸减小为原来的一半。
* 卷积层：输入6个特征图，经过16个卷积核的卷积，每个卷积核大小为5×5，得到16个特征图。
* 池化层：对每个特征图进行2×2的最大池化操作，将图像尺寸减小为原来的一半。
* 卷积层：输入16个特征图，经过120个卷积核的卷积，每个卷积核大小为5×5，得到120个特征图。
* 全连接层：将120个特征图展开成一个向量，经过全连接层，得到84个输出。
* 全连接层：输入84个输出，经过全连接层，得到最终的10个输出，表示0-9十个数字的概率分布。

LeNet5采用sigmoid激活函数，并使用反向传播算法进行训练。在手写数字识别等任务上，LeNet5取得了不错的效果，为后来的深度学习技术奠定了基础。

## 三、数据集



MNIST（Modified National Institute of Standards and Technology）数据集是一个经典的手写数字识别数据集，它由美国国家标准与技术研究所（NIST）于1980年代末期创建。

该数据集包含60000个训练样本和10000个测试样本，每个样本都是28x28的灰度图像，表示手写数字0-9。每个图像都被标记为其对应的数字，从0到9共10个类别。图像和标签的数据是分开存储的，以便于进行数据预处理和数据增强。

MNIST数据集被广泛用于测试新的机器学习和深度学习算法。它已经成为深度学习领域的标准测试集之一，许多研究论文都使用了该数据集进行实验。由于该数据集比较小，容易理解和处理，因此成为了许多学习深度学习的人入门的第一个数据集。

## 四、程序代码

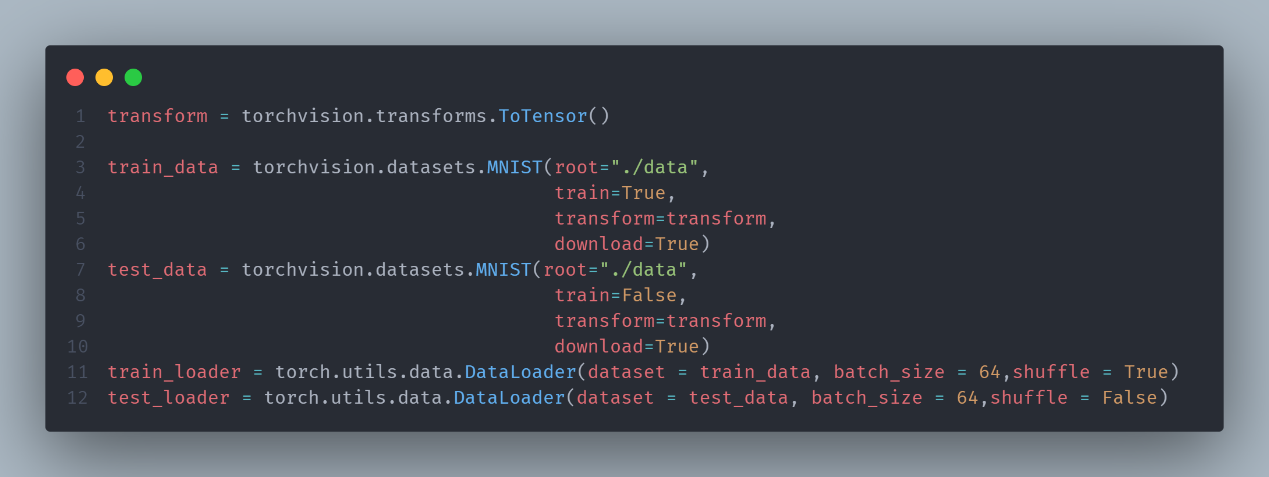
### 4.1 LeNet5模型构建

根据LeNet原始论文当中的网络结构使用Pytorch搭建相应的网络模型。



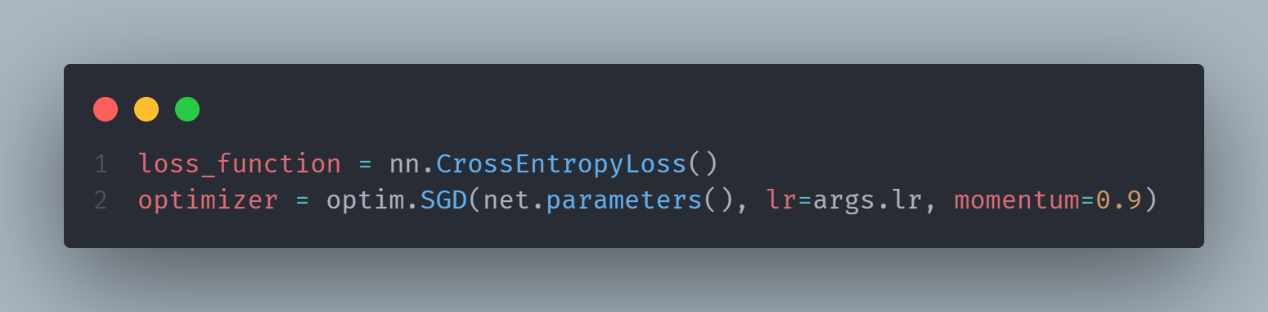
### 4.2 读取数据集

使用torchvision当中的工具下载MNIST数据集，并构建用于训练和测试的train\_loder和test\_loader。

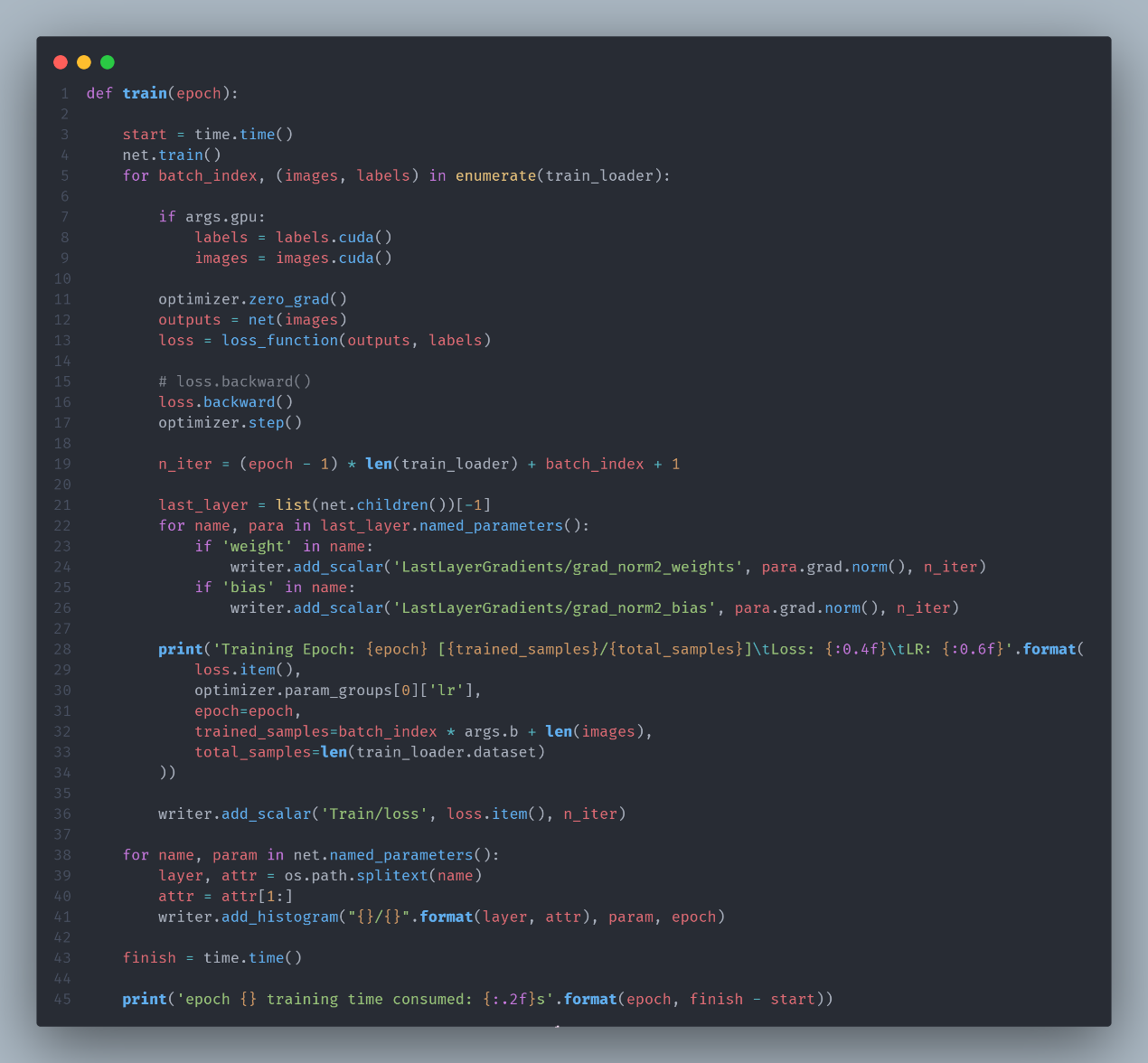


### 4.3 定义损失函数和优化器

使用交叉熵损失和SGD优化器。



### 4.4 模型的训练



在模型的训练过程当中，我们使用TensorBoard工具记录相关的损失和梯度信息，用于后续的可视化的分析。

### 4.5 模型的测试

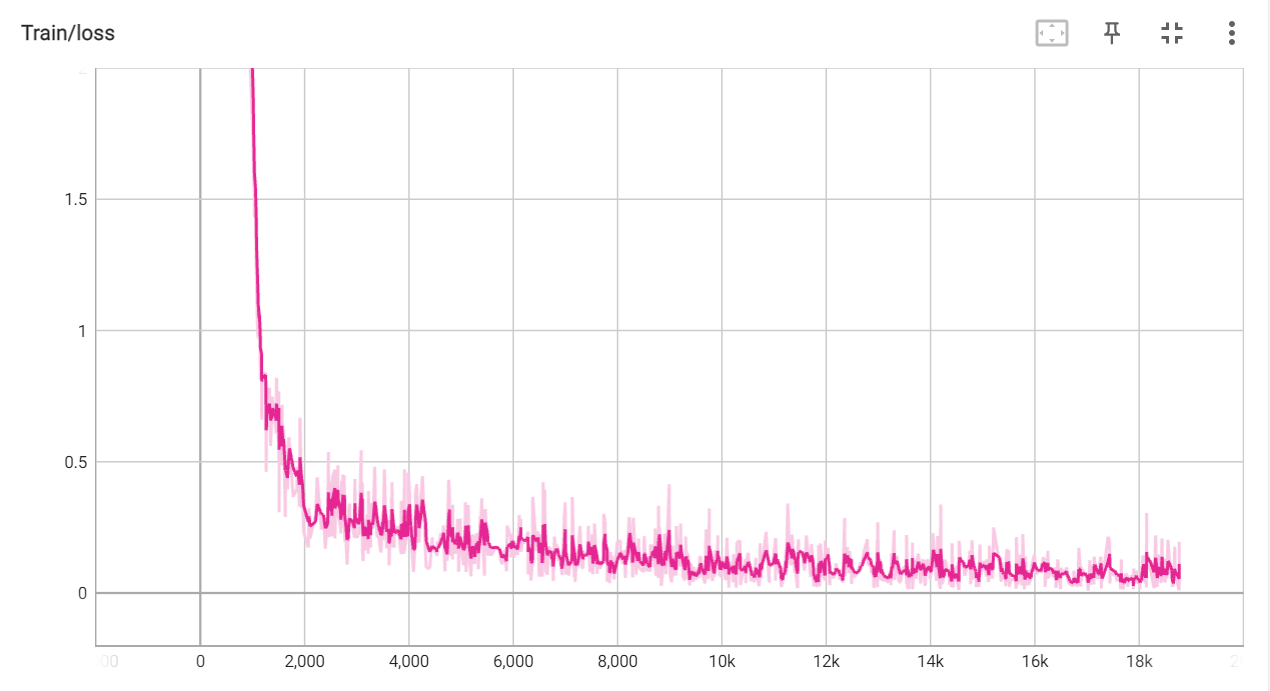


模型的测试过程和模型的训练过程比较类似，最大的区别就是在测试的时候模型是不需要计算梯度的，也无需进行梯度的反向传播。

## 五、实验结果

在给出的原始LeNet网络上，我们在MNIST数据集中训练了20个Epoch，结果如下。





可见模型已经趋近于收敛，并且达到了97.45%的准确率。

当我们将网络当中的AvgPool2d更换为MaxPool2d之后，并且对模型的参数加以一定的正则化约束，如设置优化器的weight\_delay=5e-4。发现模型的表现优于原始的LeNet结构。

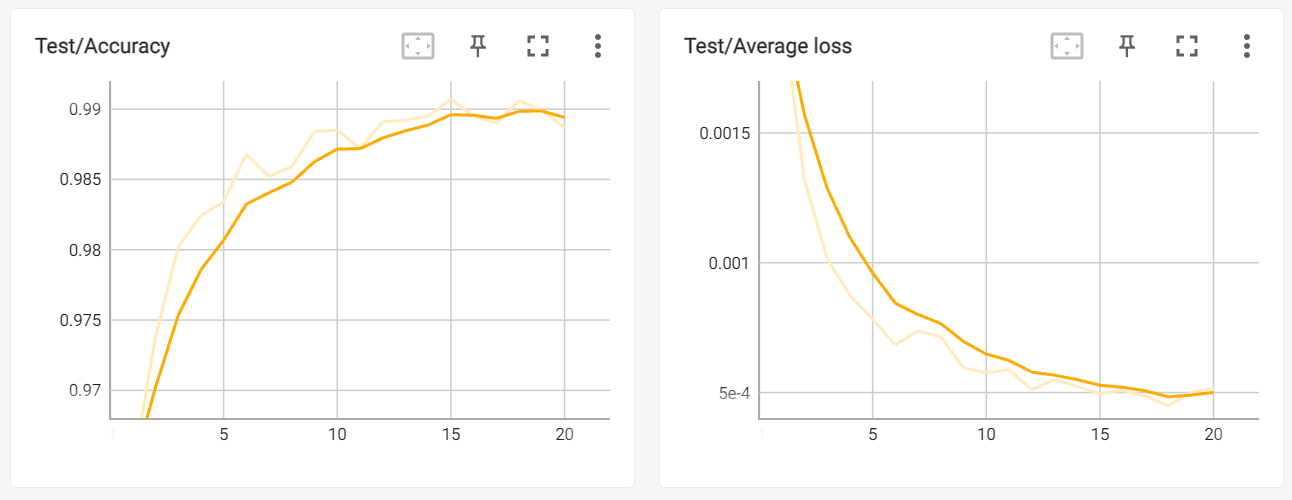


该模型最终可以达到98.55%的准确率，并且可以看出模型收敛的速度也比原始结构要快很多。

为了进一步提供模型的正确率，我们加深了原始的LeNet网络。我们多增加了两层卷积层，并且对于每一个卷积层的数据后，我们都添加了BatchNorm2d用于更好的训练。模型结构代码如下所示。

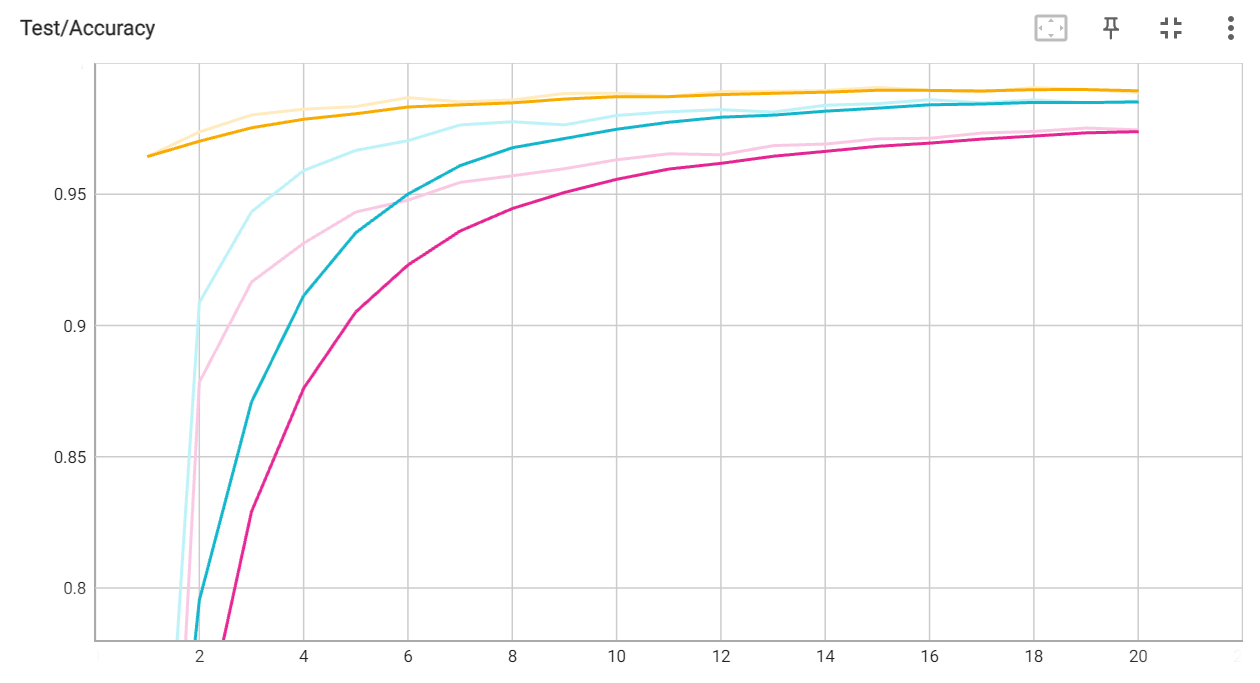


该网络在MNIST数据集中训练20个epoch的结果如下所示。



网络的收敛速度极快，在前几个epoch就可以达到95%以上的正确率，最后的准确率也达到了98.90%。

下面是这三个网络在测试集上精度的对比。



其中红色代表原始的LeNet网络。蓝色在红色的基础上改用MaxPooling，使用ReLU激活函数，对网络加以一定的正则化等措施。黄色则是在蓝色的基础上加深了两层卷积层，并且使用BatchNorm。

## 六、实验分析和总结

LeNet5是一个经典的卷积神经网络，最早应用于手写数字识别任务，而MNIST数据集正是一个手写数字识别的经典数据集。在这里，我们将使用LeNet5模型在MNIST数据集上进行实验，评估其在该数据集上的性能表现。

在实验当中我们发现LeNet5也有一些不足之处，包括：

* 缺乏深度：LeNet-5只有5层，相对于现代的深度卷积神经网络来说，它的深度非常浅。
* 局限于特定的输入尺寸：LeNet-5最初是设计用于处理32x32像素的图像，因此它对于输入尺寸的变化比较敏感，如果输入图像尺寸过大或过小，可能需要重新调整网络架构或训练数据。
* 参数较少：尽管这一点在LeNet-5的设计目的中是一个优点，但它也意味着它的表达能力相对较弱，难以处理更复杂的任务。
* 没有使用Batch Normalization等现代技术：Batch Normalization等现代技术可以加速网络训练，并改善网络的表现，而这些技术在LeNet-5被设计出来之前还未被提出。
* 激活函数不够灵活：LeNet-5使用的激活函数为sigmoid函数，在现代卷积神经网络中更常用的ReLU函数可以更好地解决梯度消失问题。

在实验当中我们对LeNet的这些不足都做了相应的改进，从实验结果也可以看出来提升的效果还是蛮明显的。

总的来说，LeNet5是一个简单而有效的卷积神经网络，能够在MNIST数据集上达到非常好的性能表现。该实验结果表明，卷积神经网络是一个有效的手写数字识别方法，并为我们在实际应用中使用更复杂的卷积神经网络奠定了基础。