Lab1 - Part2 卷积神经网络

贺劲洁 18307130370

Lab1 - Part2 卷积神经网络

一、代码基本结构

1. 文件组成
2. 主要函数

二、设计实验改进网络并论证

1. 网络结构

一层全连接层

卷积核数量

2. 优化器

简要分析

实验对比

3. 动量

实验对比

4. 归一化 Batch Normalization：

5. 数据增强

5. 学习率

三、对CNN的理解

网络结构

卷积层

原理分析

Padding

1. 作用：
2. 如何决定padding大小？

Stride 卷积步幅

卷积层优点

池化层

最大池化法 Max pooling

平均池化法

全连接层

四、对网络设计的理解

一、代码基本结构

1. 文件组成

- `cnnNetwork.py` cnn网络结构定义
- `cnnTraining.py` 网络训练结果

`loadDirData( dir_path)`

载入目标文件夹下的图片集，图片进行预处理（可选择数据增强），文件夹名为标签名

2. 主要函数

`single_train(), compare_train()`

单网络训练函数、多网络同步训练对比函数

- 网络参数配置和自定义训练选项
- 训练模型定义（可同时定义多网络同步训练）
- 加载训练集数据和验证集数据
- 逐epoch逐batch进行训练，每 `validate_freq` 个 `batch` 对测试集和验证集进行一次测试打印
- 绘制对比图（可选）
- 保存最佳的训练参数

`reload_net(data_path)`

从指定路径恢复网络模型

`test(test_dir, net_path)`

用指定路径恢复的网络对指定文件夹下的图片进行测试，打印测试结果

二、设计实验改进网络并论证

1. 网络结构

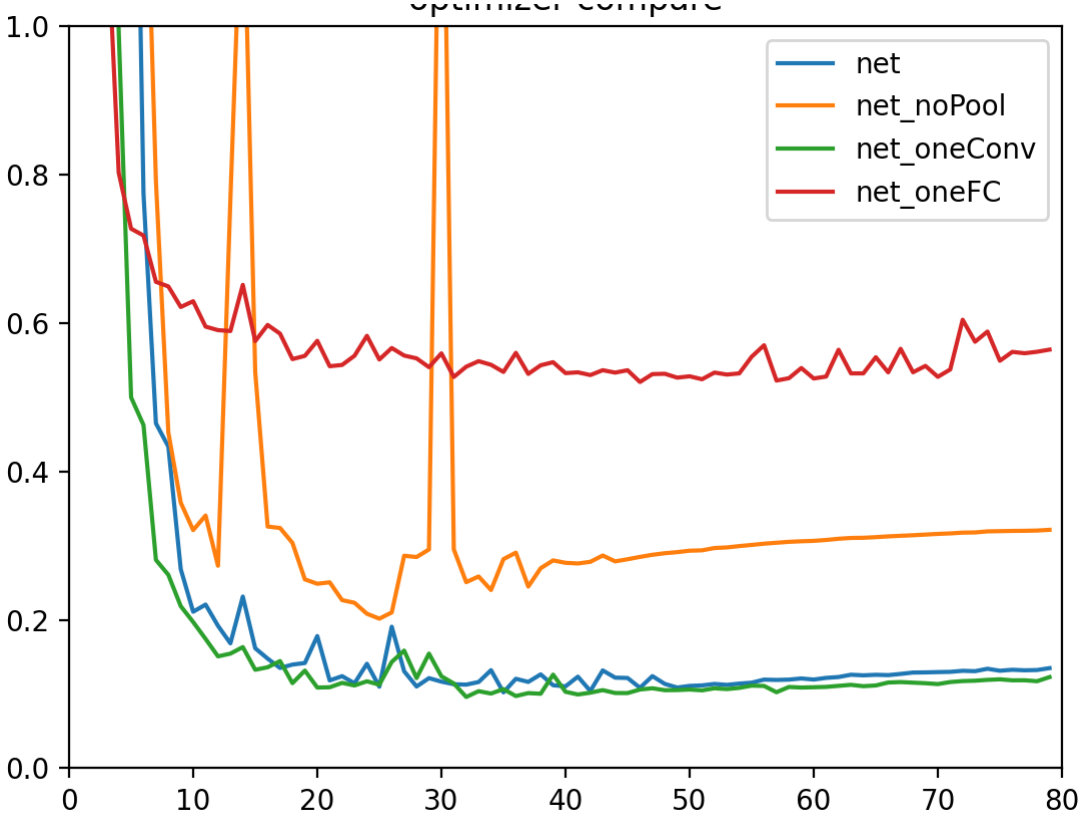
试验：对比不使用pool层、只用一个卷积层、只用一个全连接层，可看出只用一个全连接层和不用pool层的收敛效果

`net` ： 2* (Cov+Pool) + 3* FC

`net_noPool`: 2*Cov + 3 * FC

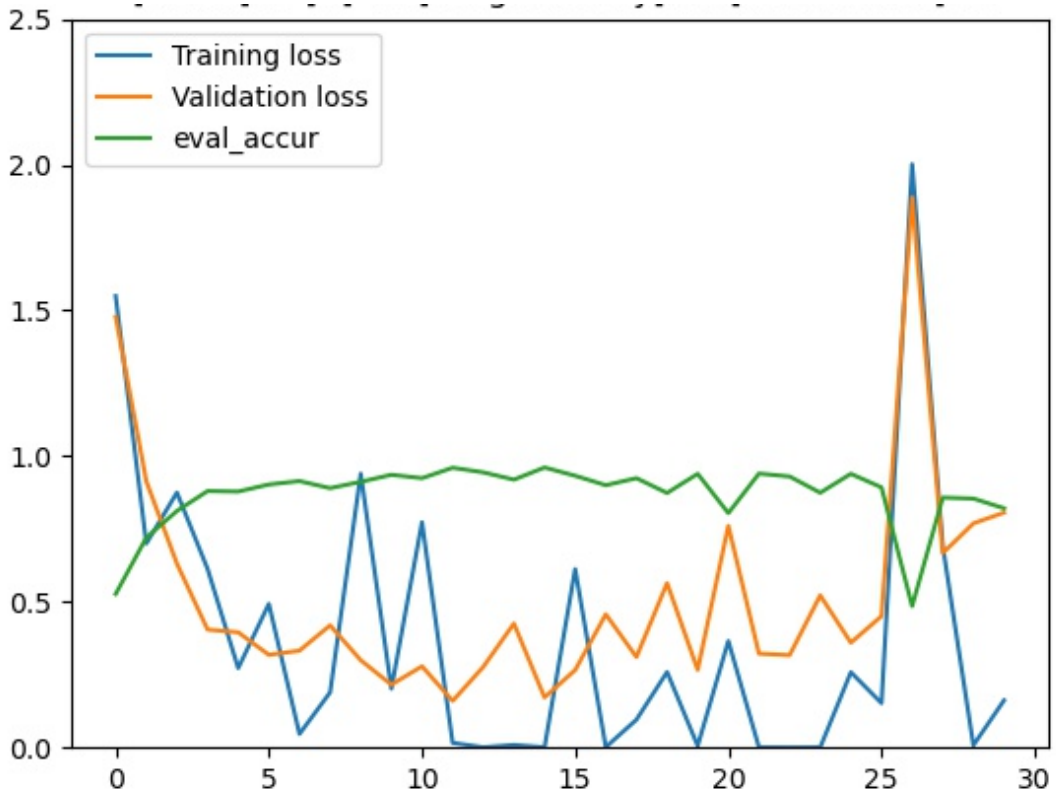
`net_oneConv`: 1*(Conv+Pool) + 3*FC

`net_oneFC`: 2* (Cov+Pool) + 1* FC



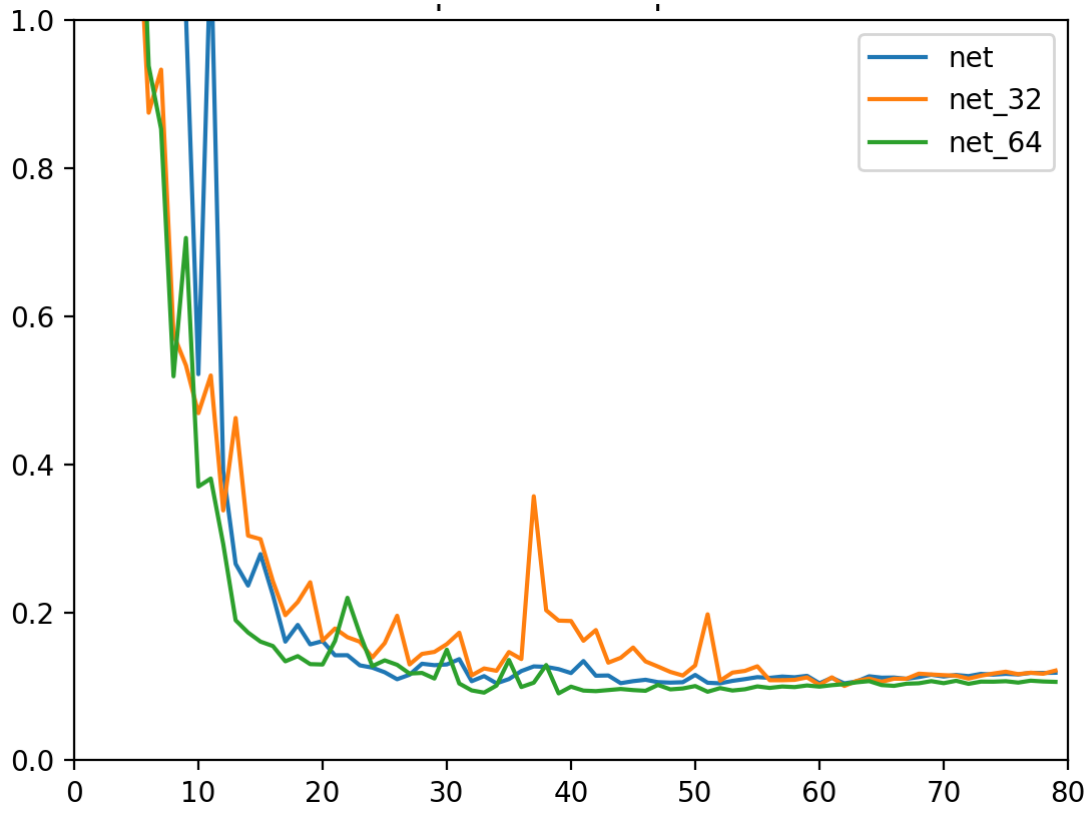
一层全连接层

只使用一层全连接层收敛速度很慢，准确率到达88-89%后无法提高，且收敛波动非常大



卷积核数量

对比了第二层卷积核数量分别为 16、32、64的情况



在该任务下没有明显差别

2. 优化器

torch中提供了许多优化器，本实验选取了常见的SGD、SGD+momentum，与自适应优化算法Adagrad、RMSProp、Adam、Adamax进行了实验对比

简要分析

SGD：随机梯度下降算法，最常见的优化，可加入动量

自适应优化算法

Adagrad

原理：累计历史梯度的平方和，对学习率进行自适应分配，历史梯度较大时，学习率较小，历史梯度较小时，学习率较大

缺点：训练后期，由于累计项非常大可能导致学习率很低

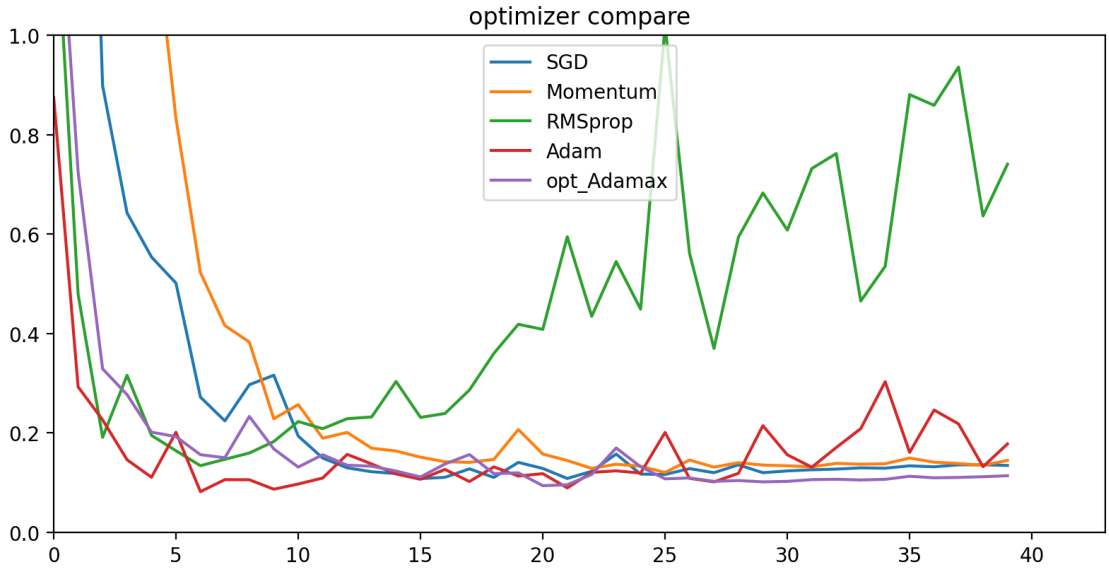
RMSProp

相比Adagrad，只累计最近一段范围内的梯度平方和，解决了Adagrad后期训练难以更新的问题

Adam：综合了RMSProp和动量

Adamax：Adam的优化，对学习率上限提供了更简单的范围

实验对比



尝试了 SGD、MOmentum、RMSprop、Adam和Adamax几种优化函数，观察其loss，发现`RMSprop`的loss 后期发散、有过拟合迹象，`Adam`也有相同趋势，相比之下，SGD、Momentum和Adamax收敛效果较好，Adamax收敛最快、收敛效果最好

3. 动量

现象：使用mini-batch时，loss有时候会出现激增

分析：由于mini-batch是分批次计算梯度进行调整，有可能朝错误方向突进

作用：减小训练调整的波动减小噪声的干扰

原理：类似增加一个物理中的“惯性”，在每一次调整的时候把之前的更新方向也考虑进来，这样能减缓大幅度的更新

使用指数加权平均的方式，引入一个“速度”的概念，当许多连续的梯度指向相同的方向时，更新最大

$$v_i = \beta \cdot v_{i-1} - rate * \delta w$$

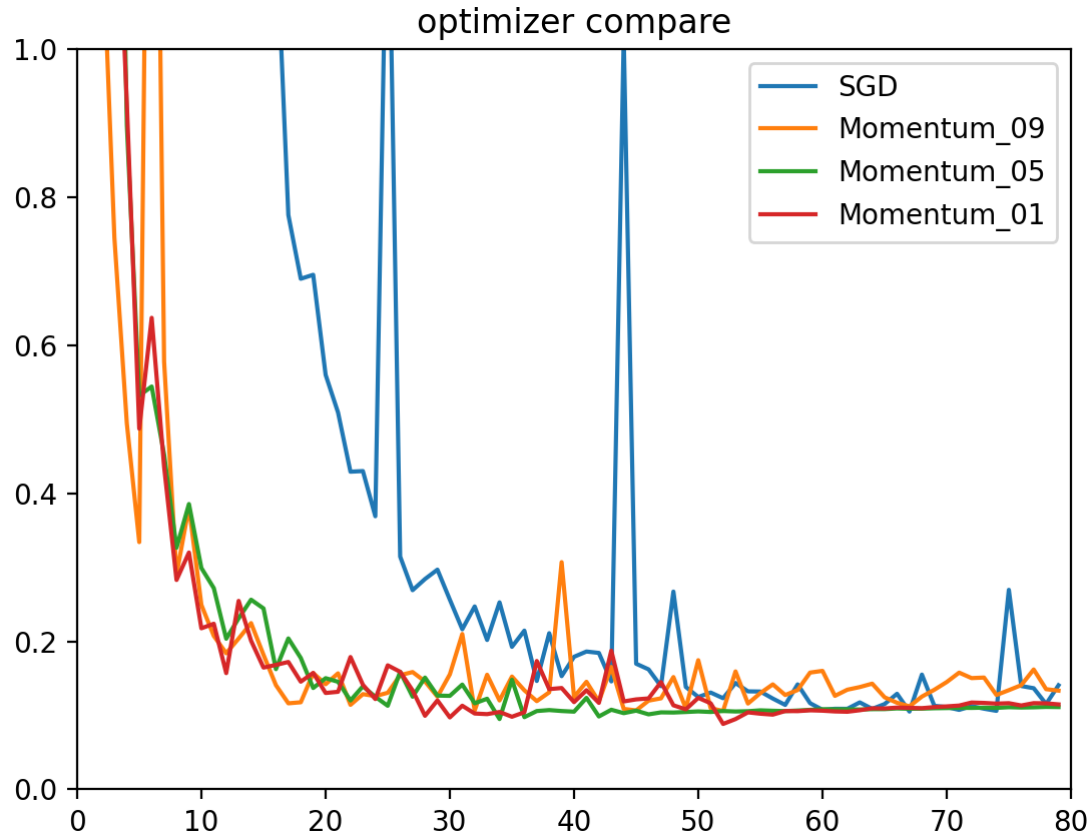
$$new\_w = w + v$$

如果之前的 v 和这次的负梯度方向相同，下降的幅度就会加大，从而加速收敛

否则下降的幅度减小，减少大幅度的错误更新

实验对比

【不使用动量、动量系数分别为0.9,0.5,0.1】

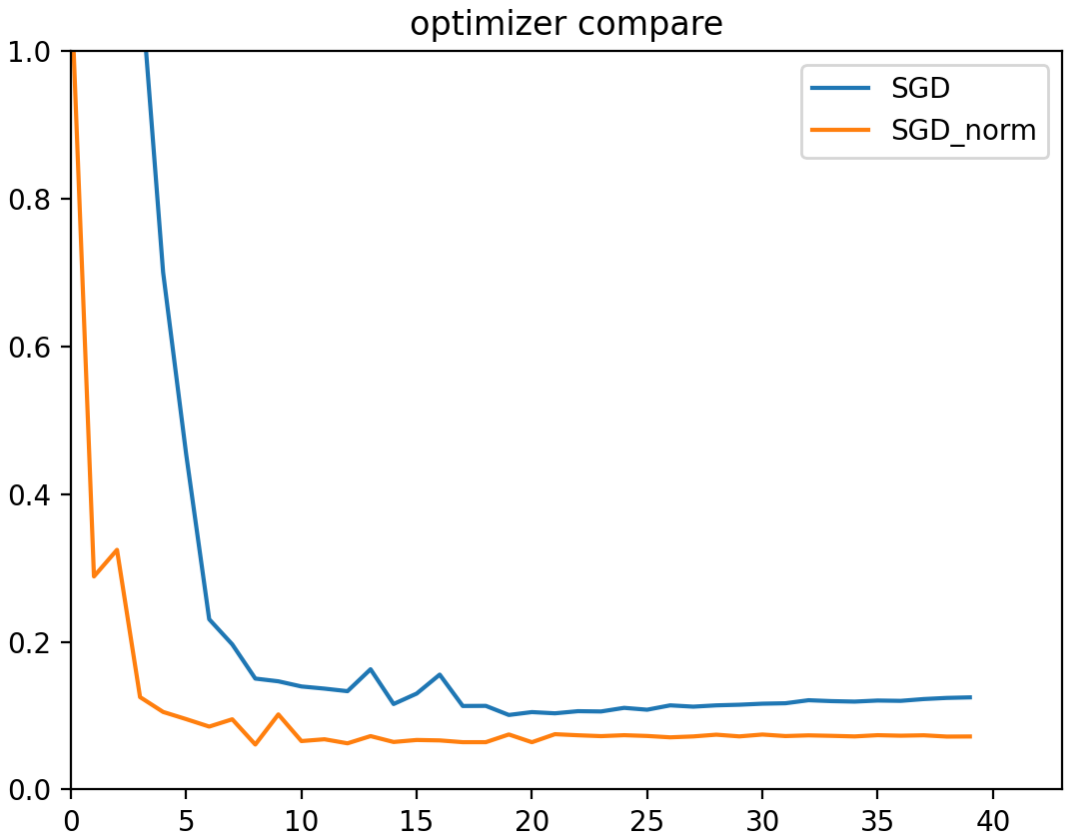


1. 相同学习率下，使用动量前期收敛速度明显优于不使用动量的模型
2. 不使用动量时，有时会出现误差的骤增，使用动量可以在梯度改变方向和整体收敛方向相差较大时减小对参数的影响

3. 加入动量后，学习可适当调低
4.

4. 归一化 Batch Normalization：

原理及作用：对数据进行归一化，使其均值为0， 方差为1，从而缓解训练中的梯度消失或爆炸现象，加快模型的训练速度



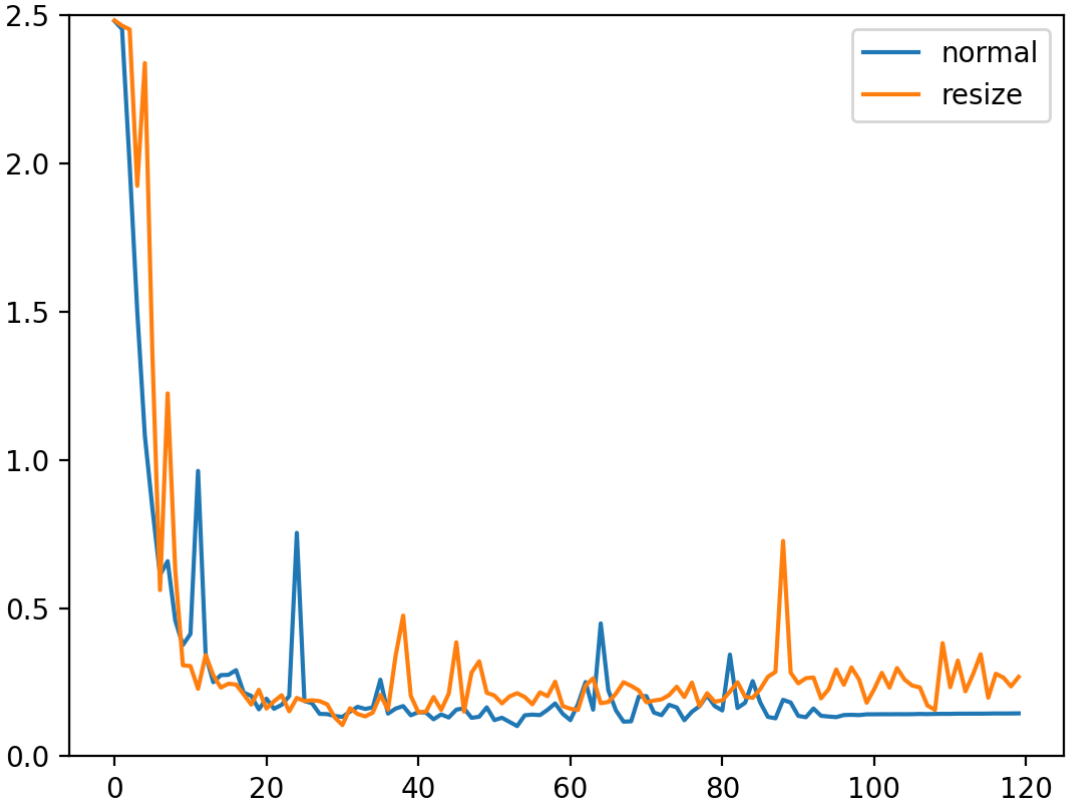
实验看出，其他条件相同的情况下，进行了归一化的模型收敛更快、效果更好

5. 数据增强

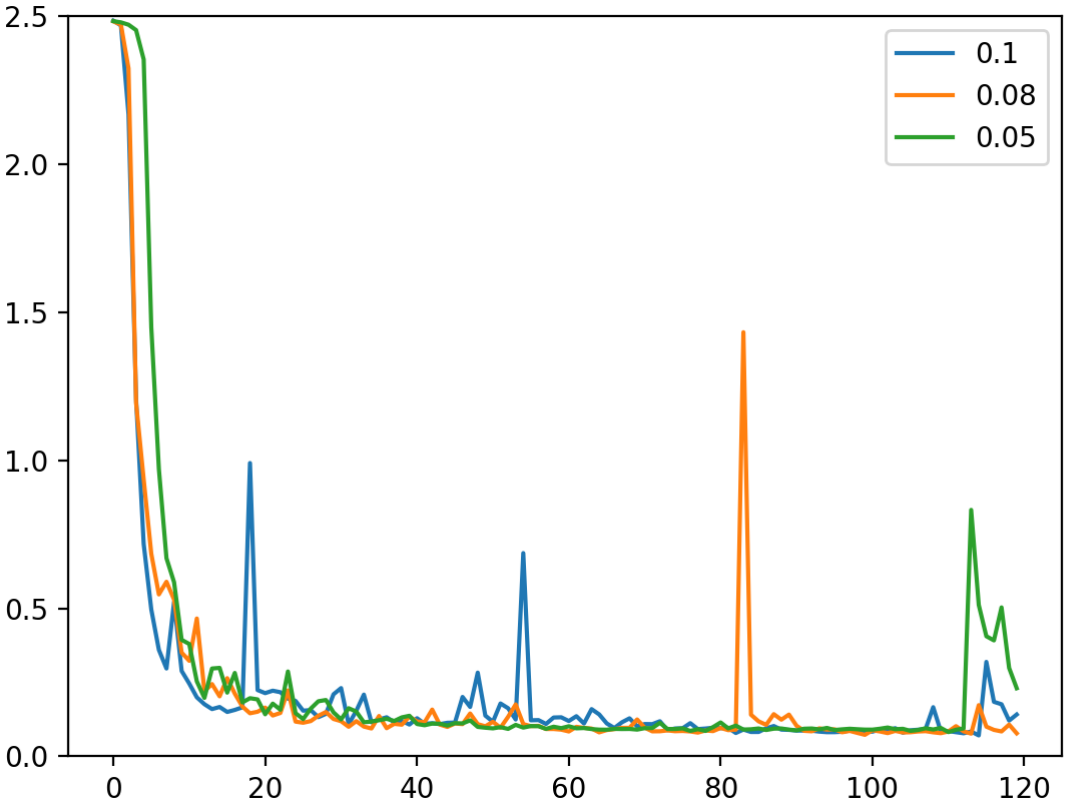
在加载数据时将图片大小 resize 为 32*32

数据增强

对这个任务好像没有太大作用，反而出现过拟合迹象



5. 学习率



收敛效果没有太大区别

三、对CNN的理解

网络结构

- 卷积层（Conv）
- 池化层 Pooling
- 全连接层 （FC）

通常，一个或多个卷积层后跟着一个池化层，过全连接层再过softmax

卷积层

原理分析

一个filter一个bias

$$
\begin{aligned}
&f^l = filtersize \\
&p^l = padding \\
&s^l = stride \\
&input: n_h^{l-1} \times n_w^{l-1} \times n_c^{l-1} \\
&output: n_h^l \times n_w^l \times n_c^l \\
&n_h^l = \frac{n_h^{l-1} + 2p^l - f^l}{s}_l + 1 \\
&n_w \text{同理} \\
&filter\text{大小}：f^l \times f^l \times n_c^{l-1} \\
&weights\text{大小}：filter\text{大小} \times filter\text{数量} \\
&bias\text{大小}：n_c^l
\end{aligned}
$$

通常，随着层数的增加，信道数增加 ， height, width 减小

Padding

在输入的周围填充一层像素

1. 作用：

1. 防止图像每经过一层都缩小，越卷越小
2. 防止角落边缘位置的信息使用较少导致的信息丢失

输出维度： $n + 2p - f + 1$

2. 如何决定padding大小?

- Valid：没有填充
- Same：填充后，输出大小 = 输入大小 p = (f - 1)/2

注：卷积核维度一般是奇数，防止不对称填充； 同时，奇数维度会有一个中心点

Stride 卷积步幅

过滤器移动的步长

输出维度：$\frac{(n+2p-f)}{s}+1$， 如果商不是整数：向下取整

注：输入和卷积核的通道数必须相同

卷积层优点

- 参数共享

  理解：统一特征检测器可以用于图像的多个区域的特征提取
- 稀疏连接

  输出单元只依赖于卷积核大小的输入单元值，其他像素不会对该输出产生影响；

  相比全连接层，参数数量很少（过滤器大小 * 通道数），计算量小、效率高

池化层

池化层没有需要学习的参数，很少用padding

f:filter size

s: stride

n' = (n - f / s) + 1

最大池化法 Max pooling

（常用）取每个区域的最大元素值

作用：只要在任何一个象限提取到某个特征，就会保留在最大池化中

平均池化法

（不常用）取平均值，一般用于很大的神经网络

全连接层

将卷积层结果展开为一维向量后，同传统神经网络

四、对网络设计的理解

不同训练任务所适合的网络结构和优化可能不同，超参有无数种组合，在参考经典网络结构的设计思路的基础上，还需要大量的实践经验积累

同时，对网络的设计和优化应该是建立在对其实现原理的深刻理解之上的，只停留在表面的理解会增加试错成本，且网络结构的参数相互关联，不深入理解很难实现更优的网络。