



山东大学  
SHANDONG UNIVERSITY

SHANDONG UNIVERSITY

---

## 机器学习第 4 次实验报告-CNN 图像识别

---

姓名: 谢子洋

学院: 网络空间安全学院 (研究院)

专业: 网络空间安全

班级: 网安三班

学号: 202100460116

2023 年 12 月 24 日

# 目录

<b>1 实验目的</b>	<b>2</b>
<b>2 实验环境</b>	<b>2</b>
<b>3 实验方法</b>	<b>2</b>
3.1 Batch . . . . .	2
3.2 数据增强 (Data Augmentation) . . . . .	2
3.3 交叉验证 (Cross Validation) . . . . .	2
3.4 深度学习中的数据标准化. . . . .	3
3.5 CNN 卷积神经网络 . . . . .	3
<b>4 结果分析与评估</b>	<b>4</b>
4.1 CNN 模型设计与训练 . . . . .	4
4.2 CNN 模型结果分析 . . . . .	6
4.3 相同参数量下深度对模型的影响 . . . . .	7
4.4 数据增强对模型训练影响 . . . . .	8
4.5 数据正规化对模型训练影响 . . . . .	8
<b>5 结论</b>	<b>9</b>
<b>参考文献</b>	<b>10</b>

# 1 实验目的

建立 CNN 神经网络进行图像识别.

# 2 实验环境

训练平台:

表 1: 训练平台信息

CPU	INTEL CORE i5-12400
Memory	DDR4 3200Hz 32GB
Gpu	RTX 4060Ti 8GB
OS	Ubuntu22.04LTS
Language	Python3.9

# 3 实验方法

## 3.1 Batch

模型训练中往往不直接使用全部的数据进行训练, 而是将训练数据分成多个 **Batch**, 然后对每个 **Batch** 进行训练. 将所有的 **Batch** 都遍历完成后, 即完成一个 **epoch**. **Batch** 的大小决定了每次迭代更新参数的样本数量, 也对模型收敛速度和效果有一定影响。

使用 **Train** 数据训练神经网络时, 每经过一次 **Epoch** 都要对 **Train** 数据进行 **shuffle**, 获得新的 **Epoch**. 而使用 **Validation** 数据集进行验证则无需进行 **shuffle**, 因为 **Validation** 数据实质上不参与模型训练.

## 3.2 数据增强 (Data Augmentation)

神经网络的训练需要足够的数据. 为了获得更多的数据, 我们可以对现有的数据集进行微小的改变充当新的数据集.

数据增强技术 (data augmentation), 主要是在训练数据上增加微小的扰动或者变化, 一方面可以增加训练数据, 从而提升模型的泛化能力, 另一方面可以增加噪声数据, 从而增强模型的鲁棒性. 主要的数据增强方法有: 翻转变换, 随机修剪、色彩抖动、平移变换、尺度变换、对比度变换、噪声扰动、旋转变换等。

## 3.3 交叉验证 (Cross Validation)

先将训练数据集划分为训练数据和验证数据, 通过验证数据判断模型是否过拟合. 在模型确定一个较优秀的参数后, 再使用全部训练数据进行训练.

### 3.4 深度学习中的数据标准化.

在深度学习中, 除了对原始数据进行标准化外, 还可以对神经网络的中间值进行标准化.

因为神经网络中大部分都是矩阵运算, 一个向量经过矩阵运算后值会越来越大, 为了网络的稳定性, 我们可以使用标准化及时把值拉回正态分布.

### 3.5 CNN 卷积神经网络

#### 1. CNN 的组成.

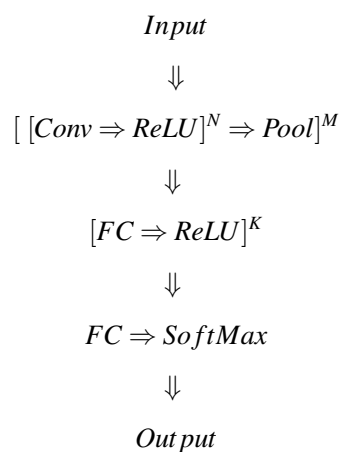
CNN 卷积神经网络主要由卷积层, 池化层和全连接层组成. 卷积神经网络中输出层的上游通常是全连接层, 因此其结构和工作原理与传统前馈神经网络中的输出层相同. 对于图像分类问题, 输出层可使用归一化指数函数 (softmax function) 输出分类标签.

- A. 卷积层 Conv layer.
- B. 池化层 Pooling layer: Max pooling.
- C. 激励层 Activation layer: ReLU, Leaky ReLU, Maxout.
- D. 全连接层 FC layer.
- E. SoftMax 层.

#### 2. CNN 模型结构.

CNN 卷积神经网络一般由多个卷积层和池化层的组合和一个全连接层组成. 卷积层和池化层是 CNN 的核心, 它们的作用是提取图片的特征. 全连接层的作用是将这些特征组合起来, 最终输出分类结果. 对于图像分类问题, CNN 需要额外添加 SoftMax 层, 将输出结果转化为概率分布.

CNN 结构一般如下:



#### 3. Loss.

针对多分类问题我们使用交叉熵作为模型 Loss.

$$L = \sum_{i=1}^n \hat{y}_i \ln y_i$$

#### 4. 反向传播.

神经网络使用反向传播进行梯度下降. 反向传播通过导数链式法则计算损失函数对各参数的梯度, 并根据梯度进行参数的更新. 在神经网络的训练过程中, 前向传播和反向传播交替进行.

## 4 结果分析与评估

### 4.1 CNN 模型设计与训练

我们基于卷积神经网络搭建模型.

#### 1. 模型架构.

CNN 卷积神经网络主要由卷积层, 激励函数层, 池化层, 全连接层组成, 本次实验中额外添加了数据标准化层以及针对多分类问题的 SoftMax 层.

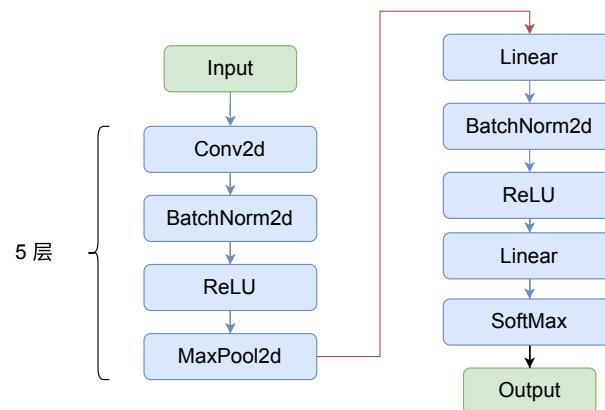


图 1: CNN 架构

实验中设计的神经网络以 (卷积 + 标准化 + 激励 + 池化) 为一个基本复合层, 并重复 5 层. 随后又连接一线性部分, 即 (线性层 + 标准化层 + 激励层 + 线性层 + SoftMax). 设计 CNN 的整体结构如图1所示.

#### 2. 模型参数.

(a) 对训练数据使用数据增强技术, 提高模型训练效果.

```

1 train_transform = transforms.Compose([
2     transforms.ToPILImage(),
3     transforms.RandomHorizontalFlip(), #依据概率p对PIL图片进行水平翻转, p默认0.5
4     transforms.RandomRotation(15),    #按照degree随机旋转一定角度
5     transforms.ToTensor(),
6 ])
  
```

(b) 标准化.

在卷积层和激励层之间添加数据标准化层BatchNorm2d.

```
1 nn.Conv2d(chan[0],chan[1],ConKernS[0], stride [0],padding[0]),
2 nn.BatchNorm2d(chan[1]),
3 nn.ReLU(),
4 nn.MaxPool2d(PoolKernS[0]), #one layer
```

(c) 设置 epoch 数.

原代码中设置训练 epoch 数为 30, 但该数值下训练不充分, 模型准确度不高. 因此我们将训练 epoch 数提高到 160, 并记录模型训练过程.

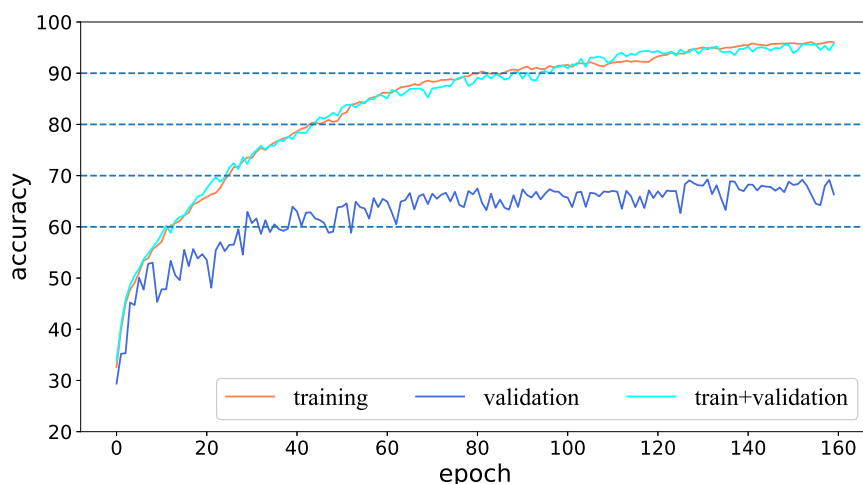


图 2: 提高 epoch 数

观察图2可知, 提高 epoch 数使得模型在训练集和验证集上的准确度都有所提升. 但随着训练次数的增多, 验证集上模型的准确率提升越来越不明显, 接近 70% 就已经到达极限. 这说明随着训练次数的不断提高, 模型逐渐过拟合. 因此我们在训练模型时要及时结束, 可以使用 early stopping 避免过拟合. 我们选取 epoch 数为 40.

3. 模型参数量.

使用 torchsummary 库中的summary 函数统计模型信息.

观察图3可知, 实验中设计的模型其参数有 863499 个.

```

1
2 -----
3
4 Layer (type)                Output Shape                Param #
5 -----
6 Conv2d-1                    [128, 128, 128, 128]      3,584
7 BatchNorm2d-2               [128, 128, 128, 128]      256
8 ReLU-3                      [128, 128, 128, 128]      0
9 MaxPool2d-4                 [128, 128, 64, 64]        0
10 Conv2d-5                    [128, 128, 64, 64]        147,584
11 BatchNorm2d-6               [128, 128, 64, 64]        256
12 ReLU-7                      [128, 128, 64, 64]        0
13 MaxPool2d-8                 [128, 128, 32, 32]        0
14 Conv2d-9                    [128, 128, 30, 30]        147,584
15 BatchNorm2d-10              [128, 128, 30, 30]        256
16 ReLU-11                     [128, 128, 30, 30]        0
17 MaxPool2d-12                [128, 128, 15, 15]        0
18 Conv2d-13                   [128, 128, 13, 13]        147,584
19 BatchNorm2d-14              [128, 128, 13, 13]        256
20 ReLU-15                     [128, 128, 13, 13]        0
21 MaxPool2d-16                [128, 128, 6, 6]          0
22 Conv2d-17                   [128, 128, 4, 4]          147,584
23 BatchNorm2d-18              [128, 128, 4, 4]          256
24 ReLU-19                     [128, 128, 4, 4]          0
25 MaxPool2d-20                [128, 128, 2, 2]          0
26 Linear-21                   [128, 512]                262,656
27 ReLU-22                     [128, 512]                0
28 Linear-23                   [128, 11]                 5,643
29 Softmax-24                  [128, 11]                 0
30 -----
31 Total params: 863,499
32 Trainable params: 863,499
33 Non-trainable params: 0
34 -----
35 Input size (MB): 24.00
36 Forward/backward pass size (MB): 8761.02
37 Params size (MB): 3.29
38 Estimated Total Size (MB): 8788.32
39 -----

```

图 3: 参数统计

## 4.2 CNN 模型结果分析

调整模型参数后最终得到结果如下:

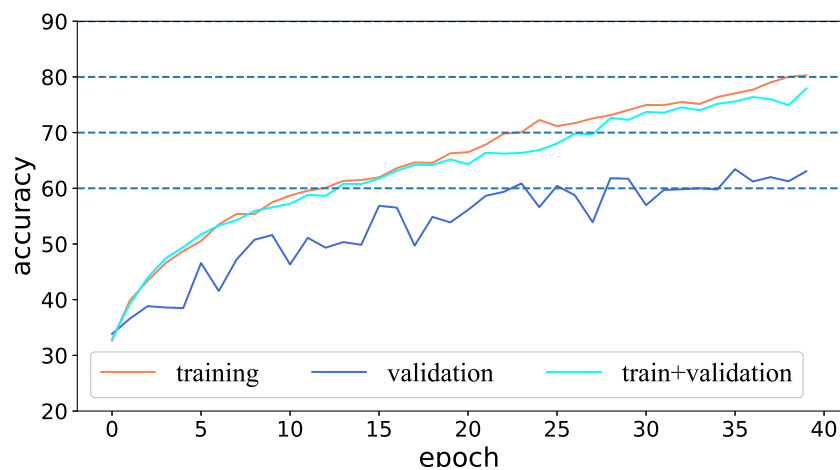
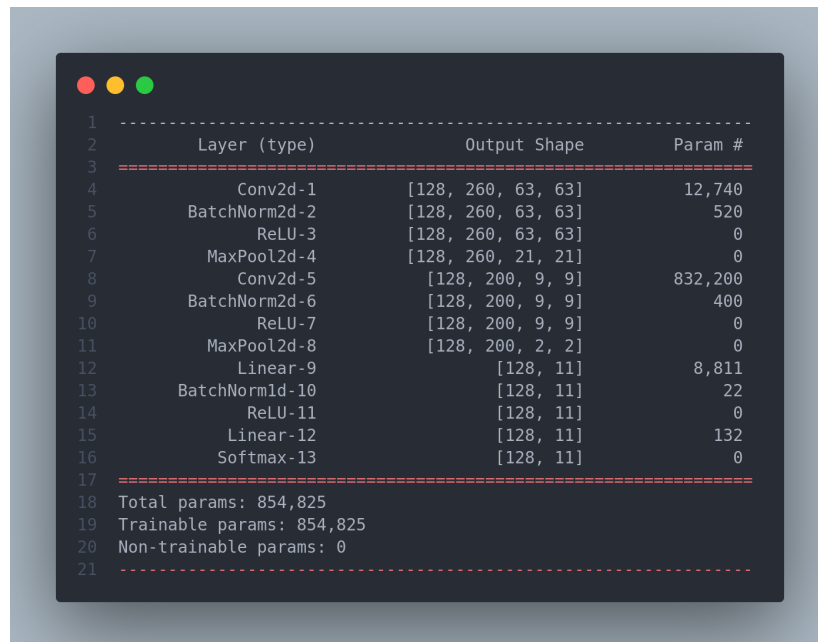


图 4: 最后结果

### 4.3 相同参数量下深度对模型的影响

在上文中我们已经搭建了一个结果较优的 CNN 图像识别模型. 为探讨相同深度下模型深度对识别结果准确率的影响, 我们将模型进行修改, 使得其层数减半但是参数量不变. 新的模型参数如图5所示.



```
1 -----
2 Layer (type)          Output Shape          Param #
3 -----
4 Conv2d-1              [128, 260, 63, 63]    12,740
5 BatchNorm2d-2         [128, 260, 63, 63]    520
6 ReLU-3                [128, 260, 63, 63]    0
7 MaxPool2d-4           [128, 260, 21, 21]    0
8 Conv2d-5              [128, 200, 9, 9]      832,200
9 BatchNorm2d-6         [128, 200, 9, 9]      400
10 ReLU-7               [128, 200, 9, 9]      0
11 MaxPool2d-8           [128, 200, 2, 2]      0
12 Linear-9             [128, 11]             8,811
13 BatchNorm1d-10       [128, 11]             22
14 ReLU-11              [128, 11]             0
15 Linear-12            [128, 11]             132
16 Softmax-13           [128, 11]             0
17 -----
18 Total params: 854,825
19 Trainable params: 854,825
20 Non-trainable params: 0
21 -----
```

图 5: 模型层数减半后参数

修改后模型的 (卷积 + 池化) 组合的层数降为 2 层, 参数总量为 854825, 与原模型的参数总量 863499 大致相同.

训练该模型并记录训练过程中正确率的变化. 比较原模型和层数减半模型的识别效果.

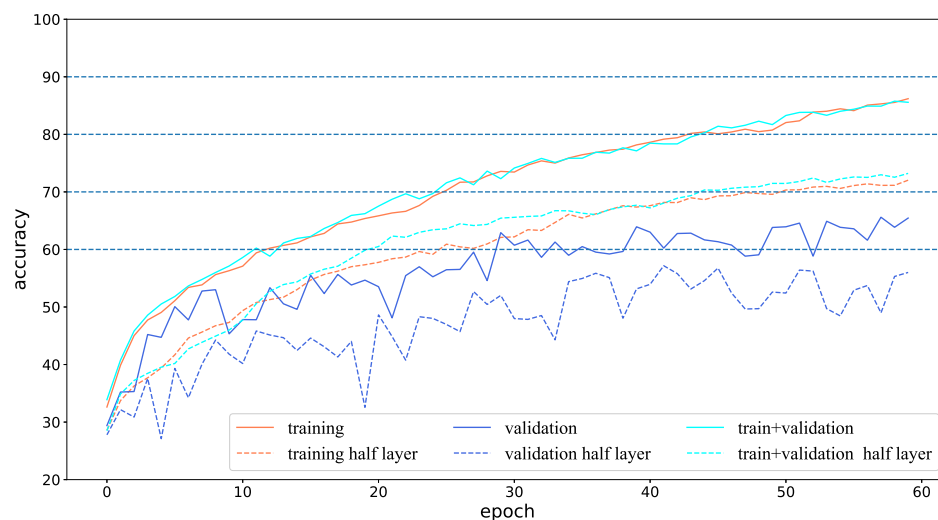


图 6: 层数减半



由图6可知, 在相同参数量下, 深度减半后模型识别率下降, 深度更高的模型识别效果更好.

#### 4.4 数据增强对模型训练影响

训练模型时分别采用原始数据和增强后的数据进行训练, 观察模型在训练集和验证集上的准确率.

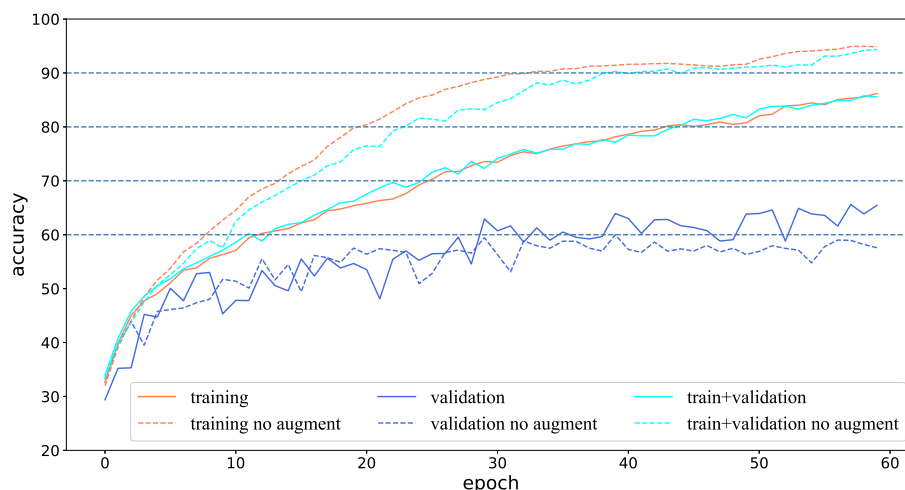


图 7: 对比数据增强效果

由7可知, 若不对数据进行数据增强, 则模型虽然会在训练数据集上准确率提升, 但在验证数据集上的准确率反而会下降, 这都是模型过拟合的特征. 数据增强后模型在训练集上准确率大幅下降, 但在验证集上的准确率却有所提升, 说明模型的泛用性通过数据增强有所提升.

使用数据增强技术能增加数据集中相关数据量, 增加训练集的多样性, 从而提升模型性能. 在使用有限的数据进行模型训练时, 数据增强操作十分有效.

#### 4.5 数据正规化对模型训练影响

训练模型时不对神经网络中间值进行标准化, 与原模型的训练过程进行对比.

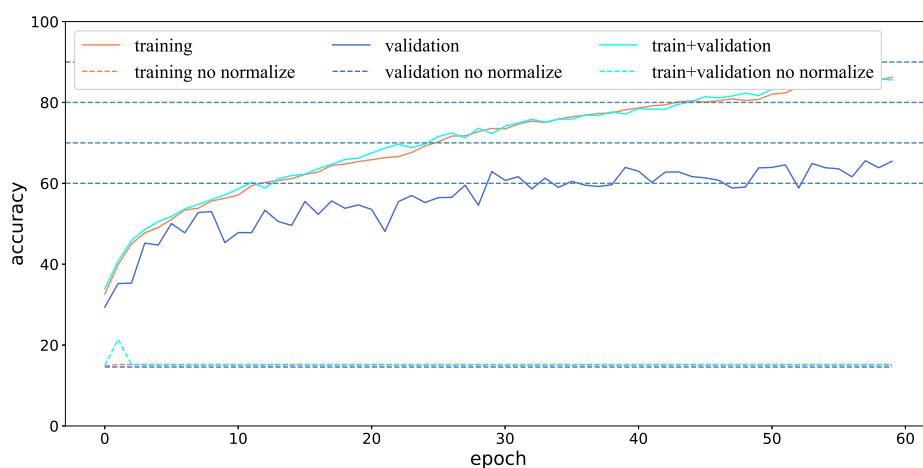


图 8: 对比数据标准化效果

可发现, 未进行数据标准化的模型, 其准确率随着训练未发生增长, 始终保持在 20% 以下. 这说明数据标准化能有效提高模型的训练效果, 使得一些原本难以训练模型变得可行.

## 5 结论

在 CNN 神经网络中, 相同参数量下更深的深度往往有更好的效果, 而单纯提升单层的参数数量并不能带来巨大的提升. 加大深度” 就相当于函数中的模块化, 就可以减少数据的需求量.

深度学习中可以使用 Normalization 提高训练效果. 标准化能使得训练过程更加鲁棒, 并避免了模型过拟合.

数据增强 (Data Augmentation, DA) 缓解了深度学习中数据不足的场景, 在图像领域首先得到广泛使用, 进而延伸到 NLP 领域, 并在许多任务上取得效果。一个主要的方向是增加训练数据的多样性, 从而提高模型泛化能力.

## 参考文献

- [1] <https://www.runoob.com/numpy/numpy-tutorial.html>