# 基于深度学习的 CIFAR-10 图片分类

自 64　　赵文亮　　2016011452

2019 年 1 月 20 日

## 1　任务介绍

CIFAR-10 数据集 [1] 中包含了 60000 张 32*32 的彩色图片，分为 10 种类别：airplane，automobile，bird，cat，deer，dog，frog，horse，ship，truck。本文将使用三种深度学习方法完成图像分类的任务。本文后续内容的结构如下：第 2 节首先介绍了本文中三种方法的基础——卷积神经网络，之后分别介绍三种方法的原理；第 3 节展示了模型的实验结果，并对其进行多方面的分析，最后对实验进行总结。

## 2　方法与模型

### 2.1　卷积神经网络

卷积神经网络在计算机视觉领域有着广泛的应用。例如，对于分类问题，一种常见的神经网络架构如图 1 所示。其中主要包括卷积层、激活函数、池化层、全连接层。不同尺度的卷积层能够有效地提取图像中不同尺度的特征，激活函数（通常为 ReLU）可以在网络中引入非线性，池化层通过下采样来降低输入尺寸，有利于加速模型的训练。最后的全连接层和 Softmax 输出层将隐藏层的结果转化为最终的分类结果（使用 One-hot 编码）。

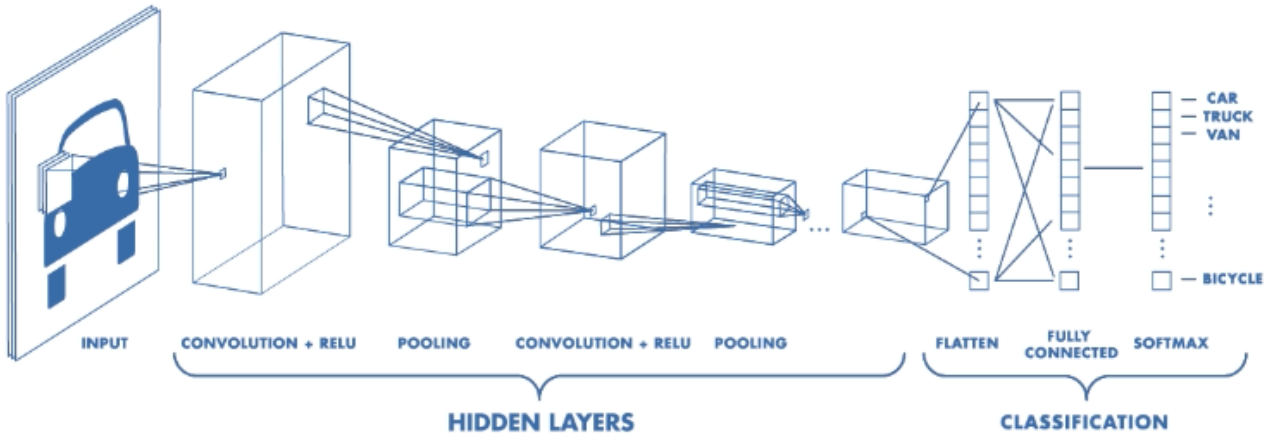



图 1: 图像分类的常见神经网络模型

本文中使用的三种分类方法都是基于卷积神经网络的。第 2.2 的方法中使用了一种有效的对网络参数的初始化方法，第 2.3 节中在卷积神经网络的基础上构造了残差网络（ResNet），第 2.4 中使用了全卷积网络（隐藏层中只有卷积层）。

## 2.2 LSUV Initialization

本节中介绍了 [2] 提出的一种新的初始化权重的方法，即 LSUV（Layer-sequential unit-variance）初始化方法。LSUV 方法可以有效地提高模型的收敛速度，并且提高模型的准确率。LSUV 方法基本思想是将权重矩阵初始化为方差为 1 的正交阵。

[3] 曾经提出过一种正交初始化的方法：

1. 将权重初始化为方差为 1 的高斯噪声
2. 对权重矩阵进行 QR 分解或 SVD 分解得到正交阵

在上述正交初始化的基础上，LSUV 方法希望每一层的输出的方差为 1，具体的实现方式相当于对第一个输入 batch 做一次批标准化（Batch Normalization[4]）。具体流程如算法 1 所示。

**Algorithm 1:** Layer-sequential unit-variance orthogonal initialization. $L$ –convolution or fullconnected layer, $W_L$ - its weights, $B_L$ - its output blob., $Tol_{var}$ - variance tolerance, $T_i$ –current trial, $T_max$ – max number of trials.

**Pre-initialize** network with orthonormal matrices as in [3]
**for** *each layer* $L$ **do**
  **while** $|Var(B_L) - 1.0| \geq Tol_{var}$ *and* $(T_i < T_{max})$ **do**
    do Forward pass with a mini-batch
    calculate $Var(B_L)$
    $W_L = W_L / \sqrt{Var(B_L)}$
  **end**
**end**

本节中在 FitNet-4[5] 的模型上使用 LSUV 初始化方法。为了叙述方便，此处借用 [6] 中神经网络的方法。FitNet-4 的网络结构如下：

$$(32C3)_3 - (48C3)_2 - MP2 - (80C3)_5 - MP2 - (128C3)_5 - MP8 - FC500 - Softmax10 \tag{1}$$

其中 $3C32$ 表示一个具有 32 个卷积核的卷积层，卷积核大小为 3x3。下标 3 表示三个这样的卷积层。MP2 表示池化核大小为 2x2 的池化层。FC500 表示具有 500 个神经元的全连接层，Softmax10 表示具有 10 个输出的 Softmax 层。

在网络构建中，为了防止梯度消失、梯度爆炸、过拟合等问题，我在每个 Pooling 层后面增加了 Batch normalization 和 Dropout。

## 2.3 ResNet

本节中的模型是由 [7] 提出的深度残差网络 ResNet。ResNet 的提出旨在解决深度神经网络难以训练的问题。ResNet 的一个构建块如图 2 所示。设输出 $\mathcal{H}(x) = \mathcal{F}(x) + x$，则 $\mathcal{F}(x) = \mathcal{H}(x) - x$。[7] 假设，对残

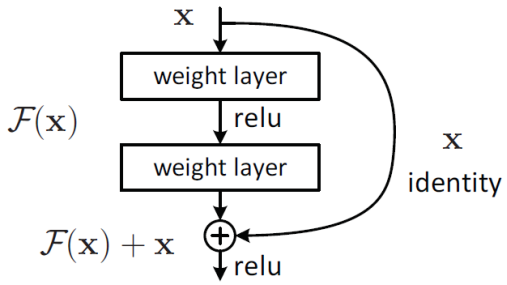



图 2: ResNet 构建块

差 $\mathcal{F}$ 的学习会比直接对 $\mathcal{H}$ 的学习容易。ResNet 模型在 *ILSVRC 2015 classification competition, ImageNet detection, ImageNet localization* 等诸多大赛中获得第一名，这些也证明了 ResNet 的普适性。

在实际的网络中，图 2 中的 weight layer 使用卷积层来实现。由于计算资源有限，本文中只使用了 20 层的 ResNet，而事实上 ResNet 可以达到上千层，并且更多的层数可以收获更高的分类精度。

### 2.4 All Convolutional Net

本节中的模型是 [8] 提出的全卷积网络。该模型的作者注意到池化层事实上可以使用步长大于 1 的卷积层来替代，全连接层可以使用卷积核为 1x1 的卷积层来代替，进而构建了全卷积的模型。整个模型中除了输入和输出层之外，完全使用卷积层构成。

仍然使用第 2.2 节的简化表述方法，该网络的模型如下：

$$(96C3)_2 - 96C^{(2)}3 - (192C3)_2 - 192C^{(2)}3 - 192C3 - 192C1 - 10C1 - GA - Softmax10 \tag{2}$$

其中上标为 2 表示该卷积层的步长为 2，GA（Global Averaging）表示全局平均的操作，即对前一层输出的每一个通道求平均。同样，在步长为 2 的卷积层（起到池化层的作用）后面使用了 Dropout 和 Batch Normalization。

## 3 实验与总结

### 3.1 模型训练

原始的 CIFAR-10 数据集中包含 50000 张训练图片和 10000 张测试图片。10000 张测试图片作为测试集使用，并将训练图片中的 1/10 拿出作为验证集，其余作为训练集。训练时，对训练集图片进行数据增强（平移、镜像、旋转等），以提高模型的鲁棒性。每轮训练结束后，在验证集上对结果进行验证，保留验证集上准确率最高的模型参数。也就是说，在整个训练过程中，模型从未看到过测试集图片。模型训练完毕后，使用测试集的图片得出最终的准确率。本文中列出的准确率都是在测试集上得出的。

本文使用 Keras[9] 搭建神经网络，三种模型都是用 SGD 方法进行训练，并使用交叉熵损失函数。学习率会随着训练的轮数增加而递减，以提高模型的准确率。LSUV、ResNet、AllConv 三个模型训练的 epoch 数分别为 230、500、500。本文使用 *NVIDIA GeForce GTX Titan Xp* 进行训练

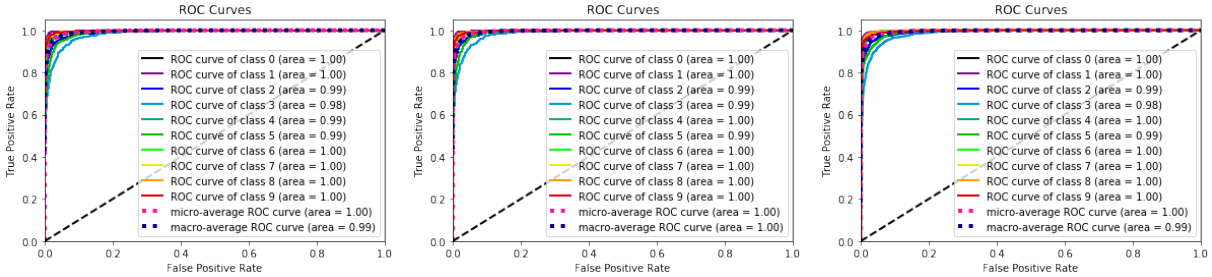
### 3.2 模型评测

各个模型的训练集和测试集的正确率如表 1 所示，可见三个模型的表现都较好。训练集正确率接近 100%，说明训练得比较充分；测试集正确率比训练集正确率低，但也相对较高，这说明了训练的过程中很好地抑制了过拟合。

表 1: 各模型测试集与训练集正确率比较

| 模型简称 | 训练集正确率 | 测试集正确率 |
|---|---|---|
| LSUV | 97.40% | 90.72% |
| ResNet | 98.09% | 91.26% |
| AllConv | 99.38% | 91.20% |

进一步绘制三个模型的 ROC 曲线，如图 3 所示。图中显示了三个模型各个类别的 ROC 曲线，以及计算得到的 ROC 曲线围成的面积（AUC）。从中可见三个模型的各个类别的 ROC 曲线均接近理想曲线，且 AUC 均接近 1。




(a) LSUV (b) ResNet (c) AllConv

图 3: 不同模型的 ROC 曲线

## 3.3 模型分析

### 3.3.1 超参数灵敏性分析

本节中将对本文中的三个模型中的超参数进行灵敏性分析。主要的超参数为学习率和 batch size，为了方便计算，本节中只比较第一轮训练的训练集准确率，如表 2至表 2 所示，这个指标可以一定程度刻画模型的收敛速度。

表 2: 不同超参数下 LSUV 模型的第一轮训练集准确率

| 初始学习率 | 第一轮训练集准确率 |
|---|---|
| 0.001 | 29.99% |
| 0.01 | 26.24% |
| 0.1 | 19.88% |

(a) 不同初始学习率（batch size = 32）

| batch size | 第一轮训练集准确率 |
|---|---|
| 32 | 29.99% |
| 64 | 26.88% |
| 128 | 23.19% |

(b) 不同 batch size（lr = 0.001）

从学习率来看，可见合适的学习率可能会提高模型收敛速度，过高的学习率可能会导致训练收敛变慢甚至无法收敛。从 batch size 来看，对于 LSUV 和 ResNet 两个模型，batch size 越大，模型收敛越快；而对于 AllConv 模型，batch size 对首先速度影响不大。

表 3: 不同超参数下 ResNet 模型的第一轮训练集准确率

| 初始学习率 | 第一轮训练集准确率 |
|---|---|
| 0.001 | 35.17% |
| 0.01 | 39.49% |
| 0.1 | 25.48% |

(a) 不同初始学习率（batch size = 32）

| batch size | 第一轮训练集准确率 |
|---|---|
| 32 | 35.17% |
| 64 | 30.82% |
| 128 | 28.63% |

(b) 不同 batch size（lr = 0.001）

表 4: 不同超参数下 AllConv 模型的第一轮训练集准确率

| 初始学习率 | 第一轮训练集准确率 |
|---|---|
| 0.005 | 37.49% |
| 0.05 | 36.84% |
| 0.1 | 10.04% |

(a) 不同初始学习率（batch size = 32）

| batch size | 第一轮训练集准确率 |
|---|---|
| 32 | 36.84% |
| 64 | 37.79% |
| 128 | 37.56% |

(b) 不同 batch size（lr = 0.05）

#### 3.3.2 模型比较

从表 1 中可以看出，三个模型的准确率差别不是很大。从训练速度来看，ResNet 的训练时间最长，LSUV 和 AllConv 的训练时间相当。这是由于 ResNet 的层数较多，训练时计算复杂度较高。在一个 CIFAR-10 数据统计网站中[1]，三个模型准确率的排名为 AllConv(95.59%)>LSUV(95.16%)>ResNet(93.57%)。本文中对三种算法的实现并没有达到上述准确率，这可能是由于超参数不同、训练轮数不够等因素导致的。

#### 3.3.3 特征分析

本文中的三种模型各有特点，但都是建立在卷积神经网络的基本架构上的。从数字图像处理的相关知识里，我们可以将卷积操作认为是对图像的一个滤波。滤波操作在图像处理中可以有多种作用。例如，高通滤波可以进行边缘检测、低通滤波可以去除噪声。另一方面，卷积 + 池化的操作与图像处理中金字塔的概念相类似 [10]。图像金字塔能够较好体现不同频段下图像的特征，例如可以使用高斯金字塔和拉普拉斯金字塔进行多频带融合 [10]。这也一定程度上解释了为什么卷积神经网络能够有效地处理大部分计算机视觉的问题。

LSUV 巧妙地通过对神经网络中的参数进行初始化，实现了模型收敛速度的提高和准确率提高。所以对于任何一个网络模型，都可以尝试使用 LSUV 初始化方法，有可能得到较好的效果。ResNet 是一种非常新颖的网络架构，它的出现使得深度神经网络的训练更加有效。当网络深度增加时，ResNet 的优势更能显现出来。AllConv 模型则是比较创新的一个想法，它试图将整个网络都用卷积层来表达，从而揭示了池化层、全连接层与卷积层之间的内在联系。

[1] http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html#43494641522d3130

### 3.4 总结

本次实验中，我使用了三种模型完成了对 CIFAR-10 图像的分类问题，虽然没有达到原论文中的准确率，但仍然取得了较好的结果。本次实验中我采用多种模型的原因一方面是想对不同的模型进行比较，另一方面是想通过这个大作业对深度学习有更深的了解。通过本次实验，我学会了自己在服务器上配置深度学习环境、使用 tensorflow 和 keras 搭建神经网络等技能，这些对我未来的科研都会很有帮助。

## 参考文献

[1] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[2] D. Mishkin and J. Matas, "All you need is a good init," *arXiv preprint arXiv:1511.06422*, 2015.

[3] A. M. Saxe, J. L. McClelland, and S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," *arXiv preprint arXiv:1312.6120*, 2013.

[4] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[5] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *arXiv preprint arXiv:1412.6550*, 2014.

[6] B. Graham, "Fractional max-pooling," *arXiv preprint arXiv:1412.6071*, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[8] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[9] F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.

[10] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, "Pyramid methods in image processing," *RCA engineer*, vol. 29, no. 6, pp. 33–41, 1984.

# A 函数说明

本文中主要使用 Keras[9] 库函数完成神经网络的搭建。一些自定义函数如下：

1. LSUVInit：LSUV 初始化函数，[2] 提供了代码链接；
2. resnet_layer：一个 resnet 的构建块（不含恒等变换回路，即为第 2.3 节中的 $\mathcal{F}(x)$）
3. resnet：ResNet 网络模型，使用 keras.layers.add 完成 $\mathcal{H}(x) = \mathcal{F}(x) + x$ 的操作得到完整的构建块，并将多个构建块堆叠形成 ResNet 结构

# B 运行方式

本文中的代码在 Jupyter Notebook 中编写。共有每个模型对应着一个目录，每个目录中的.ipynb 文件即为主代码（LSUV.ipynb, ResNet.ipynb, AllConv.ipynb）。每个目录下的 model 文件夹中为训练好的模型，可以直接加载模型并验证。