最后来一张证明是自己电脑的照片！

1. 网络结构细节：

1.1ReLU的理解：

1)

原因：深层次网络的效果超过了sigmoid，并解决了sigmoid在网络较深时的梯度弥散问题，（梯度弥散：个人理解

是到达一定程度，激活函数导数会很小，导致权重更新缓慢，不能有效学习，所以在深层网络中，就会导致网络层之间学

习效果跟没学一样！）

作用：防止过拟合

2)

1.2LRU（局部响应归一化）的理解：

1)

原因：网上说是被激活的神经元抑制相邻的神经元，因为使用ReLU之后，没有像tanh，sigmoid函数一样有个值

域区间，比如tanh完后，数据会在（0，1），sigmoid完后，数据会在（0，0.

25）之间，所以会在Relu后做一个归一化，这种抑制很管用，所以引入

2)

3)

作用：增强范化能力，

原理：

这是LRN

（局部响应归一化）的公式：粗略理解就是用对输入值a除一个数达到归一化。累加符号是在累加通道数的像素值，累加多少

由你自己设。

1.3Dropout的理解：

4)

扩充：据说在池化前使用效果很好

A、 原因：全连接层参数过多，容易过拟合

B、 实现：随机使神经元关闭，使它不能使用。

C、 原理：每一次相当于在训练一个子网络

D、 作用：消除了神经元之间的依赖，增强范化能力。

1.4重叠最大池化层的理解：

1)

作用：

避免平均池化的模糊化效果。并且AlexNet中提出让步长比池化核的尺寸小，这样池化层的输出之间会有重叠和覆

盖，提升了特征的丰富性。

2)

实现：使步长小于卷积核的大小（长宽）

2. 随机取样：

1.

实现：对数据进行随机取样，从256中取出224个样本

3. 数据增强：

1.1图片处理

1)

实现：

取图片的四个角加中间共5个位置，并进行左右翻转，一共获得10张图片，对他们进行预测并对10次结果求均值。

1.2 PCA处理图片通道

1.

2.

实现：对图像的RGB数据进行PCA处理，并对主成分做一个标准差为0.1的高斯扰动，增加一些噪声

理解：我对pca的理解就是降噪，提取重要的特征。用在这里估计是为了提取重要图片特征，减少其他因素影响。

4. 硬件组合：

GPU并行：（我无法实现，我只有一个GPU）

1)

实现：利用GPU自身内存，将数据分开分别放在不同的GPU上跑然后输出，可以直接递给下层的GPU，也可以交

叉合并给下个层。在早期的网络中，两个GPU使得网络更大，网络训练更快

5. 网络结构构造

AlexNet网络是使用五个卷积层与三个全连接层的。

结构特点：2-GPU并行，1，2，5跟随max-pooling层，两个全连接上使用dropout

前面卷积核大后面卷积核小的原因：是

一开始原图的尺寸虽然很大很冗余，但最为原始的纹理细节的特征变化就用大卷积核尽早捕捉到比较好，后面的更深的层

数害怕会丢失掉较大局部范围内的特征相关性，所以使用小卷积核。

具体：

第一个卷积层

输入：224 x 224 x 3 卷积核：11 x 11 x 3 x96(个) 步长 4

输出： （224-11+padding）/4+1 = 55 （213/4除不尽加上3变成216就除得尽）

55 x 55 x 96

第二个卷积层

输入： 将第一个卷积层的输出进行响应归一化及最大值池化

每个GPU上都是5x5x48个卷积核

卷积核：5 x 5 x 48 x 256 池化层：2 x 2

输出： （55-5+2）/2+1 = 27

27 x 27 x 256

第三个卷积层

输入：第二个卷积层两个GPU输出结果的合并

卷积核：3 x 3 x 256（通道数）x 384（个数）

第四个卷积层

输入：仅从上一个单一GPU输出

卷积核：3 x 3 x 192 x 384

第五个卷积核

输入：3 x 3 x 192 x 256

全连接：4096个神经元

最后通过softmax转化为概率，分为1000类。

6. 参数：

Dropout = 0.5（消除的神经元比例）

Batch Size = 128 （最小批量法的批量数）

Learning\_rate = 0.01 (每一定次数降低为原来的十分之一)

SGD momentum = 0.9 （动量梯度下降法中之前累加的动量所占权重）

公式

Dx = compute\_gradient(x)

（计算当前梯度）

vx =rho \* vx + dx

（

0.9就是rho，即权重，含义将上一步保存的梯度加上当下的梯度就是这一步要更新的梯度)

x+ = learning\_rate \* vx

（更新参数）

7. 在我电脑上跑的代码

对于我从该博客复制的代码而言：

输入为 28 x 28 = 784

输出为 一个10 x 1 的向量

Dropout = 0.8 batch\_size = 64 learning\_rate = 0.001

网络结构

三个卷积层：每一层都进行降采样即最大值池化层，并进行归一化然后进行dropout处理

两个全连接层，不进行dropout。

第一层的卷积核为 3 x 3 x 1 x 64

第二层的卷积核为 3 x 3 x 64 x 128

第三层的卷积核为3 x 3 x 128 x256

全连接层：神经元为1024个

VGG

1. 提出原因：

相较于alexnet的局限，vgg使网络的深度得到延伸（网络深的好处：浅层网络学到的是边缘

（Edges）、纹理（Texture）等，深层学到的是更偏向语义的信息，相当于把原本线性模型的feature

map拉长了。本质还是那么多信息，只是中间的过程更加清晰可见

），vgg的初衷是想搞清楚卷积网络深度如何影响大规模图像分类与识别精度和准确率。最初称为VGG-16（GG-Very

-Deep-16 CNN）

2. 网络架构：

2.1总体概括

◼

◼

全连接层依旧3层，根据卷积层+全连接层总数目的不同从浅到深为VGG11~

VGG19，最小的VGG11有8个卷积层与3个全连接层，最多的VGG19有16个卷积层+

3个全连接层。池化层总数依旧是5个且是2x2的max pooling，分布在不同的卷积层之下。（

卷积专注于保留空间信息前提下的channel变换，而池化则专注于空间信息的变换（下采样）

前两组卷积形式一样，每组都是：conv-relu-conv-relu-pool；

）

中间三组卷积形式一样，每组都是：conv-relu-conv-relu-conv-relu-pool；

◼

最后三个组全连接fc层，前两组fc，每组都是：fc-relu-dropout；最后一个fc仅有fc

2.2卷积核尺寸

在加深网络层属的同时避免参数过多，在所有层都采用3 x 3

的小卷积核，卷积层步长设置为1。VGG的输入设置为224 x

224大小的RGB图像，训练集对所有图象计算RGB均值，然后把图像作为输入传入VGG网络，使用3x3或1x1的filte

r，卷积步长固定为1。采用小卷积核是优点在于增多了非线性层，扩宽了网络深度，保证学习更复杂的模型，且参数更少。（

步长为1是因为实验发现这样保留feature map 的宽高情况下，性能好

）

2.2.1 多使用3 x 3 卷积核

A、 2个3 x 3 卷积核可以看作一层5 x 5 卷积层

B、 3个3 x 3 卷积核可以看作一层7 x 7 卷积层

优点

2个3 x 3 = 1 个 5 x

5，2层比1层更多一次非线性变换（Relu，会使拟合程度更好），5x5参数：5x5x输入x输出=

25x输入x输出，3x3参数：3x3x输入x输出x2=18x输入x输出，参数减少：（25-18）/25=28%

2.2.2使用1 x 1 的卷积核

优点

1：增加了非线性函数的使用（Relu）

2：1 x 1不考虑单通道上像素的局部信息，它专注于一个卷积核内部通道的信息整合。

3：可以用于降低特征映射的通道数目，比如224 x 224 x 100的图像经过20个1 x 1卷积核，会得到224 x

224 x20的输出，当卷积核数量达到上百个时，卷积计算会非常耗时，所以先用1 x 1卷积在计算3 x 3 或5 x

5卷积计算前先降低feature map的维度。

2.3池化层

池化核相比ALexnet都是3x3的maxpooling，vgg是2x2，原因可能是

2×2带来的信息损失相比3×3的比较小，相比3×3更容易捕获细小的特征变化起伏。小的卷积核

带来的是更细节的信息捕获。

2.4全连接层

网络测试阶段将训练阶段的三个全连接替换为三个卷积（FCN

：将全连接层改成7x7的卷积网络，后面两个连接曾改为1X1的卷积网络，整个网络就变成全卷积网络，故称FCN）

优点在于全卷积网络可以接收任意尺度的输入

（因为卷积核你可以自由定义，继而使之成为你想要的输出结果）。原先输入到全连接层是7x 7x512 变成 1x1x4096,

所以用7x7x512x4096的卷积核,这样就可以变成1x1x4096.

◼

FC4096-ReLU6-Drop0.5，FC为高斯分布初始化（std=0.005），bias常数初始化（0.1）

训练集上，与AlexNet相比，基本一样，超参数上只有最后一层fc有变化bias初始值从0变为0.1，该层

初始化高斯分布的标准差，由AlexNet的0.01变为0.005。◼

原因估计是这样效果更好！在bias与标准差之间做个交易。

◼

FC4096-ReLU7-Drop0.5，FC为高斯分布初始化（std=0.005），bias常数初始化（0.1）

FC1000（最后接SoftMax1000分类），FC为高斯分布初始化（std=0.005），bias常数初始化（0.1）

2.5 feature map 维度变化

对于vgg，其采用5个池化层，使得 width 和 height 逐渐变化：224->112->56->28->14->7

，

但是深度depth（或说是channel数），随着5组卷积在每次增大一倍：3->64->128->256->512->

512。特征信息从一开始输入的224x224x3被变换到7x7x512

。因为从接近25000接着输入全连接层4096这个映射过程会有点快，所以作者加了两层4096全连接层最后一个soft

max的1000. feature map维度的整体变化过程是：

先将local信息压缩，并分摊到channel层级，然后无视channel （通道）和local（局部），通过fc

这个变换再进一步压缩为稠密的feature map

，这样对于分类器而言有好处也有坏处，好处是将local信息压缩到feature

map中，坏处是信息压缩都是有损失的，相当于local信息被破坏了（分类器没有考虑到，其实对于图像任务而言，单张feat

ure map上的local信息还是有用的.）

3. 预训练特点：

为了防止不好的初始化会导致训练过程中停止学习，作者把A实验的模型的前四个卷积层和三个全连接层的参数拿

来作为模型的初始化参数，中间层的初始值从均值为0，方差为0.

01的正态分布中采样得到，bias初始化为0，学习率设定和训练A（相当于pre-train）时的一样。

1.1Pre-training介绍

Vgg是使用Pre-training方法。Pre-

training：我查了下其意思，大概就是利用之前较浅的模型（A）训练生成模型（ 可以先适应输入然后得到相较有用的特征

），然后利用生成模型得到的参数作为搜索起点，对后来的模型进行训练。 Pre-

Training的最好的理解是，它可以让模型分配到一个很好的初始搜索空间 一开始在足够大的数据上pre-

（

train，模型已经见识过了广阔的样本空间，这会带来了更广阔和丰富的特征空间，因而模型在小数据上学习时不会太过纠结

于比较片面或者偏斜的样本带来的影响），pre-trained模型的参数已经避开了用小数据train-from-scratch(

从头开始训练)的一些局部最优解。

其中有两种情况：

1)

针对数据量小的任务，把前面大部分层数的参数冻结，（

在数据集相似度不是很高下，冻结是为了弥补训练数据量不充足，这里理解为什么冻结可以弥补数据不充足是因为：

你冻结了卷积层，就为训练节省了很多时间，你不需要很多数据去训练前面冻结层的参数，只需拿数据去训练高层的

参数即可，如果你想知道应该冻结几层，可以先尝试冻结所有层，然后训练模型并查看它是如何执行的。

然后尝试解冻一个或两个较高隐藏层，让反向传播调整它们，看看性能是否提高。

您拥有的训练数据越多，您可以解冻的层数就越多 。

）保留前面的卷积层，以获取在ImageNet上得到的提取特征的能力，作为特征提取器，而只训练最后一层全连接层

。

2)

对于数据量中等的新任务需要更多的可变层来拟合新任务的数据，所以冻结前面的层，留出更多的层去拟合新数据。

1.2使用原因：

用在 ImageNet 上 pre-trained 过的模型

，

设计自己模型架构很浪费时间，尤其是不同的模型架构需要跑数据来验证性能，所以不妨使用别人在 ImageNet

上训练好的模型，然后在自己的数据和问题上在进行参数微调，收敛快精度更好。

4. 局限：

1)

VGG在加深CNN网络深度方面做出了贡献，但是并不能无限制加深，当到达一定层度后依旧会出现训练效果褪化，

梯度弥散或梯度爆炸等问题。

2)

3)

参数依旧很多，因为有三个全连接层，使用普遍的是VGG16（E）与VGG19（D）

VGG只需很少的迭代次数就开始收敛，原因一是深度和小的滤波器起到隐式规则化作用；二是一些层的 pre-

initialisation 网络A的权值W~（0,0.

（

01）的高斯分布，bias为0；由于存在大量的ReLU函数，不好的权值初始值对于网络训练影响较大。

）为了绕开这个问题，可以通过随机的方式训练最浅的网络A；然后在训练其他网络时，把A的前4个卷基层

和最后全连接层的权值当做其他网络的初始值，未赋值的中间层通过随机初始化。

5. 参数

优化方法：带动量（momentum）的小批量梯度下降

batch size：256

learning rate：0.01

和AlexNet一样，当val-acc 不下降则学习率缩小十倍，训练过程缩小了三次

momentum：0.9

weight decay（L2惩罚乘子）：5×10−4

dropout rate（前两个全连接层）：0.5

目标函数：多项式逻辑斯特回归（SoftMax）

迭代次数：37万次iteration（74 epochs）后，停止训练

6. 训练的图像尺寸

V

gg输入固定是224x224，但是这个大小是通过大于224图片裁剪得来的，vgg先使用通过256x256裁剪的图像进行

训练然后使用384x384裁剪后的进行调整网络。先说裁剪，这样可以避免过拟合吧，随机采样嘛，然后就是因为这样做了

结果效果很好。V

◼

训练图像的尺寸S固定时，设置训练图像尺寸S等于测试图像尺寸Q；

训练图像尺寸S是介于[Smin,Smax]时，设置测试图像尺寸Q=0.5(Smin+Smax)。

gg为了增大数据，zai裁剪后还对图像进行了随即水平翻转和随机的RGB颜色抖动（查了下，好像原名叫

◼

半色调技术，是通过色点大小或密度的变化造成总体强度的变化，从而表现出较多的对比度，达到了使用较少的色彩模拟表

达较多色调的目的。不知道对不对！），这两技巧AlexNet都有用到。

S:图像最小边，一般图像长宽一样，就是图像大小。

7. 实验结论

1)

2)

3)

4)

弃用LRN。Vgg在A模型里有使用到LRN，但发现性能并没有提升因此之后就没有使用。

深度增加，分类效果更好。从11层到19层，结果是越来越好。

1x1卷积核的非线性作用。C和D模型层数相同，但是D将C中3x3卷积核换成1x1卷积核而使效果得到提升。

小且多的卷积核比单个大卷积核性能好（相同视野情况下）。这是vgg最大的创新点。

8. 我使用的代码

使用的数据集是cifar-

10，网络结构是前面是九层，都是两个卷积层跟一个池化层，一共进行三次，最后接一个全连接层。输入是32x32，经过三

个相同结构的层次，因为有三个池化层，相当于降采样，所以到全连接层是32/（2x2x2）=4。训练10000次，准确率是0.

729。

ResNet

1. 背景

相较于AlexNet，vgg都在向着越深的网络层次发展，在加深层次的问题上，都遇到了加深网络层次会使训练集准确

率降低。基于深层次网络更难优化而非深层次网络学不到东西的假设，resnet提出了至少使深层网络的结果可以与浅层

网络持平，虽然增加了深度，但是不会增加误差，即学习残差的方式，而非直接去学习映射关系。

2. 创新点

2.1解决网络深度问题

Resnet使用了正则化初始化和中间的正则化层（Batch Normalization）。

BN

和卷积层一样都是一个网络层，优点加快训练速度，提高网络泛化能力，本质是一个归一化网络层，可以替代LRN

，可以打乱样本训练顺序（一张图片不会被多次选择）。

计算流程：先计算样本方差与平均值，接着对样本数据进行标准化处理，最后

进行平移和缩放处理。引入了γ和β两个参数。来训练γ和β两个参数。引入了这个可学习重构参数γ、β，让我们的网络可

以学习恢复出原始网络所要学习的特征分布。

样本是通过卷积核过滤得到一张特征图得来的，特征图有多少个特征值，就有多少个样本。

具体实现流程：

输入：待进入激活函数的变量

输出：

1.这里的K，在卷积网络中可以看作是卷积核个数，如网络中第n层有64个卷积核，就需要计算64次。

需要注意，在正向传播时，会使用γ与β使得BN层输出与输入一样。

2.在反向传播时利用γ与β求得梯度从而改变训练权值（变量）。

3.通过不断迭代直到训练结束，求得关于不同层的γ与β。

4.不断遍历训练集中的图片，取出每个batch\_

size中的γ与β，最后统计每层BN的γ与β各自的和除以图片数量得到平均直，并对其做无偏估计直作为每一层的E[x]

与Var[x]。

5.在预测的正向传播时，对测试数据求取γ与β，并使用该层的E[x]与Var[x]，通过图中11所表示的公式计算BN层输出。

正则化：

我记得机器学习时，学了两种正则化，一种是L1正则化，就是在损失函数后加一个参数绝对值的累加，L2正则化则是

加一个参数平方的累加，作用主要是限制参数大小，L1正则化因为导数是sign函数，就是在趋向于0时趋势都是直线，燃耗

在y正半轴到达x=0，然后再到（0，0）点，所以会使一部分参数值变为0，可作为特征选择用。

2.2解决网络退化问题（网络层数增加，但是在训练集上的准确率却饱和甚至下降了

）

提出一种将本身与残差一起相加后输出的全新网络，公式y=F(x)+x提出了两种mapping： 一种是identity

mapping，指的是本身，即x，另一种residual mapping，指的是差，也就是y−x

这样可以解决“随着网络加深，准确率不下降”的问题的原因是

，所以残差指的就是F(x)部分。

因为本身也加入的情况下，即使深度加深网络性能至少与之前的保持一致而不会降低！

3. Resnet结构：

有两种，如图。每个结构成为”building block“，而右图又称为”bottleneck design”

，左边的一般用于34层或者层数更少的网络，而右边的用于更深的101这样的网络，目的都是为了减少计算和参数量。之

所以分两种模型，是因为在深层网络中，右边的模型参数少于左边的。（第一个

1x1的卷积把256维channel降到64维，然后在最后通过1x1卷积恢复，整体上用的参数数目：1x1x256x64 +

3x3x64x64 + 1x1x64x256 = 69632，而不使用bottleneck的话就是两个3x3x256的卷积，参数数目:

3x3x256x256x2 = 1179648，差了16.94倍。

）

3.1问题

本身与残差需要维数一样才能进行相加，所以针对这种情况，resnet提出了两种方法

1对不足的特征维数进行补零，跟加padding一样。2称为投影法，利用1x1的卷积进行升降维。

用公式表达为：

y=F(x)+Wx

，

直接改变1x1卷积的filters数目。这种会增加参数。

实验证明，投影法优于加零的方法，在更深的网络中，除了使用图像右边的模型外，一般使用加零法，因为投影法会导致参数

过多影响计算。

4. 结论

1)

2)

Res将参数均摊到卷积层上，参数越多，学到越多，强调卷积层而弱化全连接层。

基本架构：先用一个普通卷积层，步长为2，再经过一个3x3的max-

pooling，再经过一个残差结构，没有中间的全连接层，直接到输出。

3)

残差结构使得网络需要学习的知识变少，容易学习，使得每一层的数据分布接近，容易学习。

5. 我所使用的代码

残差快的结构为2个3x3步长如果输入通道为输出通道的一半，则步长为2，如果输入输出通道数目一样则设为1，再

经过3个3x3步长为1的卷积核加上本身，如果输出通道数不同则进行zero处理，然后输出。

大体结构为输入到残差块，然后一个池化层最后一个全连接，进行十分类。

GoogleNet

1. 工程优化——同样的参数更加有效率(背景)

深层网络的问题

1更深的网络容易过拟合

2更深的网络有更大的计算量，稀疏网络虽然减少了参数但没有减少计算量。

2. 网络结构四种v1~4

V1

1一层上同时使用多卷积核，看到各种层级的feature

map。即一个模块一层可以看到多个尺度的信息，那么下一层也可以同时从不同尺度中提取特征，进行多维特征的融合。

2不同组之间的feature map不交叉计算，减少了计算量。

3额外增加2个辅助的softmax

用于向前传到梯度（辅助分类器）。辅助分类器是在中间某层输出用作分类，并按一个较小的权重0.

3加到最终的分类结果中，相当于模型融合，同时给网络增加了反向传播的梯度信号（即再分类器的较低阶段加强辨别，增加

被传播回来的梯度信号，因为在深层网络中，反向传播的梯度变得非常小，以至于网络在前几层很难学习），并提供额外的正

则化（深度网络通常会过拟合，而浅层神经网路欠拟合，将早期的分类加到后期分类，相当于均衡的作用，降低深层的过拟合）

。

V2

1引入了vgg里提出的通过3x3视野域同等卷积替换。

有效但缺乏理论的基本准则

1 . 避免表达瓶颈，特别是在网络靠前的地方。

信息流前向传播过程中显然不能经过高度压缩的层，即表达瓶颈。从input到output，feature

map的宽和高基本都会逐渐变小，但是不能一下子就变得很小。比如你上来就来个kernel = 7, stride = 5 ,

这样显然不合适。

另外输出的维度channel，一般来说会逐渐增多(每层的num\_output)

，否则网络会很难训练。（特征维度并不代表信息的多少，只是作为一种估计的手段）

2 . 高维特征更易处理。 高维特征更易区分，会加快训练。

3. 可以在低维嵌入上进行空间汇聚而无需担心丢失很多信息。

比如在进行3x3卷积之前，可以对输入先进行降维而不会产生严重的后果。假设信息可以被简单压缩，那么训练就会加快

。

4 . 平衡网络的宽度与深度。

上述的这些并不能直接用来提高网络质量，而仅用来在大环境下作指导。

V3

进行的一个改进是分解，即将nxn分解成1xn,nx1, 将7x7分解成两个一维的卷积（1x7,7x1），3x3也是一样（1x3,

3x1），这样的好处，既可以加速计算，又可以将1个卷积拆成2个卷积，使得网络深度进一步增加，增加了网络的非线性（每增

加一层都要进行ReLU）。

另外，网络输入从224x224变为了299x299。

V4

利用残差结构改进V3，即把inception与残差结构结合在一起。

3. 细节处理

1) GAP

对输入的图片进行GAP（Global Average Pooling

）替换全连接层，具体方法是对每一个feature上的所有点做平均，有n个feature就输出n个平均值作为最后的softmax

的输入。

好处：对数据在整个feature作正则化，防止拟合，不需要全连接层，减少参数，不再关注输入图像的尺寸，

因为不管是怎样的输入都是一样的平均方法，传统的全连接层要根据尺寸来选择参数数目，不具有通用性。

由图可知lobal average pooling是将每一张特征图计算所有像素点的均值，输出一个数据值，这样 n

个特征图就会输出n个数据点，将这些数据点组成一个1\*

n的向量的话，就成为一个特征向量，就可以送入到softmax的分类中计算了

2) 使用均方根反向传播(RMSProp)更新权重

1. rmsprop算法给每一个权值一个变量MeanSquare(w,t)

用来记录第t次更新步长时前t次的梯度平方的平均值。

2. 然后再用第t次的梯度除上前t次的梯度的平方的平均值，得到学习步长的更新比例。

3.

根据此比例去得到新的学习步长。如果当前得到的梯度为负，那学习步长就会减小一点点；如果当前得到的梯度

为正，那学习步长就会增大一点点。

4. 我所用的代码

网络结构输入经过一个卷积一个池化然后经过两个相同的inception块，这里使用的是V1的inception块，即一个1x1

+3x3+5x5+3x3

maxpooling。然后接一个最大池化然后经过两个相同的inception块，再接一个最大池化层，最后接一个全连接层。