華中科技大學

机 器 学 习 实 验 报 告

姓　　名　张　鑫　　学　　号　U201614582

姓　　名　岳　畅　　学　　号　U201613568

姓　　名　田祺云　　学　　号　U201614210

班　　级　种子 1601

时　　间　2019.05.19

目录

一、实验内容

本次实验的主要目标是搭建一种图片分类模型，这种模型能够识别图像块是否存在癌症组织。实验提供给了我们 96*96 像素分辨率的淋巴结组织的显微图像，我们要做的是给出在测试集图像的 32*32 的中心区域是否有癌组织的概率，这是一个图像的二元分类任务。

实验的数据是 PatchCamelyon（PCam）基准数据集的修改版本，（原始 PCam 数据集由于其概率抽样而包含重复图像，在 Kaggle 上呈现的版本不包含重复样本）。图片从较大的数字病理扫描图片中截取。其中大概有 220000 张图像用于训练，57000 张图像用于测试。

二、实验原理

问题分析:

1. 分析问题特征：

淋巴结癌细胞转移有以下特征：

① 关键特征：外来细胞群（经典位置：淋巴结缘窦）

② 具有恶性肿瘤细胞学特征的细胞

③ 核多形性(大小、形状和染色的变化)

④ 异性核：

i. 核增大

ii. 不规则核膜

iii. 不规则的染色质图案，尤指不对称

iv. 大的或不规则的核仁

v. 丰富的有丝分裂图像

⑤ 恶性肿瘤中可见的细胞排列结构：高度可变，依赖于肿瘤类型和分化

i. 腺形成

ii. 单细胞

iii. 集群的细胞

由此可得：不规则的核形状、大小或染色深浅可以提示转移，那么如何将这类数据转换以进行建模？

已知图像的标签只受中心区域（32x32px）的影响，因此只将数据裁剪到该区域对训练

模型是有益。但是如果我们将图像块切的过小，一些周围环境中的信息就会被去除。但是调查可知，32x32px 效果比 48x48px 的性能差。

2. 提升数据质量：

可以检查数据是否包含较差的数据（损坏数据或者没有特征的数据），删除他们可以提高数据的整体质量。

3. 预处理和数据扩充：

有几种方法可以避免过度拟合：数据集扩充、增强特征、正则化和选择复杂度较低的模型。每一次增强后相应的数据处理时间都会时间，这被称为测试时间增强(TTA)，如果我们对每幅图像进行多次处理并推断出平均时间，可以改善最终的预测结果。

在没有对图像进行预处理时，如（图 1）可见，正负样本在绿色通道上的差异较大，但是差异并不明显。所以我们可以通过对图像进行预处理，让图像在各个通道上的特征都有明显的变化。

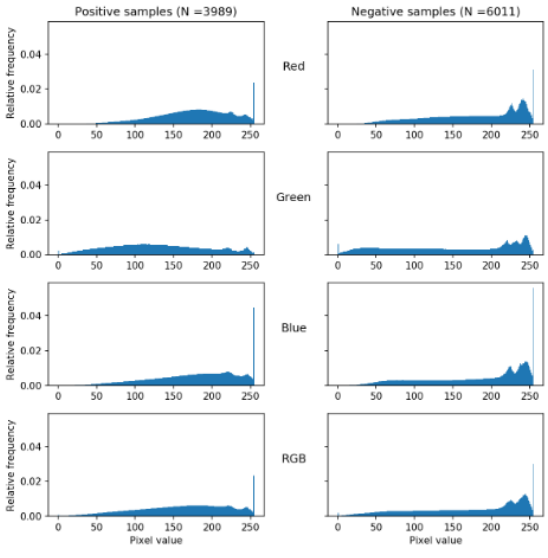


图 1 未进行预处理是数据集 RGB 分布

可以对图像进行如下处理改善数据集质量：

① 随机旋转

② 随机选择数据集

③ 随机翻转（水平竖直均可）

④ 随机缩放

⑤ 随机增亮

⑥ 高斯模糊

在本实验中我们采用了随机旋转、增加对比度、增加图片亮度的方法对数据集进行了处理，如（图 2）可见，处理之后的图像在各个通道下的特征都有明显的差异。对于正样本，高像素点处的 RGB 值较低，而对于负样本，在高像素点处的 RGB 值有明显突变。通过这样的方法，增加了正负样本的特征差异。

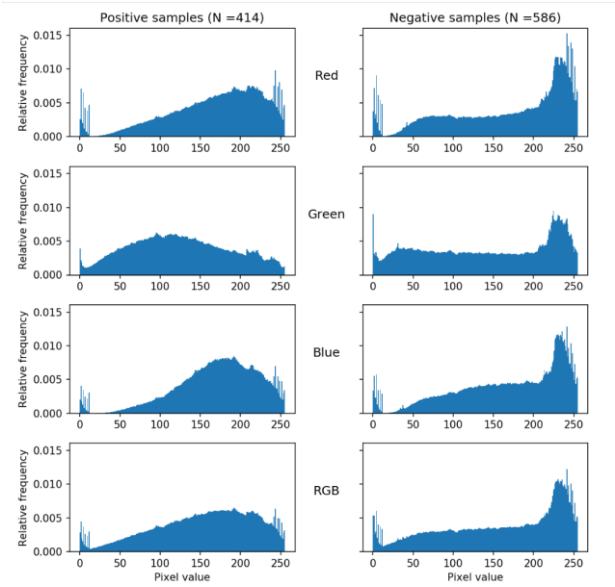


图 2 预处理后数据集 RGB 分布

图像预处理

1. 数据集预筛选：

① 数据筛选原因分析：剔除过亮或者过暗的图像：图像过暗或者过亮可能是由于曝光不良或者剪切到空白部分造成的，有部分空白的数据依旧有效，但是曝光不良的图片则损失了特征所以不能使用。

② 数据筛选方法：在 RGB 颜色通道中，0 代表黑色，255 代表白色。所以可以通过设置 RGB 阈值可以搞定。在本实验中黑色的阈值为 10，白色的阈值为 245。如果图像

的最大 RGB 值小于黑色阈值，那么可以判定这个图片为黑色，同理，如果图片的最小 RGB 值大于白色阈值，则可以判断图像为白色。处理方式如（图 3）所示。

```
if(rgb_img.max() < dark_th):
    too_dark_idx.append(i)
    continue # do not include in statistics
# is this too bright
if(rgb_img.min() > bright_th):
    too_bright_idx.append(i)

    continue # do not include in statistics
X[i] = rgb_img
y[i] = np.squeeze(df.as_matrix(columns=['label']))[i]
```

图 3 数据筛选方式

2. EDA

EDA（Exploratory Data Analysis）的目的是

① 对数据集进行粗略观察

可以让我们先对两种类别有一个大概的了解，这对于后面选择什么特征作为分类依据很重要

② 了解两种样本类型（有癌症细胞或无癌症细胞）的分布

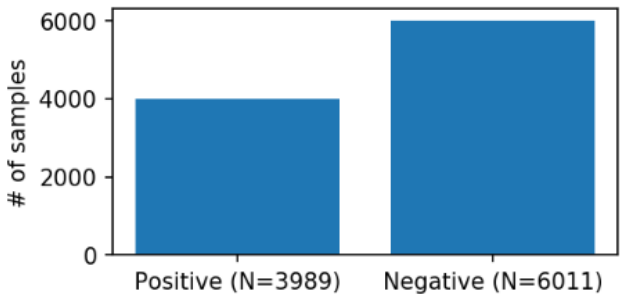


图 4 正负样本分布状态

由（图 4）可以看出，在训练集中负样本多于正样本，且正负样本数比例近似为 2：3。也就是全部判定为没有癌症细胞也能有 60%的准确率，这种样本分布可能导致分类器偏向于判断结果为负样本。为了避免分类器中的偏差并提升训练期间的稳定性，我们可以采取过采样和欠采样的方法

③ 观察图片的部分特征，比如 RGB 通道的分布，平均亮度等

i. RBG channel

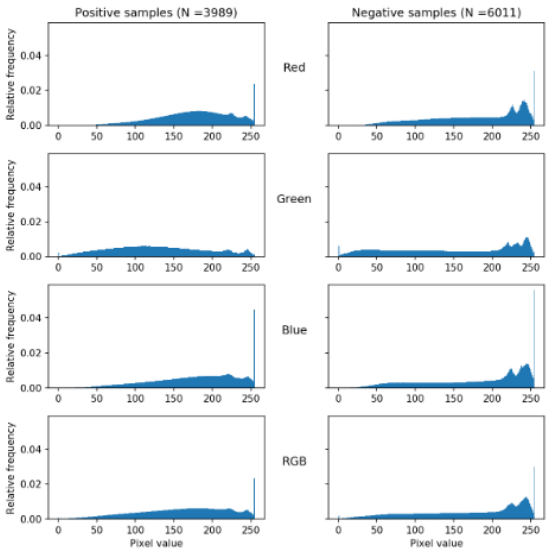


图 5 样本在 RGB 通道上的分布

由（图 5）可以看出：

1) 与正样本相比，负样本像素更多在高亮度区域，特别是在绿通道中

2) 正样本的绿通道中的像素相比其他两个通道分布在低亮度区域。

3) 负样本的三通道中的像素在高亮度区域均有峰值

4) 正负样本中都存在大部分亮白色区域

ii. 平均亮度

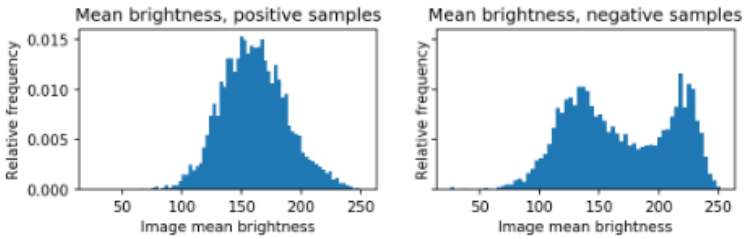


图 6 样本在 RGB 通道上的分布

由（图 6）可以看出：

- 正样本的平均亮度分布接近于均值为 150 正态分布
- 负样本近似双峰分布，峰值为 140 和 225

通过 EDA，我们可以得到以下结论：

① 正负样本在像素的分布和平均亮度的分布上有明显的区别，我们的模型可以利用这一点

② 一些图片包含了很亮的区域，可能是记录过程中的人为因素导致，我们需要找到一种方法解决他们。他们在正负样本中都有，所以不能简单当作一个特征

③ 负样本多于正样本很多，可能需要调整

3. 划分训练集与验证集

① 由于内存的限制，我们通过指定所选图片的索引来进行划分，而不用建立新的存储区。我们将 80%的数据用于训练，20%的数据用来验证我们的模型可以推广到新数据。因为两个标签中并没有十分罕见的类别，随机分割不会造成数据分类的减少。

② 为了避免之前数据排列的影响，我们重新对数据进行顺序。

训练模型

1. 建立神经网络模型

建立一个简单的卷积神经网络模型，它包含三个块，每个块中有 4 层：卷积层，组正交，池化和丢弃。在实验中，我们的神经网络模型如（图 7） 所示。Block1，Block2，Block3 结构相同，他们分别都由（卷积层、组正交、激活函数）、（卷积层、组正交、激活函数）、（池化层）和（dropout 层）组成。

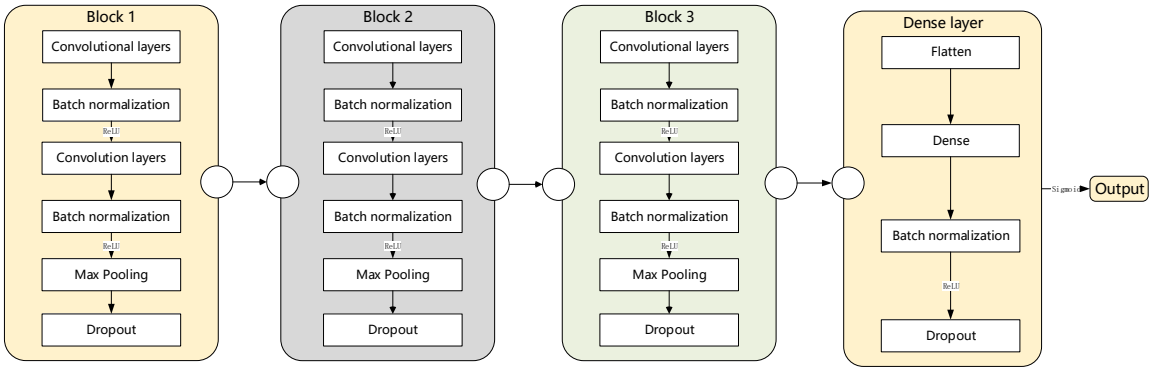


图 7 CNN 网络模型

a) 激活函数选择：可以看出我们的激活函数全部选用 ReLU，这样选择的原因是 ReLU 是一个分段函数，具有分段函数的性质，即在前传、后传、求导过程中都是分段线性的，这有助于解决深层网络收敛的问题，在深度较大的网络中，ReLU 还可以解决梯度消失的问题。但是要注意 learning rate 的选择。同等条件下，我们使用 tanh 和 sigmod 做了实验，实验的结果都不如 ReLU，原因是在一个深度较大的网络中，tanh 和 sigmod 会使梯度在反向传播中消失，层数越多，反而训练的误差越大。

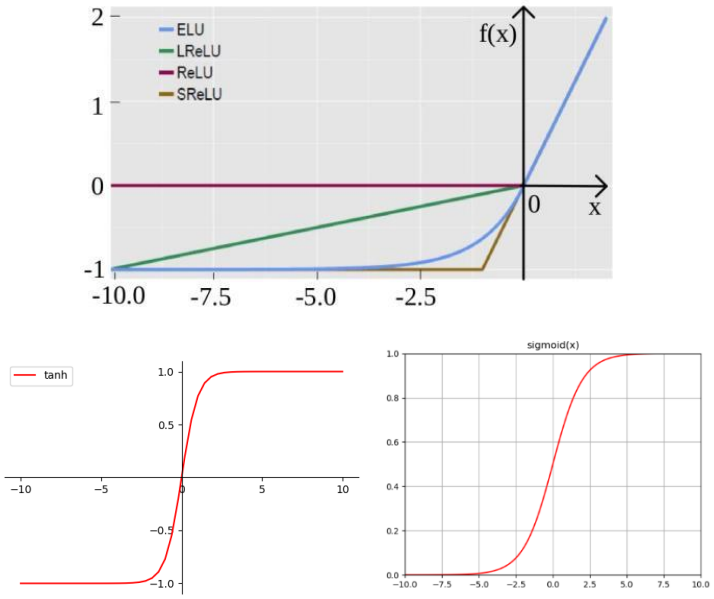


图 8 激活函数图像

b) 防止过拟合的措施：

(1) 池化层： 池化(Pooling)是在卷积神经网络中对图像特征的一种处理，通常在卷积操作之后进行。池化的目的是为了计算特征在局部的充分统计量，从而降低总体的特征数量，防止过度拟合和减少计算量

池化（上采样）缺点：

1) 内部数据结构的丢失

2) 空间层级化信息的丢失

3) 小物体信息无法重建（假设有四个 pooling layer 则 任何小于 2^4 = 16 pixel 的物体信息将理论上无法重建。）

(2) 空洞卷积：解决了池化层的弱点，在不做 pooling 损失信息的情况下，加大了感受野，让每个卷积输出都包含较大范围的信息

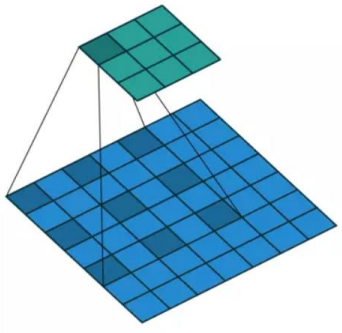


图 9 空洞卷积

(3) 选择适当的 dropout rate：放弃层(Dropout)是对该层的输入向量应用放弃策略。在模型训练更新参数的步骤中，网络的某些隐含层节点按照一定比例随机设置为不更新状态，但是权重仍然保留，从而防止过度拟合。实验中使用的数据集包含 20 万张图像，有很丰富的训练集，所以不必担心由于数据较少而带来过拟合的情况。但是随着训练层数的增加和网络的复杂度上升，训练模型会出现过拟合的情况。根据数据量以及网络层数，我们在 Block1~3 中选定 drop rate 为 0.3。对于全连接层选择 drop rate 为 0.5，这样可以在很大程度上减少计算量，并降低过拟合情况的发生

(4) 提前结束的策略：当模型训练次数过多时很有可能出现过拟合的情况，为了使模型更具普适性，在判断模型预测正确率平稳且在验证集上有较好的表现，为了防止过拟合，模型会在规定迭代次数前停止训练。

c) 优化函数

(1) 待优化参数：W(梯度),目标函数： f(w)，初始学习率 α 。

a) $g_t = \nabla f(w_t)$

b) 一阶动量： $m_t = \Phi(g_1, g_2, \dots g_t)$

c) 二阶动量： $V_t = \Psi(g_1, g_2, \dots g_t)$

d) 当前时刻下降梯度：$\eta_t = \alpha * m_t / \sqrt{V_t}$

e) 根据梯度下降进行更新：$w_{t+1} = w_t - \eta_t$

(2) SGD: Stochastic Gradient Descent,随机梯度下降

a) 特点：SGD 每次更新时对每个样本进行梯度更新，对于很大的数据集来说，可能会有相似的样本，这样 BGD 在计算梯度时会出现冗余，而 **SGD 一次只进行一次更新，没有冗余，可以新增样本。**

b) 公式：$\eta_t = \alpha * g_t$

c) 缺点：下降速度慢，可能会在沟壑两边持续震荡，停留在一个局部最优点

i. 改进：给 SGD 增加一个动量，使其冲过局部最优点：SDGM

(3) AdaGrad:

a) 特点：引入二阶动量，自适应的优化学习率

b) 公式：$V_t = \sum_{k=1}^{t} g_k^2$

c) 缺点：学习率单调递减至 0.可能会使训练提前结束，后续数据无法学到必要的知识

(4) Rmsprop:

a) 特点：改进了 adaGrad 学习率单调递减的问题，只关注过去一端是间窗口下降的梯度,避免二阶动量持续累积导致训练提前结束。

b) 公式：$V_t = \beta_2 * V_{t-1} + (1 - \beta_2) g_t$

(5) Adam:

a) 特点：同时使用一阶和二阶动量，收敛速度快，不用特别体调参

b) 缺点:没有精细调参,可能不会收敛,二阶动量在时间窗口内的累计，随着时间窗口变化，导致二姐东连变化，在后期引起学习率的震荡。

i. 可以通过限制学习率改善

c) 错过全局最优解：自适应算法可能会对前期出现的特征过拟合，后期出现的特征很难纠正前期的拟合效果。

i. 可以通过前期使用 adma，后期使用 sgd 的方法改善

(6) 优化模型处理：

a) 使用小数据集验证算法：随机梯度下降算法和数据及大小关系不大

b) 考虑不同的算法组合

c) 数据集充分打散

d) 训练过程中持续监视验证集和训练集

e) 只当合适的学习率衰减策略

d) 配置模型训练时的参数

(1) batch_size 是训练一个神经网络时很关键的参数。在训练的时候，我们将训练集分成一个个的 batch，然后用 batch 一个个的去训练我们的网络

(2) 卷积核大小：卷积核的大小选择参考 VGG 的 3*3 卷积核。选取该大小卷积核的原因如下：

1) 3*3 是最小能够捕捉像素八领域的尺寸

2) 两个 3*3 的堆卷积感受野是 5*5，三个 3*3 的堆卷积感受野是 7*7，所以可以通过小尺寸的卷积核去替代大尺寸的卷积核

3) 多个 3*3 卷积层比一个大尺寸的 filter 有更少的参数

2. 训练并且验证模型

每个 batch 训练多次，每次训练的时候对 batch 中数据的进行重排。

训练模型 ResNet

ResNet 的原理

ResNet 解决的是上述内容提到的当网络的层数增加时，由于反向传播不能进行而造成的训练误差增大的问题，即将输出可以再次与输入进行会面得出正确的引导梯度。深度残差网络(ResNet)由很多个“残差单元”组成。每一个单元(如图 X)可以表示为

$$y_l=h(x_l)+F(x_l,W_l) \quad (1)$$

$$x_{l+1}=f(y_l) \quad (2)$$

其中 x_l 和 x_{l+1} 是第 l 个单元的输入和输出， F 表示一个残差函数。h(xl)=xl 代表一个恒等映射，f 代表 ReLU。

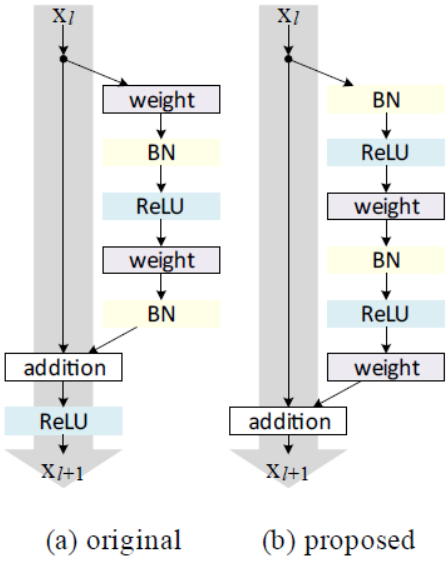


图 10 ResNet 原理图

其中 xl 是第 l 个残差单元的输入特征。$W_l=\{W_{l,k}|_{1\leqslant k\leqslant K}\}$ 是一组与第 l 个残差单元相关的权重(偏置项)，K 是每一个残差单元中的层数。F 代表残差函数。函数 f 是元素相加后的操作，函数 h 被设置为恒等映射：$h(x_l)=x_l$。

如果 f 也是一个恒等映射：$x_l+1\equiv y_l$，我们将两个等式进行整合，可以得出：

$$x_{l+1}=x_l+F(x_l,\ W_l) \quad (3)$$

通过递归 (例如，$x_{l+2}=x_{l+1}+F(x_{l+1},W_{l+1})=x_l+F(x_l,W_l)+F(x_{l+1},W_{l+1})$。)，对于任意深的单元 L

和任意浅的单元 l，可以得到一下等式：

$$x_L=x_l+\sum F(x_i, W_i), \quad (4)$$

等式（4）展现了一些良好的特性。(i) 对于任意深的单元 L 的特征 x_L 可以表达为浅层单元 l 的特征 x_l 加上一个形如 $\sum F$ 的残差函数，这表明了任意单元 L 和 l 之间都具有残差特性。(ii) 对于任意深的单元 L，它的特征 $x_L=x_0+\sum F(x_i,W_i)$，即为之前所有残差函数输出的总和(加上 x0)。

等式（4）同样也具有良好的反向传播特性。 假设损失函数为 E，从反向传播的链式法则可以得到（5）：

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l}=\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l}=\frac{\partial \mathcal{E}}{\partial \mathbf{x}_L}\left(1+\frac{\partial}{\partial \mathbf{x}_l}\sum_{i=l}^{L-1}\mathcal{F}(\mathbf{x}_i,\mathcal{W}_i)\right) \quad (5)$$

等式(5)表明了梯度$\partial E/\partial x_l$ 可以被分解成两个部分：其中$\partial E\partial x_L$ 直接传递信息而不涉及任何权重层，而另一部分$\partial E\partial x_L(\partial \sum F/\partial x_l)$表示通过权重层的传递。$\partial E\partial x_L$ 保证了信息能够直接传回任意浅层 l。

等式(5)同样表明了在一个 mini-batch 中梯度$\partial E\partial x_l$ 不可能出现消失的情况，因为通常$\partial \sum F/\partial x_l$ 对于一个 mini-batch 总的全部样本不可能都为-1。这意味着，哪怕权重是任意小的，也不可能出现梯度消失的情况。

所以经过以上分析，如果想要通过增加层数和网络复杂度来提升训练的准确度，可以使用 ResNet 来避免由于层数增加而无法将梯度反向传播的问题。

ResNet 网络模型

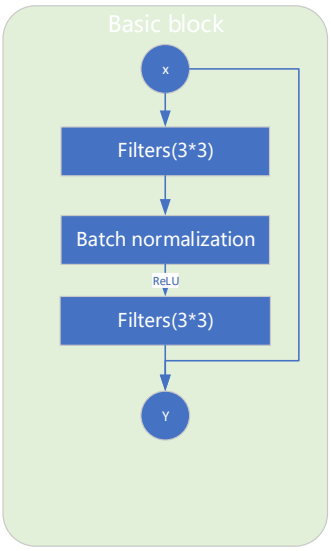


图 11 ResNet Basic Block 模型

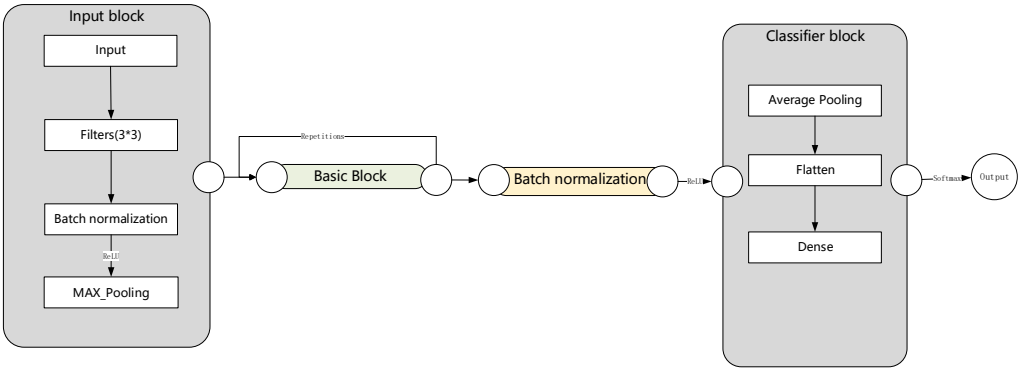


图 12 ResNet 网络模型

三、实验过程

1. 加载数据：因为数据集较大，训练时训练 10000 张图片，占据 RAM 4.7G，这种大小使得数据集并不能被随意的计算，因为 kaggle 的内存仅有 13G 在实验过程中第一次遇到了由于数据较大而产生的内存溢出问题。

   ① 在此遇到数据训练是函数选择的问题，虽然在 keras 上已经有现有的训练函数，但是在使用过程中遇到了问题。

   i. Fit :适用于数据集较小，他是将一整个数据集一次性加载到内存中进行运算，这在之前的实验中也经常被我们采用

   ii. .fit_generator 操作：

   a) Keras 调用提供给.fit_generator 的生成器函数（在本例中为 aug.flow）

b) 生成器函数为.fit_generator 函数生成一批大小为 BS 的数据

c) .fit_generator 函数接受批量数据，执行反向传播，并更新模型中的权重

d) 重复该过程直到达到期望的 epoch 数量

iii. .train_on_batch：接受单批数据，执行反向传播，然后更新模型参数

② 我们在 resnet 中选用了 fit_generator 来训练我们的模型

2. 训练集上的实验结果：

① 在建立神经模型时提过，我们的激活函数全部选用 ReLU。我们在 10000 张图片的训练集上做了实验，ReLU 作为激活函数时，实验在训练集上的经过 5 个回合，训练的准确度可以达到 84%，如（图 13）所示。若将激活函数更改，变为 tanh，进行同样条件的实验，在 5 个回合的训练之后，训练的准确度明显没有 ReLU 的准确度高，并且收敛的速度没有 ReLU 快，如（图 14）所示。

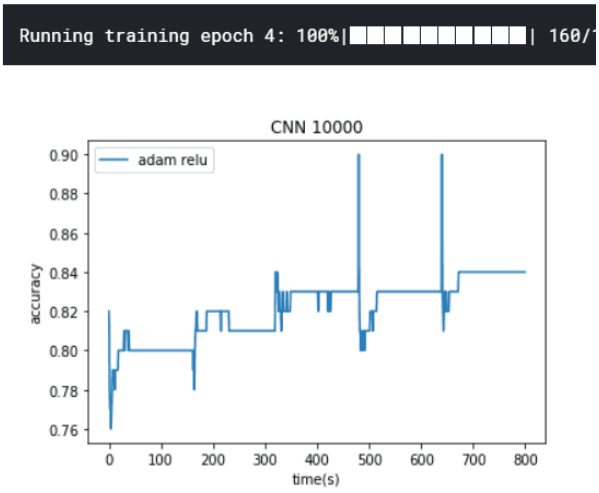


图 13 CNN 训练集训练结果 ReLU

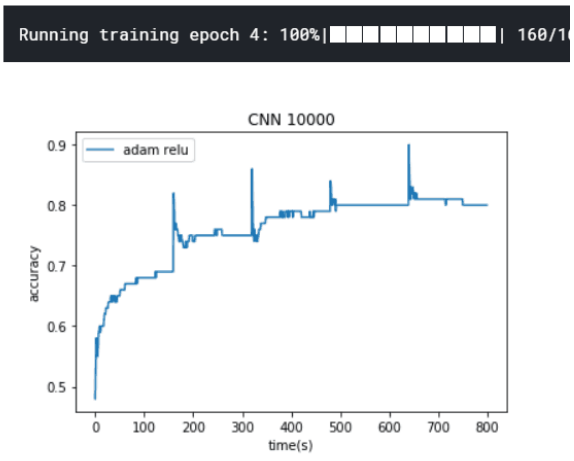


图 14 CNN 训练集训练结果 tanh

② 我们尝试使用空洞卷积来代替原来的卷积层和池化层，两模型的结果分别如图 15 和 16 所示。

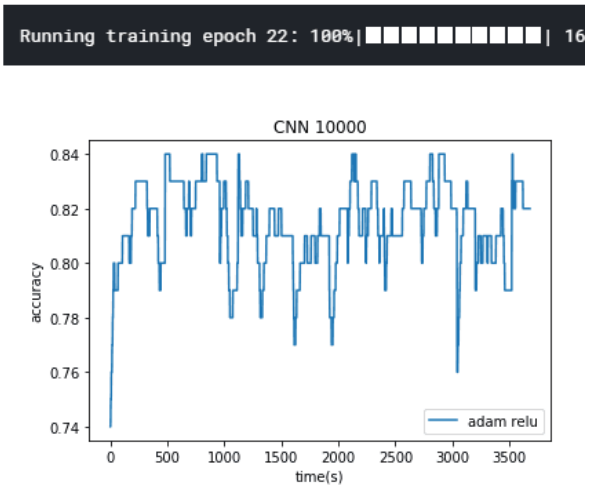


图 15 原始 CNN 验证集验证结果以及准确度随时间的变化曲线

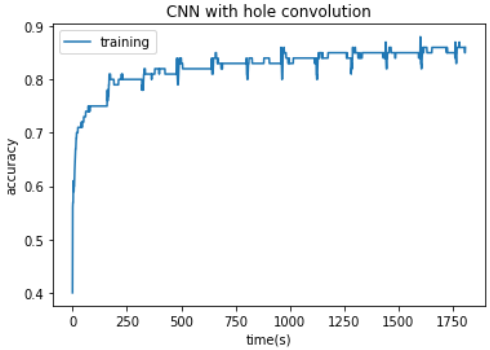


图 16 空洞卷积 CNN 验证集验证结果以及准确度随时间的变化曲线

由结果可以看出使用空洞卷积，网络准确率要比原来的要高，并且更稳定。

④ 我们更改网络层数，并观察他们在训练集和验证集上的表现，如图 17，18 所示。

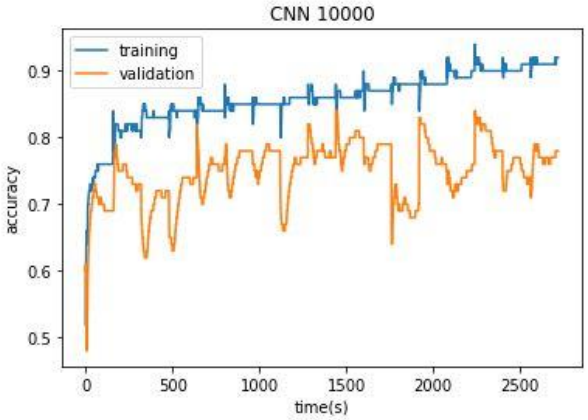


图 17 11 层网络准确率

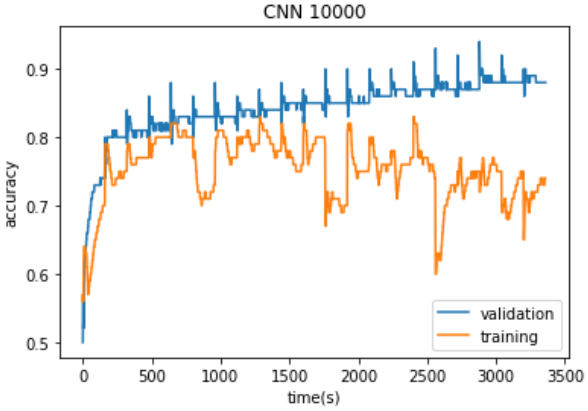


图 18 20 层网络准确率

可以看到，两个网络在训练集上，100 次左右时，准确率已趋于平稳，11 层的网络在 0.9 左右，20 层的网络在 0.85 左右，而验证集准确率较低。这说明，经典神经网络容易导致过拟合的现象出现，需要调整。而且层数越高，表现得似乎越差。

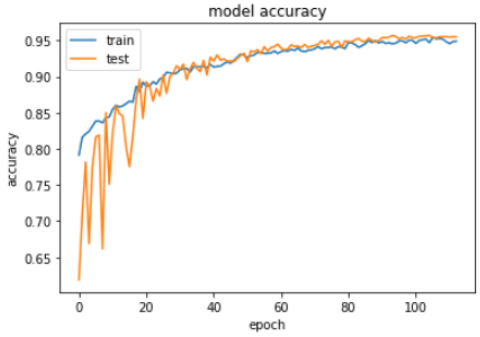
四、对比实验

由 CNN 的实验结果可以看出，CNN 对于这个模型已经有了很好的效果，准确度可以达到 90%以上。但是为了更高的提高准确度，一个直接有效的方法就是增加网络的层数。但是这样就引发了一个问题。是否层数越多准确率就会越高呢？答案是情况并不是这样的理想。在当前的模型中 CNN 使用了 11（3 block*3 layer+2 output）层神经网络，但是当训练的层数达到了 20（3*6+2）层，甚至更多的层，那么这样的多层 CNN 就会暴露出很多的弊端。例如，层数的增加以及网络模型的复杂，会导致梯度反向传播变得困难甚至梯度在反向传播中消失，使得训练的误差变大。所以为了更好的提高训练的准确度并且改变由于网络层数增加带来的反向传播梯度消失的缺陷。我们使用了 HaiMing He 提出的 ResNet。

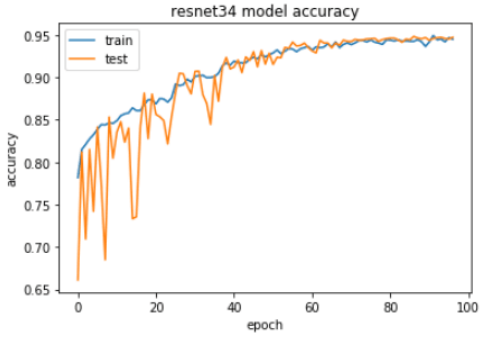
ResNet 实验结果

1）block 数对比

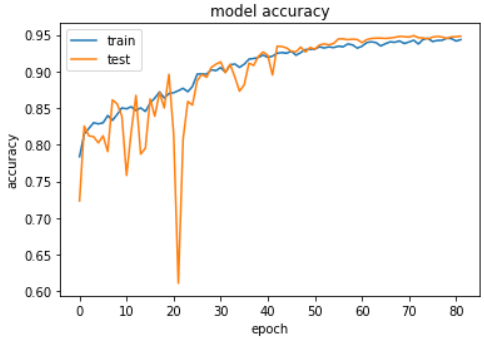
我们通过改变 ResNet 中网络的 block 数，观察这对结果的影响。



a） block=18



b） block=34

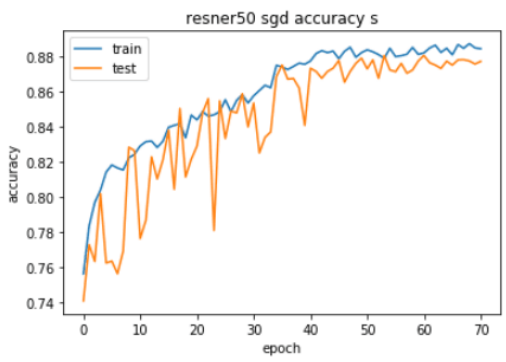


c） block=50

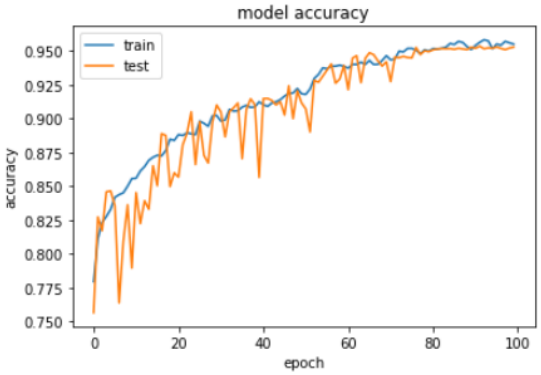
图 19 ResNet block 数变化时，training 和 test 的模型准确度

由（图 19）所示，当 block 数为 18 时，模型的收敛速度较 block 数为 50 时的收敛速度慢。并且在 block 数为 18 时，模型会在第 113 次训练时停止训练，对于 block 数为 34 时，模型会在 97 次时停止，对于 block 数为 50 时，模型会在第 80 次训练时终止训练。可见，层数越深，收敛速度越快，并且，由于 resnet 的特性，层数的加深并没有对数据的准确性产生很大的影响，最终所有函数都向 0.95 收敛。

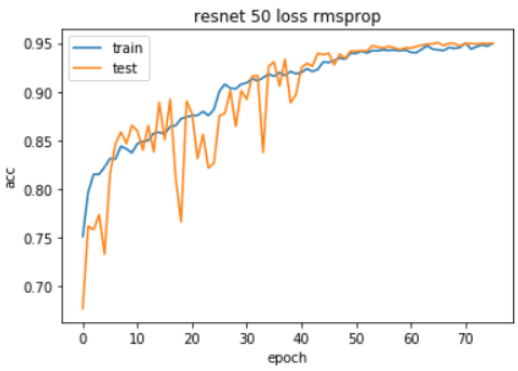
2）优化函数对比：



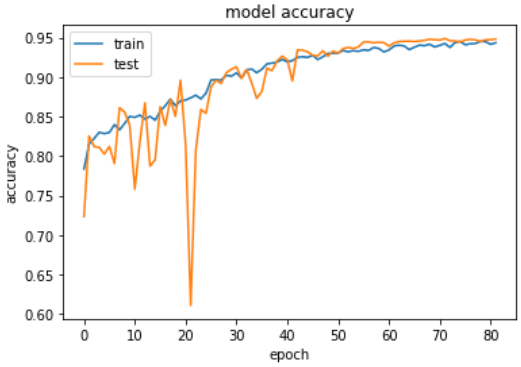
a） 优化函数 sgb



b)优化函数 adagrad



c)优化函数 rmsprop

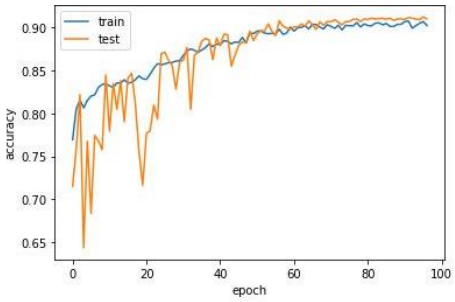


d)优化函数 adam

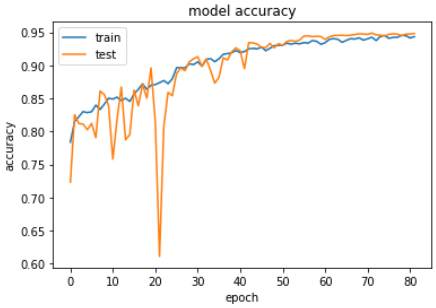
图 20 使用不同优化函数时，training 和 test 的模型准确度

可以看到，adam 的效果更好，曲线更加平滑，在 50 次左右迭代时就已经基本趋于平稳，adgrad 在训练区间内一直都有抖动的情况出现，且较为频繁，rmsprops 在初期抖动而后期区域平缓，收敛速度最快，adam 也有相似情况但是抖动要少很多， sgb 的收敛速度较慢，波动较大，结束训练后也仍然并不是十分平稳。由于步长等参数并没有特别调整过，可以看出，adam、adgrad、adam 自适应性较好。

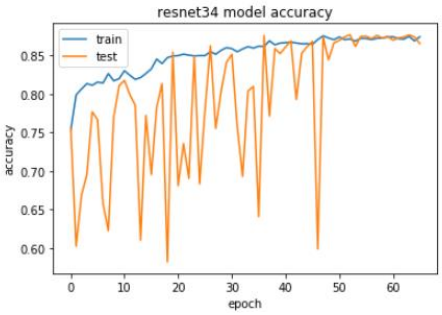
3）激活函数对比



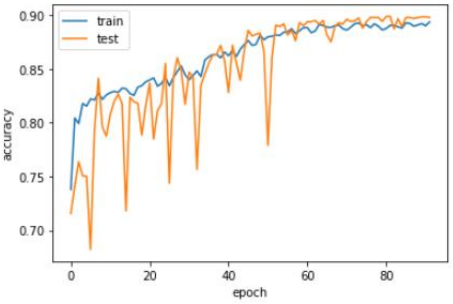
a)Elu



b) relu



c)Sigmoid

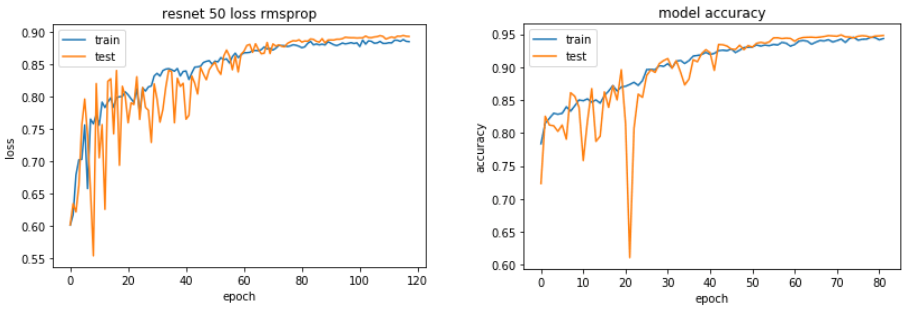


d)Tanh

图 21 使用不同激活函数时，training 和 test 的模型准确度

elu 波动较大，收敛较 relu 稍微慢一些，可以看出，整个函数保留了负数区域的特征，但是准确率较低，可能保留了不必要的特征。也证明 elu 并不总是有着比 relu 更好的表现，尤其是在这种简单分类问题上。Sigmoid 十分剧烈且最后准确率较低，初步判断是由于梯度爆炸导致训练提前终止，效果非常明显的不如 relu，tanh 则有相似的表现。

4）对比卷积方法



a)使用空洞卷积　　b)不使用空洞卷积

图 21 使用不同卷积方法时，training 和 test 的模型准确度

发现在 ResNet 中使用空洞卷积时效果没有不使用好。这有可能是因为空洞卷积替换了池化层，但是在 ResNet 中存在最大池化和平均池化两种池化层，用空洞卷积进行替换时，可能会影响原来网络的表现。

五、实验感想

这个实验上手难度不是很大，我们前期通过研究 Kaggle 上别人的代码，了解了如何读取这个数据集，并将数据集打印出来，但是由于我们从来没有接触过与癌症细胞相关的知识，看见了图片和分类也并不知道癌症细胞与非癌症细胞之间的区别。于是我们将运用数字图像处理中学到的一些知识简单解了正负样本大致的分布。之后我们学习到运用 keras 的库函数，我们可以较快的搭建好神经网络，于是我们开始了尝试。这期间也了解到 CNN 的基本框架，最终按照框架搭建起来了自己的神经网络。为了我提高网络的性能，我们将一些传统数字图像处理的方法运用了进去，例如旋转平移变换，增强亮度，裁剪等方法，但是这些方法对于最终准确率没有很好的效果。

后来我们学习了残差网络，这是一种可以使网络更深还能保证当网络深度超过最优值时，还能保持最好的性能。我们模仿 GitHub 上对于 Cifar-10 数据集搭建起来的一个残差网络建立了我们的第二个网络。后期我们对这两个网络进行调参对比，比如调整网络的深度，激活函数，优化函数等，观察它们对最后结果的影响。我们还尝试修改卷积的方式，用空洞卷

积来代替普通的卷积，并去掉池化层，这样可以避免池化对于原来图像特征的影响。在这些过程中，我们更加感受到神经网络的奇妙之处。