基于 fastai 的糖尿病视网膜病变检测

陈骋 2017202021

信息学院

日期：2020 年 7 月 4 日

摘　　要

本文借助基于 pytorch 实现的深度学习框架 fastai，并借助 albumentations 提供的图像增强技术以及 Nvidia 提供的 gpu 计算框架 cuda，在眼底图像上构建卷积神经网络模型并进行学习。最终使用训练成功的模型实现各个任务的需求。

关键词: fastai，图像增强，卷积神经网络，squeezenet

1　任务背景

1.1　糖尿病视网膜病变

糖尿病是严重影响人类健康的内分泌疾病残致死率仅次于心脑血管疾病及癌症; 其发病机制是由于胰岛素相对或绝对不足，或靶细胞敏感性下降造成糖、蛋白质、脂肪代谢失调和水电解质紊乱。其发病原因及机制尚不明确，与遗传、环境因素及其相互作用有关。临床上分为胰岛素依赖型即一型糖尿病、非胰岛素依赖型即二型糖尿病。一型糖尿病大多数为青少年发病。二型糖尿病占所有糖尿病患者的 9 0% 以上，多见于 40 岁以上者。糖尿病可引起全身多系统慢性并发症，全身性小血管和微血管病变是糖尿病常见的病理改变。糖尿病性视网膜病变 (diabetic etinopathy,DR) 是糖尿病的眼部严重并发症，随着糖尿病患者的日趋增多，DR 已成为目前特别是发达国家 20 74 岁成年人致盲的首要原因。 [1]

1.2　fastai

fastai 是一款基于 pytorch 搭建的深度学习框架，该框架下集成了许多的深度学习中的操作，因此只需调用几个函数即可完成其他框架需要大量代码才能完成的工作。并且由于 fastai 是基于 pytorch 构建的，因此可以使用 Nvidia 提供的 cuda 进行 gpu 计算，从而能显著提升计算效率。

1.3　albumentations

albumentations 是 python 的一个图像数据处理库，具有以下特点:

1. 基于高度优化的 OpenCV 库实现图像快速数据增强
2. 针对不同图像任务，如分割，检测等，超级简单的 API 接口
3. 易于个性化定制

4. 易于添加到其它框架，比如 PyTorch

其中 albumentations 自带的数据增强方法有模糊、翻转、归一化、旋转、Gamma 变换、缩放、对比度首先自适应直方图均衡化等

2 实验环境

2.1 硬件环境

cpu : Intel i7-7700HQ

gpu : Nvidia Geforce Gtx 1050Ti

内存: 16 GB

2.2 软件环境

操作系统: Windows 10 家庭版

python 版本: 3.7.0

pytorch 版本: 1.5.0

torchvision 版本: 0.6.0

fastai 版本: 1.0.61

albumentations 版本: 0.4.5

cuda 版本: 10.2

3 任务描述

3.1 任务一

对于给定的测试集图像进行 DR 分级，DR 的分级结果为 0-4，将测试图像的图像名以及具体的 DR 分级结果输出到预测文件中。该任务为一个多分类任务。

3.2 任务二

对于给定的测试集图像，进行 DR 转诊预测，当 DR 的分级达到 2 以上时，进行转诊，否则不需要转诊。其中转诊分类为 1，不转诊断分类为 0，并将图像名及分类结果输出。该任务为一个二分类问题。

3.3 任务三

对于给定的测试集图像，检测是否存在与 DR 直接或间接相关的病灶。将图像与存在的病灶信息按照概率递减的顺序输出到预测文件中。该任务为一个多标签的二分类问题。

4 病变检测

4.1 图像增强与预处理

4.1.1 使用图像增强的原因

通过观测原始图像信息，可以看到存在部分图案存在过暗或过亮的现象。在此类图片中视盘、血管等特征信息不明显，如不对其进行处理，则会对后续模型产生负面的影响。下图为原始图像中一张过暗的图片以及一张过亮的图片。

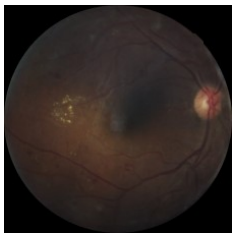

图 1: 过暗图像

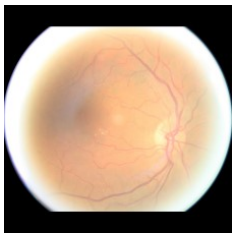

图 2: 过亮图像

4.1.2 具体方法

1. 高斯模糊
2. 对比度受限的自适应直方图均衡化
3. 伽马变换

上述方法使用 albumentations 库中的 GaussianBlur()，CLAHE()，RandomGamma() 进行实现。通过高斯模糊能减小原始图像中噪点对模型的影响，使用对比度受限的自适应直方图均衡化以及伽马变换方法能保证原始图像在对比度变化不大的情况下实现原始图像的亮度增加或者减小。

4.1.3 效果

下图中的图 3 与图 4 为图 1 与图 2 在通过上述图像增强方法进行处理后得到的新的图像

可以看到，相较于图 1 与图 2，经过图像增强操作后得到的新的图像较原始图像更为清晰可见。其中血管和试盘等特征信息较原始图像更为明显

图 3: 图 1 处理后的图像

图 4: 图 2 处理后的图像

4.1.4 图像预处理

由于原始数据集中的数据量较少，为了得到预测效果较好的模型，有必要增加数据集的数据量。而原始数据中的眼底图片大多仅有一侧的图片信息，因此将原始图像进行水平翻转并加入到原始数据中组成新的训练数据。而眼底图像的各项特征信息与位置没有关系，因而进行水平翻转显然不会提供错误的图片信息。

将经过图像增强以及水平翻转后的图像信息进行存储，并且读取原始数据的各项标签信息，根据三个不同的任务构造不同的标签文件，而由于 fastai 的多标签分类不允许标签为空，因此当原始图像中不存在病灶信息时，新增一个标签表示无病灶信息。本实验采用将测试集与训练集合并，按照一定比例重新划分的方法，因此原始的测试数据与预测数据进行处理后合并为一个新的数据集。

4.2 模型选择与训练

4.2.1 模型的选择

在实验的初期采用的模型为 k 近邻模型，得到的预测效果并不理想，主要原因便在于 k 近邻模型对于原始数据的数据格式有较高的敏感性，表现在图像处理领域上便是对图像特征的空间分布具有较高的敏感性。而卷积神经网络模型是十分适合于进行图像处理的模型，利用卷积神经网络的卷积层、池化层和全连接层等基本结构，就可以让这个网络结构自己学习和提取相关特征，并加以利用。这种特性对许多研究提供了许多便利，可以省略过往非常繁杂的建模过程。 [2] 因此此次病灶检测实验使用 fastai 中的卷积神经网络模型进行训练与预测。fastai 中集成了下列几种已预训练的卷积神经网络模型：

1. alexnet
2. densenet
3. resnet
4. squeezenet

5. vggnet

其中 densenet，resnet 以及 vggnet 属于规模较大的网络模型，而 alexnet 以及 squeezenet 属于规模较小的网络模型。而此次病变检测的数据集较小，因而 alexnet 以及 squeezenet 这类较小的网络更适合；而 densenet 等大型网络的训练消耗较大，且受限于数据集的数量，预测的效果较为一般，且容易出现过拟合现象。下图中的图 5 为 vggnet 在数据集上对任务一进行训练的效果，图 6 为 squeezenet 的训练效果

25	0.771477	0.724675	0.725490	0.645815	01:16
26	0.748599	0.709977	0.705882	0.573991	01:15
27	0.727117	0.744845	0.692810	0.608743	01:15
28	0.764293	0.763727	0.676471	0.633891	01:15
29	0.744711	0.746797	0.679739	0.638928	01:16
30	0.727857	0.693791	0.725490	0.684667	01:15
31	0.704108	0.737352	0.696078	0.692111	01:16
32	0.684519	0.769549	0.683007	0.588459	01:15
33	0.735176	0.767874	0.679739	0.543919	01:15
34	0.739270	0.787536	0.663399	0.645161	01:15
35	0.730948	0.740780	0.689542	0.583055	01:16

图 5: vggnet 训练效果

92	0.432122	0.546879	0.800000	0.757555	00:20
93	0.425987	0.515677	0.800000	0.767705	00:20
94	0.423460	0.547601	0.796078	0.746544	00:20
95	0.433310	0.547650	0.811765	0.793989	00:20
96	0.423016	0.518052	0.827451	0.830099	00:20
97	0.440513	0.538620	0.811765	0.754718	00:20
98	0.436476	0.536680	0.796078	0.741391	00:20
99	0.438847	0.580765	0.780392	0.694756	00:20

图 6: squeezenet 训练效果

上述图像中的各列分别为训练轮次，训练集损失、预测集损失、准确率、二次加权 kappa、单轮次训练时间。可以看到在经过 30 轮次左右的训练后，vggnet 的准确率无法继续升高，且观察损失信息可以看到在训练集损失值降低的情况下，预测集上的损失出现了不降反升的情况，显然出现了过拟合的问题。而 squeezenet 经过了 100 轮次的训练过后，准确率较 vggnet 更高，且各项指标都基本稳定。

并且 vggnet 单轮次的训练需要 1 分 20 秒左右，而 squeezenet 的训练仅需 20 秒左右。并且在 batch_size 为 16 时 vggnet 的显存占用值为 2.7GB 左右，而 squeezenet 仅为 1.2GB 左右。且 vggnet 的 batch_size 最大为 16，而 squeezenet 能设置为 64。所以 squeezenet 更容易收敛到一个较好的结果。

因此此次病灶检测实验采用了 squeezenet 模型进行训练以及预测，而不采用 vggnet 等大规模的模型。

4.2.2 数据的读入

在对模型进行训练之前，先对图像数据进行读取。

1. 首先读取测试数据的标签文件并建立 dataframe

2. 通过 fastai 中的 ImageList 类提供的数据读入方法 from_df 从指定路径和 dataframe 中读入图像信息
3. 使用 label_from_df 方法从 dataframe 中读取各个图像的标签，其中任务一与任务二的标签即为原始信息，任务三的标签需要设置该方法对标签进行切割从而得到多个标签信息
4. 对数据按照 0.12 的比例进行随机划分
5. 使用 fastai 的 transform 方法对图像进行缩小以及随机图像增强操作
6. 设置数据的 batch_size 大小并对每个图像进行标准化处理

4.2.3 模型的构建与训练

在读入训练数据后即可基于不同任务的数据构造相应的学习器。

fastai 的 vision 库中已封装了各种关于图像处理以及卷积神经网络的各项操作。使用 cnn_learner 函数并设置数据、初始模型、评测指标等参数即可完成卷积神经网络学习器的初始化设置。

在对训练器进行初始化后便可通过 vision 模块中的相应函数寻找适合的学习率。随后应用该学习率对学习器进行 fit 操作即可进行多轮次的训练，并且 fastai 会自动将每一轮次的训练得到的损失值、评测指标以及单轮次训练的时间输出。通过输出的结果即可清楚地了解该模型在预测集上的性能指标。

4.2.4 模型的效果

下面的图 7-图 9 为 squeezenet 在预测集上关于任务一到任务三的各项评测指标。

92	0.432122	0.546879	0.800000	0.757555	00:20
93	0.425987	0.515677	0.800000	0.767705	00:20
94	0.423460	0.547601	0.796078	0.746544	00:20
95	0.433310	0.547650	0.811765	0.793989	00:20
96	0.423016	0.518052	0.827451	0.830099	00:20
97	0.440513	0.538620	0.811765	0.754718	00:20
98	0.436476	0.536680	0.796078	0.741391	00:20
99	0.438847	0.580765	0.780392	0.694756	00:20

图 7: 任务一训练效果

25	0.373909	0.374812	0.820261	0.901669	0.843137	00:24
26	0.360346	0.361107	0.803922	0.904252	0.784314	00:24
27	0.364401	0.358037	0.816993	0.909704	0.876906	00:24
28	0.375955	0.379175	0.810458	0.898465	0.860566	00:24
29	0.375178	0.356193	0.823529	0.912478	0.901961	00:24

图 8: 任务二训练效果

其中前三行为训练轮次、训练集上的损失值、预测集上的损失值，最后一行为单轮次训练所用时间，其余各行为自定义的评测指标。其中任务一的设置的评测指标为准确率和二次加权 kappa，任务二为准确率、auc 下的 roc 面积、f1score，任务三为多标签分类的 fbeta 分数。

25	0.220704	0.226487	0.865039	00:22
26	0.218298	0.222438	0.872934	00:21
27	0.213726	0.217648	0.878169	00:21
28	0.213877	0.218751	0.870757	00:20
29	0.211675	0.214258	0.873078	00:20

图 9: 任务三训练效果

可以看到 squeezenet 通过多轮次的训练过后，在各个任务的预测集上都有不错的表现，并且模型通过 cuda 进行各项计算，相较于通过 cpu 进行计算所花费的时间大大减少。同时相较于 vggnet 等大型网络，squeezenet 能够更快、更好地收敛。

4.3 预测结果输出

4.3.1 任务一

任务一的需求为输出测试数据的 DR 分级，因此将训练完成的模型对测试集中的数据进行预测，并将预测得到的 DR 分级结果与图像名称一并输出到预测文件中。

4.3.2 任务二

任务二的需求为输出测试数据的分诊情况以及概率，因此使用训练完成的模型对测试集中的数据进行预测，将预测得到的分诊结果以及该分诊结果对应的概率一并输出到预测文件中。

4.3.3 任务三

任务三的需求为将测试数据中存在的病灶信息输出，如果没有任何病灶则仅输出图像名称。因此使用训练完成的模型对测试集中的数据进行预测，得到数据中存在的病灶标签以及所有标签对应的概率值。如果预测得到的标签为空，则认为没有病灶信息；否则将各个标签按照出现的概率值进行降序的排序，如果无病灶标签出现的概率最高，则同样认为没有病灶信息，如果不是，则将出现概率 >0.5 的病灶标签依次输出。

5 总结

5.1 实验总结

本次病变检测实验采用了通过 fastai 构建卷积神经网络模型，在对图像进行图像增强、图像翻转等预处理后使用对模型进行多轮次的训练。并将训练得到的模型按照不同的需求输出不同的预测结果。通过训练中的评测指标以及测试集上的结果可以看到本次实验的效果还是较为不错的，通过整个实验对各种方法的运用也加深了对图像处理以及深度学习的了解，因此整个实验时比较成功的。

同时通过此次实验也能感受到显卡在向量的计算上的明显优势，在本实验中显卡的计算性能相较于 CPU 能够提升两个数量级左右。因此显卡在机器学习当中充当着一个重要的角色。

5.2 不足

虽然整个实验的结果较为成功，但实验中还是存在如下不足的地方：

1. 受限于实验环境的算力，本次实验仅使用了规模较小的模型进行训练与预测。如果能够对 fastai 自带的所有模型进行训练，并依据模型的效果按照一定的置信度对测试数据进行预测，最后集成所有模型的预测结果作为最终的预测结果，那么最后得到的效果可能会更好
2. fastai 对各种操作都进行了封装，让使用者都能使用简单的几行代码进行模型的学习。但正是因为封装过深，其中的许多细节部分难以手动的更改，因此在任务三中模型对于多标签任务的分类使用的决策为概率 >0.5。所以更改决策来观测模型在预测集上的预测效果便难以实现。故而此次实验在测试集上的决策边界同样是 p>0.5
3. 通过查阅其他人的实现方法，发现设立伪标签以及使用测试时增强 (TTA) 等方法也能提升模型的预测效果，但由于 fastai 是一个比较新的框架，因此许多操作没有教程，仅能通过官方文档查阅各种 API，因此实现难度较大，故此次实验未能完成此类数据处理方法

参考文献

[1] 刘卫, 张勇进. 糖尿病性视网膜病变[J]. 国外医学. 眼科学分册, 2005(05):74-79.

[2] 卢宏涛, 张秦川. 深度卷积神经网络在计算机视觉中的应用研究综述[J]. 数据采集与处理, 2016, 31(01):1-17.