机器学习(进阶)纳米学位毕业项目

侦测走神司机项目报告

报告人：鱼 群

2018 年 6 月 10 日

1 问题的定义

1.1 项目概述

在开车时，你是否遇到过这些情况：交通灯已经变绿，但前车依然纹丝不动；正常行驶的前车，突然毫无征兆地减速并且开始左右跑偏。当你驾车超过出现上述情况的车辆时，你希望看到什么？如果你看到前车司机正在看手机(打字、沉迷于社交网络)或打电话，你一定不会感到奇怪吧。

美国疾病控制和预防中心的研究结果表明，有五分之一的交通事故是由驾驶者注意力不集中导致的，也就是说，驾驶者的分心行为会致使每年 3,000 人死亡、42,5000 人受伤。

美国“State Farm”保险公司希望通过安装在仪表盘上的摄像头来自动侦测驾驶者的分心行为并进行安全提醒，以此降低上述令人震惊的伤亡数据，同时为客户提供更好的保险服务。以仪表盘摄像头拍摄的 2D 图像作为数据集，“State Farm”保险公司希望能通过这些图像训练得到分类模型，用以区分驾驶者的正常驾驶行为和各类分心行为，这一问题也成为 Kaggle 在 2016 年的竞赛题目之一。侦测驾驶者分心行为的 Kaggle 竞赛虽已结束，但解决该问题所涉及的相关技术仍值得研究，本项目将进行与此相关的工作。

1.2 问题陈述

本项目为**监督学习**范畴内的图像多分类问题，驾驶者的行为状态包括 10 类：

- c0: 安全驾驶
- c1: 右手打字
- c2: 右手打电话
- c3: 左手打字
- c4: 左手打电话
- c5: 调收音机
- c6: 喝饮料
- c7: 拿后面的东西
- c8: 整理头发和化妆
- c9: 和其他乘客说话

这 10 类状态中只有 c0 为安全驾驶状态，其余均为驾驶者的分心行为。针对项目要求，需要解决的问题包括以下 2 个方面共 7 个要点：

Ⅰ） 使用深度学习方法构建分类模型并依此侦测驾驶者的状态，即输入 1 张彩色图片，输出 10 种状态(c0 - c9)的概率：

1) 数据分析

2) 数据预处理

3) 训练集、验证集、测试集划分

4) 分类模型构建

5) 分类模型训练及调参

6) 利用分类模型预测状态及模型评价

II） 可视化，对模型预测结果进行可视化处理，并解释其中的原因：

7) 可视化预测结果并解释原因

1.3 评价指标

本项目为多目标分类问题，Kaggle 上的评价指标采用 Multiclass Logarithmic Loss，也称 Cross Entropy(交叉熵)，计算公式为：

$$loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

式中，N 为测试集图像数量，M 为分类标记数量；y_{ij} 当测试样本 i 的真实分类属于 j 类时为 1，否则为 0；p_{ij} 为预测的样本 i 属于各分类(j)的概率。

交叉熵描述了预测分类的不确定程度，该值越小说明模型的预测分类能力越强。由信息论相关概念及计算式可知，分类模型预测越准确，交叉熵越小，极好情况下，每次预测都准确(确定)，p_{ij} 对应于正确分类时的概率值为 1，此时交叉熵为 0；反之，极坏情况下，p_{ij} 对应于正确分类时的概率值趋于 0，此时交叉熵为无穷大。

本项目训练得到的分类模型致力于在测试集上得到更小的交叉熵。

2 分析

2.1 数据的探索

本项目用到的数据来自 Kaggle 官网，包括 2 个压缩文件及 1 个 Excel 文档，其中一个压缩文件包含仪表盘摄像头拍摄的训练集、测试集图片，图片清晰可辨；另一个压缩文件以 Excel 表格形式列出了训练集图片的相关信息，包括图片产生的驾驶者 ID、图片分类、图片名称；1 个 Excel 文档是测试集图片预测结果的样例，包括测试集图片名称及预测其属于各分类的概率(概率之和为 1)，该文件作为预测结果提交至 Kaggle 官网，用于计算分类模型的交叉熵，体现为 Private Score、Public Score，得分低者排名靠前，这也是 Kaggle 竞赛的最终成绩。

2.2 探索性可视化

训练集图片共 22424 张，测试集图片共 79726 张，为防止手动分类，测试集中加入了一些不同文件大小的混淆图片，训练集、测试集图片形状均为(480,640,3)。

训练集各类图片的数量分布如图 1 所示：

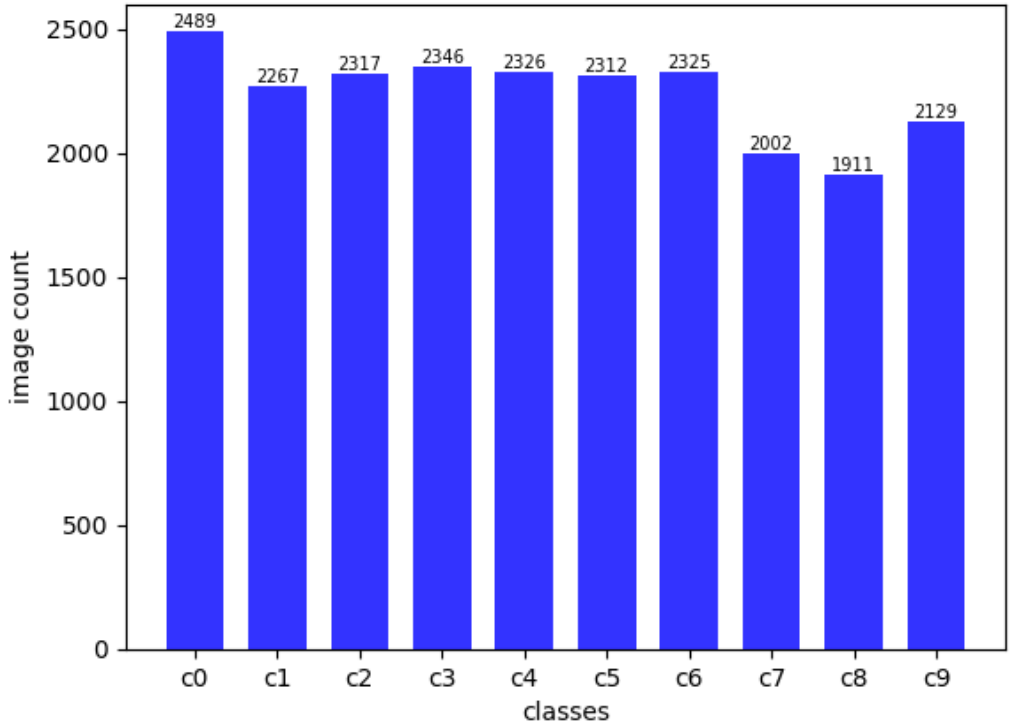


图1 训练集各类状态图片数量分布

从图中可以看出，各类状态的图片数量分布基本均衡，c0 类状态图片数量最多，为 2489 张，c8 类状态图片数量最少，为 1911 张，其余各类状态图片数量介于这二者之间。

训练集图片采集于 26 个不同的驾驶者，每个驾驶者均采集了 10 种行为状态图片。

各驾驶者的图片数量分布如图 2 所示：

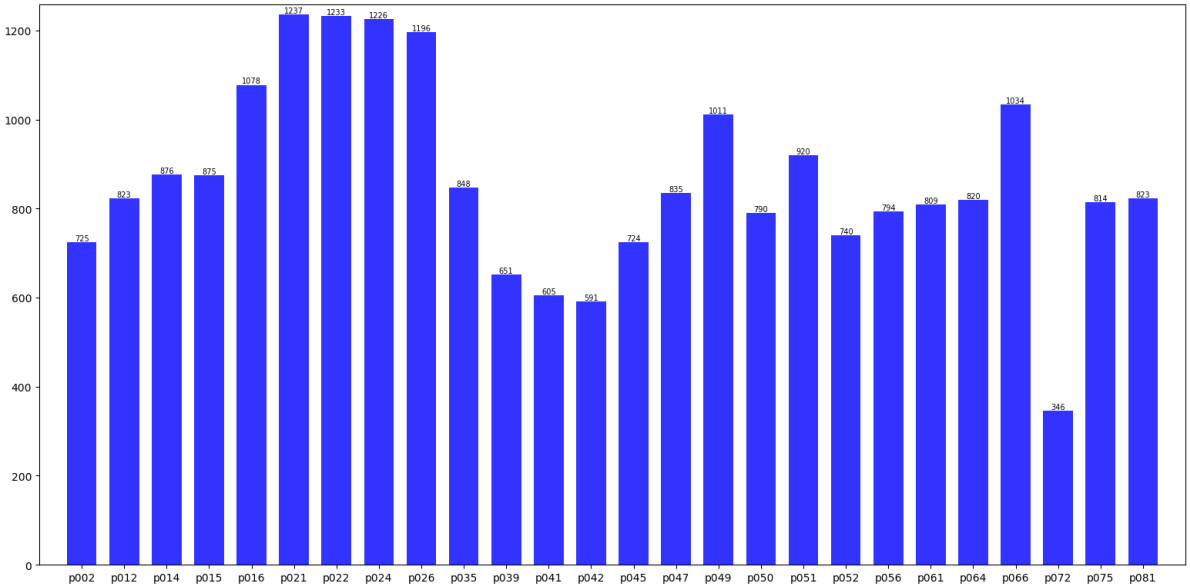


图2 各驾驶者图片数量分布

从图中可以看出，不同驾驶者的图片数量有较大区别，ID 为“p021”的驾驶者的图片数量最多，为 1237 张，“p072”驾驶者的图片数量最少，为 346 张，其余驾驶者的图片数量介于这二者之间。

对于不同驾驶者，其各行为状态的图片数量分布如图 3 所示：

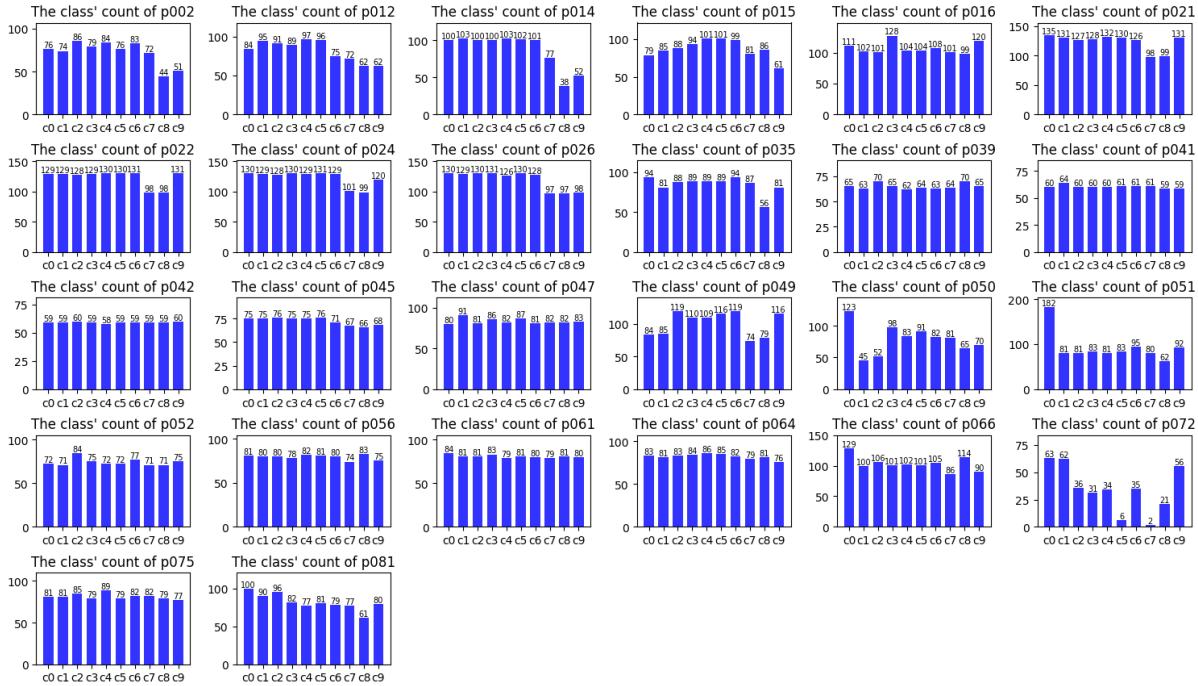


图3 不同驾驶者各类状态图片数量分布

从图中可以看出，大部分驾驶者的各类行为状态图片数量基本均衡，只有少数驾驶者的

行为状态图片数量分布不均或过少，如“p072”图片数量过少且各状态分布严重失衡；“p050”c0 类图片数量过多、c1 及 c2 类图片数量过少，“p051”的 c0 类图片数量过多，“p014”的 c8 类图片数量过少。

本项目将使用所有训练集图片作为输入数据训练卷积神经网络分类模型。

2.3 算法和技术

本项目使用 Keras 框架构建分类模型，该框架的应用模块(Applications)提供了带有预训练权值的模型，这些模型可以用来进行预测、特征提取及迁移学习[1]。目前 Keras Applications 包含的模型及其在 ImageNet 验证集上的准确率(降序排列)如表 1[2]所示：

表1 Keras Applications 预训练模型

模型名称	大小 (MB)	Top1 准确率	Top5 准确率	参数数目	深度	输入图像	模型年份
InceptionResNetV2	215	0.804	0.953	55,873,736	572	(299,299,3)	2016
Xception	88	0.790	0.945	22,910,480	126	(299,299,3)	2017
InceptionV3	92	0.788	0.944	23,851,784	159	(299,299,3)	2015
DenseNet201	80	0.770	0.933	20,242,984	201	(224,224,3)	2017
ResNet50	99	0.759	0.929	25,636,712	168	(224,224,3)	2015
DenseNet169	57	0.759	0.928	14,307,880	169	(224,224,3)	2017
DenseNet121	33	0.745	0.918	8,062,504	121	(224,224,3)	2017
VGG19	549	0.727	0.910	143,667,240	26	(224,224,3)	2014
VGG16	528	0.715	0.901	138,357,544	23	(224,224,3)	2014
MobileNet	17	0.665	0.871	4,253,864	88	(224,224,3)	2017

从表 1 中可以看出，Applications 中的模型均比较新，绝大部分为近 3 年内的模型；各模型均有很高的准确率，InceptionResNetV2 在 Top1、Top5 的准确率最高，分别为 0.804、0.953；随着模型准确率的升高，其参数数目、深度也基本呈上升趋势，准确率最高的 InceptionResNetV2 模型，其复杂程度也最高。

本项目拟采用准确率较高的模型进行迁移学习，并依此构建分类模型。此前有Udacity 学员在《司机驾驶行为检测》毕业项目中使用了 VGG16、InceptionV3 及 ResNet50 模型，取得了很好的效果；为学习、测试新模型及避免重复，本项目将依次使用 InceptionResNetV2[3]、Xception[4]、DenseNet201[5]模型进行迁移学习并构建分类模型。

2.3.1 InceptionResNetV2

2016 年 8 月，Google 团队宣布发布 InceptionResNetv2，该模型在当时的 ILSVRC 图像分类基准测试中获得了最好成绩。顾名思义，InceptionResNetV2 模型的灵感来自于 Inception 模型及 ResNet(残差网络)模型，是加入 ResNet 后由 Inception V3 模型变化而来。ResNet 的初衷在于解决“非常深度的网络在增加更多层时会表现得更差(主要是梯度消失)”这一问题，因此使用 ResNet 可以构建层数很大的深度学习框架；Inception 与 ResNet 相对，如果 ResNet 是为了更深，那么 Inception 家族就是为了更宽[6]。因而 InceptionResNetv2 具有复杂的结构，其深度最大，参数最多。InceptionResNetv2 网络结构如图 4 所示：

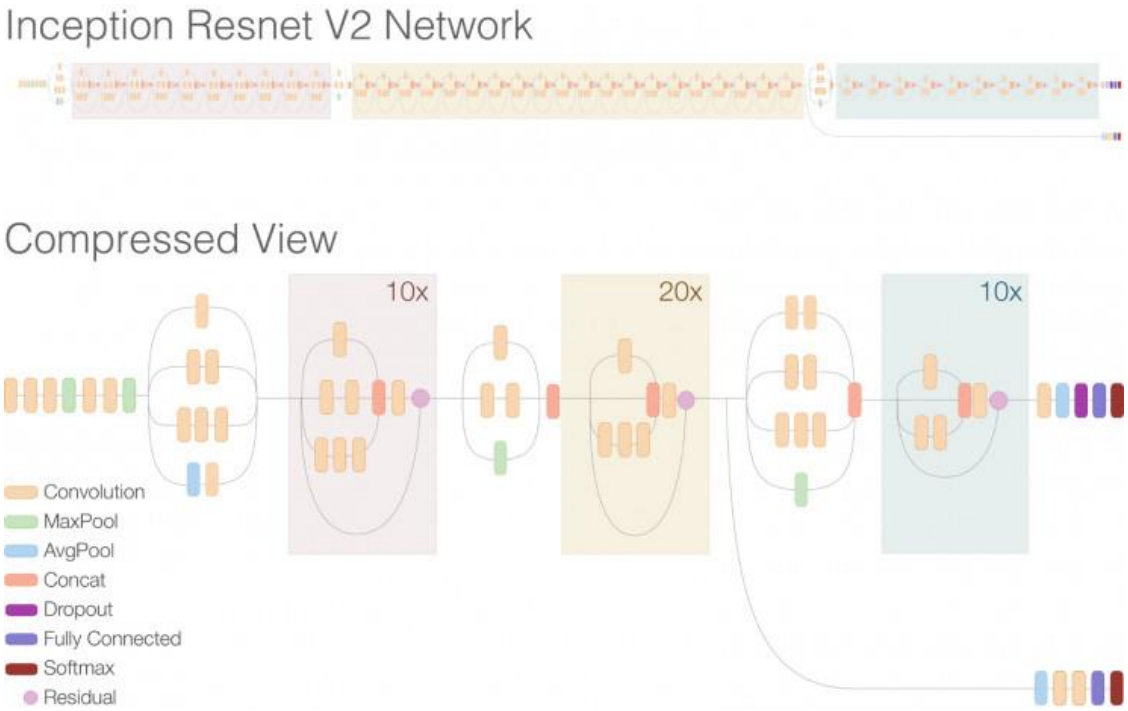


图4 InceptionResNetv2 网络结构

Keras 中的 InceptionResNetV2 模型权值训练自 ImageNet，该模型在 TensorFlow、Theano 和 CNTK 后端均可使用，且接受 channels_first 和 channels_last 两种顺序的维度输入，模型默认输入图片尺寸为 299×299。

2.3.2 Xception

通常，在一组特征图上进行卷积需要三维的卷积核，也即卷积核需要同时学习空间上的相关性和通道间的相关性。将这两种相关性显式地分离开来，是 Inception 模块的思想之一：Inception 模块首先使用 1*1 的卷积核将特征图的各个通道映射到一个新的空间，在这一过程中学习通道间的相关性；再通过常规的 3*3 或 5*5 的卷积核进行卷积，以同时学习空间上的相关性和通道间的相关性。

但此时，通道间的相关性和空间相关性仍旧没有完全分离，也即 3*3 或 5*5 的卷积核仍然是多通道输入的，那么是否可以假设它们可以被完全分离？显然，当所有 3*3 或 5*5 的卷积都作用在只有一个通道的特征图上时，通道间的相关性和空间上的相关性即达到了完全分离的效果。

Xception 由 Inception V3 演化而来，将 Inception 模块简化，仅保留包含 3*3 的卷积的分支，再将所有 1*1 的卷积进行拼接，进一步增多 3*3 的卷积的分支的数量，使它与 1*1 的卷积的输出通道数相等，此时每个 3*3 的卷积即作用于仅包含一个通道的特征图上，作者称之为“Extreme Inception”模块，这就是 Xception 的基本模块。运用“Extreme Inception”模块，作者搭建了 Xception 网络，它由一系列 SeparableConv、类似 ResNet 中的残差连接形式和一些其他常规的操作组成，结构如图 5 所示[4]：

Figure 5. The Xception architecture: the data first goes through the entry flow, then through the middle flow which is repeated eight times, and finally through the exit flow. Note that all Convolution and SeparableConvolution layers are followed by batch normalization [7] (not included in the diagram). All SeparableConvolution layers use a depth multiplier of 1 (no depth expansion).

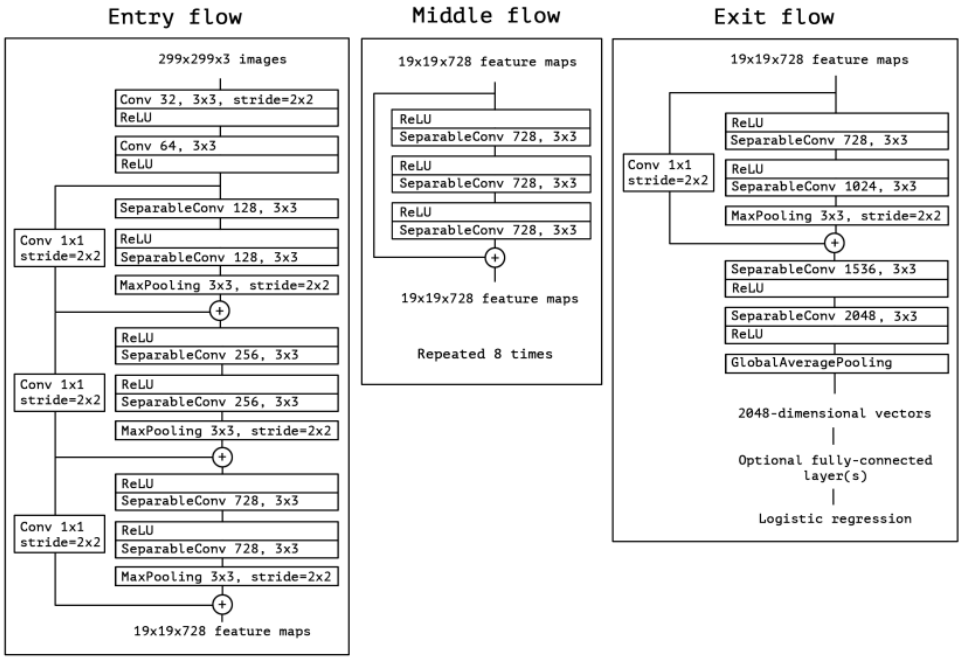


图5 Xception 模型结构

Keras 中的 Xception 亦基于 ImageNet 完成训练，该模型目前仅能以 TensorFlow 为后端使用，由于其依赖于可分离卷积层，该模型只支持 channels_last 的维度顺序(Width, Height, Channels)，默认输入图片大小为 299×299。

2.3.3 DenseNet201

DenseNet201 由 Cornell University 的团队提出并完成训练，其成果发表在 2017 年 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)上并获最佳论文奖。最近几年卷积神经网络在提高模型性能方面，向深度(如 ResNet)或广度(如 Inception)方向发展，但 DenseNet 作者则是从 feature 入手，通过对 feature 的极致利用达到更好的效果和更少的参数。DenseNet 具有减轻梯度消失、加强 feature 传递、有效利用 feature、减少参数数量等优点，其 5 层 dense block 如图 6 所示：

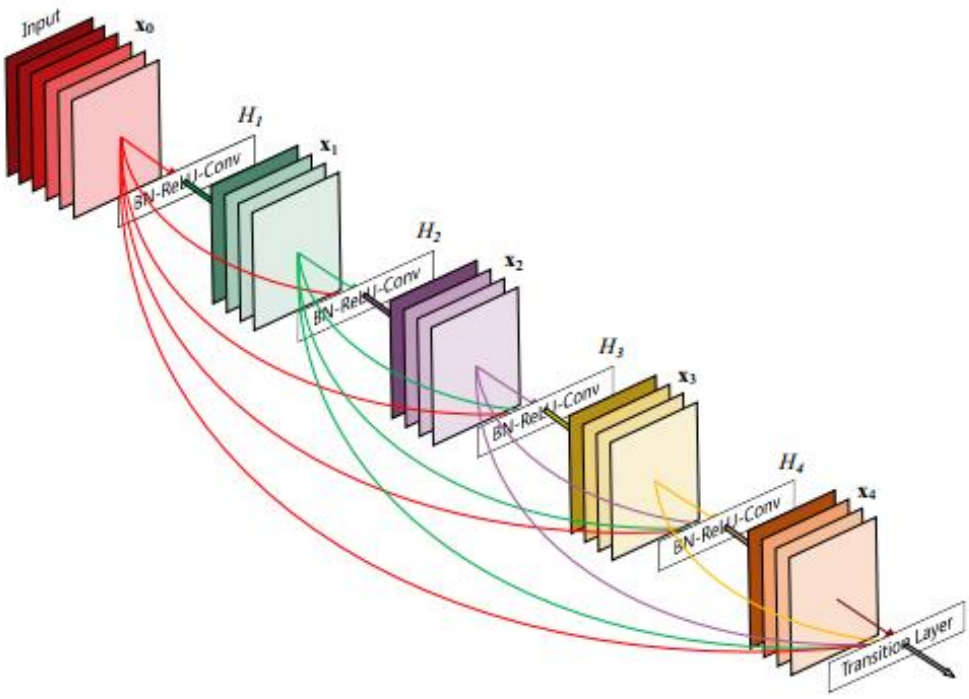


图6 5 层 dense block 结构图

DenseNet201 模型的预训练基于 ImageNet 完成，其权值兼容 TensorFlow、Theano 及 CNTK，为了在使用 TensorFlow 时获取最佳性能，推荐使用 channels_last 的维度顺序，此设置可在“~/.keras/keras.json”配置文件中修改，默认输入图片大小为 224×224。

2.4 基准模型

本项目在 Kaggle(2016)上的竞赛已经结束，共有 1440 个团队、1681 人参加了竞赛，Public Leaderboard 的评价标准使用 31% 的测试集来计算 Multiclass Logarithmic Loss，Private Leaderboard 的评价标准使用剩下的 69% 测试集来计算最终的 Multiclass Logarithmic Loss。指标越小的团队排名越高，最终的排名以 Private Leaderboard 为准。

本项目力争将测试集上的 Multiclass Logarithmic Loss 控制在 Private Leaderboard 排名前 100 以内(Private score 不高于 0.22594)，在测试集的分类结果排进 Private Leaderboard Top 10%。

3 方法

3.1 数据预处理

本项目使用 InceptionResNetV2、Xception、DenseNet201 模型实施迁移学习，该 3 个模型均使用逐样本均值消减法将图像数组元素值归一化至[-1,1]。训练集、测试集图片均为 3 通道彩色图片，形状为(480,640,3)，为使迁移学习达到较好的效果，依照模型提供的归一化方法对图片数组进行预处理，使其元素值在[-1,1]内。InceptionResNetV2、Xception 模型默认的图片输入尺寸为 299×299，DenseNet201 模型默认的图片输入尺寸为 224×224，在使用训练集、测试集图片时，也分别按照模型默认尺寸进行设置。本项目为 10 分类问题，为便于训练，将各样本标记进行 one-hot 编码。

迁移学习模型的学习能力很强大，容易学习到分类特征之外的信息从而导致过拟合，为尽量减小衣服颜色、面部表情、发型、肤色等对分类准确率的影响，在划分训练图片时以驾驶者 ID 为依据，按照 4:1 划分后形成训练集、验证集。

在实际操作过程中，要进一步提高模型的泛化能力，一般有 2 个努力方向——修改模型结构(包括提高模型复杂程度)、增加训练数据。迁移学习模型的结构已经非常有效且较为复杂，修改或融合模型结构不易实施且未必能有效提升模型泛化能力，因此本项目暂不考虑。本项目包含训练图片 22424 张，由 26 名驾驶者提供，按照 4:1 划分后，训练集图片大约有 17500 张左右；经测试，在训练参数、验证集相同的情况下，若将用以训练的同类图片左右对半拼接，使训练集图片数量增至原来的 2 倍，则分类模型的 val_loss 可由原来的 0.55 降至 0.12，val_acc 由原来的 0.85 提升至 0.96，可见增加训练

数据后模型的泛化能力大幅提升。鉴于此，为增加训练图片数量，本项目采用左右对半拼接同类图片的方式使训练集图片增至原来的 2 倍，以提高模型训练效果。

图片经数据预处理至载入训练函数的过程，有 2 种数据流转方式可以选择，一种是数据预处理后存储为数据文件，训练时再分批或一次性从文件读入输入数据，实施训练；另一种是数据预处理后在内存中滞留，直接进行训练，不产生数据文件。前者的优点在于数据预处理只需进行一次，存储为数据文件后可供多次训练使用，且外存容量比内存容量大很多，当训练、测试数据量很大时，尤其适合且只能选择该方式，缺点是数据存储、读取时比较慢，很消耗时间。后者的优点在于省略了存储环节，耗时较少，缺点在于受内存容量限制，对大数据量的情况不适用，且每次训练前均需做数据预处理，也消耗部分时间。对于本项目而言，依据驾驶者 ID 划分训练集并进行拼接、再加上验证集，图片总量大约为 4 万张，图片尺寸按照(299,299,3)计算，需要存储空间大约为 42G，该容量内存不难满足，且经过实测，等量图片的数据预处理时间远小于存储、读取数据文件时间，因此本项目采用方式二准备训练、测试数据(测试数据分 2 批读入内存)。

3.2 执行过程

针对需要解决的问题，将本项目划分为文件读入及形成样本阶段、数据预处理阶段、训练集验证集测试集划分阶段、分类模型构建阶段、分类模型训练及调参阶段、分类模型预测状态及模型评价阶段、可视化预测结果并解释原因阶段，各阶段需要实施的操作步骤如图 7 所示：

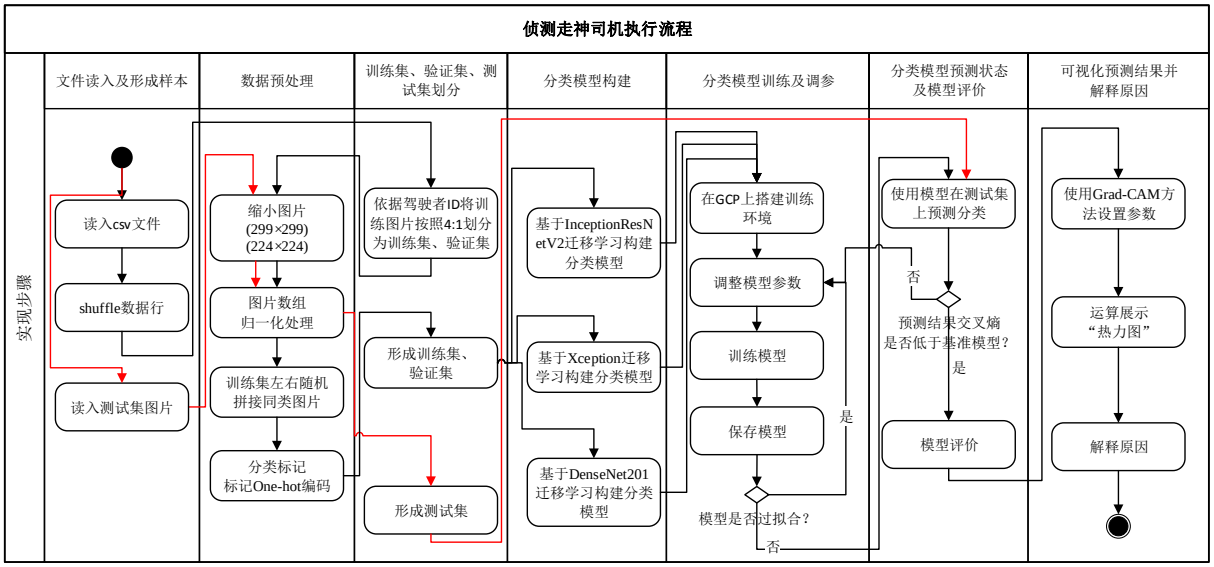


图7 司机驾驶行为检测执行流程

在训练过程中，模型可能面临的问题及解决办法如下：

1) 过拟合

解决办法：a) 设置 EarlyStopping，监测“val_loss”，若验证集损失超过 5 次不再减小，则停止训练；b) 在全连接层设置 Dropout，随机丢弃大部分节点。

2) 模型泛化能力弱、不稳定

解决办法：a) 左右随机拼接同类图片，使训练集图片数量增加 1 倍；b) 解锁迁移学习模型的部分高层、低层甚至是所有层，对其进行训练，得到新的权值；c) 增加训练轮数；d) 对同一模型使用 5-折交叉法，多次划分训练集、验证集并训练得到分类模型，选择验证集“val-loss”较低的模型分别在测试集上进行预测，取平均值作为最终预测结果；e) 综合多个验证集“val-loss”较低的迁移学习模型，联合对测试集进行预测，取平均值作为最终预测结果。

3) 训练速度慢

解决办法：选择收敛速度较快的优化器(optimizer)。

本项目使用 Google Cloud Platform 进行训练、测试，主要配置如下：

操作系统：CentOS 7

vCPU：16 核

GPU：NVIDIA Tesla K80，4 核

内存：104G

硬盘：200G

3.3 完善

本项目为监督学习范畴内的图像多分类问题，其目标为构建 10 分类模型，运用该分类模型在测试图片集上实施预测，返回单张图片分别属于 10 分类的概率。项目完善过程也即为使模型获得更小交叉熵(对应于更高准确率)的参数调整过程。

在实际建模、训练过程中，迁移学习模型待调整的参数包括编译(compile)参数及训练(fit)参数，前者主要包括优化器(optimizer)、目标函数(loss)、模型评估标准(metrics)等，后者主要包括每批数量(batch_size)、训练轮数(epochs)、K-折交叉法中的 K-fold、K-折交叉法次数(n_splits)、回调函数列表(callbacks)、解锁训练层数等。

在设置编译参数时，参考分析优化器适用性的相关资料[7]，考虑到本项目拟采用“Fine-tune”方式训练迁移学习模型，需要在初始时设置较小的学习率，综合收敛速度因素，选用“Nadam”作为优化器；对于多分类测定准确率的模型，其主要参数如下设置：

- optimizer：Nadam，学习率初始设置为 0.0002，之后每轮递减 10%
- loss：'categorical_crossentropy'
- metrics：['accuracy']

训练参数结合模型结构及硬件配置而定，设置如下：

- epochs：5
- K-fold：5
- callbacks：[EarlyStopping, ParallelModelCheckpoint ,TensorBoard, LearningRateScheduler]

重点调整的参数包括 K-折交叉法次数(n_splits)、每批数量(batch_size)、解锁训练层数，具体分 3 步进行：第一步，设定 K-折交叉法次数为 1，调整每批数量(batch_size)及解锁训练层数值，使模型在验证集上得到较低的交叉熵，得到较优模型；第二步，设定 K-折交叉法次数为 3，在较优模型上调整每批数量(batch_size)及解锁训练层数值，训练得到多个最优模型；第三步，使用多个最优模型联合对测试集图片实施预测，生成最终的预测概率 csv 文件，提交 Kaggle 官网计算 Private Score、Public Score。

3.3.1 InceptionResNetV2 参数调整

InceptionResNetV2 迁移学习模型训练总层数为 783 层，训练总参数为 54,352,106，调整 batch_size、训练层数量分别进行训练，记录不同参数训练时的 val_loss 最小值及相关结果，如表 2 所示：

表2 InceptionResNetV2 调参结果(n_splits=1)

模型序号	batch_size	训练层数	训练参数个数	训练耗时(s)	loss	acc	val_loss	val_acc
I1	64	0 ~ 7，618 ~ 782	23,567,530	850	0.1549	0.9533	1.1885	0.6400
I2	128	0 ~ 7，618 ~ 782	23,567,530	761	0.1726	0.9481	1.6307	0.4979
I3	128	0 ~ 7，287~ 782	49,261,450	923	0.1169	0.9652	0.3527	0.9270
I4	64	0 ~ 7，287~ 782	49,261,450	930	0.0023	0.9995	0.1942	0.9380

从表 2 中可以看出，训练层数较少时，模型在验证集上的泛化能力很差，但是 batch-size 设置为 64 时交叉熵更低一些；增多训练层后，模型的泛化能力大幅提升，可以得到较优模型 I4；使用不同 batch_size 训练时模型泛化能力差别不大，增加训练层数时对提高模型泛化能力非常有利。

3.3.2 Xception 参数调整

Xception 迁移学习模型训练总层数为 135 层，训练总参数为 20,881,970，调整 batch_size、训练层数量分别进行训练，记录不同参数训练时的 val_loss 最小值及相关结果，如表 3 所示：

表3 Xception 调参结果(n_splits=1)

模型序号	batch_size	训练层数	训练参数个数	训练耗时(s)	loss	acc	val_loss	val_acc
X1	32	0 ~ 7，36~134	19,740,170	684	0.0027	0.9998	0.2299	0.9442
X2	64	0 ~ 7，36~134	19,740,170	628	0.1166	0.9697	0.1379	0.9581
X3	128	0 ~ 7，36~134	19,740,170	589	0.0023	0.9998	0.1975	0.9372
X4	64	0~134	20,881,970	795	0.0027	0.9995	0.1151	0.9652

从表 3 中可以看出，训练层数相同时，batch_size 设置过大或过小均不能使模型达到较好的泛化能力，设置为 64 时模型在验证集上的交叉熵最低；当训练所有层时，可以得到较优模型 X4；保持较优模型的参数不变，设置 5-折交叉法的执行次数为 3，利用不同的训练数据进行训练，各次的训练结果如表 4 所示：

表4 Xception 5-Fold 执行 3 次结果(n_splits=3)

模型序号	batch_size	训练层数	训练参数个数	训练耗时(s)	loss	acc	val_loss	val_acc
X5	64	0~134	20,881,970	794	0.0021	0.9997	0.1522	0.9634
X6	64	0~134	20,881,970	803	0.0023	0.9996	0.1130	0.9578
X7	64	0~134	20,881,970	769	0.0044	0.9990	0.1975	0.9568

3.3.3 DenesNet201 参数调整

DenesNet201 迁移学习模型训练总层数为 709 层，训练总参数为 18,341,194，调整 batch_size、训练层数量分别进行训练，记录不同参数训练时的 val_loss 最小值及相关结果，如表 5 所示：

表5 DenesNet201 调参结果(n_splits=1)

模型序号	batch_size	训练层数	训练参数个数	训练耗时(s)	loss	acc	val_loss	val_acc
D1	32	0 ~ 7，141 ~ 708	16,692,042	924	0.0961	0.9730	0.3397	0.8866
D2	32	0 ~ 7，211 ~ 708	15,800,842	903	0.0938	0.9729	0.5432	0.8727
D3	64	0 ~ 7，211 ~ 708	15,800,842	631	0.0017	0.9998	0.4990	0.8382
D4	64	0 ~ 7，141 ~ 708	16,692,042	654	0.0016	0.9997	0.4134	0.8668
D5	32	0 ~ 708	18,341,194	791	0.0031	0.9994	0.1297	0.9597

从表中可以看出，在训练层数相同的情况下，batch_size 设置为 32 时，能获得更低的交叉熵；在相同 batch_size 的情况下，训练层数越多，模型在验证集上的交叉熵越低，当训练所有层时，获得较优模型 D5；batch_size 及训练层数对模型的泛化能力影响均比较大。

4 结果

4.1 模型的评价与验证

各迁移学习模型在训练时，loss、acc、val_loss、val_acc 随训练轮数(Epoch)的变化如图 8 所示：

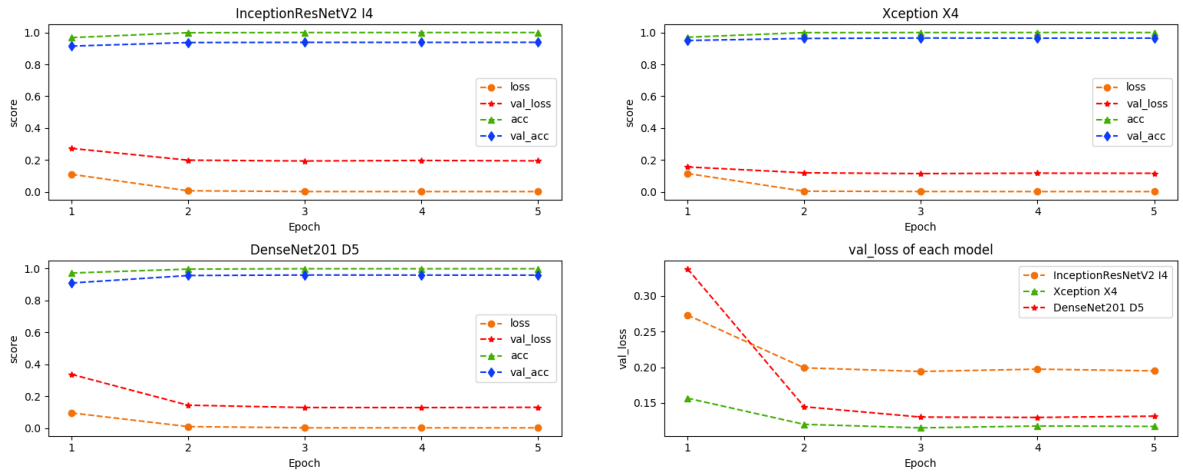


图8 各模型训练得分

从图中可以看出，loss、val_loss 随着训练轮数的增加呈下降趋势，acc、val_acc 随着训练轮数的增加呈上升趋势，且上述各值在第 2 轮后优化很慢，说明此时模型权值在微调；InceptionResNetV2 I4、Xception X4、DenseNet201 D5 的 val_loss 下降趋势一致，在本项目训练过程中，Xception X4 的 val_loss 最低，优于其他 2 个迁移学习模型。

各模型在训练时的耗时如图 9 所示，从图中可以看出，InceptionResNetV2 I4 每轮的训练耗时均大于 Xception X4、DenseNet201 D5，后二者在训练中耗时不相上下；将模型的载入、编译、训练等所有操作包含在内，统计模型单次 K-折交叉法的总耗时，InceptionResNetV2 I4 为 5452 秒(约 91 分钟)，Xception X4 为 4062 秒(约 68 分钟)，DenseNet201 D5 为 6557 秒(约 109 分钟)，DenseNet201 D5 耗时最多，Xception X4 耗时最少，InceptionResNetV2 I4 居中。

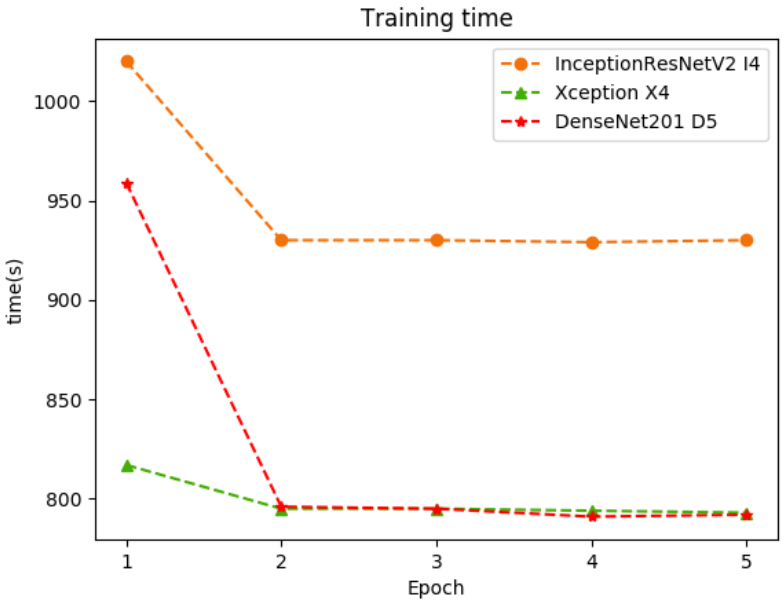


图9 各模型训练耗时

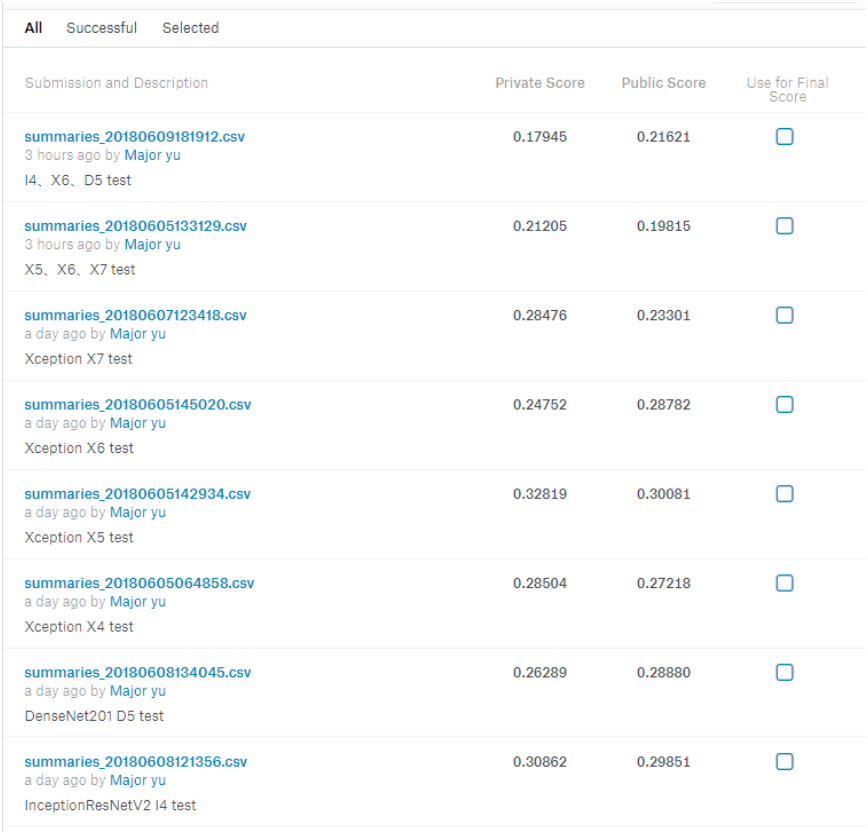
应用各调参优化后的模型分别对测试集图片进行预测，结果上传至 Kaggle 官网，Private score、Public score 如表 6 所示：

表6 模型在 Kaggle 官网得分

模型 序号	loss	acc	val_loss	val_acc	Private score	Public score
I4	0.0023	0.9995	0.1942	0.9380	0.30862	0.29851
X4	0.0027	0.9995	0.1151	0.9652	0.28504	0.27218
X5	0.0021	0.9997	0.1522	0.9634	0.32819	0.30081
X6	0.0023	0.9996	0.1130	0.9578	0.24752	0.28782
X7	0.0044	0.9990	0.1975	0.9568	0.28476	0.23301
D5	0.0031	0.9994	0.1297	0.9597	0.26289	0.28880
X5、X6、X7					**0.21205**	**0.19815**
I4、X6、D5					**0.17945**	**0.21621**

从表中可以看出，本项目训练得到的各最优迁移模型，其 Private score、Public score 均比较低，与训练时的 loss、val_loss 相比差别不是很大，说明模型没有过拟合；各模型得分均在 Kaggle 竞赛的 Leaderboard 比较靠前的位置，其中模型 X6 的 Private score 最小、排名位于 134(Leaderboard Private score 为 0.24837)，模型 X5 的 Private score 最大、排名位于 217(Leaderboard Private score 为 0.32844)；将各模型对测试集图片进行联合预测，Private score、Public score 均能得到大幅下降，Xception 经 5-折交叉法训练 3 次，得到 3 个模型，综合各模型的预测结果，Private score 降低至 0.21205，该值排名位于 82(Leaderboard Private score 为 0.21277)，综合 InceptionResNetV2、Xception、DenseNet201 训练得到的最优模型的预测结果，Private score 下降至最小值 0.17945，且该值排名位于 40(Leaderboard Private score 为 0.18114)。

模型在测试集上的联合预测结果(Private score)均已大幅低于基准模型设定的 Private score 不高于 0.22594 的要求，且在测试集的分类结果排进 Private Leaderboard Top 10%，本项目实现预期目标。各模型在 Kaggle 官网的得分如图 10 所示：



All Successful Selected

Submission and Description	Private Score	Public Score	Use for Final Score
summaries_20180609181912.csv 3 hours ago by Major yu I4、X6、D5 test	0.17945	0.21621	☐
summaries_20180605133129.csv 3 hours ago by Major yu X5、X6、X7 test	0.21205	0.19815	☐
summaries_20180607123418.csv a day ago by Major yu Xception X7 test	0.28476	0.23301	☐
summaries_20180605145020.csv a day ago by Major yu Xception X6 test	0.24752	0.28782	☐
summaries_20180605142934.csv a day ago by Major yu Xception X5 test	0.32819	0.30081	☐
summaries_20180605064858.csv a day ago by Major yu Xception X4 test	0.28504	0.27218	☐
summaries_20180608134045.csv a day ago by Major yu DenseNet201 D5 test	0.26289	0.28880	☐
summaries_20180608121356.csv a day ago by Major yu InceptionResNetV2 I4 test	0.30862	0.29851	☐

图10　各模型 Kaggle 官网得分

4.2　合理性分析

本项目为监督学习范畴内的图像多分类问题，目标在于构建泛化能力较强的分类模型。提升分类模型泛化能力一般有 2 种途径——修改模型结构(包括提高模型复杂程度)、增加训练数据。深度学习模型由模型结构及模型权值构成，迁移学习模型的结构已经很卓越，本项目无需修改，当增加训练数据、训练得到适合本分类问题的模型权值即达到了项目要求。

InceptionResNetV2、Xception、DenseNet201 模型均基于 ImageNet 预训练得到权值。ImageNet 是一个计算机视觉系统识别项目，是目前世界上图像识别最大的数据库，包括 2.2 万类共大约 1500 万张图片，比赛项目涵盖图像分类、目标定位、目标检测、视频目标检测、场景分类，但对检测司机驾驶行为相关图片未见论述及分类，也就是说，本项目涉及的图片未包含于该 3 个迁移学习模型预训练集中，迁移学习模型包含的图像分类信息在本项目未必有效。该猜测也在最初的训练中得以证实，当锁定迁移学习模型的所有层，只训练新增的全连接层、输出层并调整各参数时，模型的交叉熵始终在 1.1 以上、准确率也很低，当解锁较多层甚至对迁移学习模型全部重新进行训练时，模型的泛化能力得到大幅提升。

模型泛化能力的稳定性也是需要考虑的问题，单次划分训练集、验证集对模型进行训练时，模型的泛化能力受集合划分影响较大、时好时坏，当时用 K-折交叉法多次划分训练集、验证集训练模型时，模型泛化能力受集合划分的影响变小、趋于稳定。

InceptionResNetV2、Xception、DenseNet201 模型的设计理念和思路不尽相同，InceptionResNetV2、Xception 均源自于 Inception，前者借鉴并加入了残差，后者新创并加入了可分离卷积，DenseNet201 则是通过对 feature 的极致利用达到更好的效果和更少的参数。由于模型结构和设计理念不同，其对于同一图片提取的信息也有一定区别，训练得到分类模型后，有些图片能用 DenseNet201 模型正确分类，但在 InceptionResNetV2 或 Xception 模型上表现不佳，因此综合不同的较优分类模型对测试集进行联合预测时，预测效果提升非常明显。

5 项目结论

5.1 结果可视化

在卷积神经网络模型解释性技术方面，先后出现了 CAM(Class Activation Mapping)方法[8]及 Grad-CAM 方法[9]。CAM 借鉴了 Min Lin 等的论文 Network in Network[10]中的思路，利用 GAP(Global Average Pooling)替换掉了全连接层，把 GAP 视为一个特殊的 average pool 层，只不过其 pool size 和整个特征图一样大，据此求出每张特征图所有像素的均值，最终以类似“热力图”的形式展示它的决策依据，在图中标示出模型运算时关注的区域。CAM 的解释效果已经很不错了，但它要求修改原模型的结构，导致需要重新训练该模型，这大大限制了它的使用场景；如果模型已经上线了，或者训练的成本非常高，为此重新训练几乎是不可能的，Grad-CAM 方法针对该问题给出了解决办法。

Grad-CAM 方法的基本思路与 CAM 一致，也是通过得到每个特征图对应的权重，最后求一个加权和。但它与 CAM 的主要区别在于求权重的过程，CAM 通过替换全连接层为 GAP 层，重新训练得到权重，而 Grad-CAM 用梯度的全局平均来计算权重，从而避免了重新训练模型。事实上，经过严格的数学推导，Grad-CAM 与 CAM 计算出来的权重是等价的。

载入 Xception 模型，使用 Grad-CAM 方法显示类属 c1 的图片，如图 11 所示：

图11 使用 Grad-CAM 方法显示正确分类结果

从图中可以看出，分类模型关注的是驾驶者的“右手打字”分心行为，关注点正确，分类正确。

再来看分类错误的例子，Xception 模型将图 12(a)所示的图片(分类 c6：喝饮料)错误分类为“c8: 整理头发和化妆”；对比图 12(b)的正确分类，Xception 模型的关注区域基本相同，应该是学习到了手臂、嘴部行为，并将此作为分类依据，当对图 12(a)进行分类时，错将其分类为 c8。

(a) 分类错误　　(b) 分类正确

图12 使用 Grad-CAM 方法显示错误分类结果及比对

5.2 对项目的思考

本项目源于 2016 年的 Kaggle 竞赛，属于监督学习范畴的图像多分类问题，在实施过程中，使用深度学习中的迁移学习技术达到了预期目标。本项目的执行流程包括文件读入及形成样本阶段、数据预处理阶段、训练集验证集测试集划分阶段、分类模型构建阶段、分类模型训练及调参阶段、分类模型预测状态及模型评价阶段、可视化预测结果并解释原因阶段，通过实施本项目，进一步加深了对机器学习流程及方法的理解，掌握了运用深度学习技术解决图像学习相关问题的方法。在实施机器学习时，需注意以下方面：

1) 数据预处理很重要，直接关系到机器学习算法的训练效果，例如对于图像数据，在同一算法及参数的情况下，对比归一化处理至[-1,1]、[0,1]的训练效果，前者的泛化能力明显高于后者。
2) 使用迁移学习模型进行训练、预测时，要针对需要解决的具体问题，分析待解问题数据集与迁移学习预训练数据集之间的异同，若二者相似，迁移学习的权重可直接使用，若二者相差甚远甚至完全不同，应在新数据集上训练模型以得到更合适的模型权值。

5.3 需要作出的改进

受训练环境、时间所限，本项目在达到预期目标后即停止训练，未进一步优化分类模型。事实上，InceptionResNetV2、Xception、DenseNet201 均是很有优秀的深度学习模型，若对本项目调参得到的较优模型继续进行训练(增大 Epoch)，各模型的泛化能力应会进一步提高，或者对多个迁移模型进行 block 融合，经训练后也能有效提高模型的泛化能力，这些均是可以改进的地方。

6 参考文献

[1] Sebastian Ruder. (2017). Transfer Learning - Machine Learning's Next Frontier. Retrieved from http://ruder.io/transfer-learning

[2] Keras: The Python Deep Learning library. (2018). Applications. Retrieved from https://keras.io/applications/#densenet

[3] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, 2016. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, USA: IEEE Press, 1 - 7

[4] François Chollet, 2017. Xception: Deep Learning with Depthwise Separable Convolutions. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA: IEEE Press, 1800 – 1807
[5] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, 2017. Densely Connected Convolutional Networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA: IEEE Press, 2261 – 2269
[6] Joyce Xu. (2017). An Intuitive Guide to Deep Network Architectures. Retrieved from https://towardsdatascience.com/an-intuitive-guide-to-deep-network-architectures-65fdc477db41
[7] Sebastian Ruder. (2016). An overview of gradient descent optimization algorithms. Retrieved from http://ruder.io/optimizing-gradient-descent/
[8] B. Zhou, A. Khosla, A. Lapedriza, and etc. 2016. Learning Deep Features for Discriminative Localization. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Washington, USA: IEEE Press, 2921 - 2929
[9] Min Lin, Qiang Chen, and Shuicheng Yan. 2014. Network In Network. International Conference on Learning Representations (ICLR), Banff, Canada:1 - 10
[10] R. R. Selvaraju, M. Cogswell, A. Das, and etc. 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. IEEE International Conference on Computer Vision (ICCV), Venice, Italy: IEEE Press, 618 – 626