# 定义

## 项目背景

在实际的深度学习应用中, 数据和模型同样重要，往往准备数据和定义目标基准占据了大量的时间，而要得到一个解决实际问题的神经网络模型，调参也是一个不容忽视的环节。虽然模型的鲁棒性以及轻量化应用在实际中是最重要的，调参在实际中起的作用不是决定性的，但是不同的数据和每一种网络结构的组合都需要合适的参数才能达到最佳效果。在得到最佳网络结构的过程中，可能会出现欠拟和或者过拟和，出现过拟和和欠拟和的原因很多。出现过拟和时，往往模型过于复杂，参数空间很大，训练集和验证集本身存在偏奇，训练数据集过小。出现欠拟和时，往往模型过于简单，参数空间较小，部分参数设置不合理（dropout，学习率，卷积核size和卷积核数量等）,训练集噪声较高。出现欠拟和或者过拟和时，根据实际情况，都需要对模型进行合理调参。

本项目在MIT的无人驾驶课程中实现无人驾驶任务中，搭建根据前置摄像头所拍摄的实时录像，预测车辆转向角度的功能的深度神经网络模型的过程中，讨论调整参数对模型的影响。本项目使用MIT 6.S094 这门公开课中的Tesla数据集，训练深度卷积神经网络模型，再使用模型对车辆的前置相机所拍摄的路况图像，对车辆转向角度进行预测，在这个搭建模型过程中，探索调节参数对神经网络模型的影响。

## 问题描述

根据前置摄像头所拍摄的录像预测车辆转向角度，实际上是一个监督回归学习问题，车辆转向角度数据为-20到20之间的连续值，训练的目标，就是让模型学习不同驾驶场景下，能正确选取一个合适的转向角度。

怎样建立并调节参数得到合适的训练模型，实现对车辆转向角度预测，就是本项目的关键。训练模型可以通过从一个简单的模型开始，在训练中，根据训练情况，调整、加深，直到得到满意的结果的模型。

在训练模型的过程中，如何通过权衡使用修改学习率、加深网络、使用正则化参数、使用dropout以及使用批量标准化BN等方法，训练模型，达到模型的最佳效果，就是本项目的关键问题所在

## 评估指标

运算决定系数R2是回归分析中十分常用的统计信息，经常被当作衡量模型预测能力好坏的标准。R2的数值范围从0至1，表示目标变量的预测值和实际值之间的相关程度平方的百分比。一个模型的R2 值为0还不如直接用**平均值**来预测效果好；而一个R2 值为1的模型则可以对目标变量进行完美的预测。从0至1之间的数值，则表示该模型中目标变量中有百分之多少能够用特征来解释。模型也可能出现负值的R2，这种情况下模型所做的预测有时会比直接计算目标变量的平均值差很多。本项目通过计算测试视频的实际转向角度数据和模型预测数据之间的运算决定系数R2，判断模型效果的好坏。假设一数据集包括*y*1,...,*yn*

共*n*个观察值，相对应的模型预测值分别为*f*1,...,*fn*[8]。定义残差*ei* = *yi* − *fi*，平均观察值为



于是可以得到总平方和为：



残差平方和为：



于是，决定系数可以定义为：



# 分析

## 输入数据

本项目采用的数据，是MIT 6.S094 这门公开课中的Tesla数据集。其中视频图像是前置摄像机拍摄的10段高速公路上驾驶员行驶过程的录像，每段约50秒钟，视频图像的尺寸为1280\*720，如图1所示。由于车辆前置摄像头所拍摄的图像，有训练时用不到的冗余部分，如靠近底部显示车身的部分，和顶部有大约1/3的天空部分，需要在训练之前裁减掉，如图1下方图片所示。





图 1

由于实际训练时，使用的视频的图片是连续的，前后图片之间有很大的相关性，训练结果不利于泛化，所以在训练前要对训练图片做随机化处理。

由于需要拿出一个视频的图片作为验证集，一个视频的图片作为测试集，所以项目需要在八个数据集中做训练，最终在第十个录像中，测试最终模型结果。由于这八个训练视频，每个解压出的图片数在1350到3510之间，如果把这八个视频的图片的图片都集中起来训练，那么每轮训练需要两万多万张图片，图片大小如果在200\*200以上的话，占用的内存会非常大。所以每轮训练时，采用分别对这八个视频的图片做一次训练的方式，所有的轮次都训练完毕以后，保存训练模型，最终再将模型导入到第十段视频中，作为最终结果。

车辆转向信号数据集，来自驾驶过程中汽车内置存储记录 [1]，记录了车辆转向角度，转向时的时间戳，以及对应的视频图片的索引，如图2。转向角度数据为间隔为0.5的数字，分布在-20到20之间，部分数据如图2，图1对应的转向角度为0.5。由于转向角度都是间隔为0.5的数字，所以输出数据也要做相应处理:

1. 一个转向角度数字如果是正数，则如果它的小数部分在0到0.25的半闭半开区间，则将转向角度数字规格化为小数部分为0；如果它的小数部分在0.25到0.75的半闭半开区间，则将转向角度数字规格化为小数部分为0.5；如果它的小数部分大于等于0.75，则将它规格化为整数部分加一，小数部分为0。
2. 一个转向角度数字如果是负数，则如果它的小数部分在-0.25到0的半闭半开区间，则将转向角度数字规格化为小数部分为0；如果它的小数部分在-0.75到-0.25的半闭半开区间，则将转向角度数字规格化为小数部分为-0.5；如果它的小数部分小于等于-0.75，则将它规格化为整数部分减1，小数部分为0。

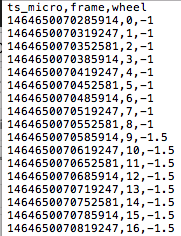


图 2

## 算法和技术

由于项目需要根据前置摄像头的画面，预测车辆转向角度，而车辆转向角度是离散的数字，所以，这个问题是一个要用卷积神经网络做回归的问题。

用于回归的算法有很多，比如卷积神经网络、决策树、SVM支持向量机等都是处理回归问题的算法，但是决策树或者SVM这一类算法对于图片数据无法直接处理。

卷积神经网络是目前用于图像分类识别的主要算法，它可以拟合任何函数[4]，对于回归问题，也可以解决。卷积神经网络和普通神经网络的区别在于，卷积神经网络包含了一个由卷积层和子采样层构成的特征抽取器。在卷积神经网络的卷积层中，一个神经元只与部分邻层神经元连接。一个卷积层中，通常包含若干个特征平面，每个特征平面由一些矩阵排列的神经元组成，同一特征平面的神经元共享权值，在网络训练的过程中，共享权值得到合理的权值。

卷积神经网络由三部分构成。第一部分是输入层。第二部分由n个卷积层和池化层的组合组成。第三部分由一个全连结的多层感知机分类器构成。在运用迁移学习的方法时，在预训练模型的基础上，根据实际图片数据和预训练模型曾经使用的数据的差别，调整训练的网络层。对于训练数据和预训练模型曾经使用的训练数据有较大相似的情况，只需微调预训练模型的最后几层，对于训练数据和预训练模型曾经使用的训练数据有较大差异的情况，则需要对模型的卷积层的大多数层再训练。

卷积神经网络中，有几个经典网络：

[VGGNet](http://www.robots.ox.ac.uk/~vgg/research/very_deep/" \t "_blank)，是2014年ILSVRC的亚军，是Simonyan and Andrew Zisserman发明的。VGGNet向我们证明了深度对于卷积网络的重要性。最终性能最优的网络是16层CONV/POOL，从输入到输出层结构相同，都是3x3卷积核2x2池化。VGG网络的优点首先是卷积核变小。实际上在VGG之前已经有一些模型开始尝试小卷积核了，VGG模型只是成功案例之中的一个。那么小卷积核有什么好处呢？首先是参数数量变少，过去一个7\*7的卷积核需要49个参数，而现在3个3\*3的卷积核有27个参数，看上去参数数量降低了不少；第二是非线性层的增加，过去7\*7的卷积层只有1层非线性层与其相配，现在有3个3\*3的卷积层有3个非线性层。非线性层的增加会使模型变得更加复杂，因此模型的表现力也有了提高。

ResNet，由He等发明，赢得了2015 ILSVRC冠军。起初plain网络结构是在VGG19的基础上做了加深和优化，而RestNet则是在plain的基础上，增加了残差结构。有证据显示，卷积神经网络层数对于训练网络的表现有至关重要的作用，然而卷积神经网络越深，越容易导致梯度消失，从而阻止模型收敛，而残差结构可以解决深度神经网络容易梯度消失的问题，从而大幅提高模型的表现。

GoogleNet，2014年ILSVR冠军，来自Google，由[Szegedy et al](http://arxiv.org/abs/1409.4842" \t "_blank)发明。它的主要贡献在于发展了Inception结构，Inception架构可以将较大size的卷积核拆分成小的卷积连接，大大减少了模型参数，却保留了精度。

在本项目中，为了探索调参对模型的影响，使用了从简单模型开始优化加深模型的办法。

从简单模型开始搭建时，采用了NVIDIA在End-to-End Deep Learning for Self-Driving Cars 论文中采用的模型，见图2。该模型需要对输入图片做标准化处理。该卷积神经网络模型使用3个卷积核大小为5\*5，stride为2\*2的卷积核，连接2个卷积核大小为3\*3的卷积核，然后连接扁平化层，和3个深度分别为100、50和10全连接层，最后连接深度为1的输出层。

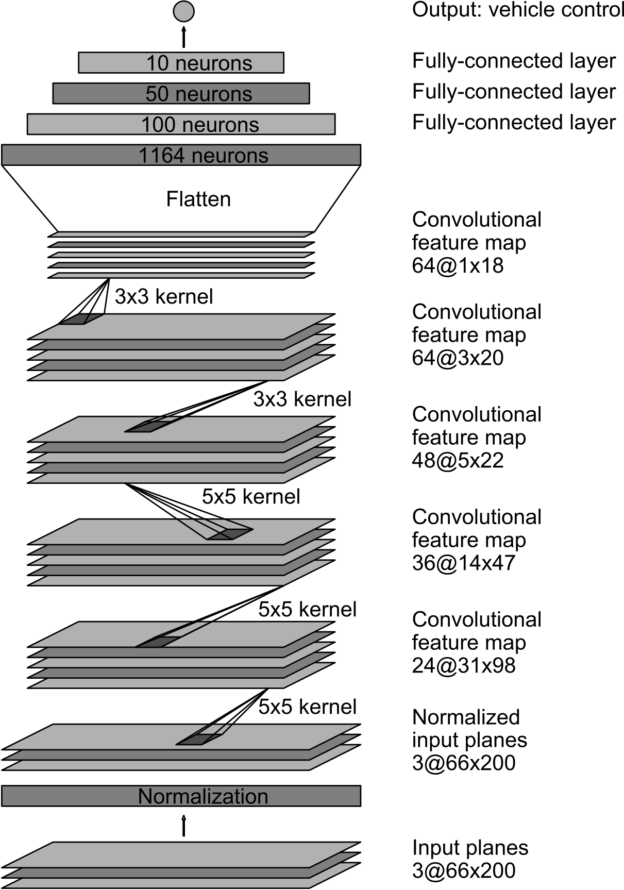


图 3

搭建模型以后，模型的调参主要调节学习率、使用正则化参数、使用dropout以及使用批量标准化输入BN，以及添加1\*1卷积。

关于学习率，运用梯度下降算法进行优化时，权重的更新规则中，在梯度项前会乘以一个系数，这个系数就叫学习速率。如果学习速率太小，会使收敛过慢，如果学习速率太大，则会导致代价函数振荡。在本项目中学习率的选择采用尝试不同数值的方法，选择模型的loss下降最多的学习率。

关于正则化参数，正是损失函数的惩则化参数分为L1正则化和L2正则化。L1正则化和L2正则化可以看做惩罚项。L1正则化可以产生稀疏权值矩阵，即产生一个稀疏模型，当模型特征众多时，让一些特征为0，这样可以用于特征选择。L2正则化可以防止模型过拟合（overfitting）；一定程度上，L1也可以防止过拟合。

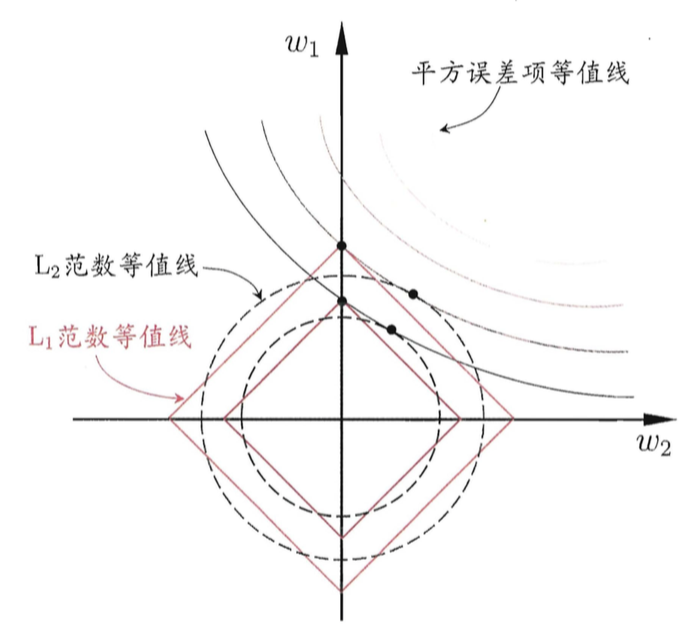


图 4

L1正则化和L2正则化的作用是怎么产生的？如图3[5]，假设有三组等值线，位于同一条等值线上的 w1 与 w2，具有相同的值（平方误差、L1-正则化或L2-正则化）。并且，对于三组等值线来说，当 (w1,w2)沿着等值线法线方向，像外扩张，则对应的值增大；反之，若沿着法线方向向内收缩，则对应的值减小。因此，对于训练模型来说，实际上是要在正则项的等值线与损失函数的等值线中寻找一个交点，使得二者的和最小。对于 L1-正则项来说，因为 L1-正则项的等值线是一组菱形，这些交点容易落在坐标轴上。因此，另一个参数的值在这个交点上就是零，从而实现了稀疏化。对于 L2-正则项来说，因为 L2-正则项的等值线是一组圆形。所以，这些交点可能落在整个平面的任意位置。所以它不能实现稀疏化。[6]

关于dropout，它由深度学习领域权威Hinton，在2012年文献：《Improving neural networks by preventing co-adaptation of feature detectors》中提出，在每次训练的时候，让一半的特征检测器停过工作，这样可以提高网络的泛化能力，Hinton把它称之为dropout。dropout在防止过拟和上，常常有显著成效。

关于批量标准化BN，BN的出现是为了解决梯度消失的问题。在批量随机梯度下降中，梯度消失的原因之一在于我们的输入数据（网络中任意层的输入）分布在激活函数（如sigmoid）收敛的区域之外时，激活函数的导数接近于0，这样梯度就消失了。BN 其实就是将数据的分布由原来激活函数的收敛区调整到梯度较大的区域，类似于数据的归一化处理，不过为了保持原来网络的特征表达能力，引入一些措施将调整后的数据又还原回去。

关于1\*1卷积，1\*1卷积在加深网络的过程中，对降低过拟和在实际操作中有非常显著的作用。1\*1卷积首先在GoogLeNet中出现，它能加深了网络，降低维度，从数学上看，它相当于做了一次全连接层的操作。全连接层和卷积层的唯一区别是卷积层的神经元只连接了输入的局部感受野，而且很多卷积层共享参数。然而，神经元在两者中，都计算点乘，因此，它们的功能是一致的。因此，全连接层和卷积层之间的转换也是可能的。对于一个1\*1卷积层，都有一个全连接层完成同样的前向传播功能。其中权重矩阵将是一个很大的矩阵，而且除了一些相同的局部感受野部分，其它地方都是0，这样就得到了一个选择特征的结构。[7]

项目的实现，使用Keras来做卷积神经网络的模型的构建、训练和保存，Keras是一个高层神经网络API，Keras由纯Python编写而成并基[Tensorflow](https://github.com/tensorflow/tensorflow)、[Theano](https://github.com/Theano/Theano)以及[CNTK](https://github.com/Microsoft/cntk)后端。

## 基准模型

本项目通过计算测试视频的实际转向角度数据和模型预测数据之间的运算决定系数R2，判断模型效果的好坏。由于本项目取八个视频的图片作为训练集，每个视频的图片张数在1350到3510之间，训练集所有图片在20000张左右，并不是很充足，本项目希望达到的运算决定系数R2目标为0.4，根据测试经验，这样绝大部分预测结果和实际结果的差距能够保持在5之内。

# 方法

## 数据预处理

在本项目中，为了探索调参对模型的影响，使用了从简单模型开始优化加深模型的方法。

从简单模型开始优化加深模型中，对图片做了裁剪掉冗余部分、根据模型要求缩放、采用了对输入数据归一化。

#### 图片裁剪

图像中，靠近底部显示车身的部分，和顶部有大约1/3的天空部分，需要在训练之前裁减掉。本项目在实现中，使用opencv库来做图片的裁剪。为了后续训练方便，还要将预处理后的图片数据存到本地磁盘。使用pickle做图片数据的存储。

#### 图片缩放

在从简单模型开始优化加深模型的办法中，将裁剪后的图片缩放成200 \* 66的大小，这是为了和NVIDIA的缩放一致，这样也能保持图片基本上和裁剪后的原始尺寸比例相当。使用opencv做图片的缩放。

#### 归一化

采用归一化输入处理，让图片均值为0，方差为1，提高梯度下降的速度。采用输入数据除以127.5，再减1的方法，将图片归一化。

### 执行过程

#### 模型搭建

采用keras搭建神经网络，先根据NVIDIA的模型搭建，该网络先依次输入3个大小为5，stride为2，卷积核个数分别为3、24和36的卷积核，两个大小为3，深度分别为64和72的卷积核，再接一个扁平化层和3个深度分别为100、50和10的全连接层，最后连接输出层。采用随机梯度下降sgd的优化器，损失函数使用mean\_squared\_error。训练中，batch size 设置为64。使用固定Epoch为20，随着epoch衰减的学习率进行训练，每一代依次对八个视频中的图片做一次训练，训练前，每个视频中的图片都随机打乱。

#### 优化调参执行过程

首先需要确定的是合适的学习率，采用分别采用0.0005，0. 00005，0.005，00006，00004，的学习速率，模型的loss在几个学习率的情况下，降到最低时，使用的学习速率为0. 00005（图5）。

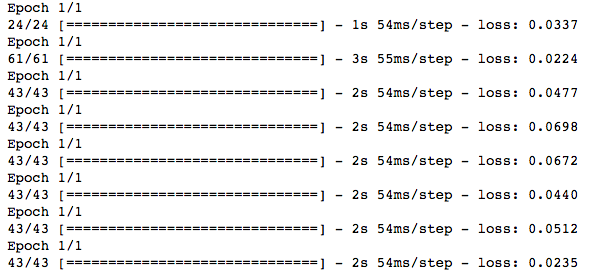
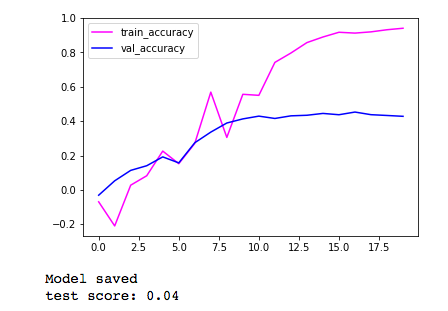


图 5

图 6

这时的训练集和验证集在20代的中的r2得分如图4，从图中可以出训练集的得分最终稳定在0.9左右，而验证集的得分稳定在0.4左右，而最终在测试集上的得分却只有0.04。说明网络的泛化能力很弱，过拟和情况严重。

这时通过在前三个5\*5的卷积层，和接下来的2个3\*3的卷积层中，添加上批量标准化BN，得到的结果如图6

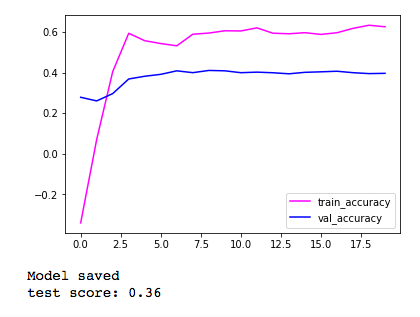


图 7

从图6可以看出，通过增加批量标准化BN，训练集的r2得分迅速达到0.6左右就收敛不再增加了，验证集的r2得分迅速达到0.4左右就收敛不再增加了。增加批量标准化BN，模型的过拟和情况得到了改善。但是这时模型的训练集的r2只有0.6左右，相对于之前0.9的得分，说明之前的模型学到了很多噪声，而加入批量标准化BN以后，模型学习的噪声减少了，学习到的真正有用的信息增加了，但只有0.6的得分，说明网络的能力还不够，模型还欠拟和，所以下一步采用增加网络深度，来增强模型的学习能力。在两层3\*3的卷积核之后，尝试添加一个大小为3\*3，没有stride的卷积核（图7），或者两个3\*3，没有stride的卷积核（图8）。

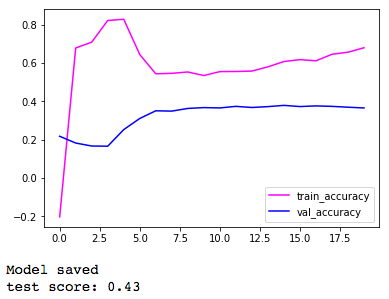


图 8

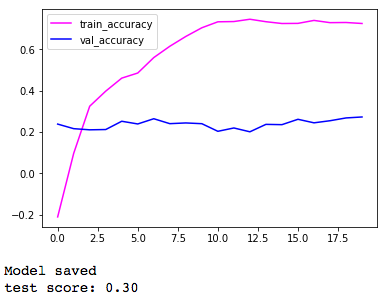


图 9

从图中可以看出，添加一个3\*3卷积核，虽然训练集的得分最终和之前相当，但是验证集的得分略有增加。添加两个3\*3卷积核之后，验证集的得分没有明显增加，但是训练集的得分有提高。训练集得分和验证集得分差距加大，过拟和情况更严重了，考虑在全连接层增加正则化参数，正则化参数加在深度为100的全连接层上，修改kernel\_regularizer和bias\_regularizer的值，分别为0.00003，0.0003，0.00005， 0.0000003，0.00002，最终，在kernel\_regularizer和bias\_regularizer都取0.00003时，得到了较好的效果，如图9。

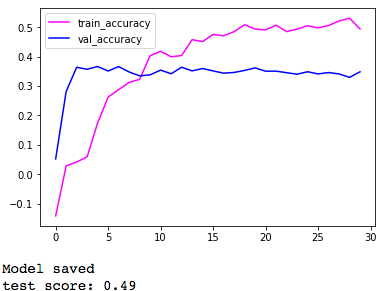


图 10

添加正则化之后，训练集和验证集的得分差距缩小，过拟和得到了改善，但是训练集和验证集得分都没有增加。接下来，考虑在新增加的2个3\*3的卷积核后面，各增加一个1\*1卷积核（如图10）。

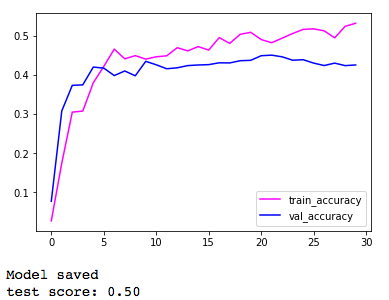
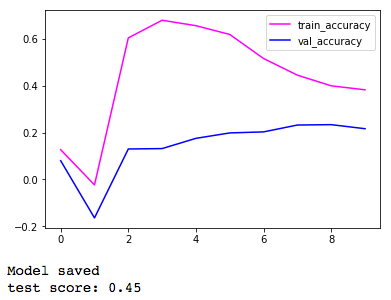
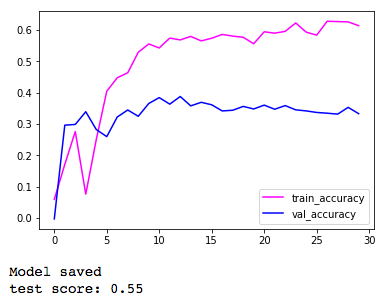
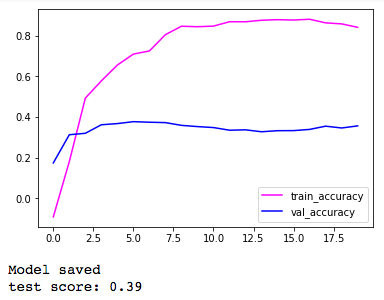


图 11

从图中可以看出，过拟和的情况进一步得到改善，训练集和验证集的得分差距又减小了，验证集的得分也有了提高。但是训练集的得分还是不高，还需要进一步提高模型能力。但是在最后增加的1\*1卷积核之后，反复尝试增加3\*3卷积、1\*1卷积，但是模型的表现都没有增加，验证集的r2得分仍然没有大的提高。过程中的部分训练图如下：

最终放弃继续拓宽网络的思路，采用调整正则化项的的方法，再加入dropout，在卷积层中增加最大池化层，得到了验证集0.48以上的结果（如图11）。



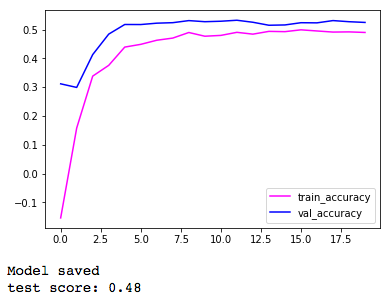


图 12

表格 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 优化步骤 | 训练集最后得分 | 验证集最后得分 | 测试集得分 | Loss |
| 采用分别采用0.0005，0. 00005，0.005，00006，00004，以及期间的学习速率， 0.00005时的结果最佳 | 0.94 | 0.43 | 0.04 | 0.09-0.38 |
| 所有卷积层添加上批量标准化BN | 0.63 | 0.4 | 0.36 | 0.023-0.069 |
| 增加3\*3，没有stride的卷积核，卷积核的个数取80到100之间的数字，84时结果最佳 | 0.73 | 0.27 | 0.30 | 0.04-0.16 |
| 在全连接层增加正则化参数，参数大小为0.0003，0.00003，0.000003，0.00005，0.00001及之间的数字，取0.00003时效果最佳 | 0.49 | 0.35 | 0.49 | 0.17-0.63 |
| 在卷积层最后添加1\*1卷积 | 0.83 | 0.31 | 0.51 | 0.038-0.21 |
| 加入最大池化层，和dropout | 0.53 | 0.48 | 0.50 | 0.15-0.62 |

### 结果

#### 模型的评价与验证

根据上一章的调参优化，得到了验证集0.45，测试集0.48的结果，如图12。所有的优化步骤见表格1，虽然这个结果并不是非常理想，但是考虑到数据集的大小，这是目前能调出的最好的结果。模型导入第十个视频得到的模拟效果截图如图14，15，图14是预测转向角度与实际相差较大时的情况，转向盘显示黄色，图15时预测转向角度与实际相差较小时的情况，转向盘显示翠绿色。从图12中可以看到，模型的训练集得分和验证集得分差别不大，模型没有太多过拟和现象，测试集得分和验证集得分也相当，说明模型泛化能力较强。图13可以看到测试数据中实际的转向角度和模型预测结果的偏差，从图中可以看出，偏差都在5以内，绝大部分都小于3。但是由于数据集有限，继续优化模型，也没能得到更好的结果。

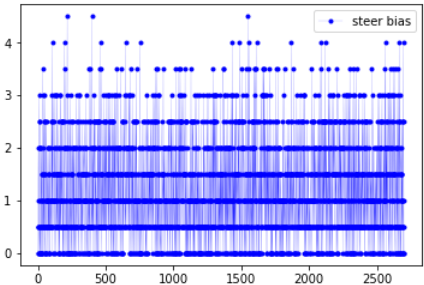


图 13



图 14



#### 后续改进

由于本项目的训练图片总数在2万张以上，若采用迁移学习，采用在别的项目中取得很好结果的模型，如resNet,Vgg19等，所需的训练时间会很长。如果采用resNet, Vgg19这样在图片分类中获得过奖项的模型，做迁移学习，可能会获得更好的效果。

由于时间有限，没有对训练过程做Cam可视化，后续增加Cam可视化分析，会对模型的优化提供更多线索依据。

参考文献

[1] MIT公开课6.S094: Deep Learning for Self-Driving Cars 讲义

[2] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna , Rethinking the Inception Architecture for Computer Vision

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Microsoft Research , Deep Residual Learning for Image Recognition

[4] <https://devblogs.nvidia.com/deep-learning-self-driving-cars/>

[5]《机器学习》（西瓜书）

[6] <https://liam0205.me/2017/03/30/L1-and-L2-regularizer/>

[7] <http://cs231n.github.io/convolutional-networks/#architectures>

[8]https://zh.wikipedia.org/wiki/%E5%86%B3%E5%AE%9A%E7%B3%BB%E6%95%B0