数据挖掘课程设计报告

姓名: 姚远　　学号: 162150109

日期：2023 年 11 月 15 日

项目网址: https://github.com/Atri2333/nuaa_data_mining_course_design

1 任务概况

1.1 数据集

本课设数据集来自于 Kaggle, 为树叶分类数据集. 该数据集包含 176 类不同的树叶, 18353 张训练样本 (每种树叶至少 50 个样本, 长尾效应并不明显) 以及 8800 张测试样本. 分辨率皆为 224×224. 可以发现这是一个简单的图片分类任务.

1.2 评测指标

Kaggle 官方直接采用 Classification Accuracy 作为竞赛评测标准. 一般在数据类别均衡的情况下, 模型的 acc 越高, 说明模型的精度越好. 本数据集由于类别很多, 数据分布较为平均 (长尾比约 3:1), 因此不考虑通过计算每个类别的 recall 值来评测模型.

1.3 解决方案

事实上这是一个课程网站的 project, 该课程是深度学习相关课程, 因此如果你去查看 Code 就会发现清一色的 Resnet 与其它深度学习方法.

本课设大致分为传统机器学习和深度学习两种方案来解决问题.

传统机器学习解决的过程大致包括：

1. 训练集验证集划分: 考虑到传统机器学习方法性能较优, 故采用 5-Fold Cross Validation, 训练验证比 4:1;
2. 数据预处理: 包括特征提取 (SIFT、HOG etc.)、动态聚类 (Kmeans)、降维 (PCA) 等, 将数据转化为统一的向量形式;
3. 模型分类: 通过采用不同的机器学习分类模型 (KNN、SVM etc.) 来测试分类效果.

深度学习的过程大致包括:

1. 找张 GPU: 使用 Autodl 上的 NVIDIA GTX 2080ti(11GB) 与校内 GPU 服务器的 NVIDIA RTX A5000(24GB) 训练深度学习模型;

2. 训练集验证集划分: 性能原因, 不考虑使用 k-fold, 直接划分. 训练验证比 4:1;
3. 数据预处理: 包括一系列的数据增强 (随机翻转、标准化) 以及采用跨图片增强 (训练过程中体现) 的 trick: MixUp、CutMix;
4. 训练验证模型: 对比采用不同的跨图片增强 trick 训练出来的深度学习模型 (resnet 及其变种) 在验证集上的效果.

1.4 DummyClassifier

直接采用 DummyClassifier 作 5 折交叉验证, 预测结果为训练样本中出现次数最多的类, 得出最优的 acc 为 0.036.

2 传统机器学习模型

事先说明: 本部分所有方法均采用 5 折交叉验证, 所提到的 acc 如未说明均为五折平均的 acc.

2.1 特征提取

2.1.1 SIFT 特征提取

SIFT(Scale-invariant feature transform) 中文名为尺度不变性特征变换, 在传统的 CV 算法中拥有很高的地位. 作为一个特征点的提取算法, 它对尺度、旋转、关照等变化不敏感, 导致其效果较优.

首先我们有必要弄明白特征点: 对于平滑的区域一般变化不大, 这不是我们关心的地方, 我们关心的是纹理复杂的地方, 例如边缘、点、角之类的区域, 而这些灰度值变换大的地方就是我们要的特征点.

算法的第一步是数据预处理. 由于彩色图是三通道的, 我们需要先将其转化为灰度图, 此时灰度图为单通道, 灰度值应该在 0-255 之间分布.

算法的第二步是构建多尺度 DoG 空间. 这里的多尺度代表对不同分辨率的图片作空间构造. 对于单个分辨率的图片, 分别作六次不同方差高斯模糊后的图像:

$$\sigma, k\sigma, k^2\sigma, k^3\sigma, k^4\sigma, k^5\sigma.$$

对于特定的方差 σ, 高斯模糊后的图像的空间灰度函数为:

$$G(x_i, y_i, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x-x_i)^2 + (y-y_i)^2}{2\sigma^2}\right)$$

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y).$$

然后需要获取高斯差分函数 (DoG). 高斯差分图像是某一相同分辨率的相邻图像作

差值得出, 然后与原图像 $I(x,y)$ 作卷积得到 DoG 函数:

$$\begin{aligned} D(x,y,\sigma) &= [G(x,y,k\sigma) - G(x,y,\sigma)] * I(x,y) \\ &= L(x,y,k\sigma) - L(x,y,\sigma). \end{aligned}$$

这样, 对于不同的分辨率, 我们有 5 个 DoG 函数. 事实上这就是高斯金字塔表达, 对于不同的尺度 (分辨率), 从下到上依次降采样, 长的很像一个金字塔.

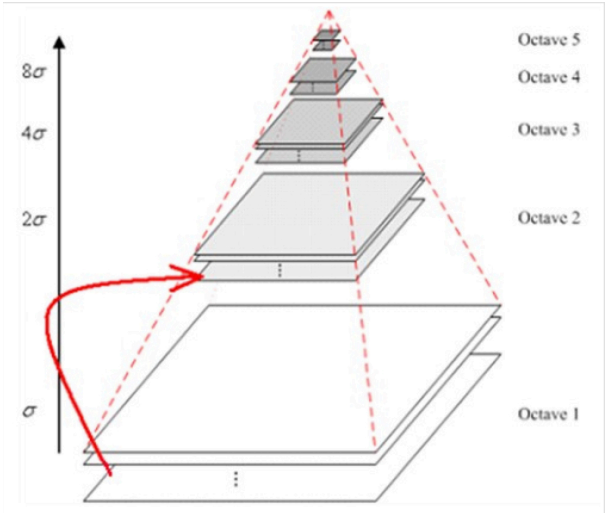


图 1: 高斯金字塔

算法的第三步是极值点检测. 寻找 DoG 函数的极值点, 最简单的想法是直接和附近点相比较. 在 SIFT 算法中我们还需要对同一组中上面和下面不同的 DoG 函数的点相比较, 这样一共需要比较 $8 + 9 \times 2 = 26$ 次, 而对于每组 5 个 DoG 函数的情况, 我们只能对中间 3 个函数进行操作.

然而这种方法找到的点的位置都是在整数空间上的. 换句话说, 这些极值点都是离散的. 因此 SIFT 算法考虑通过插值法对离散的 DoG 函数进行曲线拟合, 进一步对方程求偏导, 得到精确的极值点.

此外算法还提到了删除边缘效应的点的概念, 这里不予赘述.

算法的最后一步是产生特征点描述. 上面的过程产生的特征点只有位置信息, 我们需要通过这些特征点附近区域获取特征. 算法在检出的特征点为中心选 16×16 的空间作为特征提取空间, 然后将这些区域均分为 4×4 个子区域.

对于每个 4×4 的子区域, 计算 8 个方向的梯度方向直方图. 首先我们计算每个像素的梯度的幅值和方向, 然后对子区域进行统计, 统计 8 个方向的幅度. (类似 HOG 的特征提取)

幅度公式:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}.$$

方向公式:

$$\theta(x,y) = \arctan\left((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y))\right).$$

然后将所有子区域的幅度直方图连接起来, 即可获得特征向量, 其维度为 $4\times4\times8=128$ 维.

2.1.2 HOG 特征提取

HOG(Histogram of Oriented Gradient) 中文名为方向梯度直方图, 相比于晦涩难懂的 SIFT, 这个算法更加直接, 直接在原图的基础上产生特征向量. HOG 通过计算并统计图像局部区域内的梯度方向直方图 (SIFT 部分已经介绍过了) 来形成特征, 这导致它对图像几何和光学的变化不敏感.

与 SIFT 算法一样, HOG 也需要先将原图片转换为灰度图, 因为 HOG 提取的是纹理特征, 颜色信息并无作用.

第二步是划分子区域, 在 HOG 中叫做划分 cell, 本课设中每个 cell 为 8×8 个像素, 且相邻 cell 不重叠. 然后我们对每个像素计算其梯度幅值和方向, 将所有方向分为 9 个块, 在每个 cell 内统计梯度方向直方图.

第三步, 我们需要将多个 cell 组合成更大的连通块 (block), 将 block 内的所有 cell 的特征向量串联起来得到 HOG 的特征描述. 这里相邻的 block 之间可能重叠. 在更大的范围内统计梯度直方图, 并做归一化处理. 本课设采用 L2-norm normalization:

$$v=\frac{v}{\sqrt{|v|_2^2+\epsilon^2}}$$

最后, 假设每个 block 包括 2×2 个 cell, 相邻的 block 之间有 1 个 cell 宽度的交集. 那么对于我们分辨率为 224×224 的图片, 所产生的特征向量的维度应该为: $(224/8-1)^2\times9\times4=26244$ 维, 过于庞大. 因此我们先将图片转换为 128×64 分辨率 (因为论文就这么搞的), 这样会得到一个 $(128/8-1)\times(64/8-1)\times4\times9=3780$ 维的特征向量, 依旧庞大, 后续会进行降维处理.

2.2 特征预处理

2.2.1 KMeans 动态聚类

聚类分析又称群分析，它是研究对样品或指标进行分类的一种多元统计方法. 聚类分析要使得同一类中的对象之间的相似性比与其他类的对象的相似性更强, 目的在于使同类间对象的同质性最大化和类与类间对象的异质性最大化.

一般地, 聚类分析包括系统聚类和动态聚类. 系统聚类的缺点是每次合并不同类的时候, 结果就已经固定了. 相比系统聚类, 动态聚类法 (即 KMeans) 会根据需求, 动态地调整分类方式.

KMeans 聚类方法最简单的表达如下图所示:

KMeans 常用的一种方法是按批修改法, 它的修改原则是使如下的分类函数逐渐减

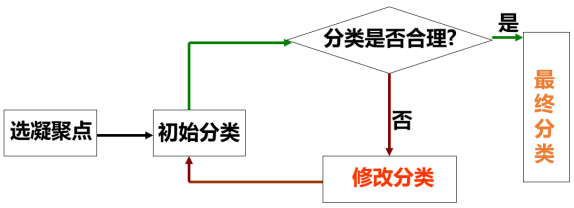


图 2: KMeans 大致过程

小:

$$l(G_1, \cdots, G_k) = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (\boldsymbol{x}_j^i - \overline{\boldsymbol{x}}^i)^\top (\boldsymbol{x}_j^i - \overline{\boldsymbol{x}}^i)$$

暴力枚举的话, 枚举复杂度为:

$$S(n,k) = \frac{1}{k!} \sum_{i=1}^{k} (-1)^{k-i} C_k^i k^n.$$

为 NP-hard 问题, 因此 KMeans 常采用迭代的方式求解.

迭代算法流程如下:

1. 令 $t = 0$, 随机选择 k 个样本点作为初始聚类中心 $m^{(0)} = (m_1^{(0)}, \cdots, m_k^{(0)})$;
2. 对样本进行聚类. 对固定的类中心 $m^{(t)} = (m_1^{(t)}, \cdots, m_k^{(t)})$, 计算每个样本到类中心的距离, 将每个样本指派到与其最近的中心的类中, 构成聚类结果 $C^{(t)}$;
3. 计算新的类中心. 对上述聚类结果 $C^{(t)}$, 计算当前各个类中的样本的均值, 作为新的类中心 $m^{(t+1)}$;
4. 若迭代收敛或聚类结果符合停止条件, 输出 $C^* = C^{(t)}$. 否则返回第二步.

由于对于每个样本图片, 我们得出的 SIFT 的特征向量的个数是不定的. 为了进一步将图片转化为单个特征向量, 本课设将训练集中所有图片的所有 SIFT 特征向量进行 KMeans 聚类 (共 64 类), 然后使用词袋模型将其转化为特征向量.

2.2.2 词袋模型

词袋模型 (Bag of Words Model, BoW) 是 nlp 领域中常用到的概念. 对于每个词, 它都对应着一个 one hot 向量, 然后一个句子就可以通过这些 one hot 向量的叠加来表示.

在本课设中, 我们将一张图片的所有 SIFT 特征在聚类中最近的类作为 word, 然后采用 one hot 向量的叠加来构造其特征向量.

2.2.3 PCA 降维

关于主成分分析, 我大二下写过一篇文章, 可以作为参考.

在提取 HOG 特征向量时, 我们发现特征向量维度过高, 若和 KNN、SVM 搭配使用会使占用内存过高, 另外有多余无效特征或相互线性强相关的特征可以去除, 于是考虑使用 PCA 进行降维.

2.3 分类器

2.3.1 KNN

我们首先采用 k 近邻分类器进行分类测试. 在这里, 我们使用 5 折交叉验证, 对于所有 $k \in [1, 20], k \in \mathbb{N}$, 记录平均 acc 信息, 得出结果如下:

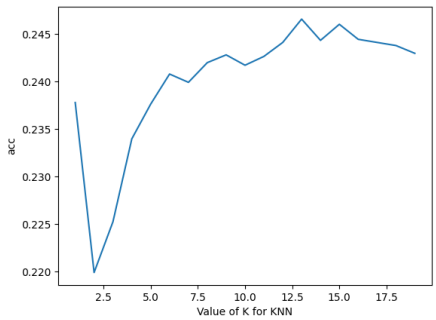


图 3: KNN4SIFT

当 $k = 13$ 的时候, 使用 sift 提取特征的 knn 能够达到平均 0.247 的正确率. 其次是 $k = 15$ 的时候能够到达 0.246 的正确率. 总的效果并不是很好.

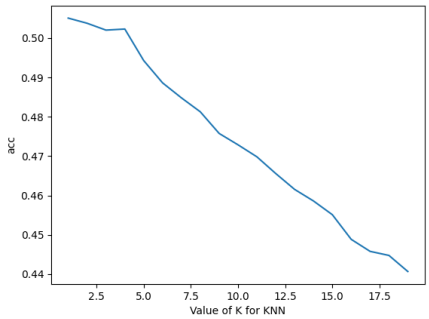


图 4: KNN4HOG

对于 HOG 的情况, 令人震惊的是它竟然在 $k = 1$ 的情况最优, acc 达到了 0.505, 而后面的 acc 相对于 k 呈递减趋势, 相比于 sift 有了巨大进步, 这可能是由于 sift 特征在转化为单一特征向量的时候经过了聚类和词袋化, 再加上图片本身的特征提取也没有 hog 那样稳定, 导致转化为特征向量的时候经过了三层的信息损失, 而 hog 特征提取只经过了 pca 降维, 导致图片特征保存地较为完全.

2.3.2 SVM

支持向量机是传统机器学习内最为著名的分类器之一, 相比于占用内存空间大的 KNN, SVM 的分类结果仅与支持向量有关, 因此占用内存较小. 在对 HOG 进行 pca 降维时可以考虑扩大信息.

model	Fold1	Fold2	Fold3	Fold4	Fold5	mean	max
svm4sift	0.352	0.320	0.321	0.345	0.328	**0.333**	0.352
svm4hog	0.580	0.543	0.564	0.554	0.533	**0.555**	0.580

表 1: svm

在本课设中, 我们采用高斯核, 并设置软间隔参数 $C = 1$, 以防止过拟合.

上表给出了 svm 在 sift 和 hog 特征上的表现, 可以发现相比于 KNN 分类器都有较大的提升.

3 深度学习模型

3.1 数据预处理

3.1.1 数据增强

由于深度神经网络极强的拟合能力, 导致其容易对某些不必要的特征过拟合, 因此对图片进行增强是很有必要的, 它能减少模型对某些属性的依赖, 从而提高网络的泛化能力.

由于本课设用到的模型全都是在 ImageNet 上预训练过的, 因此需要对图像同步在 ImageNet 上的标准化.

另外由于树叶的特性, 我们可以进行随机翻转、随机旋转等操作. 注意这些操作是在训练的时候对训练集进行的, 这样的话每个 epoch 的训练集由于数据增强的不确定性会造成不同, 这能够进一步提高模型的泛化能力.

3.2 模型与参数

本课设采用了 resnet34 及其三个变种: resnext50[1], resnest50[2] 和 densenet161[3], 均为预训练模型. 超参数设置如下:

Model	Learning rate	Weight decay	Batch size	Num epoch
resnet34	10^{-4}	10^{-3}	32	30
resnext50	10^{-3}	10^{-3}	64	50
resnest50	10^{-4}	10^{-3}	32	30
densenet161	10^{-4}	10^{-3}	32	30

表 2: hyper parameters setting

3.3 Trick

事实上这部分也可以被认为是数据预处理的范畴, 因为它们是根据跨图片增强的概念提出来的 idea, 然而可以通过修改损失函数来实现同样的效果.

3.3.1 MixUp

MixUp[4] 是一种双图片增强, 在原论文提出的核心公式为:

$$\tilde{\boldsymbol{x}} = \lambda \boldsymbol{x}_i + (1-\lambda)\boldsymbol{x}_j$$

$$\tilde{\boldsymbol{y}} = \lambda \boldsymbol{y}_i + (1-\lambda)\boldsymbol{y}_j.$$

事实上就是一种双图片按权叠加, 但是可以通过修改损失函数来实现同样效果:

$$\text{loss}^* = \lambda \text{criterion}_i + (1-\lambda)\text{criterion}_j.$$

需要注意这里的 MixUp 同样是在训练的时候进行的, 而不是提前预处理好训练数据.

3.3.2 CutMix

CutMix[5] 是一种多图片增强, 相比于 MixUp 在灰度值上的按权叠加, CutMix 更为暴力, 它直接在原图片上进行裁剪, 然后将新图片对应的区域直接贴进去.

而对于标签的修改同 MixUp. 因此损失函数同样可以写为:

$$\text{loss}^* = \lambda \text{criterion}_i + (1-\lambda)\text{criterion}_j.$$

这里 λ 为图片 i 所占新图片的面积占比. 该操作同样在训练中完成.

3.4 模型效果

由于该部分没有使用 k 折交叉验证, 因此对于每一个模型与 trick 的组合只有一个 valid acc 数据:

Model	Common	MixUp	CutMix
resnet34	0.943	0.942	**0.951**
resnext50	0.947	0.943	**0.952**
resnest50	0.960	0.960	**0.969**
densenet161	0.959	0.956	**0.965**

表 3: best valid accuracy

下图是本课设的 valid acc 曲线:

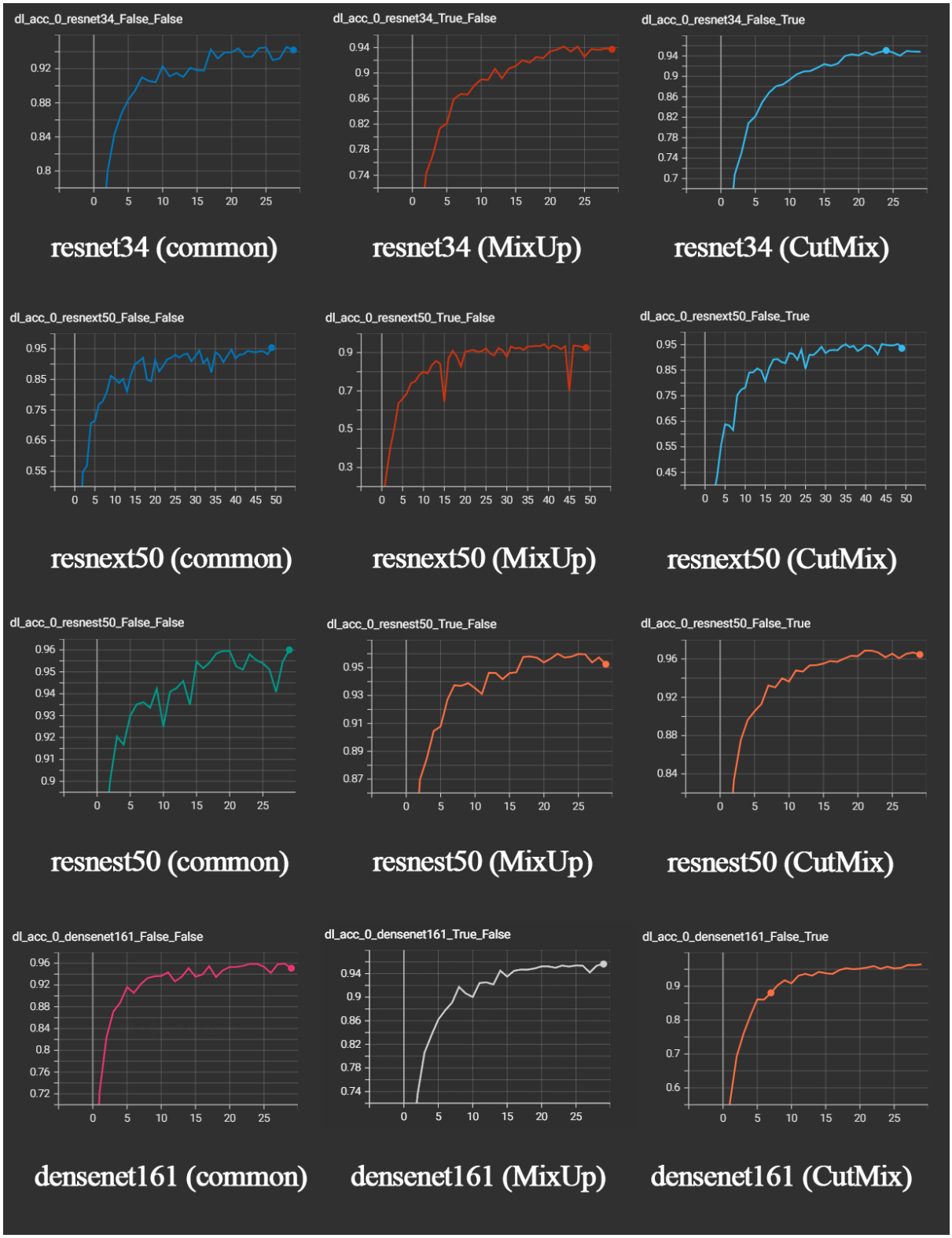


图 5: valid acc curve (by TensorBoard)

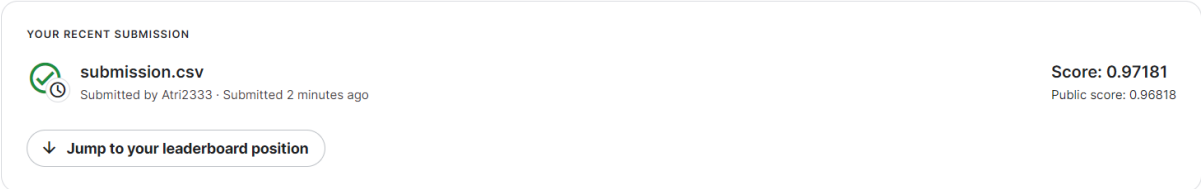
根据实验结果, 我们发现 MixUp trick 对于模型精度其实并没有提升, 相反 CutMix trick 对于模型精度有大约接近 0.01 的提升. 这可能与树叶的特性有关, 叠加树叶的灰度信息或许并不利于特征提取, 而直接叠加二者的纹理信息可能有助于网络进一步提取更高阶的特征.

在所有的模型中, resnest50 的综合表现最佳, resnext50 由于独特的网络设计 (以及我故意设的高学习率) 导致它的 acc 曲线较陡, 而即使是最原始的 resnet34, 它在验证集上的精度也远高于前面的 KNN 和 SVM 分类器在人工提取特征上的效果. 说明传统机器学习模型在这种超过一百余类的分类任务中还是不尽人意的.

3.5 集成 + 推理

我们采用 resnest50、resnext50 和 densenet161 在使用 CutMix Trick 下的训练好的模型为我们最终的提交模型. 这里我们采用投票的方式进行集成, 以验证正确率最高的 resnest50 为优先级最高的模型, 其余二者其次.

ensemble 后在 kaggle 上的评测成绩如下:



达到了 0.972 的 acc, 在并没有仔细调参, 叠 buff (trick) 的情况下还是可以接受的.

参考文献

[1] XIE S, GIRSHICK R, DOLLáR P, et al. Aggregated residual transformations for deep neural networks[A]. 2017. arXiv: 1611.05431.

[2] ZHANG H, WU C, ZHANG Z, et al. Resnest: Split-attention networks[A]. 2020. arXiv: 2004.08955.

[3] HUANG G, LIU Z, VAN DER MAATEN L, et al. Densely connected convolutional networks[A]. 2018. arXiv: 1608.06993.

[4] ZHANG H, CISSE M, DAUPHIN Y N, et al. mixup: Beyond empirical risk minimization[A]. 2018. arXiv: 1710.09412.

[5] YUN S, HAN D, OH S J, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features[A]. 2019. arXiv: 1905.04899.