梁子的周报-08-24

日期：August 24, 2020

目录

1 关于使用 sketch 作为核方法提升 SVM 精度的问题

上一周在做实验的时候，发现基于 sketch 的核方法效果没有线性分类器好，这个结果与预期不符。因此又重新做了一系列实验，终于搞清楚了这个问题。

1.1 SVM 基本思路

SVM 的基本思路是在特征空间中找到一个超平面将数据点进行分割。但是数据往往不是线性可分的，所以线性 SVM 的效果在现实数据集中表现并不理想（因为现实数据并不是线性可分）。

核方法是为了解决这种问题的一种思路：既然当前特征维度下数据不是线性可分，那么可以将特征映射到更高的维度，这样就可以构造出线性可分的超平面了。因而效果就会比较好。

1.2 把 sketch 作为非线性核的处理手段

sketch 方法原本的应用场景是：

1. 数据降维。生成短小紧凑或者大而稀疏的特征向量，以节省内存；
2. 快速使用。sketch 可以作为中间过程快速应用在近似搜索等问题上；
3. 应对特征维度不同的数据，与其他方法不同，sketch 方法可以将数据映射成相同长度的“指纹”；

而在本工作的场景下，sketch 除了具有上述的特征 1 之外，本身还充当了一种非线性核方法。

1.3 基于 sketch 的核方法分类精度与线性分类器的分类精度比较

在上述两条的基础上，应有：

1. 如果 sketch 方法没有降低原始特征的维度，也没有使用诸如 b-bit 的方法节省内存，那么得到的精度要高于线性分类器的分类精度；
2. 显然，sketch 方法如果进行了降维或 bbit 这一类的手段，精度则会比不降维的 sketch 要差；
3. 使用了降维的 sketch 方法在高维数据上的表现与线性分类器相比，具体的结果取决于压缩的比率。

我的实验结果和雷润泽的不一样，经过思考，主要原因是：数据集中特征维度不一致。这是雷润泽使用的数据集、特征维度、sketch 长度：

dataset	feature	k
pendigits	16	32,64,128,256,512,1024,2048
satimage	36	32,64,128,256,512,1024,2048
segment	19	32,64,128,256,512,1024,2048
splice	60	32,64,128,256,512,1024,2048
letter	16	32,64,128,256,512,1024,2048

我没有对所有的数据集进行重新实验，单单放上最后一个数据集的实验结果：

method	acc	feature	备注
linear svm	66.3	16	
gpmh-kernel+svm	94.96	512	对生成的 sketch 每一个 bucket 用 6-bit 近似存储
gpmh-kernel+svm	49.18	16	同上

而按照微信群里的要求，我的数据集与基本的维度特征是这样的：

dataset	feature	k
rcv1	47,236	32,64,128,256,512,1024,2048
kdda	20,216,830	32,64,128,256,512,1024,2048

面对这样的高维数据集，在 sketch 长度没有增长的前提下，似乎精度的下跌也是无法避免的。在 rcv1 上进行实验的结果如下：

method	acc	feature	备注
linear svm	97.67	47,236	
pmh-kernel+svm	91.83	64	对生成的 sketch 每一个 bucket 用 6-bit 近似存储
pmh-kernel+svm	95.4	128	同上
pmh-kernel+svm	96.35	256	tongshang
pmh-kernel+svm	96.28	512	tongshang
cws-kernel+svm	94.43	128	tongshang

从表中可以看出：

1. sketch 的方法很大的降低了特征维度，同时精度的降低并不是特别大。
2. 时间原因，上述实验只进行了一次，不排除一些偶然性。比如 pmh 下 512 维得到的效果不如 256 维好。

综上所述，实验的结果和理论的分析是互相验证的。

其他的一些实验参数设定：

参数	描述
k	k 就是 hash 的次数，也就是 sketch 的长度，也就是上面使用了 sketch 方法的 feature 个数
b	每个 bucket 的存储位数
c	svm 的一个超参数，实验中全部用的 1

本来试图网格化参数放在服务器上跑，结果直接卡住了，在李、张、程学长的帮助下才解决。。耽误了一些时间。因此变为手动增加线程去做。还有一些维度比较高的实验没有出来结果，随时补充。

除此之外：对于 sketch 降维会降低精度的问题，雷润泽曾经讲过一篇论文，flyhash，是将数据映射到高维稀疏空间的仿生 sketch 方法。这个方法没有降维，那会不会有较好的效果呢？虽然会提升特征的维度，但是后面 W 的乘法操作也就变成了加法，或许也不会增加后面的负担。

2 对“可解释性”这个想法的验证

2.1 代码进度

实现这个思路主要是三步：

1. sketch-kernel 与 linear svm 的实现【已完成】
2. 将 batch learning 改为 online learning 【正在做】
3. 添加上新的互学习的部分【没完成】

目前做到了第二步，由于接手的代码 svm 部分直接调用的 liblinear 库，而那个库默认就一个 train() 函数，因此修改为 online learning 模式需要对 C++ 代码进行一些修改。我对 C++ 不是很熟悉，所以看起来稍稍有点慢。并且，对于参数的优化该库使用了 newton 迭代法，这方法明明是用来求解非线性方程组的，我还没看明白。因此，由于语言的原因，这里的进度会稍稍慢一点。（不过也不能躲避，C++ 是数据挖掘里那么常用的语言）而根据我对 online learning 的调研，以及对 SIGMOD2018 那篇论文源代码的阅读，基本的改进方法明白了。

对于第三步，我阅读了互学习那篇论文，同知识蒸馏类似，该论文是在交叉熵损失函数的基础上提出的。而目前 SVM 使用的损失函数并不是交叉熵。如何去适配是将来需要思考的。

3 online learning 的调研

还没弄完。。这周日晚上是我的小报告，我准备就到时候把它汇报了吧。0.0