自然语言理解大作业
——基于 HMM 的分词以及基于 SVM 的文本分类

冯亚伟 陈呈辉 祥保庆

一. 作业框架

基于 SVM 实现文本分类的主要步骤包括：1、文本分词处理，2、特征选择，3、特征权重计算，3、文本特征向量表示，4、基于训练文本的特征向量数据，训练 SVM 模型，5、对于测试集进行特征向量表示，代入训练得到的 SVM 模型中进行预测分类。

在这次大作业中，我们使用 HMM 模型实现了一个分词程序，其中使用了 Python 的 numpy 库，然后对搜狗语料库中的文本进行分词处理。在这里我们对搜狗预料库进行了裁剪，只选择了 2200 个文本，每个类别 220 个文本，其中前 200 个文本用作训练，后 20 个文本用作测试。在这次大作业中，我们实现的分词程序的正确率在 80%左右，具体分词正确率结果在 ChineseSegmentation\PKU_GB\score.txt 文件中

特征选择我们采用的是开方检验的算法,选择的特征在 SVMFeature.txt 文件中，每个类别选取 1000 个特征，10 个类别 10000 特征，由于重复，计算出来的特征为 9508 个。

特征权重计算，采用的是 TF*IDF 计算算法，训练文本的特征向量表示数据在 train.svm 文件中，测试文本的特征向量表示数据在 test.svm 中。

对 train.svm 对于模型训练，和对于 test.svm 模型预测，使用的是 LIBSVM 库，链接为 http://www.csie.ntu.edu.tw/~cjlin/libsvm/，文本分类正确率为 164/200=0.82。文本分类具体结果在 LIBSVM 文件夹中,具体测试命令在这个文件夹下的“测试命令.txt”文件中。

二. 背景知识

1. 隐马尔科夫模型(HMM)

HMM 是在 Markov 链基础上发展起来的,是一个双重随机过程,其中之一是 Markov 链,是基本的随机过程,描述状态的转移;另一个随机过程描述状态和观察值之间的统计对应关系。站在观察者的角度,只能看到观察值,不能直接看到状态,是通过一个随机过程去感知状态的存在及特性。因此称为“隐”Markov 模型,即 HMM。

HMM 可以记为 $\lambda=(N,M,\pi,A,B)$,简写 $\lambda=(\pi,A,B)$。更形象的说,HMM 分为两部分:一个 Markov 链,由 π, A 描述, 产生的输出为状态序列;另一个随机过程,由 B 描述,产生输出观察值序列。

HMM 模型具体可以由下列参数描述:

(1) N :模型中 Markov 链状态数目。记 N 个状态为 S1,,SN ,记 t 时刻 Markov 链所处状态为 qt ,显然 qt ∈(S1,,SN)。

(2)M :每个状态对应的可能的观察值数目。记 M 个观察值为 V1,…,VM ,记 t 时刻的观察值为 Ot , Ot ∈(V1,,VM)。

(3)π:初始状态概率矢量,π=(π1,,πN)。其中,

$$\begin{cases} \pi_i = P(q_1 = S_i), 1 \le i \le N, \\ \pi_i \ge 0, \\ \sum_{i=1}^{N} \pi_i = 1 。 \end{cases}$$

(4)A 为状态转移概率矩阵,A=(aij)N×N 。其中,

$$\begin{cases} a_{ij} = p(q_{t+1} = S_j / q_t = S_i), 1 \le i, j \le N, \\ a_{ij} \ge 0, \\ \sum_{j=1}^{N} a_{ij} = 1 。 \end{cases}$$

(5)B 为观察值概率矩阵,B=(bjk)N×M 。其中,

$$\begin{cases} b_j(k) = p(O_t = V_k / q_t = S_j) = p(V_k \mid S_j), 1 \le j \le N, 1 \le k \le M, \\ b_j(k) \ge 0, \\ \sum_{k=1}^{M} b_j(k) = 1 。 \end{cases}$$

应用 HMM 模型的步骤如下：

(1)得到观察序列 O = O1, …, O n 和模型 λ = (π, A, B), 利用前向-后向算法快速计算出在该模型下,观察事件序列发生的概率 P(O/λ)。(评估问题)。

(2)利用 Viterbi 算法选择对应的状态序列 S = q1 ,, qn ,使 S 能够合理的解释观察序列 O 。即揭开模型的隐含部分,在优化准则下找到最优状态序列。(解码问题)。

(3)利用 Baum-Welch 算法调整模型参数 λ = (π, A, B) , 即得到模型中的五个参数,使得 P(O / λ) 最大。(学习问题)。

2．支持向量机方法（SVM）

SVM 方法适合大样本集的分类,特别是文本分类,它将降维和分类结合在一起。SVM 算法基于结构风险最小化原理,将原始数据集合压缩到支持向量集合,然后用子集学习得到新知识,同时也给出由这些支持向量决定的规则。并且可得到学习错误的概率上界。假设线性分类面的形式为:

$$g(D)=\omega\cdot D+b=0$$

其中 w 为分类面的权系数向量,b 为分类阈值,可用任一支持向量求得,或者通过两类中任一对支持向量取中值求得。将判别函数归一化,使得所有样本都满足 $|g(D)|=1$,即 $y_i[(\omega\cdot D_i)+b]-1\geq 0,i=1,2,...,N$,Yi 是样本的类别标记,即当样本属于类 C 时 Yi=1,否则 Yi=-1;Di 是相应的样本。这样样本的分类间隔就等于 $2/\|\omega\|$,设计的目标就是要使得这个间隔值最小。据此定义 Lagrange 函数:

$$L(\omega,b,\alpha)=\frac{1}{2}(\omega\cdot\omega)-\sum_{i=1}^{n}\alpha_i\{y_i[\omega\cdot D_i+b]-1\}$$

其中为 ai>0 为 Lagrange 乘数,对 w 和 b 求偏微分并令其为 0,原问题转换成如下对偶问题:

在约束条件 $\sum_{i=1}^{n}y_i\alpha_i=0,\alpha_i>0,i=1,2,...,n$ 下对 ai 求解下列函数的最大值:

$$Q(\alpha)=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j(D_i,D_j)$$

如果 α_i^* 为最优解，再由公式

$$\omega^*=\sum_{i=1}^{n}\alpha_i^*\times y_i D$$

得出最优分类面的权系数向量。为了判断某个样本是否属于类 C,计算如下最优分类函数:

$$f(D)=sign\{(\omega^*\cdot D)+b^*\}=sign\{\sum_{i=1}^{n}\alpha_i^* * y_i(D_i\cdot D)+b^*\}$$

若 f(D)=1,D 就属于该类;否则就不属于该类。对于线性不可分的情况,可以引入松弛因子,在求最优解的限制条件中加入对松弛因子的惩罚函数。

三．具体实现

1. 基于 HMM 的分词程序

这个方法的基本思想是把输入字串(String)S 作为 HMM：m的输入;切分后的单词串 Sw 为状态的输出，即观察序列。字的性质 Sc 为状态序列，其标注有四种情况：B 词开头，M 词中间，E 词结尾，S 符号。分词程序可分成两步骤：

1.1、训练隐状态之间的转移概率矩阵和状态发射矩阵。

这部分代码位于 ChineseSegmentation/PreHMM.py 中。前者的函数为 trainTransProb(trainFileName, tranFileName)，即算出状态序列四种情况两两之间的转移概率，采用简单的统计即可。其结果矩阵如下：

-10000000000000000	-1.91884919336	-0.15873490297	-10000000000000000
-10000000000000000	-1.06226695342	-0.424145455474	-10000000000000000
-0.720477807421	-10000000000000000	-10000000000000000	-0.666543687887
-0.557415611005	-10000000000000000	-10000000000000000	-0.850243439952

后者的函数为 trainEmitProb(trainFileName, emitFileName)，其结果即为 BMES 四种状态发射到字典（worddict.txt）中每个字的概率。

1.2、读取上述参数，基于 Viterbi 算法对 String 进行切分，并计算分词准确率。

这部分代码位于 ChineseSegmentation/Viterbi.py 中。核心函数即是 cutResult = viterbi(cutStr, IniProb, TransMatrix, EmitMatrix, WordDict)。其中， WordDict（字典）为整理得到，TransMatrix、EmitMatrix 为上述步骤得到，cutStr 为待分词的句子，IniProb 为初始转移概率。

2. 基于 SVM 进行文本分类

如上所述，这次文本分类中，我们做了如下步骤：

2.1 特征选择

特征选择我们选择用卡方统计方法。这部分代码位于 FeatureSelecion.py 中。在构建每个类别的词向量后，对每一类的每一个单词进行其卡方统计值的计算。首先对卡方检验所需的 a、b、c、d 进行计算。其中，a 为在这个分类下包含这个词的文档数量；b 为不在该分类下包含这个词的文档数量；c 为在这个分类下不包含这个词的文档数量；d 为不在该分类下，且不包含这个词的文档数量。然后得到该类中该词的卡方统计值，公式为 float(pow((a*d - b*c), 2)) /float((a+c) * (a+b) * (b+d) * (c+d))。对每一类别的所有词按卡方值进行排序，取前 k 个作为该类的特征值，这里我们取 k 为 1000。10 个类别 10000 特征，由于重复，计算出来的特征为 9508 个。

2.2 特征权重计算

采用的是 TF*IDF 计算算法，它的优点是每个词的权重与特征项在文档中出现的频率成正比，与在整个语料中出现该特征项的文档数成反比。训练文本的特征向量表示数据在 train.svm 文件中，测试文本的特征向量表示数据在 test.svm 中。这个逻辑比较简单，就是套公式就好。

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

2.3 基于 libsvm 库进行文本分类

对上述的按 libsvm 要求整理好的文件进行训练和测试。具体为对 train.svm 对于模

型训练，对于 test.svm 模型预测，使用的是 LIBSVM 库，文本分类正确率为 164/200=0.82。文本分类具体结果在 LIBSVM 文件夹中,具体测试命令在这个文件夹下的测试命令.txt 文件中。

四．分工合作

冯亚伟同学独立负责了分词部分。陈呈辉同学和祥保庆同学负责了后续步骤，冯亚伟参与了 debug 和完善。冯亚伟同学的别名为代码第一行的 auther 中的 shadowwalker。最后一同合作完成了报告书写。

每个人的联系方式如下：

冯亚伟 中科院网络中心 1450596322@qq.com

陈呈辉 中科院网络中心 369131259@qq.com tel:13261562202

祥保庆 中科院网络中心 1416398057@qq.com

五． 参考文献

1. 基于隐马尔可夫模型(HMM)的人脸表情识别，王冲鹊，通讯技术，2007

2. SVM 在文本分类中的应用，叶志刚，硕士论文，2006

3. 统计自然语言处理，宗成庆，清华大学出版社，2013

4. http://www.csie.ntu.edu.tw/~cjlin/libsvm/，林智仁，台湾大学，libsvm 库

5. 零零碎碎从互联网上看了很多博客、文档，这里就不一一列举了，对他们表示诚挚的感谢。