基于 PPM 算法的垃圾邮件过滤方法①

Spam Email Filtering Based on PPM Algorithm

王海晓 彭 鹏 徐从富 (浙江大学 计算机学院 浙江 杭州 310027)

摘 要: 本文在简要介绍 PPM 数据压缩算法及其改进的基础上，着重论述该算法在垃圾邮件过滤中的应用。首先将样本邮件进行文本预处理，并对正常邮件和垃圾邮件训练集进行训练，分别建立上下文模型；然后输入待过滤邮件，与 PPM 压缩模型进行比较，分别计算交叉熵以判断邮件类型；最后，测试结果表明，该算法达到较好效果。

关键字: 垃圾邮件过滤 PPM 数据压缩 上下文模型交叉熵

1 引言

电子邮件已成为人们通讯的重要工具，但日益泛滥的垃圾邮件正严重地影响着人们的工作和学习。据中国互联网协会反垃圾邮件中心(http://www.anti-spam.cn/)最新发布的 2007 年第四季度反垃圾邮件状况调查报告中，用户平均每周发送 12.63 封邮件，而平均每周收到 16.71 封垃圾邮件，且 45.29%企业用户认为反垃圾邮件产品防范效果不明显，因此反垃圾邮件技术的研究越来越受到人们的关注。

传统的黑白名单、关键词过滤、邮件路由等反垃圾邮件技术，面临动态 IP 跟踪难、过滤率低、误判率高、网络流量大、规则维护工作量大的技术瓶颈[1]。因此，目前的垃圾邮件过滤系统逐渐倾向于引入基于内容的机器学习判别方法。基于内容的垃圾邮件判别方法大体分为基于规则的方法，如 Ripper、决策树、Boosting、粗糙集等方法和基于概率统计的方法，如贝叶斯过滤、SVM、winnow、Logistic 回归等[2]。其中，基于字节熵的 PPM（Prediction by Partial Matching）[3]、DMC 数据压缩算法[3]在垃圾邮件过滤中得到了很好的实验结果，具有广阔的应用前景。

本文在简要介绍 PPM 及其改进算法原理的基础上，将 PPM*算法应用于垃圾邮件过滤，通过建立上下文模型，计算交叉熵以预测邮件的类型，得到了较好的实验结果。

2 PPM算法及其改进[6,7]

2.1 PPM 算法原理

PPM 是一种基于上下文统计模型的技术，由 Cleary 等[4]在 1984 年首次提出，而后 Moffat[5]对 PPM 算法做了许多改进，在过去 10 年已成为无损压缩的性能标准。

PPM 的基本思想就是利用最近输入的几个字符(叫做上下文模型)，来预测下一个字符。其中，上下文模型的长度 k 可以从 0 到已输入字符的最大长度 km 不等。对于 k 长度的上下文模型来说，首先要计算在输入字符串中，每个 k 长度的字符串后面不同字符出现的次数，然后可以得到该上下文模型的预测概率。得出预测概率后，用算术编码对该字符进行编码。而且 PPM 是一种自适应技术，每个上下文模型的预测概率将会随着输入字符串的改变而随时改变。

定义一个已知字符串模型的最大长度为 k，当下一个输入的字符已经由该上下文模型预测出时，该输入字符出现的概率就是预测概率。而当下一个字符不能由上下文模型预测出时，输入字符的概率无法得到，这时，就需要用到“逃逸(escape)”概率。添加逃逸上下文预测模型是为了防止零概率字符的情况出现。“逃逸”概率将模型从 k 跳到 k-1，如果 k-1 长度的模型能预测出该字符，就能得到相应的预测概率。如果不能，该过程将一直进行直到某个模型可以预测出该字符。逃逸上下文预测模型概率的计算方法有多种，

① 基金项目:国家 863 计划项目(2007AA01Z197)
收稿时间:2008-08-25

每一种方法对应 PPM 一种类型，如 PPMA、PPMC、PPMD、PPMP 等。

在实际应用中人们还发现 k 值为 4 或 5 时，PPM 的性能达到最高。这是由于 k 值越大，虽然预测模型增加了，但也增加了新字符的逃逸次数，而逃逸会使编码的效率降低。

2.2 改进的 PPM*算法

PPM 使用的是最大长度固定的上下文模型，效率不是最高。改进的 PPM 算法，允许上下文模型的长度可以根据编码的情况进行改变，可以在预测下一个字符的时候，选择合适的上下文模型，从而可以减少“逃逸”的次数，提高算法的效率。这种使用可变上下文模型的算法称为 PPM*。

由于 PPM*算法允许自由选择上下文模型，它的一个主要的问题就是，怎么选择预测的上下文模型。使用“确定上下文”可以解决这个问题。“确定上下文”就是指该上下文只能预测一个字符。选择可变上下文模型有一个主要问题，就是需要大量的内存和时间。在实际应用中 PPM 算法中的 k 值不能太高，因为所需的内存和时间会按指数规律急剧上升。而这个问题对于 PPM*算法尤为突出，因为它要访问所有相关的上下文，以确定使用哪一个确定性上下文。

解决这个问题的方法是建立上下文树。该树能够把上下文模型保存起来，并且能够用指针回溯到输入的字符串。该树的每个叶节点代表了一个唯一的上下文。因此如果要想扩展上下文，只要把叶节点向下扩展就可以了。根据输入数据更新上下文树以后，就可以利用确定性上下文得到新输入数据出现的概率，然后用算术编码对其进行编码就可以了。

3 基于PPM*算法的垃圾邮件过滤

3.1 最小交叉熵分类法

本质上，压缩算法应用到文本分类中，是通过训练文本建立一个压缩模型，然后使用这些模型去评估目标文件。分类标准有以下两种方法，最小交叉熵(minimum cross-entropy)方法和最小描述长度(minimum description length)方法。根据在 SEWM2008 大规模中文垃圾邮件比赛中使用 PPM 的实验结果[8]，最小交叉熵方法获得了比最小描述长度更好的结果，所以本系统采用了最小交叉熵作为文本压缩模型与待分类文本的比较方法。

交叉熵 H(X,M)决定了在使用压缩模型 M 对源 X 生成的信息进行加密时候对于每个字符所需要的平均比特数：

$$H(X,M)=E_{x\sim P}\left(\frac{1}{|x|}L(x|M)\right) \tag{1}$$

公式(1)中，交叉熵期望源 X 在概率分布为 P 时，获得相应的值。L(x|M))表示在 M 模型下字符串 x 的理想编码长度，需要注意的是，H(X,M)≥H(X)总是成立的，这是因为在最理想的条件下，最好的可能模型可以达到等同于熵的压缩率。

精确的交叉熵很难计算，因其必须知道源 X 的概率分布 P。不过，当假设运用模型 M 拥有足够长的字符序列，并且期望这些序列代表了所有可能由源产生的系列样本时，公式(1)可以近似成如下公式：

$$H(X,M)\approx\frac{1}{|x|}L(x|M) \tag{2}$$

当|x|越大，假设源遍历各种类型，那么这个估值将接近于真实的交叉熵。假设马尔科夫模型 M 的上下文长度为 k，得到

$$L(x|M)=-\log\prod_{i=1}^{|x|}f(x_i|x_{i-k}^{i-1},M) \tag{3}$$

其中，$f(x_i|x_{i-k}^{i-1},M)$ 是在给定模型 M 和 x_{i-k}^{i-1} 的情况下赋予 xi 的概率值。

根据 Cleary 等人[9]的工作，把在目标文档中 d 的交叉熵估计为文本交叉熵 H(X ,M, d)。其实就是将公式(2)中的 x 替换为 d。我们期望目标文档获得的较小交叉熵模型，近似在真实数据源产生的文档中所获得的熵值。

下面就是 PPM 的最终进行分类判定的函数：

$$c(d)=\arg\min_{c\in C}H(X,M_c,d)=\arg\min_{c\in C}-\frac{1}{|d|}\log\prod_{i=1}^{|d|}f(d_i|d_{i-k}^{i-1},M_c) \tag{4}$$

其中，M_c 代表了由所有的类别为 c 的训练样本训练所得的压缩模型。

在邮件过滤时，将待判定文本与两个分类模型分别使用最小交叉熵方法计算，然后将邮件判定为所得的熵值比较小的那个类别。

3.2 过滤原理

PPM 压缩算法应用到邮件过滤中，首先就是要将

测试邮件做文本预处理，见图 2 所示。

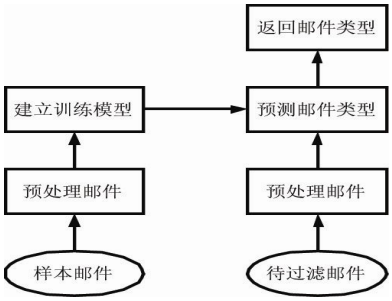

图 1　PPM 算法邮件过滤结构

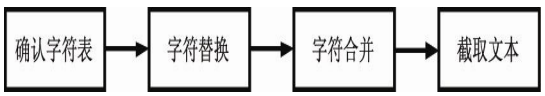

图 2　文本预处理

文本预处理是建立训练模型的基础，因为 PPM 算法建立的上下文模型需要学习全部样本邮件，需要巨大的计算资源，因此用于建模的文本如何预处理十分关键。为了减少逃逸概率，选取 ASCII 码中的 72 个字符作为字符表，包括 ASCII 码中代码从 32~127 中除了大写字母 A~Z 的字符。确定字符表后，接着做字符替换，若文本中出现字符表外的字符，用某个特殊字符进行替换。然后对文本中的换行、首行空格等空字符进行合并。对于一些较长的文本再做截取，为了获得较好的计算结果，取每封邮件的前 3000 个字符作为测试文本。

例如取任意邮件，该邮件的邮件头信息为：

Received: from webmail (unknown [156.117.187.38])

by jnv2tys0k3v2.a9hj.edu.cn (Postfix) with SMTP

id B5BF15BB03; Wed, 25 Oct 2006 13:45:11 +0800 (HKT)……

首先，替换邮件头中非字符表内的字符，如大写英文字母，然后合并空格，处理后的邮件头为：

received: from webmail (unknown [156.117.187.38])

□by jnv2tys0k3v2.a9hj.edu.cn (postfix) with smtp

□id b5bf15bb03; wed, 25 oct 2006 13:45:11 +0800 (hkt)……

假如该邮件为中文邮件，那么正文中的中文字符全部用某一特殊字符替换，但要注意保留其他非中文字符，如数字和标点符号等。

从文本预处理过程中，不难发现基于字符熵的 PPM 算法不同于基于内容的过滤算法。基于内容的过滤算法，如 Winnow 算法需要对文本内容进行解析、分词等复杂的文本训练过程[10]。PPM 算法在经过样本邮件预处理后，需要建立训练模型。把样本邮件按照索引顺序依次输入学习，分别对正常邮件训练集和垃圾邮件训练集进行训练，并由此建立 ham 上下文模型和 spam 上下文模型。然后再输入经过预处理的待过滤邮件，与 PPM 压缩模型进行比较，分别计算交叉熵，取交叉熵较小值对应的模型类型，从而进行分类。

3.3 实验结果

本实验数据集采用公共垃圾邮件语料库 SEWM2007,共 60000 封邮件参与了训练和测试试验。其中垃圾邮件 45000 封，正常邮件 15000 封，分别取 30000 封邮件作为训练和测试数据集。

本系统参照二元判别结果(垃圾邮件还是正常邮件)为原始数据，根据人为判断后得出的结果(index 文件中的判别)为答案，计算出过滤器的误过滤 ham%(正常邮件误判为垃圾邮件的误过滤率)、spam%(垃圾邮件误判为正常邮件的误过滤率)，根据 ham%、spam%得到两个指标：平均误过滤 lam%、ROCA。

(1-ROCA)%：以 ham%为横坐标，以 spam%为纵坐标，做 ROC 曲线，求 ROC 曲线上方面积。ROC 曲线下面部分的面积反映了在所有可能值上过滤器效率的一个累计度量，从而避免用单一的 ham % 或 spam % 进行衡量的局限性。两个指标数值越小，表示垃圾邮件过滤系统性能越好；最后将根据(1-ROCA)%值为最终判断，lam%作为参考。

表 2 列举了 PPM 算法与基于内容的 Logistic 回归和 Winnow 算法在邮件过滤中的性能比较。

表 2　PPM、LR、Winnow 算法在邮件过滤中的性能比较

	PPM	Logistic 回归	Winnow
ham%	1.07(0.85-1.32)	6.32(5.78-6.89)	8.77(8.13-9.44)
spam%	0.09(0.05-0.14)	0.59(0.49-0.70)	5.84(5.54-6.16)
lam%	0.31(0.25-0.38)	1.96(1.81-2.14)	7.17(6.83-7.52)
(1-ROCA)%	0.0388(0.0262-0.0575)	0.8668(0.7518-0.9992)	5.2797(4.9696-5.6081)

假设字符串 $s_1^n = s_1 \ldots s_n \in \Sigma^*$，现有文本 $d \in \Sigma^*$，该文本上下文模型长度为 k，那么在 s_{i-k}^{i-1} 的情况下赋予 s_i 的概率值为[11]：

$$P(d) = \prod_{i=1}^{|d|} P(s_i \mid s_1^{i-1}) \approx \prod_{i=1}^{|d|} P(s_i \mid s_{i-k}^{i-1}) \qquad (5)$$

通过公式(5)，可得 $P(s \mid spam)$ 和 $P(s \mid ham)$ 的值，并对运算结果进行归一化处理，计算 score 值，取临界值 0.5，score>0.5 为 spam，score<0.5 为 ham[11]：

$$spam\_score(d) = \frac{p(d \mid spam)^{1/|d|}}{p(d \mid spam)^{1/|d|} + p(d \mid ham)^{1/|d|}} \qquad (6)$$

在公式(6)中，|d|表示文本长度。

根据 score 值，描绘 ROC 曲线如图 3 所示。由图 3 可知，PPM 算法比 Logistic 回归、Winnow 算法曲线下方的面积大，由此可知 PPM 算法对应的参考指标(1-ROCA)%数值最小，表明三种算法中 PPM 算法在邮件过滤中性能最佳。

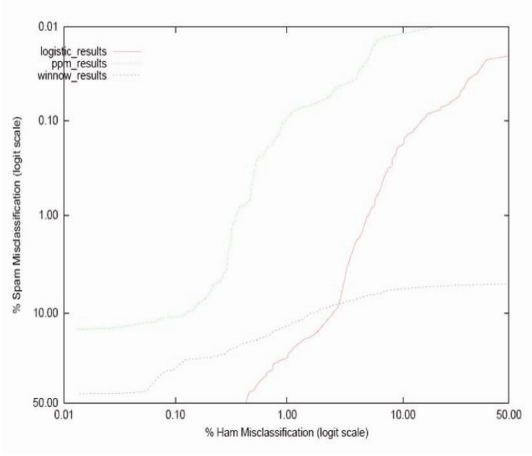

图 3　PPM、LR、Winnow 算法在邮件过滤中的 ROC 比较

4　结束语

PPM 算法采用统计数据压缩技术，将邮件作为字节流，提取字符或二进制流层次上的特征作为过滤标准。无需常规的文本解析、分词等步骤，大大简化了文本的预处理过程。用于训练的上下文树可以动态生成，在预测过程中如有新的样本邮件加入，只需在该上下文树中增加新的节点，从而动态更新模型即可，避免了重建过程，因此大大提高了执行效率。PPM 算法的缺点是需要的内存等计算资源较大，但其过滤效果明显优于 Winnow 等算法，具有广阔的应用前景。

参考文献

1 裴亚辉,熊盛武.垃圾邮件与反垃圾邮件技术.网络通讯与安全, 2007,6:1251－1252.

2 王斌,潘文锋.基于内容的垃圾邮件过滤技术综述.中文信息学报,2005,19(5):1－10.

3 Bratko A, Cormack GV. Spam filtering using statistical data compression models. Journal of Machine Learning Research, 2006,7: 2673－2698.

4 Cleary JG, Written IH. Data compressing using adaptive coding and partial string matching. IEEE Transactions on Communications, 1984, 32(4):396－402.

5 Moffat A. Implementing the PPM data compressing scheme. IEEE Transactions on Communications, 1990,38(11):1917－1921.

6 周小四,杨杰,王淑华.改进的 PPM 数据压缩算法及性能分析和比较.上海交通大学学报,2002,36(12):1841－1845.

7 李晓翔,王淑华,赵正校.PPM 压缩算法在图像压缩中的应用.计算机工程, 2002,28(7):175－177.

8 SEWM2008 垃圾邮件过滤系统评测. 华南理工大学信息网络工程研究中心和广东省计算机网络重点实验室邮件评测小组, 2008: http://www2.scut. edu.cn/antispam/ppt.html.

9 Cleary JG, Teahan WJ. Unbounded length contexts for PPM. The Computer Journal, 1997,40(2/3): 67－75.

10 张丽,黄东.基于 Winnow 算法的反垃圾邮件引擎的设计与实现.计算机技术与发展,2006,16(4): 170－172,175.

11 Bratko A, Filipic B. Spam Filtering using Character-le-vel Markov Models: Experiments for the TREC2005 Spam Track. The Fourteenth Text REtrieval Conference (TREC 2006) Proceedings. http://trec.nist.gov/pubs/trec14/papers/jozef-stefan.bratko.spam.pdf.