一种寻找最近子串的快速种子集求精算法

张懿璞 茹 锋 王 飚
(长安大学电子控制与工程学院 西安 710064)

摘 要 寻找序列中的最近子串对挖掘基因中特殊的功能位点和了解基因调控关系有着重要意义。提出了一种新的基于种子集求精的改进的期望最大化算法SCEM来对基因序列中的最近子串进行寻找。通过对输入序列聚类,将数据集分解为若干种子集,再使用改进的期望最大化算法对各种子集进行求精,SCEM最终可寻找到序列中的最近子串。真实数据和模拟数据实验表明,SCEM算法可以寻找到真实的最近子串,与随机投影等流行算法相比也能保证较高的性能和效率,并且可以有效解决较长的最近子串寻找问题。

关键词 最近子串,种子集,聚类求精,期望最大化

中图法分类号 TP301.6 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.5.049

Fast Seed Set Refinement Algorithm for Closest Substrings Discovery

ZHANG Yi-pu RU Feng WANG Biao
(School of Electronics and Control Engineering, Chang'an University, Xi'an 710064, China)

Abstract Finding the closest substrings in sequences is crucial for mining the specific functional sites in gene and understanding the gene regulatory relationship. This paper proposed a novel algorithm namely SCEM based on the improved expectation maximization algorithm of the seed sets refinement. SCEM divides the dataset into a series of seed sets by clustering the input sequences, then uses the improved EM algorithm to refine these seed sets and finds the closest substrings finally. Experiments using the real datasets and the simulated datasets demonstrat our algorithm can find the real closest substrings and has the high performance and efficiency comparing with the popular algorithm such as Random Projection. Moreover, SCEM also can solve the long closest substrings finding problem effectively.

Keywords Closest substring, Seed set, Cluster refinement, Expectation maximization

1 引言

最近子串查找问题一直是计算机科学家和生物信息学家关注的焦点之一,可用于识别基因序列或蛋白质序列中诸多具有特殊功能的模式。挖掘基因序列中特殊的最近子串也被称为植入模体识别问题,对了解基因的转录活性及理解基因表达有着重要意义,是现今生物信息学中研究最为广泛的问题之一[1-3]。

近年来许多植入模式识别的算法被提出[10-13],这些算法根据模式的定义方式被分为精确方法和近似方法两大类,但随着序列规模的增长和模式长度及保守性的变化,大多算法都存在准确度或效率方面的问题[4]。在之前的研究中,我们提出了基于聚类的求精算法CEM(Cluster Expectation Maximization algorithm)[5]。CEM克服了传统算法的缺点,在性能和效率上有了明显的改善;但是,对于大规模序列中的长模体,在收敛过程中需要对各子串进行大量计算,算法效率有待进一步提高。

本文提出了一种基于改进的期望最大化方法的种子集算法SCEM(Seed-Cluster Expectation Maximization),该算法将输入序列分解为若干子集,筛选出其中合格的种子集,并使用改进的EM算法对种子集进行求精。真实数据和模拟数据实验表明,改进的种子集求精方法加速了收敛,使得SCEM算法在处理较长植入模式时保持了较高的性能和效率。

2 相关工作

2.1 问题与符号定义

最近子串查找问题(植入模式识别问题)的具体定义如下[6]:假设模体是一个未知的长度为 l 的子序列,在 t 条长度均为 n 的输入序列中,每一条序列都包含一个植入的模体变异片段,这个模体变异片段与原模体长度相同且在至多 d 个随机选择的独立位置上发生了碱基的变异(替换),这些发生过碱基变异的片段也被称为模体实例。

表1列出本文中所使用符号的定义。

到稿日期:2015-11-18 返修日期:2016-01-17 本文受国家自然科学青年基金(61501058),陕西省自然科学基础研究计划项目(2014JM8328,2014JQ7269,2016JQ6075),中央高校基本业务费(310832161008)资助。

张懿璞(1985-),男,博士,讲师,主要研究方向为数据挖掘、模式识别,E-mail:zyipu.chd@gmail.com;茹 锋(1969-),男,博士,教授,主要研究方向为图像处理、机器视觉;王 飚(1969-),男,博士,副教授,主要研究方向为大系统优化、自适应算法。

表1 符号定义

N	集合中 l-mers 总数	T	迭代次数
$\mathbf{X}$	包含所有 l-mers 的集合	$\mathbf{\Sigma}$	字符集
n_i	第 i 条序列的长度	Z_{ij}	表示 x_{ij} 是否为模体的隐变量
l	模体长度	θ_{BS}	模体成分概率分布
t	序列条数	θ_{BG}	背景成分概率分布
d	替换字符数	w	子序列中任一位置
k	任一字符	S	$\{S_i \mid i=1,\cdots,t\}$ 序列集
$x_{ij}=(s_{i,j},\cdots,s_{i,j+w},\cdots,s_{i,j+l-1})$		从第 i 行第 j 个位置开始长度为 l 的子序列	

2.2 期望最大化算法

MEME(the Multiple Expectation Maximization for Motif Elicitation)使用 EM 算法对模体所在位置进行迭代估计，在每一次迭代中，模体的位置权重矩阵通过计算输入序列中每个子序列的似然估计值来进行更新。EM 算法可以保证似然函数收敛到一个局部最大值，但是往往对初始条件较为敏感。为了减小初始位点的影响，MEME 通常针对不同的初始位点多次运行 EM 算法[7]。

EM 算法模型将输入序列看作两个多项式成分的混合，一个是模体成分 $\theta_{BS}=(f_{1k},\cdots,f_{wk},\cdots,f_{lk})$；另一个是背景成分 $\theta_{BG}=\{f_{0k}\}(k\in\mathbf{\Sigma})$。$\mathbf{Z}=\{Z_{ij}\}$ 为一组隐变量，用于表示每个长度为 l 的子序列(也称为 l-mer)是由模体成分组成还是由背景成分组成。当 $Z_{ij}=1$ 时，表示这个子序列完全由模体成分组成；$Z_{ij}=0$ 时，则完全由背景成分组成。这样，序列中每一个长度为 l 的子序列 x_{ij} 都可以看作是完全由模体成分或背景成分构成，或由两个成分混合构成，可以表示为：

$$p(x_{ij}\mid Z_{ij},\theta_{BS},\theta_{BG})=p(x_{ij}\mid\theta_{BS})^{Z_{ij}}\,p(x_{ij}\mid\theta_{BG})^{1-Z_{ij}} \tag{1}$$

位置权重矩阵 Θ(PWM)可由 θ_{BS} 和 θ_{BG} 得到：

$$\theta_{wk}=\log(f_{wk}/f_{0k}) \tag{2}$$

θ_{wk} 表示字符 k 在模体位置 w 上的信息权重。则整个数据集的期望似然值可以表示为：

$$LL=\log(p(x,Z\mid\Theta))$$
$$=\sum_{i=1}^{t}\sum_{j=1}^{n_i-l+1}Z_{ij}\log p(x_{ij}\mid Z_{ij},\theta_{BS},\theta_{BG}) \tag{3}$$

期望最大化算法正是通过不断更新子序列的权重 Z 和位置权重矩阵 Θ 来使期望似然值 LL 最大化，具体分为 E 步和 M 步。

E-step：通过序列中 l-mer 的似然值估计每一个 l-mer 的隐变量 Z。

$$Z_{ij}^{(T)}=\frac{p(x\mid Z_{ij},\Theta^{(T)})}{\sum_{n_i-l+1}p(x\mid Z_{ij},\Theta^{(T)})} \tag{4}$$

M-step：根据每一个 l-mer 的隐变量 Z 重新估计模体的位置权重矩阵 Θ。

$$\Theta^{(T+1)}=\arg\max E[\log p(x,Z\mid\Theta^{(T)})] \tag{5}$$

其中

$$\theta_{BS}\rightarrow\arg\max_{\theta_{BS}}\sum Z_{ij}\log p(x_{ij}\mid\theta_{BS}) \tag{6}$$

$$\theta_{BG}\rightarrow\arg\max_{\theta_{BG}}\sum(1-Z_{ij})\log p(x_{ij}\mid\theta_{BG}) \tag{7}$$

这里 $\theta_{BS}=\{f_{wk}\}$，f_{wk} 表示字符 k 在模体中第 w 个位置的概率。

$$p(x_{ij}\mid\theta_{BS})=\prod_{w}f_{wk}\mid(s_{i,j+w-1}=k) \tag{8}$$

其中，$s_{i,j+w-1}$ 表示输入序列中第 i 行第 j 个 l-mer 中第 w 个位置的字符。概率更新方程为：

$$f_{wk}=\frac{\sum Z_{ij}\mid(s_{i,j+w-1}=k)}{\sum Z_{ij}}=\frac{c_{wk}+\beta_k}{\sum\mid c_{wk}+\beta_k\mid} \tag{9}$$

c_{wk} 为字符 k 在模体位置 w 上出现次数的期望值，β_k 为防止零概率出现的伪计数。

MEME 算法默认使用零阶马尔可夫模型作为背景，在由 M 步得到新的位置权重矩阵 $\Theta^{(T+1)}$ 后，返回 E 步估计新的隐变量 $Z^{(T+1)}$，并重复 E 步和 M 步，直至收敛。但是，由于 EM 算法迭代时需要计算序列中全部 l-mer 的似然估计值，对于较大的数据集，计算时间过长且收敛速度变慢的缺陷不能避免。针对这一缺点，本文设计了如下的 SCEM 算法。

3 SCEM 算法

之前的研究中已提出了 CEM 算法，通过选取参考子序列将数据集分解为若干子集，再使用 EM 算法对分解后的子集进行求精来寻找植入子序列的位置。CEM 算法很好地解决了植入模式寻找问题中的一般问题和挑战问题，但是其迭代训练只是通过分解数据减小了搜索空间；在解决挑战问题时，面对大量的背景干扰，其仍存在运行速度较慢的缺陷。本文在 CEM 算法的基础上，提出了改进的 SCEM 算法，通过对期望最大化算法中隐变量 Z 的更新，加速迭代求精过程。SCEM 分为如下 3 个步骤：(1)选取参考序列生成种子集；(2)使用加速的 EM 算法求精各种子集；(3)定位植入模式的位置。具体流程如下。

1. 生成种子集

如果 x_1 和 x_2 是由子序列 x 在 d 个位置上进行字符替换后得到，那么 x_1 和 x_2 之间的 Hamming 距离一定小于或等于 $2d$，记作 $d_H(x_1,x_2)\leqslant 2d$。对于任一输入序列如 S_1，设 S_1 为参照序列，S_1 中的 $n-l+1$ 个子序列为参考子序列。通过每个参考子序列对剩余序列中的子序列进行搜索，寻找到每条序列中与参考子序列 Hamming 距离小于或等于 $2d$ 的子序列，则针对这 $n-l+1$ 个参考子序列就可以生成 $n-l+1$ 个子集，这些子集也称为聚类子集。

令 $B_{i'}(2\leqslant i'\leqslant t)$ 表示由参照子序列 x_{1j} 在序列 $S_{i'}(2\leqslant i'\leqslant t)$ 中选出的 l-mer 组成的集合，那么聚类子集 $m_j(1\leqslant j\leqslant n-l+1)$ 就是由参照子序列 x_{1j} 和 $B_{i'j}$ 构成，$m_j=\{x_{1j},B_{2j},\cdots,B_{tj}\}$。图 1 给出了一个聚类子集构成过程的样例。

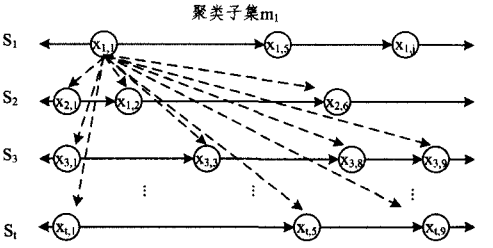

图 1 聚类子集的组成

由植入模式问题的定义可以知道，植入的模式一定在这 $n-l+1$ 个聚类子集中，但是，这些聚类子集并不可能都含有植入模体实例，其中有大量由背景成分造成的冗余。对于某些参考子序列，我们发现存储选出 l-mer 的集合 $B_{i'j}$ 常为空集。也就是说，假设这个参照子序列是一个植入模体实例，如果在剩余的某一条序列中不存在与它相应的植入模体实例，那么这个参照子序列本身就不是真实的模体实例，假设失败。

由此，我们认为那些含有任一空集 $B_{i'j}$ 的聚类子集不合格并将其剔除，则剩余的聚类子集被称为种子集，记为$\{M_j\}$。

2. 求精种子集

对于第 j 个种子集 M_j，B_{2j}，…，B_{tj} 为从相应序列中选出 l-mer 的集合。注意到，每个 $B_{i'j}$ 中选出 l-mer 的数量都不相同，为了避免这些由选出 l-mer 数量的不同所带来的计算差异，从 B_{2j}，…，B_{tj} 中分别选择一个 l-mer 组成位置概率矩阵，并使用期望最大化算法求精。在求精迭代开始前，初始状态由参照子序列 x_{1j} 和从每个 $B_{i'j}$ 中随机选出的一个 l-mer 组成，由初始状态即可得到分布 $\Theta^{(0)}$。

在初始状态确定后，我们就可以使用 EM 算法对各种子集进行求精。注意到，在 EM 算法中进行参数更新时，式(4)引入了数据集中所有 l-mer 的似然估计值用于更新隐变量 Z。但是在 EM 算法的每次迭代参数更新中，似然值的计算却主要依赖于模体成分 θ_{BS}，即一部分非模体子序列的权值 Z 很小，对整个参数更新的贡献也非常小。基于这个观察，在 EM 的参数迭代更新中进行一些改进，通过忽略掉那些 Z 值非常小的 l-mer 来加速迭代收敛。

通过 Z 值定义一个新的子集 V_δ，用于表示那些与模体成分估计值所匹配的 l-mers，$V_\delta=\{n: Z_n>\delta, 1\leqslant n\leqslant N\}$，定义 f_{wk}^*、c_{wk}^* 为方程(9)中 f_{wk}、c_{wk} 的近似值：

$$f_{wk}^*=\frac{\sum\limits_{n\in V_\delta} Z_{ij}\,|\,(s_{i,j+w-1}=k)}{\sum\limits_{n\in V_\delta} Z_{ij}}=\frac{c_{wk}^*+\beta_k}{\sum|c_{uk}^*+\beta_k|} \tag{10}$$

同时，为了方便表示，还定义：

$$c_{wk}^\wedge=\sum_{n\notin V_\delta} Z_{ij}\,|\,(s_{i,j+w-1}=k) \tag{11}$$

$$N^\wedge=N-|V_\delta| \tag{12}$$

这里，$c_{wk}=c_{wk}^*+c_{wk}^\wedge$，$f_{wk}^*$ 的相对误差 ε_δ 为

$$\begin{aligned}\varepsilon_\delta&=\frac{f_{wk}-f_{wk}^*}{f_{wk}}\\&=\frac{c_{wk}-\sum|c_{wk}+\beta_k|/\sum|c_{wk}^*+\beta_k|\times c_{wk}^*}{c_{wk}}\end{aligned} \tag{13}$$

$$c_{wk}\varepsilon_\delta=c_{wk}^\wedge-\sum|c_{wk}^\wedge+\beta_k|/\sum|c_{wk}^*+\beta_k|\times c_{wk}^* \tag{14}$$

注意到，$\sum|c_{wk}^\wedge+\beta_k|$和$\sum|c_{wk}^*+\beta_k|$分别表示那些在第 w 个位置上开始的 Z 值大于或小于 δ 的 l-mers 的期望个数。令 $X^\wedge=\sum|c_{wk}^\wedge|$，$X^*=\sum|c_{wk}^*|$，那么

$$\varepsilon_\delta\leqslant\max\{\frac{c_{wk}^\wedge}{c_{wk}},\frac{X^\wedge c_{wk}^*}{X^* c_{wk}}\} \tag{15}$$

根据 V_δ 的定义，$X^\wedge\leqslant\delta\cdot N^\wedge$，则有 $c_{wk}^\wedge\leqslant\delta\cdot N^\wedge$，且

$$|\varepsilon_\delta|\leqslant\frac{\delta N^\wedge}{c_{wb}^\wedge} \tag{16}$$

根据 $c_{wk}^\wedge$ 即可设置相应的阈值 δ 来保证模体成分估计值小于相对误差。一般地，取相对误差 $\varepsilon_\delta=0.4$[7]。通过式(10)的参数更新方法，使用改进的 EM 算法对每个种子集进行求精，令 $|IC^{(T+1)}/IC^{(T)}-1|<10^{-6}$时，迭代停止，这里

$$IC=\sum_{w=1}^{l}\sum_{k\in\Sigma} f_{wk}\log(\frac{f_{wk}}{f_{0k}}) \tag{17}$$

3. 定位植入模式

通过 EM 迭代，可以得到每一个求精子集的 IC，令 $Q=\max\{IC_j\}$，Θ_Q 为 Q 所在种子集对应的模体分布。

通过聚类子集的构建可以知道，真实的植入模体实例一定存在于某一个种子集当中，所以对于每条序列，由 Θ_Q 计算出具有最大对数似然值的 l-mer 即为植入模式。

$$\log p(x_{ij}\,|\,\Theta_Q)=\max\sum_{w=1}^{l}\log\theta_w\ (1\leqslant i\leqslant t) \tag{18}$$

将计算得到的植入模体位置存入集合 L 中作为最终输出。

SCEM 算法的伪代码如下。

```
输入：t,l,d,ε_δ,S={S_1,…,S_t}
输出：motif,L
1. FOR each l-mer x_1j (1≤j≤n-l+1)
2.     M_j←Φ
3.     FOR each l-mer x_i'j in S_i' (2≤i'≤t)
4.         B_i'j←Φ
5.         IF d_H(x_1j, x_i'j)≤2d
6.             B_i'j← B_i'j ∪ x_i'j
7.         WHILE B_i'j≠ Φ
8.                 M_j←M_j ∪B_i'j ∪x_1j
9. FOR each M_j
10.     use improved EM algorithm calculate IC_j and Θ_j
11. Q←max{IC_j}
12. get motif from Q Θ_Q
13. FOR each sequence S_i (1≤i≤t)
14.     p'←0, L_i←0
15.     FOR each l-mer x_ij
16.         evaluate log p(x_ij |Θ_max)
17.         IF log p(x_ij |Θ_max)>p'
18.             p' ← log p(x_ij |Θ_max), L_i← j
19.     L←L∪L_i
20. RETURN motif and L
```

4 实验分析

首先使用 5 组真实基因数据对算法进行测试，包括：CRP binding site in E. coli (E. coli CRP)、dihydrofolate reductase (DHFR)、LexA、c-fos、Yeast ECB[4]。测试时，假设模型中每条序列内只有一个植入模式，通过变换不同的 l、d 参数来找到真实模体。表 2 为使用不同(l,d)参数的真实基因数据的测试结果。

表 2 真实基因数据测试结果

基因	预测结果	真实模体	(l,d)
DHFR	TCGCGCCAAACTT	TTCGCGCCAAACT	(13,2)
LexA	TACTGTATATAT ATACAG	TACTGTATATAT ATACAG	(18,5)
c-fos	CCATATTAGGACATCT	CATATTAGGACATCTG	(16,5)
Yeast ECB	TTACCCGTTTAGGAAA	TTACCCNNTNAGGAAA	(16,5)
E. coli CRP	TGTGAAATAGTTCACA	TGTGANNNNGNTCACA	(16,5)

此外，还使用人工生成的序列对算法进行测试。在 20 条长度为 600 个字符的每条序列中随机植入一个长度为 l、变异数量为 d 的模式，并将本文算法在相同环境中(2.3GHz CPU 和 4GB 内存)与随机投影算法[8]和 Matlab 下的 CEM 算法进行对比。

表 3 为使用 3 种算法对人工模拟数据进行测试的结果，除了一般的(11,2)、(15,4)、(18,6)、(19,7)实例外，还使用长植入模式(27,8)和(36,11)对算法进行测试。设置随机投影算法迭代 200 次，CEM 算法迭代 30 次。

表 3 人工模拟数据测试结果

(l,d)	Algorithms		
	PROJECTION	CEM	SCEM
(11,2)	0.88(10s)	0.96(10s)	0.97(10s)
(15,4)	0.94(4m)	0.98(45s)	1(26s)
(18,6)	0.89(33m)	0.98(9.8m)	0.98(5.8m)
(19,7)	0.86(1.5h)	0.95(18.2m)	0.96(8.9m)
(27,8)	0.84(2h)	0.94(31.2m)	0.96(20.1m)
(36,11)	0.78(2.5h)	0.91(40.5m)	0.96(34.6m)

通过表 3 可以看到，相比随机投影和 CEM，SCEM 不但能保证较高的算法精度，并且具有较低的时间开销。在寻找长植入模式时，其算法精度都保持在 95%以上，时间开销远小于随机投影。

结束语 本文提出的算法可以在合理的时间内有效解决 (l,d)植入模体发现问题，并且在解决长植入模体时也展现出了良好的性能，与目前流行的算法相比极具竞争力。通过对期望最大化算法中隐变量 Z 的约束，加速各种子集收敛求精，并且由于各种子集的运算相互独立，所提算法也易于并行化。而结合 ChIP-seq 等新技术，处理更大规模的数据，则是我们后续所关注的研究内容。

参 考 文 献

[1] Park P J. ChIP-seq: advantages and challenges of a maturing technology[J]. Nat Rev Genet, 2009, 10(10): 669-80

[2] Zhang Yi-pu, Wang Ping. A Fast Cluster Motif Finding Algorithm for ChIP-Seq Data Sets[C]// BioMed Research International, 2015. 2015

[3] Bailey T L, Elkan C. Fitting a mixture model by expectation maximization to discover motifs in bipolymers[C]// Second International Conference on Intelligent Systems for Molecurlar Biology. 1994: 28-36

[4] Zambelli F, Pesole G, Pavesi G. Motif discovery and transcription factor binding sites before and after the next-generation sequencing era[J]. Briefings in Bioinformatics, 2013, 14(2): 225-237

[5] Zhang Yi-pu, Huo H, Yu Qiang. A Heuristic Cluster-based EM Algorithm for the Planted(l,d) Problem[J]. Journal of Bioinformatics and Computational Biology, 2013, 11(4): 1350009

[6] Pevzner P A, Sze S H. Combinatorial approaches to finding subtle signals in DNA sequences[C]// ISMB. 2000: 269-278

[7] Reid J, Wernisch L. STEME: efficient EM to find motifs in large data sets[J]. Nucleic Acids Research, 2011, 39(18): 126

[8] Buhler J, Tompa M. Finding motifs using random projections [J]. Journal of Computational Biology, 2002, 9(2): 225-242

[9] Chen Kun, Zhang Xiao-jun. MCI Clustering Algorithm Solving Planted (l,d) Motif Identification[J]. Journal of Henan University(Natural Science), 2015, 45(1): 102-107(in Chinese)
陈昆，张小骏. MCL 聚类算法求解植入(l,d)模体识别问题[J]. 河南大学学报(自然科学版)，2015, 45(1): 102-107

[10] Sun De-cai, Wang Xiao-xia. Research on Filtering Algorithm of Approximate String Matching[J]. Computer Technology and Development, 2015(4): 171-176(in Chinese)
孙德才，王晓霞. 近似串匹配过滤算法研究[J]. 计算机技术与发展，2015(4): 171-176

[11] Xu Yong-kang, Yang Guang-lu, Lu Song-feng, et al. Approximate string matching algorithm based on compressed suffixarray [J]. Computer Engineering and Applications, 2015, 51(23): 139-142(in Chinese)
胥永康，杨光露，路松峰，等. 基于压缩后缀数组的近似字符串匹配算法[J]. 计算机工程与应用，2015, 51(23): 139-142

[12] Chan H L, Lam T W, Sung W K, et al. Compressed indexes for approximate string matching[J]. Algorithmica, 2010, 58(2): 263-281

[13] Atallah M J, Grigorescu E, Wu Y. A lower-variance randomized algorithm for approximate string matching[J]. Information Processing Letters, 2013, 113(18): 690-692

(上接第 256 页)

[9] Krawczyk B, Woźniak M, Cyganek B. Clustering-based ensembles for one-class classification[J]. Information Sciences, 2014, 264: 182-195

[10] Zhou Z H, Wu J X, Tang W. Ensembling neural networks: many could be better than all[J]. Artificial Intelligence, 2002, 137(1/2): 239-263

[11] Wang L, Li Q. Effective selective ensemble algorithms for support vector machines[J]. Neurocomputing, 2010: 287-295

[12] Li N, Zhou Z. Selective ensemble under regularization framework[J]. Lecture Notes in Computer Science, 2009, 5519: 293-303

[13] Zhang L, Zhou W. Sparse ensembles using weighted combination methods based on linear programming[J]. Pattern Recognition, 2011, 44(1): 97-106

[14] Krawczyk B. One-class classifier ensemble pruning and weighting with firefly algorithm[J]. Neurocomputing, 2015, 150(B): 490-500

[15] Vapnik V N. Statistical Learning Theory[M]. New York: Wiley, 1998

[16] Principe J C. Information Theory Learning: Renyi's Entropy and Kernel Perspectives[M]. Springer, 2012

[17] Tang E K, Suganthan P N, Yao X. An analysis of diversity measures[J]. Machine Learning, 2006, 65(1): 247-271

[18] Yin X C, Huang K, Yang C, et al. Convex ensemble learning with sparsity and diversity[J]. Information Fusion, 2014, 20: 49-59

[19] Vedelsby J, Krogh A. Neural network ensembles, cross-validation and active learning[J]. Advances in Neural Information Processing Systems, 1995, 7: 231-238

[20] He R, Hu B G, Zheng W S, et al. Robust principal component analysis based on maximum correntropy criterion[J]. IEEE Transactions on Image Processing, 2011, 20(6): 1485-1494

[21] Eockfellar R. Convex analysis[M]. Princeton University Press, Princeton, 1970

[22] Wu M, Ye J. A small sphere and large margin approach for novelty detection using training data with outliers[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009, 31(11): 2088-2092

[23] Frank A, Asuncion A. UCI machine learning repository[OL]. University of California, Irvine, School of Information and Computer Science, Irvine, CA. http://archive.ics.uci.edu/ml