# 一种双向挖掘频繁项的有效方法*)

王晓峰　张松筠
（上海海事大学信息工程学院　上海 200135）

**摘　要**　Apriori算法已成为关联规则挖掘的一个经典方法，广泛地被应用于如贸易决策、银行信用评估、金融保险等诸多领域。这种自底向上方法挖掘短频繁项集时效果较好，当频繁项集较长时，其时间复杂度量呈指数增长态势。本文结合自顶向下和自底向上搜索两种方法，提出一种能更好解决长、短频繁项集问题的双向挖掘方法。通过计算复杂度分析的实验表明，所提出的方法是有效可行的。

**关键词**　数据挖掘，频繁项集，双向搜索，自顶向下挖掘

## The Research and Implementation of Double Search Data - Mining Method for Frequents

WANG Xiao-Feng　ZHANG Song-Jun
(School of Information Engineering, Shanghai Mariting University, Shanghia 200135)

**Abstract**　The Apriori algorithm has become a classic method for mining association rules. It is widely applied to various fields such as trade decision-making, bank evaluating credit, finance insurance, etc. This method is an effective down-top algorithm for minging frequents, but it will come across time-consuming huge computing problems in mining long pattern frequent itemsets(e.g. 100 items). A new ideal method of top-down mining frequent itemsets, which adopts some new concepts such as transaction and itemset association information tables, key-items and reduction items, projection DB, etc is presented. It is very effective, especially when being used to mine long items. This paper propses a new method, combining the top-down search method and bottom-up search method, but its main search strategy is still top-down method. This algorithm can better solve problems of long&short frequents, the validity and effectiveness of the proposed algorithm is proved through the analysis of computing complexity and experimentation.

**Keywords**　Data mining, Frequents, Double search, Top-down mining

## 1　引言

Apriori算法采用自底向上的方法遴选频繁项，从长度为1的小项目集出发，在满足最小支持度的条件下，逐步构造长度较大一点的项目集，直至得到最大的频繁项。这种方法挖掘短频繁项集时效果较好，当频繁项较长时，算法的时间复杂度则呈指数增长态势，例如，要发现一个长度为100的频繁项，将产生$2^{100}\approx10^{30}$个以上的候选项。为克服Apriori算法在长频繁项挖掘时遇到的问题，人们提出了各种改进方法，1998年Roberto和Bayardo利用自底向上搜索和项目集排序的方法建立了一种挖掘长型频繁项的Max-Miner算法；Lin和Kedem提出了一种双向钳形搜索Pincer-Search方法，利用自底向上搜索产生的非频繁项集来约束和修剪自顶向下方向的最大候选频繁集，候选频繁项集来自于Apriori方法。文[1]中提出了一种新的自顶向下挖掘长频繁项的有效方法，该方法对长频繁项具有相当好的挖掘效果，但对短频繁项挖掘略有不足。

本文组合了自顶向下和自底向上两种方法，采用一种双向挖掘策略。主挖掘方向是自顶向下搜索，结合信息系统约简，直接把事务数据库的项目集作为候选项目集，寻找满足最小支持度的最大项目集。在搜索过程中，利用自顶向下挖掘方法，把大数据库投影到小数据库上，利用约简项作为启发和约束信息，压缩频繁项搜索空间；同时，利用自底向上方法及时修剪候选集，减少候选集的规模和数量，较好地解决了长、短频繁项的挖掘问题。

## 2　主要概念和定义

**定义1**　事务-项目相关信息表：给定项目集：$I=\{i_1,i_2,\cdots,i_m\}$，事务数据库$D=(U,A)$，其中$U=(X_1,X_2,\cdots,X_n)$，$A=T\cup C$，$C=(C_1,C_2,\cdots,C_n)$，项目集合$C_i\subseteq I$，$T=\{t_1,t_2,\cdots,t_n\}$是事务标识符集合，则每一对象$X_i\in U$有唯一一个事务标识TID和一个项目集对应。为精确描述这种事务和项目间的对应关系，称$S=(U,A,V,f)$为事务项目相关信息表，简称信息表。这里，$V=(0,1)$，$f:T\times C\to V$。

**例1**　给定一事务数据库的事务记录如表1，则对应的事务项目相关信息表如表2。比较事务数据库和信息表，不难发现信息表是事务数据库的扩张，事务数据库是信息表的子集。所有事务数据库关于事务和项目的有关信息在信息表中都有所对应。事务标识在库中与表中是一样的；数据库的一个记录（或对象）对应表中的一行；数据库的项目与信息表的项目（或属性）相对应：

表1　事务数据库

| TID | Items |
|---|---|
| T1 | A,C,D |
| T2 | B,C,E |
| T3 | A,B,C,E |
| T4 | B,E |

表2　事务-项目信息表

| TID | A | B | C | D | E |
|---|---|---|---|---|---|
| T1 | 1 | 0 | 1 | 1 | 0 |
| T2 | 0 | 1 | 1 | 0 | 1 |
| T3 | 1 | 1 | 1 | 0 | 1 |
| T4 | 0 | 1 | 0 | 0 | 1 |

数据库中$T_i$标识的项目集与信息表$T_i$行项目属性值集相对应，但信息表每个项目属性有两个属性值：1表示数据

*)本文获上海市教委科研基金和上海市重点学科建设项目资助（编号：T0602）。**王晓峰**　博士，教授，研究方向：数据挖掘与知识发现，智能信息处理；**张松筠**　硕士研究生，研究方向：数据挖掘与知识发现。

库对应行中有该项目，0 表示数据库对应行无该项目。

信息表中除事务标识外，每一行的和等于对应数据库中项目集的大小；每一列的和对应数据库相应项目的支持度。信息表的属性值不论取 1 还是 0，对于标识事务都有意义，但对应数据库的支持度计算，只有取 1 的项目才有意义。

利用信息表和数据库间的这种对应关系，有一个事务数据库就有一个信息表，反之亦然。所以本文中除特别声明外，不再区分数据库与信息表之间的同异细节。进一步，可称信息表的 Discernibility 矩阵为事务库的 Discernibility 矩阵，信息表的约简为事务数据库的约简，相应的信息表的核就是事务数据库的核。

**例 2** 例 1 的 Discernibility 矩阵见图 1。由 Discernibility 矩阵的性质：

$\because (B \vee D \vee E) \wedge A \wedge C = BAC \vee DAC \vee EAC$

| | $T_1$ | $T_2$ | $T_3$ |
|---|---|---|---|
| $T_1$ | | | |
| $T_2$ | ABDE | | |
| $T_3$ | BDE | A | |
| $T_4$ | ABCDE | C | AC |

图 1 Discernibility 矩阵

$\therefore$ 信息表的约简为{B,A,C}，{D,A,C}，{E,A,C}，核为{A,C}。

**定义 2** 事务数据库的投影：给定一个事务数据库的项目子集，那么该子集的项目与其对应的事务形成了原数据库的一个子集，称这一过程为对数据库投影，所生成的新数据库为原数据库在项目子集_L 的投影数据库。

**定义 3** 项目集的核与约简：给定一个事务数据库的项目子集，就有一个对应的投影数据库。称这个投影数据库的核与约简项为该项目子集的核与约简项。项目子集的大小对应了信息表属性的个数。

**定义 4** 约简数据库：设 X 是给定的项目集，$Y$ 是 $X$ 的约简集。那么含有 $X$ 各项目的事务集 $T$ 连同 $X$ 集合的各项目元素一起构成了事务据库的投影库 $A$，而约简集 $Y$ 的各项目组合连同 $T$ 又构成了 $A$ 的投影数据库，称为约简数据库。约简库 $B$ 和投影库 $A$ 有共同的事务项。约简库的各项目集是投影库 $A$ 中的项目子集，投影库 $A$ 中的项目集是约简库 $B$ 项目的超集。

仍就例 1 来说，原数据库是 $(T,C)$，其中，$T$ 是事务标识集，$C$ 是所有项目的集合。假设给定项目集 $X$ 为 ABCE，显然 $X \subseteq C$，则事务库 $(T,C)$ 在 $X$ 上的投影库是 $(Tx,X)$，$Tx \subseteq T$，就相当于把原来数据库去掉项 $D$ 后形成的数据库子集。计算投影库 $(Tx,X)$ 的约简，有约简项 ABC。再次投影得 $(Tx,\{ABC\})$ 为 $(Tx,X)$ 的约简库。

如果投影库 A 中的项目集不重复，那么投影库和约简库是对应的，对于 A 中的每一个事务项目集在约简库中都有唯一一个约简子集与之相对应。这样，利用事务数据库在项目集上的投影结果，可以将项目集合分解为约简项和非约简项。项目集约简项具有下列性质，从而可以利用项目集的分解来解决最大频繁项的支持度计算问题。

项目集约简项的性质：

**定理 1** 设有项目集 $X$，其支持度 $\text{support}(X)=m$，$Y$ 是 $X$ 的约简，则 $\text{support}(Y)=m$。证明（见文[1]）。

**推论 1** 若给定项目集 $X$ 的支持度 $\text{support}(X)=m$，$Y$ 是 $X$ 的约简，$Z=X-Y$。则 $\text{support}(X) \leqslant \text{support}(Z)$，特别地，当 Z 为约简项时，有 $\text{support}(X)=\text{support}(Z)$。

**推论 2** 若给定项目集 $X$ 的支持度 $\text{support}(X)=m$，$Y$ 是 $X$ 的约简，$Y \subseteq X_1 \subseteq X$，$X_2 \subseteq X$，但 $Y$ 不是 $X_2$ 的子集。则 $\text{support}(X_1) \leqslant \text{support}(X_2)$。

根据投影库和约简库的一一对应关系我们还可以知道：在约简库中，约简子集的支持度最小，所以在投影库中含有约简集的超集的支持度也最小。又根据核的定义，核是所有约简项的公共部分，也是标识各项目集必不可少的关键部分。所以核对所在项目集的支持度影响最大。因此有下述定义存在。

**定义 5** 关键项目集：若一个事务数据库的项目子集作为候选频繁项集，那么其核与约简项就是候选集的关键项目集。

**定义 6** 最大频繁项：给定一个候选项目集，其中满足用户最小支持度的最大频繁项目集叫做最大频繁项。

## 3 项目集的约简

计算一项目（属性）集计算约简和核，有两种方法：一是利用 Discernibility Matrix（可辨识矩阵），通过逻辑计算可计算出所有约简项，求出项目集的核；另一种实际上利用约简项的定义和启发式信息计算出一组特定的约简项。例如，胡小华、苗夺谦等人先后提出了不同的启发式约简算法，通过有效值计算直接找出一组约简，计算复杂度为 $M \times \bigcirc(N' \times N')$，其中，$M$ 是项目数，$N'$ 是指关系数据库所生成的元组数，最大为 $N \times M$，$N$ 为事务数据库的记录数。

对于信息表系统来说，问题将更简单，根据约简项的定义和信息表的特点对约简算法作适当修改即得到一种方便的计算方法。

设 $T=(U,C,D)$ 是给定的信息系统，其中，$U=\{X_1,\cdots,X_m\}$ 是对象集（论域），$C$ 是条件属性集，$D$ 是决策属性集。那么，$POSc(D)=\bigcup_{x \in U/D} \underline{C}X$，其中，$POSc(D)$ 表示 $D$ 的 $C$-正区域，$U/D$ 表示 $U$ 关于 $D$ 的商集，$\underline{C}X=\underline{IND}(C)X$。所以，$POSc(D)=\bigcup X$，这里，$X \in U/IND(C)$，且 $X \subseteq Y$，$Y \in U/D$。

在信息表系统中，$U/D=\{\{t_1\},\{t_2\},\cdots,\{t_n\}\}$，$POSc(D)$ 集合是事务数据库中非重复出现的项目所对应的事务标识集，即 $POSc(D)$ 集合是支持度为 1 的项目对应的事务标识集。所以通过计算事务数据库中非重复出现的项目集，就可以得到 $POSc(D)$。

## 4 Double_Search 挖掘算法基本思想及描述

### 4.1 基本思想

本算法的基本思想是自顶向下和自底向上双向挖掘，主搜索方向是自顶向下。它基于以下两个性质：

**性质 1** 如果一个项目集是频繁的，那么，它的所有子集都是频繁的；

**性质 2** 如果一个项目不是频繁的，那么，它的所有超集都不是频繁的。

### 4.2 自底向上挖掘——改进经典 Apriori 算法

```
Gen_itemset //生成非频繁项和频繁项
Input:L_k,
Output: new candidate set C_{K+1},
        new frequent set L_{k+1}
        new inferequent set S_{k+1}
Method:Gen_infrequent(L_k,C_{k+1},S_{k+1},L_{k+1})
1. 调用连接(join)子程序
2. 调用修剪(prune)子程序
```

连接子程序：

```
Input:L_K//在第 K 次搜索中发现的频繁项集，并被排序
Output: C_{k+1}//初始候选集
Method: Join(L_k,C_{K+1})
For i from 1 to |L_k-1|
    For j from i+1 to |L_k|
        If L_K.itemset_i and L_k.itemset_j have
              the same (K-1)_prefix
          C_{K+1}:=C_{k+1} ∪ {L_k.itemset_i ∪ L_k.itemset_j}
```

```
        Else
          Break
    End
End
```

修剪子程序：

```
Input: Ck+1//连接过程中生成的初始候选集
Output: Ck+1//修剪后的候选集,用于下一次连接过程
        Sk+1//修剪过程中产生的非频繁项集
Method:Prune(Ck+1,SK+1)
For all itemsets c in Ck+1
    For all k_subsets s of c
        If s∉Lk
          Delete c from Ck+1
          Store c in Sk+1
    End
End
```

### 4.3 自顶向下挖掘——投影约简挖掘算法

```
Gen_candidate //生成候选集算法
Input: 给定的项目集 X={X1,X2…Xm.}
Output: 项目集 X 对应的的候选集C(含有长度为 m-1 的候选项);
        X 对应的频繁项 Z。
Method: Gen_candidate(X,Z,C)
Q=∅
Z=X-Q
While(support(Z)<minsupport and (Z)≠∅)
  Ql =gen_keyitems(Z)
  Q= QUQl
  Z= X-Q
End while
C=∅
S=|Q|
For i=1 to S
Yi= X-{qi} // q∈Q,生成 s 个长度为 m-1 的组合项
If (Yi⊆Lk)//当Yi 是已发现的频繁项的子集
    Delete Yi  删除该项目集
      else
  Ci=CU{Yi} //生成新的候选集
  End If
End For
End Gen_candidate
Gen_frequent 生成频繁项算法
Input:项目集 C
Out: C相应的频繁项 L
Method: gen_frequent(C,L)
N1=|C|;
For i=1 to N1
    If support(Ci)<minsup then
      Gen_candidate(Ci,L,Y);
      Gen_frequent(Y,L);
    Else
      L=LUC;
    End if
End
End Gen_frequent
```

在 Gen_frequent 生成频繁项算法中，进行的是项目集是否为频繁项的判定工作，若满足最小支持度，则加入频繁项集 L 中；否则将调用 Gen_ca ndidate 生成新的候选集，并递归调用 Gen_frequent。当生成的候选集都是频繁项或没有候选项时，算法结束。

### 4.4 双向挖掘

```
Double_Search 频繁项挖掘算法
Input:数据库 T, minsup
Out: 频繁项集 L
Method: Double_Search(T,L)
L=∅;
K1=1;
C1=Tk.CK//项目集中包含的频繁 1-项集
While Ck≠∅ begin
    read database and count support for itemsests in Ck
    Lk:={frequent itemsets in Ck}
    Sk:={infrequent itemsets in Ck}
    call Gen_itemset(Lk,Ck+1,Sk+1,Lk+1)
    k:=k+1;
    For m=l to N
        X=Tk.C ; // X=事务 Tk 中包含的项目候选集
        Del_minitems(X,Sk); //删除 X 中的非频繁项集 Sk
        If L≠∅ then
          Gen_candidate(X,L,C);
        end if
        Gen_frequent(C ,L);
    End For;
    End While
Out L ;
EndDouble_Search
```

| TID | Items |
|---|---|
| T1 | a, b, c, d, f |
| T2 | b, c, d, f, g, h |
| T3 | a, c, d, e, f |
| T4 | c, e, f, g |

图 2 事务数据库

| Item | a | B | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| Supp | 2 | 2 | 4 | 3 | 2 | 4 | 2 | 1 |

图 3 单项目支持度

## 5 举例说明

设给定的事务数据库如图 2，若用户给定的最小支持度 minsup=3，可得项目支持度如图 3。当 $k=1$，$X=T_1=$abcdf，调用 Gen_candidate，得 $Q_1=$ab，$L_1=$cdf；所以候选集为 $C_1=${bcdf，acdf}，如图 4 所示。

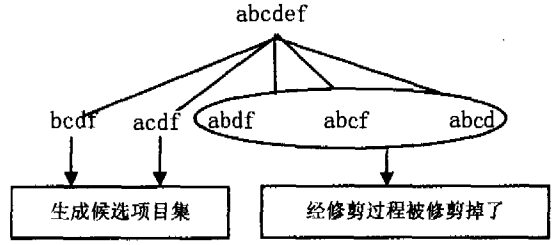


图 4 项目集的自顶向下计算修剪结果

由图 4 中可以看到：对于 5-项目集 abcdf，本来应该有 5 个 4-项目子集，但是经过算法中的 while 循环后，我们所考虑的 4-项目候选集仅剩下了 bcdf，acdf 两个(图 4)中带下划线)，而其余的 3 个 4-项目子集(图 4)椭圆中部分)则被修剪掉了，因为它们含有约简项{ab}。由自底向上挖掘过程有频繁一项{c}，{d}，{f}。非频繁项集有{a}，{b}，{e}，{g}，{h}。

对{bcdf}运用自顶向下分解，有子集{bcd}，{bcf}，{cdf}，{bdf}，但在自底向上搜索中，发现{bcdf}子集{bc}、{bd}、{bf}支持度为 2，小于最小支持度 3，是非频繁的，根据性质 2，即非频繁项的超集一定是非频繁的，所以应剪去{bcf}，{bdf}，{bcd}，只余下{cdf}。同理，对{acdf}运用自顶向下分解，有子集{acd}，{acf}，{cdf}，{adf}，因为{acdf}的子集{ac}、{ad}、{af}也是非频繁项(其支持度为 2)，所以应剪去{acd}，{acf}，{adf}，而{cdf}又包含于{bcdf}中，故不再重复计算。因此，{cdf}为 $X=T_1=$abcdf 时的最大频繁项集。进一步计算可知该项为唯一的最大频繁项集。

在迅驰Ⅱ/512M 的计算机上，我们分别实现了 Apriori、Top_Down 和 Double_Search 算法。根据数据库中事务数以及不同的最小支持度，运行算法进行比较试验，得出以下结论：最小支持度递减，Top_Down 算法运行时间也是递减的，Apriori 算法确是递增的，但两种算法都有各自擅长的应用范围。Double_Search 算法融合两者的长处，具有更广的应用范围，较好地解决了长、短频繁项的挖掘问题。

**结束语** Double_Search 算法融合了 Apriori 算法的自底向上挖掘策略，在挖掘长频繁项时，又采用自顶向下挖掘策略，基本上根据实际情况，解决了长、短频繁项的挖掘问题。同时，由于算法采用了粗糙集中的某些理论和方法，故而使关联规则挖掘和分类规则挖掘方法得到了融合。

由于时间、条件和水平的限制，本算法还有待于进一步完善和深入，例如，如何使后次挖掘尽量充分利用前次的挖掘结

果，怎样使约简数达到最少，以及在此基础上开发增量式的挖掘算法、并行挖掘算法等都是今后有用的研究方向。

## 参 考 文 献


1 王晓峰，王天然. 一种自顶向下挖掘频繁项的有效方法. 计算机研究与发展，2004(1)
2 Lin D I, Kedem Z M. Pincer-search: A New Algorithm for Discovering the Maximum Frequent Set [C]. In: Proc. of the 6th European Conf. on extending database technology, 1998
3 Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large d tabases. In: Proceedings of the ACM SIGMOD Conference on Management of data, 1993. 207～216
4 Lin J L, Dunham M H. Mining association rules: Anti-skewalgorithms. In: Proceedings of the International Conference on DataEngingeering, Orlando, Florida, February 1998
5 Brin S, Motwani R, Ullman J D, et al. Dynamic Itemset counting and implication rules for market basket data. In: ACM-SIGMOD International Conference on the Management of Data, 1997
6 冯玉才，冯剑琳. 关联规则的增量式更新算法[J]. 软件学报，1998(4)：3010
7 Roberto J, Bayardo Jr. Eficiently mining long patterns from databases, [C]. In: Proc. of the'98 ACM- SIGMOD int. Conf. On Management of Data(SIGMOD'98), 1998. 85～93
8 王晓峰，王天然. 相关测度与增量式支持度和信任度的计算[J]. 软件学报，2002(11)：2208～2214


(上接第 163 页)

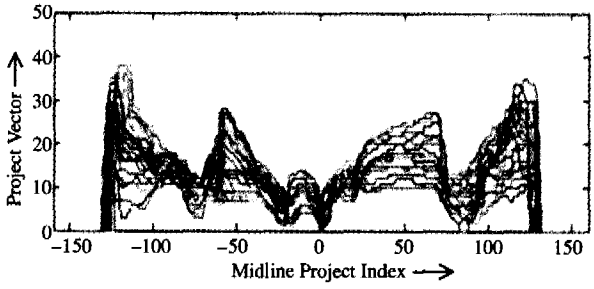


图 5 轮廓沿中线投影叠加效果

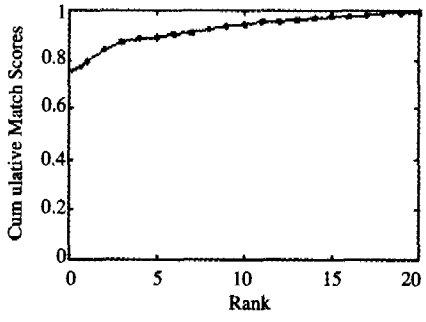


图 6 累积匹配分值

### 4.3 实验结果分析和总结

本文讨论了将人体轮廓沿中线投影的新颖的步态特征提取方法，此方法包括了更多的人体特征，如手的摆动情况。与文[6]相比，本文的特征提取方法更加新颖，方法更易实现，且计算量低，从识别性能中，相比指纹、虹膜等生理特征，步态、笔迹等行为特征的识别能力偏弱的情况下，该方法的正确分类率 CCR(Correct Classification Rate)达到如图 6 所示，实验结果比较令人满意。实验表明该算法不仅获得了令人鼓舞的识别性能，而且拥有相对较低的计算代价，是一种有效的步态特征提取与识别方法。

## 参 考 文 献


1 Abdelkader C B, Cutler R, Nanda H, et al. EigenGait: Motion-Based Recognition Using Image Self-Similarity. LNCS, 2001, 2091：289～294
2 Hayfron-Acquah J B, Nixon M S, Carter J N. Automatic Gait Recognition by Symmetry Analysis. Pattern Recog Letters, 2003, 24：2175～2183
3 Lee L, Grimson WEL. Gait Analysis for Recognition and classification. In: Proceeding of the IEEE Conference on Face and Gesture Recognition, 2002. 155～161
4 Collins R, Gross R, Shi J. Silhouette-based Human Identification from Body Shape and Gait. In: Proc IEEE Conf, 2002
5 Wang L(王亮), Tan T N, Hu W M, et al. Silhouette analysis-based gait recognition for human identification. IEEE, 2003. 1505～1518
6 王亮，胡卫明，谭铁牛. 基于步态的身份识别. 计算机学报，2006, 26(3)：353～360
7 http://www.sinobiometrics.com


(上接第 176 页)

实验结果如表 3 所示，不难看出，HSDD_S 的正确识别率均分别优于其他 5 种特征抽取方法，基于散度差准则的特征抽取方法的特征抽取速度也优于基于 Fisher 准则的 LDA 和 KFD 方法。

表 3 6 种方法在最近邻分类器下的识别率(%)、特征抽取和识别时间(s)

| 特征抽取方法 | 识别率 | 特征抽取时间 | 分类时间 | 总时间 |
|---|---|---|---|---|
| LDA | 68.0 | 569.24 | 76.36 | 645.60 |
| KFD | 72.5 | 744.14 | 70.47 | 814.61 |
| SDD | 68.4 | 351.27 | 75.45 | 426.72 |
| HSDD_P | 74.5 | 514.89 | 72.39 | 587.28 |
| HSDD_G | 74.5 | 517.12 | 72.19 | 589.31 |
| HSDD_S | 75.1 | 516.46 | 72.08 | 588.54 |

**结论** 本文提出了一种新的非线性鉴别分析方法——基于散度差准则的隐空间图像特征抽取方法。对于图像的特征抽取来说，该方法具有以下优点：一是计算更高效，特征抽取的速度有了显著的提高；二是避免了高维小样本问题所引起的类内散布矩阵奇异的问题；三是不仅能够抽取非线性特征，同时，与现有的其他核方法相比，本文的方法不需要核函数满足正定条件，扩大了核函数的选取范围，这样可以选择更加适合具体问题的核函数。在 ORL 人脸数据库和 AR 人脸数据库上的实验结果验证了以上的结论。

## 参 考 文 献


1 Fisher R A. The use of multiple measurements in taxonomic problems. Annals of Eugenics, 1936, 7：178～188
2 Belhumeur P N, et al. Eigenfaces vs Fisherfaces: Recognition using class specific linear projections. IEEE Trans Pattern Anal Machine Intell, 1997, 19(7)：711～720
3 Hong Z Q, Yang J Y, et al. Optimal discriminant plane for a small number of samples and design method of classifier on the plane. Pattern Recognition, 1991, 24(4)：317～324
4 Liu K, Yang J Y, et al. An efficient algorithm for Foley-sammon optimal set of discriminant vectors by algebranic method. International Journal of Pattern Recognition and Artificial Intelligence, 1992, 6(5)：817～829
5 金忠，杨静宇，陆建峰. 一种具有统计不相关性的最优鉴别向量集. 计算机学报，1999, 22(10)：1105～1108
6 杨健. 线性投影分析的理论与算法及其在特征抽取中的应用研究：[学位论文]. 南京：南京理工大学，2002
7 Yang J, Zhang D, et al. Two-dimensional PCA: a new approach to appearance-based face representation and recognition. IEEE Trans Pattern Anal Machine Intell, 2004, 26(1)：131～137
8 Yang J, Yang J Y. From image vector to matrix: a straightforward image projectionIMPCA vs PCA. Pattern Recognition, 2002, 35(9)：1997～1999
9 宋枫溪，程科，杨静宇. 最大散度差和大间距线性投影与支持向量. 自动化学报，2004, 30(11)：890～896
10 程云鹏. 矩阵论. 西安：西北工业大学出版社，1999, 294～302