基于增量核主成分分析的数据流在线分类框架

吴 枫 [1] 仲 妍 [1] 吴泉源 [1]

摘 要 核主成分分析 (Kernel principal component analysis, KPCA) 是一种非线性降维工具, 在降低数据流分类处理量方面发挥着积极作用. 然而, 由于复杂性太高, 导致 KPCA 的降维能力有限. 为此, 本文给出了一种增量核主成分分析算法 (Incremental KPCA for dimensionality-reduction, IKDR), 该算法在每步迭代估计中只需线性内存开销, 大大降低了复杂性. 在 IKDR 的基础上, 结合 BP (Back propagation) 神经网络提出了数据流在线分类框架: IKOCFrame (Online classification frame based on IKDR). 通过一系列真实和人工数据集上的实验, 检验了 IKDR 算法的收敛性, 并且验证了 IKOCFrame 相对于同类基于成分分析的分类算法的优越性.

关键词 降维技术, 数据流分类, 增量核主成分分析, 独立成分分析

DOI 10.3724/SP.J.1004.2010.00534

Online Classification Framework for Data Stream Based on Incremental Kernel Principal Component Analysis

WU Feng[1] ZHONG Yan[1] WU Quan-Yuan[1]

Abstract Kernel principal component analysis (KPCA) has been suggested for various data stream classification tasks requiring a nonlinear transformation scheme to reduce dimensions. However, the dimensionality reduction ability is restricted because of its high complexity. Therefore this paper proposes an incremental kernel principal component analysis algorithm: IKDR, which iteratively estimates the kernel principal components with only linear order storage complexity per iteration. On the basis of IKDR, this paper proposes an online classification framework for data stream: IKOCFrame. Extensive experiments on real and artificial datasets validate the convergence of IKDR and confirm the superiority of IKOCFrame over other recent classification schemes based on component analysis.

Key words Dimensionality reduction, data stream classification, incremental kernel PCA (IKPCA), independent component analysis (ICA)

为保障网络和信息系统安全, 需要对网络监控数据流进行实时、高效地在线分类分析. 然而, 监控数据流具有无限、高速、连续等特点, 由此产生了高效处理需求与有限的计算、存储、网络带宽等资源之间的矛盾. 为此, 各种基于成分分析 (Component analysis, CA) 的降维技术被广泛应用于数据流分析中, 并在降低处理量方面发挥着重要作用[1]. 相关工作介绍如下:

主成分分析 (Principal component analysis, PCA)[2] 是一种有效的非监督降维技术. 它利用输入样本点空间的主成分元素, 归纳和提取其中的结构信息, 从而最终实现降维. 然而, 它在计算主成分时需要完整的样本点空间. 增量主成分分析 (Incremental PCA, IPCA)[3−5], 在计算主成分时不需要完整的样本点空间, 但同 PCA 一样, 无法处理线性不可分的问题.

核主成分分析 (Kernel PCA, KPCA)[2, 6] 是一种非线性主成分分析方法. 首先通过非线性映射 $\Phi: \mathbf{R}^N \to F$, $x \to X$, 将输入样本点空间 (原始空间) 的非线性问题变换到高维特征空间 F 中, 然后运用线性 PCA 方法对其进行处理. 它允许提取 l (原始空间中样本点个数) 个核成分, 相对于 PCA 而言, 能够更加完整、精确地归纳和表达原始空间的结构信息. 然而, KPCA 的计算复杂性很高, 只能处理极其有限的样本点.

独立成分分析 (Independent component analysis, ICA)[6−7] 是伴随盲信号分离问题发展起来的一项新的统计信号处理技术. ICA 假设观测数据是由一些相互独立的、服从非高斯分布的独立成分组合而成.

本文针对 PCA、ICA 不能处理线性不可分问题的弊端和 KPCA 复杂性太高的问题, 给出了增量核主成分分析算法 (Incremental KPCA for dimensionality-reduction, IKDR), 它首先将当前输

收稿日期 2008-12-26 录用日期 2009-10-22
Manuscript received December 26, 2008; accepted October 22, 2009
国家高技术研究发展计划 (863 计划) (2006AA01Z451, 2007AA01Z474) 资助
Supported by National High Technology Research and Development Program of China (863 Program) (2006AA01Z451, 2007AA01Z474)
1. 国防科学技术大学计算机学院 长沙 410073
1. School of Computers, National University of Defense Technology, Changsha 410073

入向量空间 (原始空间) 映射到高维特征空间, 然后增量提取当前特征空间的核成分, 最后对核向量进行降维. 在 IKDR 基础上, 结合 BP (Back propagation) 神经网络提出了数据流在线分类框架 IKOCFrame. 通过实验, 检验了 IKDR 算法的收敛性, 并且验证了 IKOCFrame (Online classification frame based on IRDR) 能够有效克服各种成分分析算法在数据流分类应用中存在的弊端.

本文组织结构如下: 在本文作者已有工作[8] 的基础上, 第 1 节给出 IKDR 算法的推导, 并对该算法进行了直观性解释和复杂性分析; 第 2 节以 IKDR 算法为基础, 提出 IKOCFrame; 第 3 节给出统计实验结果和分析.

1　增量核主成分分析算法 IKDR

1.1　IKDR 算法推导

令输入向量序列依时序到达: $\boldsymbol{\phi}(1),\boldsymbol{\phi}(2),\cdots$, 其中, $\boldsymbol{\phi}(t)$, $t=1,2,\cdots$ 为 L 维核化升维向量 (L 足够大); 不失一般性, 假设 $\mathrm{E}\{\boldsymbol{\phi}(t)\}=0$, $A=\mathrm{E}\{\boldsymbol{\phi}(t)\boldsymbol{\phi}(t)^{\mathrm{T}}\}$ 为未知 $L\times L$ 协方差矩阵; 令 $\Phi=(\boldsymbol{\phi}(1),\boldsymbol{\phi}(2),\cdots,\boldsymbol{\phi}(n),\cdots)$, $K=\Phi^{\mathrm{T}}\Phi$ 为核矩阵.

首先, 给出协方差矩阵 A 的主特征向量的迭代估计式. 令 $\boldsymbol{x}$ 是矩阵 A 的特征向量, λ 为对应的特征值, 则有 $\lambda\boldsymbol{x}=A\boldsymbol{x}$; 替换 A 为样本协方差矩阵, 并用第 n 步估计向量 $\boldsymbol{x}(n)$, $\boldsymbol{v}(n)=\lambda(n)\boldsymbol{x}(n)$ 分别替换 $\boldsymbol{x},\lambda\boldsymbol{x}$, 可以得到 $\boldsymbol{v}(n)=\dfrac{1}{n}\boldsymbol{v}(n)+\dfrac{(n-1)\sum\limits_{i=1}^{n}\boldsymbol{\phi}(i)\boldsymbol{\phi}(i)^{\mathrm{T}}\boldsymbol{v}(n)}{n^2\|\boldsymbol{v}(n)\|}$, 将其改写为迭代形式, 得到式 (1):

$$\boldsymbol{v}(n)=\frac{1}{n}\boldsymbol{v}(n-1)+\frac{(n-1)\sum\limits_{i=1}^{n}\boldsymbol{\phi}(i)\boldsymbol{\phi}(i)^{\mathrm{T}}\boldsymbol{v}(n-1)}{n^2\|\boldsymbol{v}(n-1)\|} \tag{1}$$

式 (1) 仅估计矩阵 A 的第一阶主特征向量. 由于特征向量相互正交, 于是可以在已估算出的低阶主特征空间的补空间中, 产生估计高阶主特征向量所需的输入向量. 例如: 第 n 步迭代时, 假定 $\boldsymbol{\varphi}_1(n)$ 为计算 $\boldsymbol{v}_1(n)$ (第一主特征向量的估计值) 所需的输入向量, 其中 $\boldsymbol{\varphi}_1(n)\boldsymbol{\varphi}_1(n)^{\mathrm{T}}=(\sum_{k=1}^{n}\boldsymbol{\phi}(k)\boldsymbol{\phi}(k)^{\mathrm{T}})/n$; 为计算 $\boldsymbol{v}_2(n)$ (第二主特征向量的估计值), 可以利用式 (2) 使得 $\boldsymbol{\varphi}_1(n)$ 与 $\boldsymbol{v}_1(n)$ 正交化, 从而产生计算 $\boldsymbol{v}_2(n)$ 所需的输入向量 $\boldsymbol{\varphi}_2(n)$, 具体正交化过程[3] 如下:

$$\boldsymbol{\varphi}_2(n)=\boldsymbol{\varphi}_1(n)-\boldsymbol{\varphi}_1(n)^{\mathrm{T}}\frac{\boldsymbol{v}_1(n)}{\|\boldsymbol{v}_1(n)\|}\frac{\boldsymbol{v}_1(n)}{\|\boldsymbol{v}_1(n)\|} \tag{2}$$

综上, 估计协方差矩阵 A 的第 i ($i\geq 1$) 阶主特征向量的迭代式为:

$$\boldsymbol{v}_i(n)=\frac{1}{n}\boldsymbol{v}_i(n-1)+\frac{(n-1)\boldsymbol{\varphi}_i(n)\boldsymbol{\varphi}_i(n)^{\mathrm{T}}\boldsymbol{v}_i(n-1)}{n\|\boldsymbol{v}_i(n-1)\|} \tag{3}$$

其中,

$$\boldsymbol{\varphi}_{i+1}(n)=\boldsymbol{\varphi}_i(n)-\boldsymbol{\varphi}_i^{\mathrm{T}}(n)\frac{\boldsymbol{v}_i(n)}{\|\boldsymbol{v}_i(n)\|}\frac{\boldsymbol{v}_i(n)}{\|\boldsymbol{v}_i(n)\|}=\prod_{j=1}^{i}\left[I-\frac{\boldsymbol{v}_j(n)\boldsymbol{v}_j^{\mathrm{T}}(n)}{\|\boldsymbol{v}_j(n)\|^2}\right]\boldsymbol{\varphi}_1(n)$$

接下来, 对式 (3) 进行等价变换, 给出核矩阵 K 的核特征向量的迭代估计式. 令 $\lambda\boldsymbol{x}$ 可以由 $\boldsymbol{\phi}(j)$ 线性表示为 $\lambda\boldsymbol{x}=\Phi\boldsymbol{a}$, 其中 $\boldsymbol{a}$ 为向量; 将其代入 $\lambda\boldsymbol{x}=A\boldsymbol{x}$ 中, 得到 $\Phi\Phi^{\mathrm{T}}\Phi\boldsymbol{a}/\lambda=A\boldsymbol{x}=\lambda\boldsymbol{x}=\Phi\boldsymbol{a}$, 即 $K\boldsymbol{a}=\lambda\boldsymbol{a}$, 显然, $\boldsymbol{a}/\|\boldsymbol{a}\|$ 就是 K 的核特征向量; 令 $\boldsymbol{v}_i(n)=\Phi(n)\boldsymbol{a}_i(n)$, $\Phi(n)=(\boldsymbol{\phi}(1),\cdots,\boldsymbol{\phi}(n))$, $\boldsymbol{a}_i(n)=(\boldsymbol{a}_{i1},\cdots,\boldsymbol{a}_{in})^{\mathrm{T}}$, 那么, $\boldsymbol{a}_i(n)/\|\boldsymbol{a}_i(n)\|$ 即为核特征向量的估计向量. 具体地, 分两种情形进行推导:

1) 当 $i=1$ 时, 将 $\boldsymbol{v}_1(n)=\Phi(n)\boldsymbol{a}_1(n)$ 代入式 (3) 中, 得

$$\Phi(n)\boldsymbol{a}_1(n)=\frac{1}{n}\Phi(n)(\boldsymbol{a}_1(n-1)^{\mathrm{T}},0)^{\mathrm{T}}+\frac{(n-1)\Phi(n)\Phi(n)^{\mathrm{T}}\Phi(n)(\boldsymbol{a}_1(n-1)^{\mathrm{T}},0)^{\mathrm{T}}}{n^2\|\Phi(n-1)\boldsymbol{a}_1(n-1)\|} \tag{4}$$

因为

$$\|\Phi(n-1)\boldsymbol{a}_1(n-1)\|=\sqrt{\boldsymbol{a}_1(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_1(n-1)} \tag{5}$$

将式 (5) 代入式 (4) 中, 得

$$\boldsymbol{a}_1(n)=\frac{1}{n}(\boldsymbol{a}_1(n-1)^{\mathrm{T}},0)^{\mathrm{T}}+\frac{(n-1)K(n)(\boldsymbol{a}_1(n-1)^{\mathrm{T}},0)^{\mathrm{T}}}{n^2\sqrt{\boldsymbol{a}_1(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_1(n-1)}} \tag{6}$$

2) 定义 $\Pi_i(n)=\prod_{j=1}^{i-1}\left[I-\dfrac{\boldsymbol{v}_j(n)\boldsymbol{v}_j^{\mathrm{T}}(n)}{\|\boldsymbol{v}_j(n)\|^2}\right]$, 在式 (3) 中, 当 $i>1$ 时, 有:

$$\boldsymbol{v}_i(n)=\frac{n-1}{n}\times\left(\frac{\dfrac{\Phi(n)\Phi(n)^{\mathrm{T}}}{n}-\sum\limits_{j=1}^{i-1}\dfrac{\boldsymbol{v}_j(n)\boldsymbol{v}_j(n)^{\mathrm{T}}}{\|\boldsymbol{v}_j(n)\|}}{\|\boldsymbol{v}_i(n-1)\|}-I\right)\times\boldsymbol{v}_i(n-1)+\boldsymbol{v}_i(n-1)\tag{7}$$

$$\sum_{j=1}^{i-1}\frac{\boldsymbol{v}_j(n)\boldsymbol{v}_j(n)^{\mathrm{T}}}{\|\boldsymbol{v}_j(n)\|}=\Phi(n)\left(\frac{\boldsymbol{a}_1(n)}{\sqrt[4]{\boldsymbol{a}_1(n)^{\mathrm{T}}K(n)\boldsymbol{a}_1(n)}},\cdots,\frac{\boldsymbol{a}_{i-1}(n)}{\sqrt[4]{\boldsymbol{a}_{i-1}(n)^{\mathrm{T}}K(n)\boldsymbol{a}_{i-1}(n)}}\right)\cdot\left(\frac{\boldsymbol{a}_1(n)}{\sqrt[4]{\boldsymbol{a}_1(n)^{\mathrm{T}}K(n)\boldsymbol{a}_1(n)}},\cdots,\frac{\boldsymbol{a}_{i-1}(n)}{\sqrt[4]{\boldsymbol{a}_{i-1}(n)^{\mathrm{T}}K(n)\boldsymbol{a}_{i-1}(n)}}\right)^{\mathrm{T}}\cdot\Phi(n)^{\mathrm{T}}\tag{8}$$

令

$$A_{i-1}(n)=\left(\frac{\boldsymbol{a}_1(n)}{\sqrt[4]{\boldsymbol{a}_1(n)^{\mathrm{T}}K(n)\boldsymbol{a}_1(n)}},\cdots,\frac{\boldsymbol{a}_{i-1}(n)}{\sqrt[4]{\boldsymbol{a}_{i-1}(n)^{\mathrm{T}}K(n)\boldsymbol{a}_{i-1}(n)}}\right)^{\mathrm{T}}\tag{9}$$

将式 (8) 和式 (9) 代入式 (7) 中, 得到式 (10):

$$\boldsymbol{a}_i(n)=\frac{1}{n}(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}}+\left(\frac{(n-1)I}{n^2\sqrt{\boldsymbol{a}_i(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_i(n-1)}}-\frac{(n-1)A_{i-1}(n)^{\mathrm{T}}A_{i-1}(n)}{n\sqrt{\boldsymbol{a}_i(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_i(n-1)}}\right)\times K(n)(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}}\tag{10}$$

综合情形 1) 和 2) 得到 IKDR 算法: 首先, 根据式 (6) 更新 $\boldsymbol{a}_1(n)$, 其中 $A_0=0$, $\boldsymbol{a}_1(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_1(n-1)$ 由前次迭代过程计算得到; 接着, 利用式 (10) 更新 $\boldsymbol{a}_2(n)$, 其中 $A_1(n)$ 需要由 $\boldsymbol{a}_1(n)$ 和 $K(n)$ 计算得到; 最后, 类似于 $\boldsymbol{a}_2(n)$, 依次更新 $\boldsymbol{a}_i(n)$, $k\geq i\geq 3$. 不难看出, IKDR 算法能够有效利用前次迭代结果进行增量估计, 从而降低计算复杂性.

算法 1. IKDR 算法

Input 新达到核矩阵 $K(n)$ $(n\geq 2)$; $\boldsymbol{a}_1(1)=\cdots=\boldsymbol{a}_k(1)=1$; $K(1)=\boldsymbol{\phi}(1)^{\mathrm{T}}\boldsymbol{\phi}(1)$.

Output 第 n 个核向量到达时, 亦即第 n 步迭代时, 计算前 k 个核成分: $\boldsymbol{a}_1(n)/\|\boldsymbol{a}_1(n)\|,\cdots,\boldsymbol{a}_k(n)/\|\boldsymbol{a}_k(n)\|$.

Method

For $i=1, 2, \cdots, k$ do

1) 计算 $A_{i-1}(n)$, 其中 $A_0=0$:

$$A_{i-1}(n)=\left(\frac{\boldsymbol{a}_1(n)}{\sqrt[4]{\boldsymbol{a}_1(n)^{\mathrm{T}}K(n)\boldsymbol{a}_1(n)}},\cdots,\frac{\boldsymbol{a}_{i-1}(n)}{\sqrt[4]{\boldsymbol{a}_{i-1}(n)^{\mathrm{T}}K(n)\boldsymbol{a}_{i-1}(n)}}\right)^{\mathrm{T}}$$

2) 更新第 i 个核成分 $\boldsymbol{a}_i(n)$:

$$\boldsymbol{a}_i(n)=\frac{1}{n}(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}}+\left(\frac{(n-1)I}{n^2\sqrt{\boldsymbol{a}_i(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_i(n-1)}}-\frac{(n-1)A_{i-1}(n)^{\mathrm{T}}A_{i-1}(n)}{n\sqrt{\boldsymbol{a}_i(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_i(n-1)}}\right)\times K(n)(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}}$$

End For

1.2 IKDR 算法举例

本节中, 我们将通过一个例子阐释 IKDR 算法的工作原理. 该例子中, 我们将估算前 2 个核主成分, 即 $k=2$. 令第 n 个输入核矩阵为 $K(n)$, $K(n)$ 为 $n\times n$ 阶矩阵, $K(1)=1$, $K(2)=\begin{bmatrix}1&2\\2&3\end{bmatrix}$; 初始化 $\boldsymbol{a}_1(1)=\cdots=\boldsymbol{a}_k(1)=1$.

当第 $n=2$ 个输入核矩阵为 $K(2)$ 到达时, 调用 IKDR 算法, 进入循环过程:

1) 当 $i=1$ 时, 计算 $\boldsymbol{a}_1(2)$:

$$\boldsymbol{a}_1(2)=\frac{1}{2}(\boldsymbol{a}_1(1)^{\mathrm{T}},0)^{\mathrm{T}}+\frac{K(2)(\boldsymbol{a}_1(1)^{\mathrm{T}},0)^{\mathrm{T}}}{2^2\sqrt{\boldsymbol{a}_1(1)^{\mathrm{T}}K(1)\boldsymbol{a}_1(1)}}=[0.75,0.5]^{\mathrm{T}};$$

2) 当 $i=2$ 时, 计算 $\boldsymbol{a}_2(2)$:

a) 首先计算 $A_1(2)$:

$$A_1(2)=\frac{\boldsymbol{a}_1(2)^{\mathrm{T}}}{\sqrt[4]{\boldsymbol{a}_1(2)^{\mathrm{T}}K(2)\boldsymbol{a}_1(2)}}=[0.58,0.38];$$

b) 接下来计算 $\boldsymbol{a}_2(2)$:

$$\boldsymbol{a}_2(2)=\frac{1}{2}(\boldsymbol{a}_2(1)^{\mathrm{T}},0)^{\mathrm{T}}+\left(\frac{I}{2^2\sqrt{\boldsymbol{a}_2(1)^{\mathrm{T}}K(1)\boldsymbol{a}_2(1)}}-\frac{A_1(2)^{\mathrm{T}}A_1(2)}{2\sqrt{\boldsymbol{a}_2(1)^{\mathrm{T}}K(1)\boldsymbol{a}_2(1)}}\right)\times K(2)(\boldsymbol{a}_2(1)^{\mathrm{T}},0)^{\mathrm{T}}=[0.365,0.25]^{\mathrm{T}}$$

当 $\boldsymbol{a}_1(2)$ 和 $\boldsymbol{a}_2(2)$ 计算完毕, 循环结束.

当第 $n=3$ 个输入核矩阵为 $K(3)=\begin{bmatrix}1&2&2\\2&3&3\\2&3&5\end{bmatrix}$ 到达时, 重复上述过程计算 $\boldsymbol{a}_1(3)$ 和 $\boldsymbol{a}_2(3)$, 此时 $\boldsymbol{a}_1(3)$ 和 $\boldsymbol{a}_2(3)$ 的维数将变为 3.

1.3 IKDR 的直观解释

考虑二维向量集的情况. 令 $\boldsymbol{v}_1, \boldsymbol{v}_2$ 分别代表该向量集的第一阶和第二阶主特征向量, 如图 1 所示, $\boldsymbol{v}_1, \boldsymbol{v}_2$ 分别与椭圆长轴和短轴平行; 令 $\boldsymbol{v}_1(n-1)$ 代表 $\boldsymbol{v}_1$ 的第 $n-1$ 步估计向量, 虚线 l 与 $\boldsymbol{v}_1(n-1)$ 垂直, 并将整个椭圆平面划分为上下两部分. 因为椭圆是中心对称的, 可以将下平面中的任意向量 $\boldsymbol{\phi}_l$ 等价旋转至上平面 (产生的新向量记为 $-\boldsymbol{\phi}_l$), 这样只需分析上平面的情况.

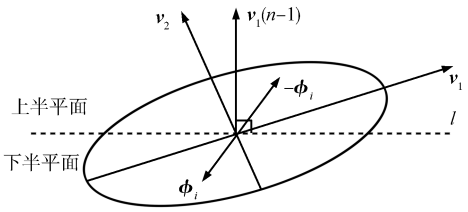


图 1　IKDR 直观解释[3]

Fig. 1　Intuitive explanation of IKDR[3]

当 n 足够大时, 结合文献 [8] 中的收敛性分析, 式 (3) $(i=1)$ 可以近似改写为:

$$\boldsymbol{v}_1(n)=\left(\frac{1}{n}+\frac{(n-1)^2}{n^2}\right)\boldsymbol{v}_1(n-1)+\frac{(n-1)\boldsymbol{\phi}(n)\boldsymbol{\phi}(n)^{\mathrm{T}}\boldsymbol{v}_1(n-1)}{n^2\|\boldsymbol{v}_1(n-1)\|}$$

注意到 $\dfrac{(n-1)\boldsymbol{\phi}(n)^{\mathrm{T}}\boldsymbol{v}_1(n-1)}{n^2\|\boldsymbol{v}_1(n-1)\|}$ 实际上是个标量, 所以 $\dfrac{(n-1)\boldsymbol{\phi}(n)\boldsymbol{\phi}(n)^{\mathrm{T}}\boldsymbol{v}_1(n-1)}{n^2\|\boldsymbol{v}_1(n-1)\|}$ 实际上是向量 $\boldsymbol{\phi}(n)$ 的某个常数倍. 由此可见, $\boldsymbol{v}_1(n)$ 就是向量 $\boldsymbol{v}_1(n-1)$ 和 $\boldsymbol{\phi}(n)$ 的加权和. 因为下平面的任意向量 $\boldsymbol{\phi}_l$ 与 $\boldsymbol{v}_1(n-1)$ 的夹角均为钝角, 所以, $\boldsymbol{\phi}_l^{\mathrm{T}}\boldsymbol{v}_1(n-1)$ 为负值, 上式可以改写为:

$$\boldsymbol{v}_1(n)=\left(\frac{1}{n}+\frac{(n-1)^2}{n^2}\right)\boldsymbol{v}_1(n-1)+\frac{(n-1)}{n^2}\times\left|\frac{\boldsymbol{\phi}_l^{\mathrm{T}}\boldsymbol{v}_1(n-1)}{\|\boldsymbol{v}_1(n-1)\|}\right|\cdot(-\boldsymbol{\phi}_l)$$

根据上式可知, 任意向量 $-\boldsymbol{\phi}_l$ 对 $\boldsymbol{v}_1(n-1)$ 的纯作用力均表现为拉力 (即, 使得 $\boldsymbol{v}_1$ 下一步的估计向量更加靠近 $-\boldsymbol{\phi}_l$), 那么综合考虑上平面所有向量的合力, 容易知道只要两个主特征值大小不同, $\boldsymbol{v}_1(n-1)$ 必定向 $\boldsymbol{v}_1$ 方向收敛, 类似于文献 [3] 中第 2.2 节的分析, $\boldsymbol{v}_1(n)$ 将收敛于第一主特征向量.

1.4 IKDR 复杂性分析

在式 (10) 的每一步更新中, $A_k(n)$ 的计算量主要集中在 $A(n)^{\mathrm{T}}K(n)$ 上面, 其计算开销为 $\mathrm{O}(k\times n^2)$. 因为 $\boldsymbol{a}_i(n-1)^{\mathrm{T}}K(n-1)\boldsymbol{a}_i(n-1)$ 能够通过迭代得到, 并且等式 $K(n)(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}}=((K(n-1)\boldsymbol{a}_i(n-1))^{\mathrm{T}},K(n)_{n,*}\cdot(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}})^{\mathrm{T}}$ ($K(n)_{n,*}$ 代表 $K(n)$ 的第 n 行) 仅仅需要计算 $K(n)_{n,*}\cdot(\boldsymbol{a}_i(n-1)^{\mathrm{T}},0)^{\mathrm{T}}$ 便可得出, 所以两者的开销为 $\mathrm{O}(n)$. 综上, 更新所有的 $\boldsymbol{a}_i(n)$ 的开销为 $\mathrm{O}(k\times n^2)$. 如果假设核函数的计算开销为 $\mathrm{O}(L)$ (针对高斯核和多项式核), 那么在 IKDR 每步迭代中, 总的计算开销为 $\mathrm{O}(k\times n\times L+k\times n^2)=O(k\times n^2)$. 相对于 KPCA 的时间复杂度 $\mathrm{O}(n^3)$ 而言, 具有明显的优势.

在式 (10) 的每一步更新中, 存储系数矩阵的内存开销为 $\mathrm{O}(k\times n)$ (k 为需要计算的主成分的个数, n 为输入向量的个数), 存储训练集的开销为 $\mathrm{O}(n\times L)$ (L 为原始空间的维度), 还有存储 $(K(n-1)\boldsymbol{a}_i(n-1))^{\mathrm{T}}$ $(i=1,\cdots,k)$ 的开销为 $\mathrm{O}(k\times n)$. 忽略小的开销, 那么在 IKDR 每步迭代中, 总的内存开销为 $\mathrm{O}(k\times n+n\times L+k\times n)$. 相对于 KPCA 的空间复杂度 $\mathrm{O}(n^2)$ 而言, 具有明显的优势.

2 数据流在线分类框架 IKOCFrame

数据流在线分类算法 IKOCFrame 的工作原理如下: 1) 首先对 BP 神经网络进行初始化: 学习速率设定为 0.05, 期望误差设定为 0.01, 隐层初始权值全部设定为 0.01, 输出层初始权值设定为 +1, −1 的权值数相等. 2) 初始化过程结束以后, 调用 IKDR 算法对神经网络进行训练, 同时建立特征库, 算法中采用的核函数为高斯核, 参数 $\sigma=1$; 系统初始化过程和 BP 神经网络训练过程均离线完成. 3) 在线分类过程开始后, 系统实时对采集到的数据流进行核化变换, 得到特征数据流, 然后对其进行降维处理后, 送入 BP 神经网络分类引擎进行分类. IKOCFrame 的具体工作流程见图 2.

设 $\{\boldsymbol{x}(1),\cdots,\boldsymbol{x}(n),\cdots\}$ 为原始空间中无穷可列的输入向量集; $\{\boldsymbol{\phi}(1),\cdots,\boldsymbol{\phi}(n),\cdots\}$ 为核化升维后的核向量集; n 个核向量形成的核矩阵定义为 $K(n)$, 其中, $K(n)_{i,j}=(\boldsymbol{\phi}(i),\boldsymbol{\phi}(j))$. 有限集 $\{c_1,\cdots,c_m\}$ 为原始空间中输入向量所属的 m 个类的集合. 对每个 $\boldsymbol{x}$, $M(\boldsymbol{x})$ 为 $\boldsymbol{x}$ 的所属类号, 例如, $\boldsymbol{x}(i)$

$\in c_j$, 则 $M(\boldsymbol{x}(i)) = j$.

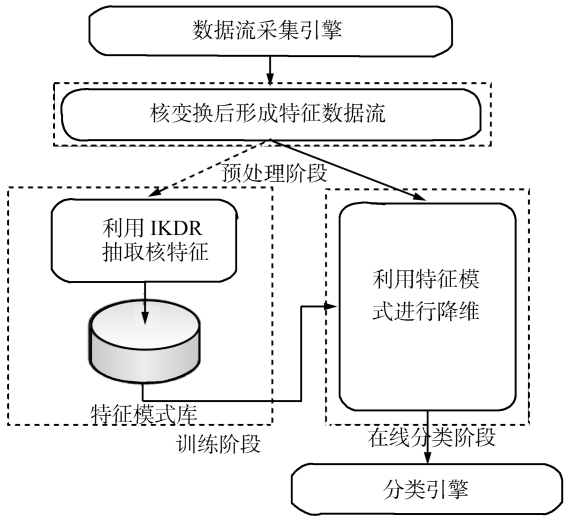


图 2 IKOCFrame 流程图

Fig. 2 The flow chart of the IKOCFrame

算法 2. IKOCFrame 算法

Input 新到达第 n 个输入向量: $\boldsymbol{x}(n)$; 训练矩阵为 $D_{\text{train}} = [t(1), t(2), \cdots, t(N)]^{\text{T}}_{N\times L}$, 其中, N 为训练数据集的大小, L 为训练样本点 $t(i)$ $(i = 1, \cdots, N)$ 维数. 采用高斯核: $\sigma = 1$.

Output $M(\boldsymbol{x}(n))$.

Method

1) 训练阶段

利用选定的特殊训练集对 BP 神经网络进行训练, 训练知识以权值的方式被保存下来:

a) 计算核矩阵 D_{train}:

$$K_{\text{train}} = kernel\_matrix(D_{\text{train}}, \ 'RBF\_kernel', \sigma, D_{\text{train}});$$

b) 通过 IKDR 算法, 计算 K_{train} 的前 k 个核主成分: $\boldsymbol{a}_1/\|\boldsymbol{a}_1\|, \cdots, \boldsymbol{a}_k/\|\boldsymbol{a}_k\|$, 然后对 $K^{\text{T}}_{\text{train}}$ 进行降维, 其中, $C = K^{\text{T}}_{\text{train}} \cdot PVvec \cdot PCcoeff$ 且

$$PCcoeff = \begin{bmatrix} \|\boldsymbol{a}_1\| & 0 & \cdots & 0 \\ 0 & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \|\boldsymbol{a}_k\| \end{bmatrix};$$

$$[C, PCvec, PCcoeff] = IKDR(N, K_{\text{train}});$$

c) 置换矩阵 C: $P = C^{\text{T}}$;

d) 利用 D_{train} 的类标记向量 $\boldsymbol{L}_{\text{train}}$ 和 C 训练 BP 神经网络.

2) 在线分类阶段

a) 定义到达的第 n 个输入向量为 $\boldsymbol{x}(n)$;

b) 计算 D_{train} 和 $\boldsymbol{x}(n)$ 之间的核矩阵:

$$K_{\text{classify},n} = kernel\_matrix(D_{\text{train}}, \ 'RBF\_kernel', \sigma, \boldsymbol{x}(n)^{\text{T}});$$

c) 利用 $PCvec$ 和 $PCcoeff$ 对 $K_{\text{classify},n}$ 进行降维:

$$C_1 = K^{\text{T}}_{\text{classify},n} \cdot PCvec \cdot PCcoeff;$$

d) 转置 C_1: $P_1 = C_1^{\text{T}}$;

e) 利用 BP 神经网络和 P_1 对 $\boldsymbol{x}(n)$ 进行分类, 得到分类标记 $M(x(n))$.

3 统计实验

所有实验均在一台 AMD Athlon Dual Core CPU 2.2 GHz 的 PC 机上进行, 操作系统为 Windows XP, 算法采用 Matlab 6.x 实现.

3.1 IKDR 收敛性验证

本节实验旨在验证 IKDR 算法的收敛性, 亦即验证式 (3) 的收敛性. 为检验式 (3) 对特征向量的估计效果, 在每步迭代过程中, 对特征向量的估计值 $\boldsymbol{v}/\|\boldsymbol{v}\|$ 与真实值 e 之间的相关系数 (即内积 $(\boldsymbol{v}/\|\boldsymbol{v}\|, e)$) 进行统计, 显然, 相关系数越接近 1, 则估计效果越好. 为检验特征值的估计效果, 通过计算 $\|\boldsymbol{v}_i\|/\lambda_i$ 来衡量, 显然, 比率系数越接近 1, 则估计效果越好.

通过 4 组实验验证式 (3) 的收敛性. 每组实验中, 采用相同大小、不同维数和不同来源的数据集, 其中, A 中数据集由 Matlab 随机生成, 维数为 100; B 中数据集由 Matlab 随机生成, 维数为 50; C 中数据集为 KDDCUP 99 (10 %) 数据集, 维数为 41. 实验 1 ∼ 4 组中数据集大小分别为 1 000, 2 000, 5 000 和 10 000. 前 5 个主特征向量和特征值的收敛性实验结果见图 3.

由上述 4 组实验结果可以看出: 绝大多数时候, 前五个主特征向量的估计值与真实值之间的相关系数均保持在 80 % 以上; 任何时候, 前五个主特征值的估计值与真实值之间的比率系数均保持在 0.95 ∼ 1.05 之间, 其中前三个主特征向量的估计值与真实值之间的比率系数均接近 1. 由此可见, 算法的收敛性能良好. 另外, 发现算法的收敛速度随着维数的减少而提高, 并且, 收敛效果也随着维数的减少而更好. 对算法整体性能 (收敛速度和效果) 而言, 具有良好结构的数据集 (KDDCUP 99 (10 %) 数据集) 比不具备任何结构的数据集 (随机数据) 优势明显.

3.2 IKOCFrame 有效性验证

实验数据采用 KDDCUP 99 数据集[9−10], 六种不同领域[10−12]、不同维数的真实数据集以及若干满足一系列高斯分布的人工数据集, 并采用如下标记进行命名[13]: 'B' 表示数据集中数据点个数, 'C' 和 'D' 分别表示类的个数和数据点的维数.

3.2.1 分类质量

本小节对 KOCFrame、IKOCFrame 和 FIOCFrame 算法的分类质量进行比较测试, 其中, KOC-

Frame 和 FIOCFrame 分别是用 KPCA 和 FastICA 算法替换 IKOCFrame 中 IKDR 算法得到的. 实验中, 基于核的成分分析算法均采用高斯核, 参数 $\sigma = 1$; KPCA, IKDR 和 FastICA 均将 41 个特征降为 3 个特征.

首先, 利用 KDDCUP 99 数据集对比验证三种方法在网络入侵检测中的有效性. 实验中, 采用相同训练集分别训练 KOCFrame、IKOCFrame 和 FIOCFrame, 训练集大小 N_{train} 分别取为 800, 1 000, 1 200, 1 500, 2 000 和 2 500; 然后, 使用 N_{test} 取为 400 000 的测试集评估 3 种方法的假阴性率 (False negative rate, FNR)、假阳性率 (False positive rate, FPR) 和准确率 (Accurate, ACC), 其具体定义如下:

$$FNR = \frac{disDet}{attackNum},\quad FNR = \frac{misDet}{normalNum},$$
$$ACC = \frac{Det}{totalNum}$$

其中, $attackNum$ 代表数据集中攻击行为的数量; $disDet$ 代表在数据集的所有攻击行为中, 算法未能准确判断其攻击类型的数量; $normalNum$ 代表数

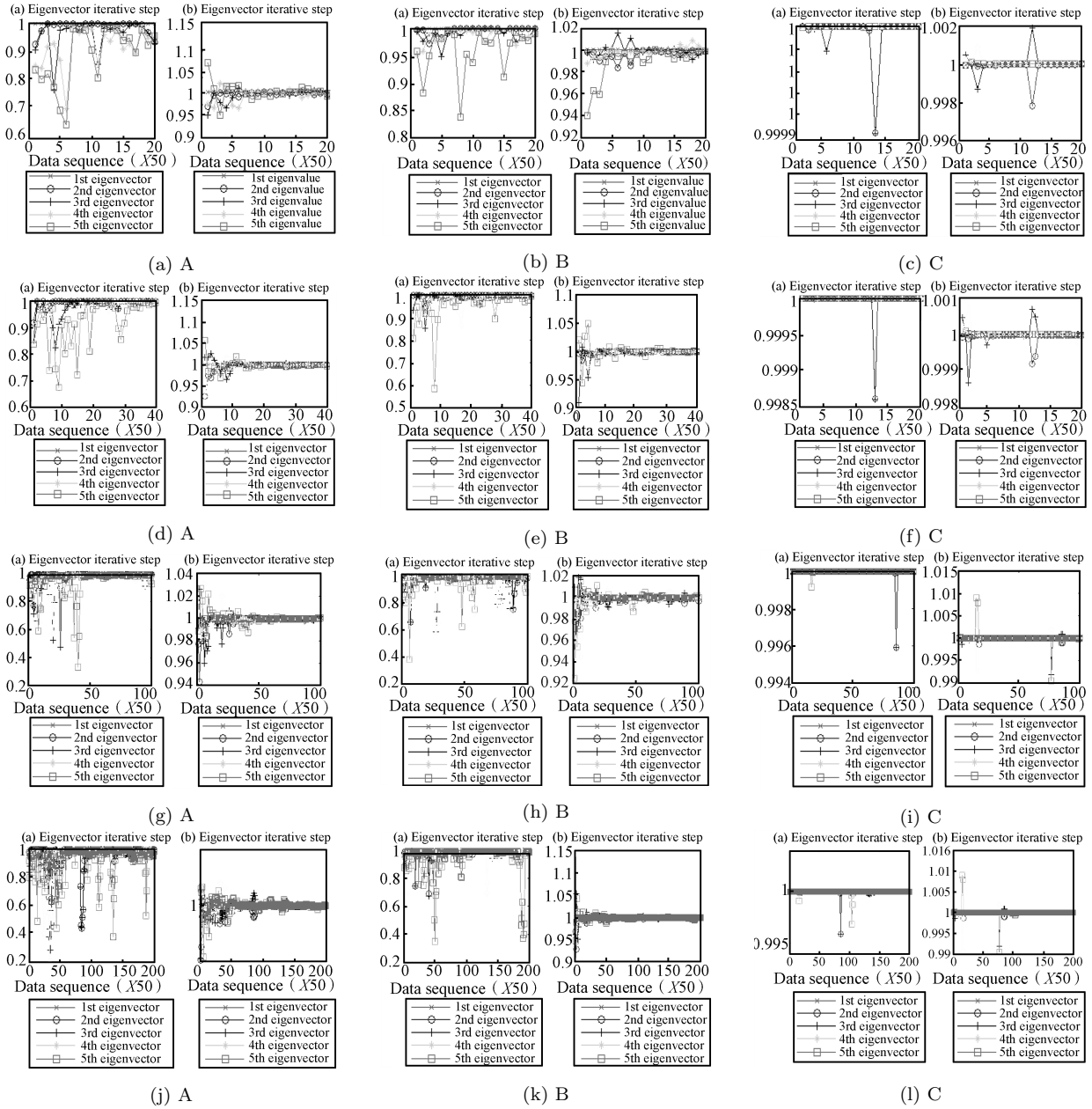


图 3　4 组收敛性分析

Fig. 3　The convergence analysis in 4 groups

据集中正常行为的数量; $misDet$ 代表在数据集的所有正常行为中, 算法误判为攻击行为的数量; $totalNum$ 代表数据集中所有行为的总和; Det 代表在数据集的所有行为中, 算法准确判断其行为类型的数量. 不难看出, 假阴性率可理解为狭义的漏检率, 假阳性率可理解为虚警率. 重复 5 次随机实验, 平均实验结果见表 1.

结果显示, IKOCFrame、KOCFrame 和 FIOCFrame 的 FNR 平均值分别为 1.214 %, 1.043 %, 0.918 %; FPR 平均值分别为 5.48 %, 8.862 %, 13.415 %; ACC 平均值分别为 97.9 %, 97.326 %, 96.475 %. 可以看出, IKOCFrame 在攻击行为检测方面, 尽管检测性能最差, 但是与其他两种方法相比, 实际差距并不明显; 在正常行为识别和行为分类方面, 性能明显占优. 整体上而言, 在分类质量方面, IKOCFrame 最优, FIOCFrame 最差. 另外, 值得一提的是, 在训练集大小取 2 000 时的某次分类实验中, KOCFrame 的 FPR 居然达到了 38.66 %, 而 IKOCFrame 仅为 8.3 %; 这是由于 KPCA 复杂性过高导致的: 随着核矩阵的阶数不断增加, 算法的内存开销和计算复杂度急剧上升, 使得处理速度严重下降, 性能也变得很不稳定; 而 IKDR 可以通过增量求取、迭代逼近的方法, 降低内存开销, 从而有效地克服了 KPCA 的不稳定因素. 最后, 通过观察发现, 算法的分类质量并没有随训练样本点的增加而呈现出递增趋势, 这说明分类质量与训练集的选择具有密切的关联, 只要选取得当, 即使较小的训练集也能训练出性能良好的分类器.

其次, 采用不同领域、不同维数的数据集对比验证三种方法的有效性. 验证方式采用交叉验证法, 实验结果见表 2.

结果显示, IKOCFrame、KOCFrame 和 FIOCFrame 的 ACC 平均值分别为 85.80 %, 75.82 %, 83.84 %. 不难看出, IKOCFrame 的分类质量最高; 另外, IKOCFrame 不仅在 5 个数据集中具有最高的分类精度, 而且在 3 个数据集中 ACC 达到了 90 % 以上, 说明即使面对多样化的应用领域, IKOCFrame 仍然能够保持良好的分类质量; 最后, 观察发现, 在 Coffee 数据集中 KOCFrame 的 ACC 居然只有 35.19 %, 说明该方法的分类性能极为不稳定.

3.2.2 分类效率

分类效率通过分类执行时间来衡量. 本小节在不同长度、不同维数和不同类个数的数据流上检验 IKOCFrame 的分类效率. 特别地, 为考察算法处理高速数据流的能力, 选取的数据规模均在 10 万条以上. 实验中, IKOCFrame 采用高斯核, 参数 $\sigma = 1$. N_{train} 从 1 000 到 2 500 变化.

从 100 K 到 400 K 变化元组个数, 同时固定元组的维数和类的个数, 考察数据流长度变化对算法效率的影响. 图 4 (a) 给出了 KDDCUP 99 数据集上

表 1 KDDCUP 99 数据集下, IKOCFrame、KOCFrame 和 FIOCFrame 间的比较 (%)

Table 1 Comparison among IKOCFrame\KOCFrame\FIOCFrame under the KDDCUP 99 dataset (%)

算法		800	1 000	1 200	1 500	2 000	2 500
IKOCFrame	FNR	1.03	0.898	1.122	2.08	0.938	1.214
	FPR	7.48	8.808	3.754	6.08	5.126	1.63
	ACC	97.626	97.452	98.33	97.086	98.19	98.696
KOCFrame	FNR	1.068	1.236	1.07	0.874	0.922	1.09
	FPR	5.15	8.926	8.738	10.21	12.256	7.896
	ACC	98.08	97.16	97.332	97.178	96.714	97.49
FIOCFrame	FNR	0.87	0.854	0.886	0.984	0.862	1.052
	FPR	13.06	17.529	14.144	8.716	13.278	13.764
	ACC	96.588	95.654	96.354	97.404	96.552	96.3

表 2 六种数据集下, IKOCFrame、KOCFrame 和 FIOCFrame 间的比较 (%)

Table 2 Comparison among IKOCFrame\KOCFrame\FIOCFrame under six dtatsets (%)

数据集	维数	ACC		
		IKOCFrame	KOCFrame	FIOCFrame
Gun	151	82.78	70.56	83.33
Pima	9	73.57	73.05	71.88
Flare	11	79.98	78.76	77.47
PageBlocks	11	90.13	89.33	90.00
Mafer	153	95.75	90.00	95.17
Coffee	287	92.59	35.19	85.19
Average ACC		85.80	75.82	83.84

的实验结果. 结果显示, 随着数据流长度的增加, 执行时间呈线性增长趋势, 说明算法效率随着数据流的进行保持平稳; 而且, 7 秒内能够处理 400 K 个元组, 说明算法可以满足高速数据流的处理要求.

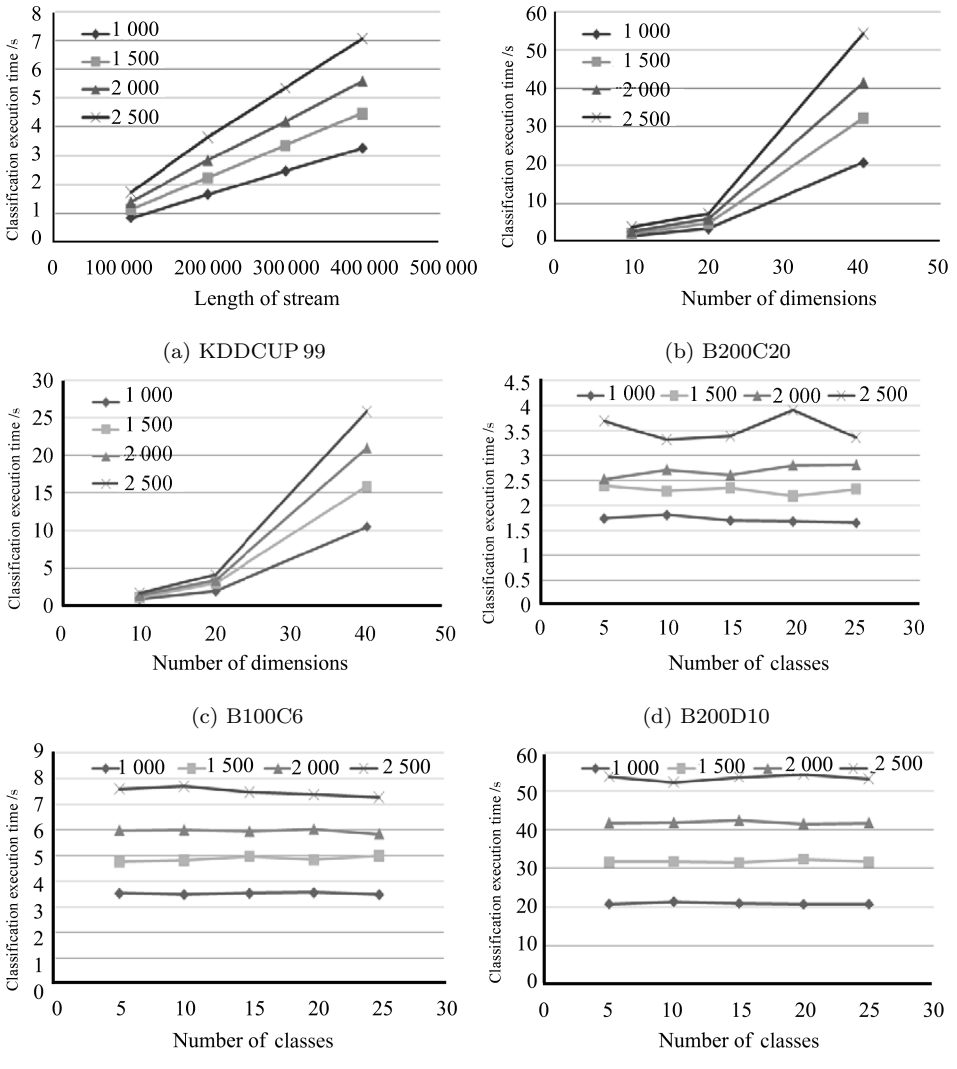
从 10 到 40 变化元组的维数, 同时固定元组的个数和类的个数, 考察维数变化对算法效率的影响. 图 4 (b) ∼ 4 (c) 给出了 B200C20 和 B100C5 数据集上的实验结果. 结果显示, 对所有数据集而言, 随着维数的增加, 执行时间呈现相同的非线性增长趋势; 当维数为 10 和 20 时, 执行时间相差无几, 但当维数大于 25 时, 执行时间增长幅度突然变大. 这说明, 随着维数的增加, 算法效率会随之下降.

从 5 到 25 变化类的个数, 同时固定元组的个数和维数, 考察类的个数变化对算法效率的影响. 图 4 (d) ∼ 4 (f) 给出了 B200D10、B200D20 和 B200D40 数据集上的实验结果. 结果显示, 对所有 3 个数据集而言, 随着类个数的增加, 执行时间基本保持不变, 这说明算法效率与类的个数无关.

3.2.3　核函数选择和参数敏感性测试

本小节针对 IKOCFrame 中核函数的选取问题进行讨论. 核函数的确定对于特定数据分布的特征提取具有关键性作用, 但如何选择合适的核函数, 没有统一的标准可循, 本文主要通过实验的方法对核函数进行选择和验证. 实验中, 采用 2 种常用核函数: 高斯核和多项式核, 分别对 IKOCFrame 的假阴性率、假阳性率和准确率进行统计 (见表 3). 数据集采用 KDDCUP 99 数据集, 训练集大小取为 800, 1 000, 1 200; 测试集共 400 000 条记录. 重复 5 次随机实验, 平均实验结果见表 3.

从高斯核和多项式核的对比中可以看到, 在采用高斯核时, 参数 σ 对 IKOCFrame 的分类质量影



(a) KDDCUP 99　(b) B200C20

(c) B100C6　(d) B200D10

(e) B200D40　(f) B200D40

图 4　执行时间分别随数据流长度、维数和类个数变化

Fig. 4　Execution time vs. length of stream\number of dimensions\number of classes

表 3 基于各种核函数的 IKOCFrame 性能比较 (%)

Table 3 The performance comparison among IKOCFrames based on different kernel functions (%)

高斯核	FNR	FPR	ACC	多项式核	FNR	FPR	ACC
$\sigma=0.1$	2.0187	12.292	95.8380	$d=1$	0.7167	17.756	95.734
$\sigma=1$	1.0167	6.6793	97.8027	$d=2$	2.5413	6.9227	96.546
$\sigma=10$	0.932	13.834	96.3793	$d=3$	0.9368	15.0482	96.1232
$\sigma=100$	0.8667	15.864	96.0087	$d=4$	2.3135	7.563	96.5922

响较大, 并集中体现在 FPR 方面; 当 σ 为 1 时, 分类质量最好, σ 过大或过小时, 分类质量较差. 采用多项式核时, 阶数 d 对 IKOCFrame 的分类质量影响类似于 σ; 当 $d=1$ 时, 分类质量最差. 另外, 通过实验发现, 阶数过高也会导致分类质量急剧下降.

4 总结

本文针对 PCA、ICA 不能处理线性不可分问题的弊端和 KPCA 复杂性太高的问题, 给出了增量核主成分分析算法 IKDR, 并在此基础上结合 BP 神经网络提出了数据流在线分类框架 IKOCFrame. 通过实验分析, 检验了 IKDR 算法的收敛性; 同时, 验证了 IKOCFrame 具有很好的可扩展性, 并能在不同领域的应用中, 保持稳定、良好的分类质量. 总体来说, IKOCFrame 能够有效克服各种成分分析算法在数据流分类应用中存在的弊端.

References

1 Martinez A M, Kak A C. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2001, **23**(2): 228−233

2 Mika S, Scholkopf B, Smola A, Muller K R, Scholz M, Ratsch G. Kernel PCA and de-noising in feature spaces. In: Proceedings of the Conference on Advances in Neural Information Processing Systems. Denver, Colorado: MIT Press, 1999. 536−542

3 Weng J Y, Zhang Y L, Hwang W S. Candid covariance-free incremental principal componet analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003, **25**(8): 1034−1040

4 Zhang Y L, Weng J Y. Convergence Analysis of Complementary Candid Incremental Principal Component Analysis, Technical Report MSU-CSE-01-23, Michigan State University, USA, 2001

5 Papadimitriou S, Sun J, Faloutsos C. Streaming pattern discovery in multiple time-series. In: Proceedings of the 31st International Conference on Very Large Data Bases. Trondheim, Norway: ACM, 2005. 697−708

6 Scholkopf B, Smola A, Muller K R. Nonliear component analysis as a kernel eigenvalue problem. *Neural Computation*, 1998, **10**(5): 1299−1319

7 Hyvavinen A, Oja E. Independent component analysis: algorithms and applications. *Neural Networks*, 2000, **13**(4-5): 411−430

8 Wu Feng. The Construction of Resilient Functions and the Research of Intelligentized IDS Methods [Master dissertation], National University of Defense Technology, China, 2005
(吴枫. 弹性函数构造与智能入侵检测方法研究 [硕士学位论文], 国防科学技术大学, 中国, 2005)

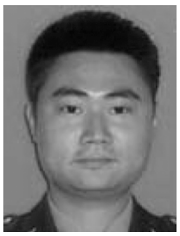
9 Li Ming, Zhou Zhi-Hua. Online semi-supervised learning with multi-kernel ensembe. *Journal of Computer Research and Development*, 2008, **45**(12): 2060−2068
(黎铭, 周志华. 基于多核集成的在线半监督学习方法. 计算机研究与发展, 2008, **45**(12): 2060−2068)

10 Blake C L, Merz C J. UCI repository of machine learning databases [Online], available: http://www.ics.uci.edu/~mlearn/MLRepository.html, November 7, 2007

11 Coenen F. LUCS-KDD DN software [Online], available: http://www.csc.liv.ac.uk/~frans/KDD/software/LUCS_K-DD_DN/, November 7, 2007

12 Keogh E, Xi X P, Li W, Ratanamahatana C. The UCR time series data mining archive [Online], available: http://www.cs.ucr.edu/~eamonn/TSDMA/index.html, November 7, 2007

13 Cao F, Ester M, Qian W, Zhou A. Density-based clustering over an evolving data stream with noise. In: Proceedings of the 6th International Conference on Data Mining. Bethesda, USA: SIAM, 2006. 326−377



吴 枫 国防科学技术大学计算机学院博士研究生. 主要研究方向为数据挖掘, 网络安全和人工智能. 本文通信作者.
E-mail: wfttyy_2000@163.com
(**WU Feng** Ph. D. candidate at the School of Computers, National University of Defense Technology. His research interest covers data mining, network security, and artificial intelligence. Corresponding author of this paper.)

仲 妍 国防科学技术大学计算机学院博士研究生. 主要研究方向为高性能计算和数据挖掘.
E-mail: mandy_zh82 @163.com
(**ZHONG Yan** Ph. D. candidate at the School of Computers, National University of Defense Technology. Her research interest covers high performance computing and data mining.)

吴泉源 国防科学技术大学计算机学院教授. 主要研究方向为分布计算和人工智能. E-mail: mandy_zh82@hotmail.com
(**WU Quan-Yuan** Professor at the School of Computers, National University of Defense Technology. His research interest covers distributed computing and artificial intelligence.)