基于多核字典学习的软件缺陷预测

王铁建[1] 吴 飞[2] 荆晓远[1]
(武汉大学计算机学院软件工程国家重点实验室 武汉 430072)[1]
(南京邮电大学自动化学院 南京 210023)[2]

摘 要 提出一种多核字典学习方法,用以对软件模块是否存在缺陷进行预测。用于软件缺陷预测的历史数据具有结构复杂、类不平衡的特点,用多个核函数构成的合成核将这些数据映射到一个高维特征空间,通过对多核字典基的选择,得到一个类别平衡的多核字典,用以对新的软件模块进行分类和预测,并判定其中是否存在缺陷。在NASA MDP数据集上的实验表明,与其他软件缺陷预测方法相比,多核字典学习方法能够针对软件缺陷历史数据结构复杂、类不平衡的特点,较好地解决软件缺陷预测问题。

关键词 软件缺陷预测,多核学习,字典学习,类不平衡
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.12.026

Multiple Kernel Dictionary Learning for Software Defect Prediction

WANG Tie-jian[1] WU Fei[2] JING Xiao-yuan[1]
(State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan 430072, China)[1]
(School of Automation, Nanjing University of Posts and Telecommunications, Nanjing 210023, China)[2]

Abstract A multiple kernel dictionary learning approach for software defect prediction was proposed. Software historical defect data have complicated structure and marked characteristic of class-imbalance. Multiple kernel learning is an effective technique in the field of machine learning which can map the historical defect data to a higher-dimensional feature space and make them express better. We got a multiple kernel dictionary learning classifier which has the advantages of both multiple kernel learning and dictionary learning. The widely used datasets from NASA MDP datasets are employed as test data to evaluate the performance of all compared methods. Experimental results demonstrate the effectiveness of the proposed multiple kernel dictionary learning approach for the software defect prediction task.

Keywords Software detect prediction, Multiple kernel learning, Dictionary learning, Class-imbalance

1 引言

软件缺陷预测是软件工程中的一个非常重要的研究课题。基于度量的静态软件缺陷预测技术借助从已有软件模块中获得的历史数据,对新的软件模块进行缺陷预测,以判断它们是否存在缺陷,从而为软件项目提供决策支持[1-3]。软件缺陷预测一般包括如下步骤:1)标记模块类别,软件模块可以被划分为有缺陷模块和无缺陷模块两个类别;2)提取模块属性,利用McCabe度量[4]、Halstead度量[5]等方法对软件模块进行度量,得到软件模块的属性;3)建立预测模型,根据软件模块的类别和属性信息,利用机器学习的方法,通过学习获得一个分类器;4)对新模块进行预测,利用该分类器,根据新软件模块的属性信息,对它的属性进行预测,从而判断该模块是否含有缺陷。

静态软件预测所使用的历史数据结构复杂,如果将常用的分类技术直接应用在这些缺陷数据上,分类效果往往不是很理想。此外,在软件缺陷预测中,有缺陷的软件模块的数目往往远少于无缺陷的软件模块的数目,这使得软件缺陷历史数据具有显著的"类不平衡"特点[6-7]。如何充分分析、利用已有的软件模块历史数据,构造出准确、高效的软件模块分类器,是研究者当前比较关心的问题。核方法[8]可以将这些历史数据映射到一个能代表原始数据分布的高维特征空间中,从而更好地发掘数据信息,提高分类器的分类性能。与单核相比,多核学习方法[9-11]能够将不同特性的核函数进行组合,获得多类核函数的优点,得到更优的映射性能,从而使数据在新的特征空间中得到更好的表达,提高预测精度。同时,多核学习还能够解决诸如样本规模庞大、多维数据不规则、样本分布不平坦、数据信息异构等实际应用问题。字典学习[12-13]在稀疏表示的理论基础上,通过对初始字典进行学习,得到了一个更加紧凑的字典,这使其拥有了更强的表示能力,从而减小

到稿日期:2016-10-11 返修日期:2016-12-22
王铁建(1982-),男,博士生,主要研究方向为模式识别、机器学习、软件工程;吴 飞(1989-),男,博士生,主要研究方向为模式识别、机器学习、软件工程;荆晓远(1971-),男,博士,教授,博士生导师,CCF会员,主要研究方向为模式识别、机器学习、软件工程。

了重构误差，能更好地对样本进行稀疏编码。

本文提出了一种利用多核字典学习进行软件缺陷预测的方法，目的是将多核学习与字典学习相结合。这样不仅能够将软件缺陷历史数据映射到一个能代表原始数据分布的高维特征空间中，更好地发掘其中的信息，而且还能够通过对多核字典基的选择，构造出一个类别平衡的多核字典，从而解决软件缺陷历史数据本身存在的“类不平衡”问题，提高软件缺陷的预测效果。

2 相关工作

许多传统的机器学习方法，比如支持向量机、贝叶斯、决策树、神经网络、代价敏感和字典学习等，已经被人们成功地应用于软件缺陷预测领域。Elish 等人[14]和 Turhan 等人[15]分别利用支持向量机和权重贝叶斯分类器对软件缺陷进行预测。Wang 等人[16]提出了一种秩相关决策树缺陷预测方法，即把秩相关系数加入到决策树节点选择中，以提高预测精度。

在软件缺陷预测中，无缺陷软件模块的数目远远多于有缺陷模块的数目，因此，在此过程中必须考虑数据集的“类不平衡”问题。为了解决这一问题，Sun 等人[17]提出了一种集成学习方法(Coding based Ensemble Learning，CEL)，通过特定的编码过程来解决类不平衡问题。Ma 等人[18]利用单核函数把软件缺陷数据映射到核空间中，然后使用核主成分分析进行软件缺陷预测，其做法是在核空间中构造一个非对称分类器，通过补偿“类不平衡”导致的回归模型偏移来提高软件缺陷预测的效果。

此外，在软件缺陷预测的过程中把有缺陷模块预测为无缺陷模块的风险成本要远远高于把无缺陷模块预测为有缺陷模块。为了降低有缺陷模块被误分的代价，Zheng[19]提出了代价敏感的神经网络增强算法，加大了对将缺陷样本预测为无缺陷样本的惩罚，以提高分类器预测缺陷样本的能力。Jing 等人[13]提出将代价敏感的字典学习方法应用于缺陷预测，改善了缺陷预测的效果。

3 多核字典学习

3.1 多核学习与稀疏表示

给定一个监督的机器学习问题$(x_i,y_i)\in X\times Y$，其中输入空间$X\subseteq\mathbb{R}^d$，输出空间$Y\in\{-1,+1\}$，可以通过一个非线性映射$(\Phi:X\to F,x\to\Phi(x))$把输入数据映射到一个新的特征空间$F=\{\Phi(x)\mid x\in X\}$中，然后利用新的数据表示方法考虑原来的学习问题。所谓“核方法”，就是用核函数$k(x,x')$来代替核空间中的内积$\langle\Phi(x),\Phi(x')\rangle$。常用的核函数有多项式核、高斯核等，其中，多项式核为$k(x,x')=(\langle x,x'\rangle+\theta)^p$，RBF 高斯核为$k(x,x')=\exp(-\gamma\|x-x'\|_2^2)$。将不同特性的核函数进行组合，获得多类核函数的优点，以得到更优的映射性能，即为“多核学习”。

与单核方法相比，多核学习不仅能够组合利用各基本核的特征映射能力，很好地解决核函数的选择问题，而且还能将不同特性的核函数进行组合，获得多类核函数的优点，得到更优的映射性能，从而使数据在新的特征空间中得到更好的表达，提高预测精度。多核学习方法不仅能够解决诸如样本规模庞大、多维数据不规则、样本分布不平坦、数据信息异构等实际问题，而且还能够作为一种方法来解释学习结果，使得应用问题得到更深入的理解。多核学习中的合成核是多个核函数的凸组合，可以表示为各个核函数的加权和：

$$K(x,x')=\sum_{m=1}^{M}\beta_m k_m(x,x') \tag{1}$$

其中，β_m 是合成核中各基本核的权系数，$\beta_m\geqslant 0$，$\sum_{m=1}^{M}\beta_m=1$。在多核框架下，将样本在特征空间中的表示问题转化为基本核与权系数的选择问题。多核学习的目标就是通过最优化方法求取合成核的参数，可以将其转化为如下最优化形式：

$$\min_{\theta\in\Delta}\min_{f\in H_{K(\theta)}}\frac{1}{2}\|f\|_{H_{K(\theta)}}^2+C\sum_{i=1}^{N}l(f(x_i),y_i) \tag{2}$$

其中，$\Delta=\{\theta\in\mathbb{R}_+^M\mid\theta^{\mathrm{T}}eM=1\}$，$K(\theta)(\cdot,\cdot)=\sum_{j=1}^{M}\theta_j k_j(\cdot,\cdot)$，$l(f(x_i),y_i)=\max(0,1-y_i f(x_i))$。用 e_M 表示 M 维的单位向量，上式可以转化为：

$$\min_{\theta\in\Delta}\max_{\alpha\in\Xi}\left\{\alpha^{\mathrm{T}}e_N-\frac{1}{2}(\alpha\circ y)^{\mathrm{T}}\left(\sum_{j=1}^{N}\theta_j,K^j\right)(\alpha\circ y)\right\} \tag{3}$$

其中，$K^j\in\mathbb{R}^{N\times N}$，$K^j_{p,q}=k_j(x_p,x_q)$，$\Xi=\{\alpha\mid\alpha\in[0,C]^N\}$，$\circ$为两个向量的内积。这是一个最优化问题，可以通过构建拉格朗日方程求得合成核中各基本核的权系数 β_m。

稀疏编码是字典学习的基础，它是用一组超完备基向量来表示样本。假设训练集为$A=[A_1,\cdots,A_i,\cdots,A_c]$，将这些样本的线性组合构成一个字典，根据线性子空间原理，测试样本 y 可稀疏表示为该类训练样本的线性组合：

$$\hat{\alpha}=\arg\min_{\alpha}\|y-A\alpha\|_2^2,\ \text{s.t.}\ \|\alpha\|_0\leqslant T \tag{4}$$

其中，$\hat{\alpha}$是测试样本 y 的稀疏表示系数，T 是稀疏阈值。测试样本 y 可以归类为系数最大的类别，即相关性最高的类别：

$$identity(y)=\arg\min_{i}\|y-A_i\alpha\|_2 \tag{5}$$

其中，A_i 是第 i 类样本的字典基。假设原始的训练样本集为$A=[A_1,\cdots,A_i,\cdots,A_c]\in\mathbb{R}^{m\times n}$，通过字典学习得到的字典为$D=[D_1,\cdots,D_i,\cdots,D_c]$，$X=[X_1,\cdots,X_i,\cdots,X_c]$是样本 A 在字典 D 上的稀疏编码，则样本 A 的稀疏表示为 $A\approx DX$。

3.2 多核字典学习

稀疏表示中的字典是由原始训练样本直接构成的，为了获得更加稀疏、精确的表示，可以对这些原始训练样本加以学习，以构造最优字典基，这就是字典学习方法。在软件缺陷预测中，假设原始的训练样本集为$Y=[Y_1,\cdots,Y_i,\cdots,Y_c]$，非线性映射 $\Phi(\cdot):R^m\to F$ 为多核映射，多核字典学习的目标就是在特征空间 F 中得到一个多核字典 $\Phi(D)=[\Phi(D_1),\cdots,\Phi(D_i),\cdots,\Phi(D_c)]$，使得：

$$J_{(\Phi(D),X)}=\arg\min_{(\Phi(D),X)}\{r(\Phi(Y),\Phi(D),X)+\lambda\|X\|_1\} \tag{6}$$

其中，$r(\Phi(Y),\Phi(D),X)$为重构误差，$\|X\|_1$ 是稀疏约束，λ 为平衡因子。假设 $X_i=[X_i^1,X_i^2,\cdots,X_i^c]$是 $\Phi(Y_i)$在多核字典 $\Phi(D)$上的子编码矩阵，为了能够同时保证字典的表示能力和鉴别能力，将 $r(\Phi(Y),\Phi(D),X)$写为：

$$r(\Phi(Y_i),\Phi(D),X_i)=\|\Phi(Y_i)-\Phi(D)X_i\|_F^2+\|\Phi(Y_i)-\Phi(D_i)X_i^i\|_F^2+\sum_{j=1,j\neq i}^{c}\|\Phi(D_j)X_i^j\|_F^2 \tag{7}$$

其中，第一项保证了多核字典的表示能力，使多核字典 $\Phi(D)$ 能够很好地表示 $\Phi(Y_i)$，即令 $\| \Phi(Y_i)-\Phi(D)X_i \|_F^2$ 最小化；第二项和第三项保证了多核字典的鉴别能力，$\Phi(Y_i)$ 由其相应的子字典 $\Phi(D_j)$ 来稀疏表示，而不是由其他子字典表示，即令 $\| \Phi(Y_i)-\Phi(D_i)X_i^i \|_F^2$ 和 $\| \Phi(D_j)X_i^j \|_F^2$ 最小化。

多核字典学习的过程就是利用迭代方法反复求解式(6)中的 $\Phi(D)$ 和 X 的过程[20-21]。首先将多核字典 $\Phi(D)$ 固定，求解稀疏编码 X；然后固定稀疏编码 X，优化多核字典 $\Phi(D)$；之后将多核字典 $\Phi(D)$ 固定，优化稀疏编码 X；接下来再将稀疏编码 X 固定，优化多核字典 $\Phi(D)$。经过反复迭代，就能够得到最优化的多核字典 $\Phi(D)$ 和稀疏编码 X_i。利用多核字典 $\Phi(D)$ 和稀疏编码 X_i，能够将测试样本稀疏表示出来；同时，根据稀疏表示时的重构误差，能够对测试样本进行分类。

完整的多核字典学习流程如算法 1 所示。

算法 1　多核字典学习方法

输入：训练集 $\{y_1, y_2, \cdots, y_N\}$，多核函数 $K(x,x')=\sum_{m=1}^{M}\beta_m k_m(x,x')$

步骤 1　核化

通过多核函数 K 进行非线性映射，把训练样本、测试样本映射到高维特征空间中；

步骤 2　初始化 Φ(D)

利用 PCA 方法选取数量相等的有缺陷样本和无缺陷样本，初始化多核字典 Φ(D)；

步骤 3　更新稀疏编码 X

固定 Φ(D)，求出稀疏编码 X；

步骤 4　更新多核字典 Φ(D)

固定 X，求出优化后的多核字典 Φ(D)；

步骤 5　输出 Φ(D) 与 X_i

重复步骤 3 和步骤 4，直到 $J_{(\Phi(D),X)}$ 收敛，输出 Φ(D) 和 X_i；

步骤 6　缺陷预测

利用优化后的多核字典 Φ(D) 与稀疏编码 X_i 进行软件缺陷预测。

4　实验

为验证多核字典学习方法的预测效果，我们进行了如下实验。实验使用的数据集是 NASA Metrics Data Program (NASA MDP)项目数据集，每个数据集代表美国宇航局的软件系统或子系统，包含每个组成模块的静态代码度量和相应的缺陷标记数据。NASA MDP 数据集的规模如表 1 所列。

表 1　NASA MDP 数据集

数据集	有缺陷模块数	无缺陷模块数	属性个数	有缺陷与无缺陷模块的比例
CM1	42	302	38	1∶7.19
JM1	1672	6170	22	1∶3.69
KC1	314	889	22	1∶2.83
KC3	36	158	40	1∶4.39
MC2	44	81	40	1∶1.84
MW1	27	228	38	1∶8.44
PC1	61	650	38	1∶10.66
PC3	134	945	38	1∶7.05
PC4	177	1111	38	1∶6.28
PC5	471	1252	39	1∶2.65

从有缺陷模块与无缺陷模块的比例可以看出，这些数据集具有显著的“类不平衡”特点，在 PC1 数据集中，有缺陷模块与无缺陷模块的数目比例达到了 1∶10.66。

预测模型的评估指标有召回率、误报率、查准率和精确度。如表 2 所列，假设被正确地预测为有缺陷(即实际为有缺陷且被分类器预测为有缺陷)的模块数为 *TP*(True Positives)，被错误地预测为有缺陷(即实际为无缺陷但被分类器预测为有缺陷)的模块数为 *FP*(False Positives)，被错误地预测为无缺陷(即实际为有缺陷但被分类器预测为无缺陷)的模块数为 *FN*(False Negatives)，被正确地预测为无缺陷(即实际为无缺陷且被分类器预测为无缺陷)的模块数为 *TN*(True Negatives)，则召回率、误报率、查准率、精确度的定义如下。

表 2　缺陷预测评估指标

	预测为有缺陷	预测为无缺陷
实际有缺陷	*TP*	*FN*
实际无缺陷	*FP*	*TN*

召回率(Probability of Detection，pd)：正确预测为有缺陷的模块数与真实有缺陷的模块数的比值，即 $pd=TP/(TP+FN)$。该比值对软件缺陷预测来说很重要，因为预测模型的目的是尽可能地找出有缺陷的模块。

误报率(Probability of False alarm，pf)：错误预测为有缺陷模块数与实际无缺陷模块数的比值，即 $pf=FP/(FP+TN)$。

查准率(Precision)：正确被预测为有缺陷的模块数与预测为有缺陷模块数的比值，即 $precision=TP/(TP+FP)$。该比值评估的是模型预测的正确程度。

精确度(Accuracy)：正确预测模块数与总模块数之比，即 $Accuracy=(TP+TN)/(TP+FN+FP+TN)$。该比值是所有数据挖掘分类器应用中广泛使用的指标。

F-measure 指标就是将召回率(Recall rate)与查准率(Precision)结合起来评价。F-measure 的计算公式为 $F\text{-}measure=2*pd*precision/(pd+precision)$。AUC 值(Area Under Curve)被定义为 ROC 曲线(Receiver Operating Characteristic)下的面积，使用 AUC 值可以评估二分类问题分类效果的优劣。AUC 值的计算公式为 $AUC=(pd-pf+1)/2$。F-measure 值和 AUC 值的取值都为 0～1，并且数值越大，表示软件缺陷预测模型的预测性能越好。

本文采用十折交叉验证(10-fold CV)方法进行实验。将每个 NASA MDP 数据集随机均分为 10 组，轮流将其中 9 组做训练，剩余的 1 组做预测，取 10 次实验度量值的平均值作为算法的运行结果进行比较。实验所使用的基本单核函数有 20 个，其中高斯核函数 13 个，参数范围为 $(2^{-5}, 2^{-5}, \cdots, 2^{6})$；多项式核 7 个，参数范围为 1～7。对于字典学习公式中的 λ 参数，由文献[22]可知，当 $0.000001 \leqslant \lambda \leqslant 0.1$ 时，会达到较好的预测效果，因此实验中 λ 取值为 0.01。对比方法是近年来提出的缺陷预测方法，包括支持向量机(Support Vector Machine，SVM)[14]、权重贝叶斯(Weighted Naïve Bayes，WNB)[15]、秩相关决策树(Compressed C4.5 decision tree，CC4.5)[16]、基于编码的集成学习(Coding based Ensemble Learning，CEL)[17]、非对称核主成分分析(Asymmetric Kernel

Principal Component Classification, AKPCC)[18] 和代价敏感的字典学习(Cost-sensitive Discriminative Dictionary Learning, CDDL)[13]。

表 3 列出以上几种软件缺陷预测方法在 NASA MDP 数据集上基于 F-measure 值和 AUC 值的实验结果。与其他软件缺陷预测算法相比,多核字典学习(Multiple Kernel Dictionary Learning, MKDL)在所有的 NASA MDP 数据集上都获得了最高的 F-measure 值。从平均值的角度分析,与字典学习方法相比,多核字典学习方法将缺陷预测的 F-measure 值最少提高了 12.86% ((0.79−0.69)/0.69);与支持向量机方法相比,多核字典学习方法将缺陷预测的 F-measure 值最多提高了 29.51% ((0.79−0.61)/0.61)。这说明多核字典学习方法取得了较好的预测效果。

表 3 基于 F-measure 值和 AUC 值的实验结果

Dataset		SVM	WNB	CC4.5	CEL	AKPCC	CDDL	MKDL
CM1	*F*	0.46	0.55	0.49	0.50	0.37	0.60	0.69
	A	0.71	0.69	0.70	0.72	0.76	0.74	0.83
JM1	*F*	0.54	0.56	0.58	0.61	0.59	0.67	0.78
	A	0.76	0.73	0.69	0.78	0.81	0.76	0.85
KC1	*F*	0.71	0.69	0.65	0.68	0.70	0.72	0.73
	A	0.75	0.78	0.75	0.85	0.75	0.81	0.87
KC3	*F*	0.62	0.75	0.67	0.62	0.53	0.70	0.88
	A	0.86	0.85	0.76	0.82	0.83	0.88	0.89
MC2	*F*	0.83	0.70	0.76	0.88	0.82	0.79	0.93
	A	0.91	0.91	0.90	0.94	0.86	0.91	0.96
MW1	*F*	0.60	0.59	0.55	0.52	0.59	0.66	0.70
	A	0.70	0.78	0.73	0.71	0.72	0.76	0.81
PC1	*F*	0.53	0.58	0.50	0.60	0.60	0.65	0.68
	A	0.80	0.79	0.79	0.83	0.78	0.85	0.88
PC3	*F*	0.55	0.57	0.59	0.68	0.63	0.71	0.76
	A	0.81	0.85	0.88	0.84	0.75	0.81	0.86
PC4	*F*	0.65	0.71	0.86	0.90	0.82	0.86	0.91
	A	0.82	0.84	0.89	0.92	0.89	0.89	0.96
PC5	*F*	0.60	0.68	0.83	0.71	0.69	0.62	0.85
	A	0.89	0.93	0.91	0.94	0.91	0.92	0.95
Average	*F*	0.61	0.64	0.65	0.67	0.63	0.70	0.79
	A	0.80	0.82	0.80	0.84	0.81	0.83	0.89

注:*F* 表示 F-measure 值,*A* 表示 AUC 值

从 AUC 值上也能够看到,与其他软件缺陷预测算法相比,多核字典学习(Multiple Kernel Dictionary Learning, MKDL)在所有的 NASA MDP 数据集上都获得了最高的 AUC 值。同样地,从平均值的角度分析,与基于编码的集成学习方法相比,多核字典学习方法将缺陷预测的 AUC 值最少提高了 5.95%((0.89−0.84)/0.84);与支持向量机方法和秩相关决策树方法相比,多核字典学习方法将缺陷预测的 AUC 值最多提高了 11.25%((0.89−0.80)/0.80)。这同样说明多核字典学习方法取得了较好的预测效果。

结束语 针对软件缺陷历史数据结构复杂、类不平衡的特点,提出了一种多核字典学习软件缺陷预测方法,将多核学习与字典学习相结合,把软件缺陷历史数据映射到一个能代表原始数据分布的高维特征空间中,并通过对多核字典基的选择,得到一个类别平衡的多核字典,其中有缺陷核子字典和无缺陷核子字典的字典基相同,从而解决了软件缺陷历史数据本身存在的"类不平衡"问题。在 NASA MDP 数据集上的对比实验表明,与其他几种有代表性的软件缺陷预测方法相比,多核字典学习方法获得了更高的 F-measure 值和 AUC 值,取得了较好的软件缺陷预测效果。这说明多核字典学习方法能够针对软件缺陷历史数据结构复杂、类不平衡的特点,较好地解决软件缺陷预测问题。

参 考 文 献

[1] LYU M R. Software Reliability Engineering: A Roadmap[C]// Future of Software Engineering. 2007: 153-170.

[2] LESSMANN S, BAESENS B, MUES C, et al. Benchmarking classification models for software defect prediction: A proposed framework and novel findings[J]. IEEE Transaction Software Engineering, 2008, 34(4): 485-496.

[3] NAM J, PANY S J, KIM S. Transfer Defect Learning[C]// International Confernece on Software Engineering. 2013: 382-391.

[4] MCCABE T J. A complexity measure[J]. IEEE Transactions Software Engineering, 1976, SE-2(4): 308-320.

[5] HALSTEAD M H. Elements of Software Science (Operating and programming systems series)[M]. New York: Elsevier North-Holland, 1977.

[6] KHOSHGOFTAAR M T, GAO K, SELIYA N. Attribute Selection and Imbalanced Data: Problems in Software Defect Prediction[C]// IEEE International Conference on Tools with Artificial Intelligence. 2010: 137-144,.

[7] GAO K, KHOSHGOFTAAR T M, NAPOLITANO A. A Hybrid Approach to Coping with High Dimensionality and Class Imbalance for Software Defect Prediction[C]// Machine Learning and Applications. 2012: 281-288.

[8] MULLER K R, MIKA S, RATSCH G, et al. An introduction to kernel based learning algorithms[J]. IEEE Transactions on Neural Networks, 2001, 12(2): 181-201.

[9] RAKOTOMAMONJY A, BACH F, CANU S. More efficiency in multiple kernel learning[C]// International Conference on Machine Learning. 2007: 775-782.

[10] GEHLER P V, NOWOZIN S. On Feature Combination for Multiclass Object Classification[C]// IEEE 12th International Conference on Computer Vision. 2009: 221-228.

[11] KEMBHAVI A, SIDDIQUIE B, MIEZIANKO R. Incremental Multiple Kernel Learning for Object Recognition[C]// IEEE 12th International Conference on Computer Vision. 2009: 638-645.

[12] YANG M, ZHANG L, FENG X, et al. Fisher Discrimination Dictionary Learning for sparse representation[C]// IEEE International Conference on Computer Vision. 2011: 543-550.

[13] JING X Y, YING S, ZHANG Z W, et al. Dictionary learning based software defect prediction[C]// International Conference on Software Engineering. 2014: 792-802.

[14] ELISH K, ELISH M. Predicting Defect-prone Software Modules Using Support Vector Machines[J]. Journal Systems and Software, 2008, 81(5): 649-660.

(下转第 168 页)

参 考 文 献

[1] SINGH S, CHANA I. Cloud resource provisioning: survey, status and future research directions[J]. Knowledge & Information Systems, 2016, 49(3): 1-65.

[2] Amazon Cloud Services [EB/OL]. http://aws.amazon.com.

[3] LI Q, ZHENG X. Research Survey of Cloud Computing[J]. Computer Science, 2011, 38(4): 32-37. (in Chinese)
李乔, 郑啸. 云计算研究现状综述[J]. 计算机科学, 2011, 38(4): 32-37.

[4] KONDO D, JAVADI B, MALECOT P, et al. Cost-benefit analysis of Cloud Computing versus desktop grids[C]// Proceedings of the 2009 IEEE International Symposium on Parallel and Distributed Processing (IPDPS 2009). Washington DC, 2009: 1-12.

[5] CHEN C L P, ZHANG C Y. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data[J]. Information Sciences, 2014, 275(11): 314-347.

[6] YANG X, WALLOM D, WADDINGTON S, et al. Cloud computing in e-Science: research challenges and opportunities[J]. Journal of Supercomputing, 2014, 70(1): 408-464.

[7] GUNDA P K, RAVINDRANATH L, THEKKATH C A, et al. Nectar: automatic management of data and computation in datacenters[C]// Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI 2010). Berkeley, 2010: 1-8.

[8] DEELMAN E, SINGH G, LIVNY M, et al. The cost of doing science on the cloud: The Montage example[C]// International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2008). Austin: IEEE, 2008: 1-12.

[9] ADAMS I F, LONG D D E, MILLER E L, et al. Maximizing efficiency by trading storage for computation[C]// Proceedings of the 2009 Conference on Hot topics in Cloud Computing (HotCloud'2009). Berkeley, 2009: 1-5.

[10] YUAN D, YANG Y, LIU X, et al. A cost-effective strategy for intermediate data storage in scientific cloud workflow systems [C]// 2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS). Atlanta, 2010: 1-12.

[11] YUAN D, YANG Y, LIU X, et al. A data dependency based strategy for intermediate data storage in scientific cloud workflow systems[J]. Concurrency and Computation: Practice & Experience, 2012, 24(9): 956-976.

[12] YUAN D, YANG Y, LIU X, et al. On-demand minimum cost benchmarking for intermediate dataset storage in scientific cloud workflow systems[J]. Journal of Parallel and Distributed Computing, 2011, 71(2): 316-332.

[13] YUAN D, YANG Y, LIU X, et al. A Local-Optimisation Based Strategy for Cost-Effective Datasets Storage of Scientific Applications in the Cloud[C]// Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing (Cloud 2011). Washington DC, 2011: 179-186.

[14] YUAN D, YANG Y, LIU X, et al. A Highly Practical Approach toward Achieving Minimum Data Sets Storage Cost in the Cloud [J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(6): 1234-1244.

[15] YUAN D, CUI L, LI W, et al. An Algorithm for Finding the Minimum Cost of Storing and Regenerating Datasets in Multiple Clouds[J]. IEEE Transactions on Cloud Computing, 2015(99): 1.

[16] YUAN D, LIU X, YANG Y. Dynamic On-the-Fly Minimum Cost Benchmarking for Storing Generated Scientific Datasets in the Cloud[J]. IEEE Transactions on Computers, 2015, 64(10): 2781-2795.

[17] YUAN D, YANG Y, LIU X, et al. Computation and Storage Trade-Off for Cost-Effectively Storing Scientific Datasets in the Cloud[M]// Handbook of Data Intensive Computing. Springer New York, 2011: 129-153.

(上接第 134 页)

[15] TURHAN B, BENER A. Analysis of naïve bayes' assumptions on software fault data: An empirical study[J]. Data Knowledge Engineering, 2009, 68(2): 278-290.

[16] WANG J, SHEN B J, CHEN Y T. Compressed C4.5 Models for Software Defect Prediction[C]// International Conference on Quality Software. 2012: 13-16.

[17] SUN Z B, SONG Q B, ZHU X Y. Using Coding Based Ensemble Learning to Improve Software Defect Prediction[J]. IEEE Transactions on Systems Man and Cybernetics Part C, 2012, 42(6): 1806-1817.

[18] MA G, LUO C, CHEN H. Kernel Based Asymmetric Learning for Software Defect Prediction[J]. IEICE Transactions on Information and Systems, 2012, E95-D(1): 215-226.

[19] ZHENG J. Cost-sensitive boosting neural networks for software defect prediction [J]. Expert Systems with Applications, 2010, 37(6): 4537-4543.

[20] ROSASCO L, VERRI A, SANTORO M, et al. Iterative projection methods for structured sparsity regularization[OL]. http://dspace.mit.edu/bitstream/handle/1721.1/49428/MIT-CSAIL-TR-2009-050.pdf?sequence=1.

[21] YANG M, ZHANG L, YANG J, et al. Metaface learning for sparse representation based face recognition[C]// International Conference on Image Processing. 2010: 1601-1604.

[22] ZHANG D, YANG M, FENG X. Sparse representation or collabortive representation: which helps face recognition? [C]// IEEE International Conference on Computer Vision. 2011: 471-478.