一种新的多任务朴素贝叶斯学习方法*

孙立健，周　鋆，张维明

(国防科技大学 信息系统工程重点实验室，长沙 410072)

通讯作者：周鋆，E-mail: zhouyun@nudt.edu.cn

摘要：本文针对现有多任务学习方法运行效率低和数据信息利用不足的问题，提出了一种新的多任务朴素贝叶斯学习方法。该方法基于朴素贝叶斯原理，将经典的单任务朴素贝叶斯模型引入到多任务学习问题中，通过两种新的更新策略，更好地利用总体数据的先验信息来提高模型的分类性能和泛化能力；该方法与目前的多任务学习模型相比，运行效率显著提升。实验结果表明，该方法优于只针对单任务的朴素贝叶斯学习方法，并在部分数据集上优于当前主流的多任务学习方法。

关键词：多任务学习；朴素贝叶斯；机器学习；数据集预处理

中图法分类号：TP391.4　　**文献标识码**：A　　**文章编号**：

A New Implementation of Multi-task Naïve Bayes Learning

SUN Lijian, ZHOU Yun, ZHANG Weiming

(Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410072, China)

Abstract: The multi-task learning models often exist the problems of low efficiency and insufficient use of shared data information, this paper proposed a new implementation of multi-task naïve Bayes learning method. Based on the naïve Bayesian principle, this method introduces the classical single-task naïve Bayesian learning model into multi-task learning problem. Through two new updating strategies, the prior information of the overall data is better utilized to improve the classification performance and generalization ability of the model. Compared with the current multi-task learning model, the method has significantly improved the operating efficiency of the current model. The experimental results show that this method is superior to the single-task naïve Bayes learning method and is superior to current state-of-the-art multi-task learning methods in some datasets.

Key words: multi-task learning; naïve Bayes; machine learning; data preprocessing

网络空间正在从单一的信息域发展到涵盖物理、信息、认知和社会四域的第五维空间，通过信息传递，影响、控制实体行为，形成虚实结合、人机一体、多域融合的复杂空间，网络空间的安全问题成为备受关注的热点问题，维护网络空间安全是国家的核心战略之一。网络空间中的很多安全问题，如攻击检测、行为识别等都可以看成机器学习中的分类问题[1] 。然而由于某些高级攻击（如 APT 攻击）较难发现，相关样本少，且不同分类任务（检测阶段）间难以建立有效的联系，因此机器学习界提出了多任务学习方法来解决这类问题。

多任务学习是同时学习多个相关任务的方法，它可以更加充分的利用数据之间的共享信息，从而获得更好的模型性能。现实世界中很多问题不能简单地分解为一个个独立的子问题，

* **收稿时间**：0000-00-00; **修回时间**：0000-00-00

基金项目：国家自然科学基金(61703416); 湖南省自然科学基金(2018JJ3614)

作者简介：孙立健(1995-)，男，硕士生，主要研究方向为情报学和多任务学习；周鋆（1987-），男，讲师，主要研究方向为机器学习、网络空间安全；张维明（19620），男，教授，主要研究方向为指挥信息系统、体系工程理论。

这些子问题通常是相互关联的，即通过一些共享因素或共享表示联系在一起。多任务学习(Multi-task Learning，MTL)方法同时对这些任务进行学习，通过提取和利用任务之间的共享信息，分类器中的参数更新相互影响，从而改善分类器的泛化性能[2] 。当任务中已知标记样本数较少时，多任务学习可以有效地增加样本数，从而使分类结果更加准确。

遗憾的是，在现有的多任务学习方法中，普遍存在模型运行效率低，对数据的先验知识利用不充分等问题。本文提出了一种新的多任务朴素贝叶斯学习方法，该方法可以显著提高多任务学习方法的效率。本文采用两种全新的更新共享信息策略对多任务学习中的参数更新，第一种是将不同任务上的参数逐渐向所有任务的总体参数逼近，即总体更新算法（Overall Iteration algorithm）；另一种是将不同任务上的参数逐渐向分类结果最优任务的参数逼近，即最优更新算法（Optimal Iteration algorithm）。通过两种更新策略，NB-MTL 模型可以更加充分相关任务和总体数据之间的先验信息，且利用较少的计算资源便可以达到更佳的分类效果。

1 相关工作

多任务学习方法表示同时学习多个相关任务，而不是独立地对各个任务进行单独学习[3] ,[4] 。人类在学习一个新任务时往往可以从相似任务中获得启发，多任务学习方法采用类人学习机制，从而使学习变得更为简单、容易、准确。同时，多任务学习方法充分考虑和利用了任务间的共享信息，因此模型的泛化性能和鲁棒性得到有效提升。

近年来，多任务学习已成为机器学习领域的研究热点之一。2013 年，Oyen 等人通过迁移学习来利用相关任务之间的数据[5] ，从而有效提高了学习贝叶斯网络结构的鲁棒性。2017 年，Zhou 等人将多任务学习运用到概率图模型上[6] ，有效利用了有向无环图的共享隐藏结构。2018 年，Zhang 等人提出 L2MT 模型[7] ，为多任务问题构建多任务模型，从而提高对历史多任务模型的利用程度。同时，多任务学习方法在网络空间等多个领域发挥了越来越大的作用。2019 年，Zhou 等人[8] 通过在 XSS 攻击检测中运用贝叶斯网络和多任务学习方法，有效的提高了 XSS 攻击威胁的识别率。

一些早期的多任务学习方法假设不同任务的目标函数参数是相似的[9] 或多个相关任务共享同一特征子集[10] ，这些多任务方法均通过正则项约束使相关任务之间的不同尽可能小。目前，多任务学习主要基于稀疏表示。2008 年，Argyriou 等人提出 MTL-FEAT 模型[11] ，该模型通过学习多任务之间的稀疏表示来共享信息。2011 年，Kang 等人对 MTL-FEAT 模型的约束进行松弛，提出了 DG-MTL 模型[12] ，该模型通过将多任务学习问题转化为混合整数规划问题，显著提高了多任务学习模型的性能。

在 MTL-FEAT 和 DG-MTL 模型的基础上，2012 年 Abhishek 等人提出 GO-MTL 模型[13] 。GO-MTL 模型采用一种新的多任务学习分组和重叠组结构，每个任务组的参数位于一个低维子空间中，不同分组的任务通过共享一个或多个潜在的基任务来共享信息。2018 年，Jeong 等人在上述模型的基础上进行了改进，提出 VSTG-MTL 模型[14] ，该模型在学习任务间重叠组结构的同时引入变量选择，模型将系数矩阵分解成两个低秩矩阵的乘积，从而更加合理地利用多个相关任务之间的共享信息。

2017 年，一种区别于传统稀疏表示策略的自步多任务学习 spMTFL 模型被提出[15] ，Murugesan 等人采用一种类人学习策略，将自定步长的任务选择方法引入到多任务学习中，模型通过迭代选择最合适的任务来学习任务参数和更新共享信息。

本文通过采用一种全新的更新共享信息策略，提出基于贝叶斯原理的多任务朴素贝叶斯学习模型(Multi-task Naïve Bayes Learning，NB-MTL)。NB-MTL 模型首先基于单任务朴素贝叶斯学习模型(Single-task Naïve Bayes Learning，NB-STL)初始化，然后通过均值迭代算法（Average

Iteration algorithm）或最优值迭代算法（Optimal Iteration algorithm）进行参数更新，充分相关任务和总体数据之间的先验信息，直至算法收敛。本文创新地基于贝叶斯原理进行多任务学习，并在此基础上，提出了多任务朴素贝叶斯学习算法，基于两种朴素贝叶斯的算法更新策略，多个任务之间可以共享参数信息。针对模型特点，本文提出一种简洁但灵活的学习率增长方式，有效的契合了模型的更新特性；由于朴素贝叶斯计算效率高，本方法较传统多任务学习方法在计算效率上有较大优势。

2 单任务朴素贝叶斯学习模型

朴素贝叶斯是一种特殊的贝叶斯网络，该方法主要用于分类[16] 。贝叶斯分类器是基于贝叶斯原理的统计学方法，该方法通过利用先验信息和样本数据来确定事件的后验概率，从而完成对新数据的分类任务。

相比于一般贝叶斯网络，朴素贝叶斯拥有着极强的独立性假设。令 y 表示类标签，它包含在一个可数集 $\{C_1,C_2,\ldots,C_K\}$ 中， K 表示类别的个数， $u_1,u_2,\ldots,u_D$ 表示每一个类标签所对应的随机变量。朴素贝叶斯模型假设各个变量之间是相互独立的，即 $u_1,u_2,\ldots,u_D$ 相互独立。此时，随机变量的联合概率分布为：

$$P(C_k \mid u_1,u_2,\ldots,u_D) = P(C_k)\cdot P(u_1,u_2,\ldots,u_D \mid C_k) = P(C_k)\prod_{d=1}^{D} P(u_d \mid C_k) \qquad 2.1$$

当给定类标签时，所有特征变量的似然即可计算得到。因此，在测试集上，每一个类标签的后验概率可以表示为类标签的先验和所有变量的似然的乘积，即

$$P(C_k \mid u_1,u_2,\ldots,u_D) = P(C_k)\cdot P(u_1,u_2,\ldots,u_D \mid C_k) / P(u_1,u_2,\ldots,u_D) \qquad 2.2$$

根据最大后验准则，新的测试实例从各个标签中选取后验概率最大的类标签作为它的类标签。由于给定测试实例时，对于所有的类别，$P(u_1,u_2,\ldots,u_D)$ 均为常数，故在计算过程中通常省略计算。新实例的类标签 $\hat{y}$ 最终的计算方式为：

$$\hat{y} = \arg\max_{C_k}\left(P(C_k)\prod_{d=1}^{D} P(u_d \mid C_k)\right) \qquad 2.3$$

朴素贝叶斯分类器的训练过程就是估计类标签的先验概率 $P(C_k)$ 和每个特征变量的似然 $P(u_d \mid C_k)$ 的过程。在进行概率估计时，令 λ 表示训练集中实例的数量，λ_k 表示训练集中第 k 类样本的数量，由于样本呈独立同分布，故类标签 C_k 的先验概率为 $P(C_k)=\lambda_k/\lambda$ 。对于离散属性，特征变量 u_d 在第 k 类样本中的取值分别为 $\mu_{k,d}=\{\mu_{k,1},\mu_{k,2},\ldots,\mu_{k,D}\}$， 似然 $P(u_d \mid C_k)$ 可估计为 $P(u_d \mid C_k)=\mu_{k,d}/\left(\sum_{d=1}^{D}\mu_{k,d}\right)$。为避免其它特征变量所携带的信息被训练集中未出现的特征变量所“抹去”，在估计概率值时采用拉普拉斯修正上式转变为 $P(C_k)=(\lambda_k+1)/(\lambda+K)$， $P(u_d \mid C_k)=(\mu_{k,d}+1)/(\sum_{d=1}^{D}\mu_{k,d}+\lambda_k)$。

在多任务学习条件下，简单的单任务朴素贝叶斯学习模型没有考虑任务之间的相关性，它通过朴素贝叶斯模型对不同任务进行逐个学习，得到每一个任务上所有实例的分类结果。在本

文中记该模型为单任务朴素贝叶斯学习模型，即 NB-STL 模型。

3 多任务朴素贝叶斯学习模型

3.1 多任务学习环境

假定有 D 个变量和 $\mathcal{T}$ 个任务的监督学习任务，其中第 t 个任务的输入矩阵为 $\mathbf{X}_t=\left[\left(\mathbf{x}_{t,1}\right)^T,\left(\mathbf{x}_{t,2}\right)^T,\ldots,\left(\mathbf{x}_{t,N_t}\right)^T\right]^T\in\mathbb{R}^{N_t\times D},\mathbf{x}_{t,n}=[u_{t,n,1},u_{t,n,2},\ldots,u_{t,n,D}]\in\mathbb{R}^D$，输出向量为 $\mathbf{y}_t=\left[y_{t,1},y_{t,2},\ldots,y_{t,N_t}\right]^T\in\mathbb{R}^{N_t},y_{t,n}\in\{C_1,C_2,\ldots,C_K\}$。朴素贝叶斯模型可以用来描述输入和输出之间的关系，即

$$\begin{aligned}\hat{\mathbf{y}}_t^*&=\arg\max_{\mathbf{y}_t}\left(P\left(\mathbf{y}_t\mid\mathbf{X}_t\right)\right)\\&=\arg\max_{\mathbf{y}_t}\left(P\left(\mathbf{y}_t\mid\mathbf{u}_{t,1},\mathbf{u}_{t,2},\ldots,\mathbf{u}_{t,D}\right)\right)\end{aligned}\tag{3.1}$$

对于第 t 个任务中的第 n 个实例，它的最大后验概率为

$$\hat{y}_{t,n}=\arg\max_{C_k}\left(P_t\left(C_k\right)\cdot\prod_{d=1}^{D}P_t\left(u_d\mid C_k\right)\right)=\arg\max_{C_k}\left(P_t\left(C_k\right)\cdot\prod_{d=1}^{D}P_t\left(\mu_{n,k,d}\mid C_k\right)\right)\tag{3.2}$$

在第 t 个任务中，$P_t\left(C_k\right)$是类标签的先验概率，$P_t\left(u_d\mid C_k\right)$是给定类标签 C_k 的特征变量 u_d 的似然，$\mu_{k,d}$ 为 u_d 的取值。由于 $P_t\left(u_1,u_2,\ldots,u_D\right)$ 是不依赖于 $P_t\left(C_k\right)$ 的常量，在省略计算 $P_t\left(u_1,u_2,\ldots,u_D\right)$ 的情况下后验概率 $P\left(C_k\mid u_1,u_2,\ldots,u_D\right)$ 可以描述为 $P_t\left(C_k\right)$ 和 $P_t\left(u_d\mid C_k\right)$ 的乘积。为了防止连乘操作造成浮点下溢，通常采用对数似然的方式进行计算，使乘法转换为加法。此时，计算公式为

$$\log\left(P_t\left(C_k\mid u_1,u_2,\ldots,u_D\right)\right)=\log\left(P_t\left(C_k\right)\right)+\sum_{d=1}^{D}P_t\left(u_d\mid C_k\right)\tag{3.3}$$

3.2 NB-MTL算法

单任务朴素贝叶斯学习模型在参数学习时没有考虑任务之间的相关性，故本文提出一种新的多任务朴素贝叶斯学习方法来提高模型的分类效果和泛化性能。算法总体流程如下，NB-MTL 算法首先基于 NB-STL 算法对各个任务上的参数进行初始化，学习得到第 $m=0$ 迭代时的先验 $P_t^0\left(C_k\right)$ 和似然 $P_t\left(u_d\mid C_k\right)$。初始化后，对所有任务采用总体迭代算法或最优迭代算法进行参数更新，直到收敛，如 3.2.1~3.2.3 节所述。此时，迭代输出为每个任务的最佳类标签先验 $P_t^*\left(C_k\right)$。最后，基于 $P_t^*\left(C_k\right)$ 和 $P_t\left(u_d\mid C_k\right)$ 完成对各个任务测试数据集的分类。接下来，本节对总体迭代算法、最优迭代算法和收敛条件展开详细介绍。

3.2.1 NB-MTL(Overall Iteration)

在 NB-MTL 总体迭代算法中，模型首先基于输入 $\mathbf{X}_t$ 和 $\mathbf{y}_t$ 训练得到 $m=0$ 时的每一个任务的先验 $P_t^0\left(C_k\right)$ 和似然 $P_t\left(u_d\mid C_k\right)$，它们被用来预测输入 $\mathbf{X}_t$ 的类标签 $\hat{\mathbf{y}}_t$。令 $\hat{\mathbf{y}}_t^*=f\left(\mathbf{X}_t,\ P_t^0\left(C_k\right),P_t\left(u_d\mid C_k\right)\right)$，函数 $f\left(\mathbf{X},P\left(C_k\right),P\left(u_d\mid C_k\right)\right)$ 表示基于类标签的先验

$P(C_k)$和似然$P(u_d \mid C_k)$预测数据集$\mathbf{X}$的类标签$\hat{\mathbf{y}}$。接下来，令共享任务$\mathcal{T}'$的数据集为$\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}^m_{\mathcal{T}'}\right)=\left([\mathbf{X}_1;\mathbf{X}_2;\ldots;\mathbf{X}_{\mathcal{T}}], [\hat{\mathbf{y}}^m_1;\hat{\mathbf{y}}^m_2;\ldots;\hat{\mathbf{y}}^m_{\mathcal{T}}]\right)$，基于数据集$\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}^m_{\mathcal{T}'}\right)$计算共享任务$\mathcal{T}'$的先验$P^m_{\mathcal{T}'}(C_k)$，即所有任务的总体先验。各个任务第$m$次迭代的先验$P^m_t(C_k)$以$\alpha$为学习率增加共享任务$\mathcal{T}'$的先验信息，即$P^{m+1}_t(C_k)=\alpha\cdot P^m_{\mathcal{T}'}(C_k)+(1-\alpha)\cdot P^m_t(C_k)$，其中$P^m_t(C_k)$表示第$m$次迭代过程中第$t$个任务上第$k$个类标签的先验。如果第$m$次迭代过程中基于$P^m_t(C_k)$和$P_t(u_d \mid C_k)$的分类结果总评分$\xi\left(\left[\hat{\mathbf{y}}^m_1, \hat{\mathbf{y}}^m_2, \ldots, \hat{\mathbf{y}}^m_{\mathcal{T}}\right]\right)$大于$\xi\left(\left[\hat{\mathbf{y}}^*_1, \hat{\mathbf{y}}^*_2, \ldots, \hat{\mathbf{y}}^*_{\mathcal{T}}\right]\right)$，则将最佳先验$P^*_t(C_k)$和最佳分类结果总评分$\xi\left(\left[\hat{\mathbf{y}}^*_1, \hat{\mathbf{y}}^*_2, \ldots, \hat{\mathbf{y}}^*_{\mathcal{T}}\right]\right)$更新为$P^m_t(C_k)$和$\xi\left(\left[\hat{\mathbf{y}}^m_1, \hat{\mathbf{y}}^m_2, \ldots, \hat{\mathbf{y}}^m_{\mathcal{T}}\right]\right)$，完成后进入下一次迭代。在具体的计算过程中，分类结果总评分ξ可采用准确率或 f1 值进行计算。总体迭代算法如表 1 所示。其中函数$g(\mathbf{X},\mathbf{y})$表示基于数据集$\mathbf{X}$和类标签$\mathbf{y}$计算得到类标签的先验$P(C_k)$。

表 1 总体迭代算法

Table 1 Overall Iteration algorithm

输入： $\mathbf{X}_t$，$\mathbf{y}_t$：任务$t=1,2,\ldots,\mathcal{T}$的训练集

ε：收敛阈值

α：学习率

输出： $P^*_t(C_k)$：最佳类标签先验

1. 基于 NB-STL 初始化，基于$\mathbf{X}_t, \mathbf{y}_t$得到$m=0$时的先验$P^0_t(C_k)$和似然$P_t(u_d \mid C_k)$，即$\left[P^0_t(C_k), P_t(u_d \mid C_k)\right]=g(\mathbf{X}_t, \mathbf{y}_t)$
2. $P^*_t(C_k) \leftarrow P^0_t(C_k)$
3. $\hat{\mathbf{y}}^*_t = f\left(\mathbf{X}_t, P^0_t(C_k), P_t(u_d \mid C_k)\right)$
4. repeat
5. $\quad \hat{\mathbf{y}}^m_t = f\left(\mathbf{X}_t, P^m_t(C_k), P_t(u_d \mid C_k)\right)$，$t=1,2,\ldots,\mathcal{T}$
6. $\quad \left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}^m_{\mathcal{T}'}\right)=\left([\mathbf{X}_1;\mathbf{X}_2;\ldots;\mathbf{X}_{\mathcal{T}}], [\hat{\mathbf{y}}^m_1;\hat{\mathbf{y}}^m_2;\ldots;\hat{\mathbf{y}}^m_{\mathcal{T}}]\right)$
7. $\quad P^m_{\mathcal{T}'}(C_k) = g\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}^m_{\mathcal{T}'}\right)$
8. $\quad P^{m+1}_t(C_k)=\alpha\cdot P^m_{\mathcal{T}'}(C_k)+(1-\alpha)\cdot P^m_t(C_k)$

9. $\hat{\mathbf{y}}_t^{m+1} = f\left(\mathbf{X}_t, P_t^{m+1}\left(C_k\right), P_t\left(u_d \mid C_k\right)\right)$， $t = 1,2,\ldots,\mathcal{T}$

10. if $\xi\left(\left[\hat{\mathbf{y}}_1^{m+1}, \hat{\mathbf{y}}_2^{m+1}, \ldots, \hat{\mathbf{y}}_{\mathcal{T}}^{m+1}\right]\right) > \xi\left(\left[\hat{\mathbf{y}}_1^{*}, \hat{\mathbf{y}}_2^{*}, \ldots, \hat{\mathbf{y}}_{\mathcal{T}}^{*}\right]\right)$

11. $P_t^{*}\left(C_k\right) = P_t^{m+1}\left(C_k\right)$

12. endif

13. $m = m+1$

14. $\Delta = \sum_{t=1}^{\mathcal{T}} \sum_{k=1}^{K} \left| P_t^{m+1}\left(C_k\right) - P_t^{m}\left(C_k\right) \right|$

15. until $\Delta < \varepsilon$

3.2.2 NB-MTL(Optimal Iteration)

NB-MTL 最优迭代算法与总体迭代算法的区别是它不再从所有任务的总体水平中获取平均意义上的共享先验信息，而是从分类效果最好的任务中获取共享信息。NB-MTL 最优迭代算法首先采用与总体迭代算法相一致的方法获取验证集 $\mathbf{X}_t$ 上每一个任务的类标签 $\hat{\mathbf{y}}_t$ 并计算各个任务的分类结果评分 $\xi\left(\hat{\mathbf{y}}_t^m\right)$。此时，NB-MTL 最优迭代算法将 $\xi\left(\hat{\mathbf{y}}_t^m\right)$ 最大的任务 t 作为任务 $\mathcal{T}'$。当存在多个最大评分值时，随机从该任务集中选择任务 $\mathcal{T}'$。令任务 $\mathcal{T}'$ 的数据集为 $\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}_{\mathcal{T}'}^m\right) = \left(\mathbf{X}_t, \hat{\mathbf{y}}_t^m\right)$ 并基于 $\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}_{\mathcal{T}'}^m\right)$ 计算最优任务 $\mathcal{T}'$ 的先验 $P_{\mathcal{T}'}^m\left(C_k\right)$，接下来的步骤同总体迭代算法得到最佳先验 $P_t^{*}\left(C_k\right)$。最优迭代算法如表 2 所示。

表 2 最优迭代算法

Table 2 Optimal Iteration algorithm

输入： $\mathbf{X}_t$， $\mathbf{y}_t$：任务 $t = 1,2,\ldots,\mathcal{T}$ 的训练集

ε：收敛阈值

α：学习率

输出： $P_t^{*}\left(C_k\right)$：最佳类标签先验

1~4：同表 1 的 1-4 行

5. $\hat{\mathbf{y}}_t^m = f\left(\mathbf{X}_t, P_t^m\left(C_k\right), P_t\left(u_d \mid C_k\right)\right)$， $t = 1,2,\ldots,\mathcal{T}$

6. $t' = \arg\max_t \left(\xi\left(\hat{\mathbf{y}}_t^m\right)\right)$, $\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}_{\mathcal{T}'}^m\right) = \left(\mathbf{X}_{t'}, \hat{\mathbf{y}}_{t'}^m\right)$

7. $P_{\mathcal{T}'}^m\left(C_k\right) = g\left(\mathbf{X}_{\mathcal{T}'}, \hat{\mathbf{y}}_{\mathcal{T}'}^m\right)$

8~15：同表 1 的 8-15 行

最优迭代算法与总体迭代算法的区别主要在表 1 和表 2 的第 6 行。

3.2.3 收敛判定

从 NB-MTL 算法的更新策略中可以看出，模型参数更新的过程实质上也是各个任务上先验 $P_{\mathcal{T}'}^{m}\left(C_k\right)$ 的比重不断累积的过程。随着迭代次数的增多，各个任务的先验 $P_t^m\left(C_k\right)$ 将趋于一致。基于上述分析，本文给定超参数 ε 作为算法的收敛阈值，在迭代过程中，计算不同任务上先后两次迭代的各个类标签的先验之差 Δ 的绝对值之和，即 $\Delta=\sum_{t=1}^{\mathcal{T}}\sum_{k=1}^{K}\left|P_t^m\left(C_k\right)-P_t^{m-1}\left(C_k\right)\right|$。当 $\Delta<\varepsilon$ 时算法收敛，获得最佳先验 $P_t^*\left(C_k\right)$，此时算法收敛，对应到表 1 的第 14 和 15 行。

最后，基于最佳先验 $P_t^*\left(C_k\right)$ 和似然 $P_t\left(u_d\mid C_k\right)$，根据公式 3.1 和 3.2 即可计算得到测试集上的最终分类标签。

3.3 理论分析

3.2 节对 NB-MTL 算法进行了详述，具体说明了 NB-MTL 算法的实现细节。本节将从理论分析的角度说明 NB-MTL 算法优于传统的 NB-STL 算法。

从 NB-MTL 算法的实现过程可看出，NB-MTL 算法首先基于 NB-STL 算法进行初始化，从而获得与 NB-STL 算法相一致的任务先验和似然，从而保证在 NB-MTL 算法在没有更新时，NB-MTL 算法和 NB-STL 算法具有相同的性能。

开始更新后，NB-MTL 算法采用从总体样本或分类效果最好的任务样本中获取它们的先验信息，从而使得各个任务获得了先验补充信息。NB-MTL 算法依据模型分类效果判断该信息的有效性。当通过增加该共享信息使得模型的分类效果提升时，接受该信息，反之则拒绝。通过上述机制，可以保证 NB-MTL 算法在进行更新时总是接受正信息，拒绝负信息。通过正信息的不断累加，NB-MTL 算法可以逐步在 NB-STL 算法的基础上提升模型的分类效果，从而保证了 NB-MTL 算法的优越性。

4 实验结果与分析

4.1 实验设置

为验证 NB-MTL 算法的有效性，本文采用 F1 值对各个算法的分类效果进行评估，即模型分类结果评分 ξ 由 F1 值表示。NB-MTL 算法的超参数 ε 设置为 10^{-5}，α 设置为 0.3。实验在处理器为 Inter Core i5-4200M，运行内存为 6GB 的 64 位电脑上运行。NB-MTL 的两种迭代算法 NB-MTL(Overall Iteration)，NB-MTL(Optimal Iteration)）与下述三种方法进行对比。

(1) NB-STL

该算法参考本文第二节中的单任务朴素贝叶斯学习模型。为了防止连乘操作造成浮点下溢，后验概率计算方式与 NB-MTL 一致，采用对数似然的方式，即公式 3.3。

(2) VSTG-MTL

VSTG-MTL 将模型中的权重矩阵分解成两个低秩矩阵的乘积，这些矩阵同时进行任务之间的变量选择和学习任务间的重叠组结构，进而提高模型的泛化性能。模型中的参数参考文献[14] 中进行设置.

(3) spMTFL

spMTFL 从不同的任务中由易到难的进行选择，学习任务参数，更新共享知识，从而优化新的双凸损失函数。模型中的参数参考文献[15] 中进行设置。

4.2 合成数据集及结果

本实验在 5 个 UCI 公开数据集上进行，数据集分别为 colic(368)，credit-a(690)，ionosphere(351)，kr-vs-kp(3196)，mushroom(8124)，sick(3772)，spambase(4601)，其中括号内的数值为每个数据集的数据量大小。首先，基于 Weka 对上述数据集进行预处理，处理缺失值并对数据进行离散化和二元化。处理完成后，对单任务数据集进行不放回随机采样。最后对每组数据随机去除 50%的特征信息以制造多任务条件：考虑 VSTG-MTL 和 spMTFL 的模型运算机制，保留特征属性，特征数据采用该特征数据的均值进行替换；NB-STL 和 NB-MTL 用于处理离散数据，因此直接剔除对应特征数据。

完成数据预处理后，采用 5 折交叉验证法对不同模型进行测试并计算不同任务上最终分类结果的总 F1 值，实验结果如表 3 所示。

表 3 不同模型在合成数据集上的实验结果

Table 3 Experimental Results of Different Models on Synthetic Datasets

F1 值	VSTG-MTL	spMTFL	NB-STL	NB-MTL (Optimal)	NB-MTL (Average)
colic	0.8156	0.80902	0.7905	0.8201	**0.8274**
credit	0.6943	0.72532	0.7245	**0.7560**	0.7295
ionosphere	0.9023	0.89992	0.9003	**0.9068**	0.9008
kr-vs-kp	0.7538	**0.77700**	0.7053	0.7441	0.7409
mushroom	0.8971	**0.96982**	0.9173	0.9178	0.9175
sick	**0.9830**	0.97452	0.9702	0.9703	0.9705
spambase	0.7943	**0.8658**	0.8274	0.8300	0.8280
Ave.	0.8343	**0.8602**	0.8336	0.8493	0.8449

表 4 不同模型在合成数据集上的运行时间

Table 4 Running Time of Different Models on Synthetic Datasets

Time(s)	VSTG-MTL	spMTFL	NB-STL	NB-MTL (Optimal)	NB-MTL (Average)
colic	13.4880	45.2250	**0.2190**	13.7350	12.0850
credit	7.5310	41.5830	**0.2650**	12.4760	11.4630
ionosphere	23.2760	61.1700	**0.1250**	11.9380	12.0930
kr-vs-kp	17.8560	41.1670	**0.2340**	16.1520	16.4460
mushroom	100.6760	99.9720	**0.3130**	22.9940	21.5480
sick	18.0980	37.4340	**0.2190**	17.3880	17.5100
spambase	53.0560	83.5450	**0.4220**	17.5870	17.3910
Ave.	33.4259	58.5851	**0.2567**	16.0386	15.5051

从表 3 可以看出，NB-MTL 模型在小数据集 colic、credit、ionosphere 上取得了最佳的模型效果，体现了当数据规模有限时，贝叶斯模型可以更加有效地弥补数据信息的不足，从而取得更好的分类效果。同时，NB-MTL(Optimal)模型在 colic、credit、ionosphere、mushroom、spambase 数据集上均优于 VSTG-MTL 模型，证明 NB-MTL(Optimal)模型较目前主流的基于稀疏表示的多任务学习算法具有良好的性能。除 colic 和 sick 数据集外，NB-MTL(Optimal)模型优于 NB-MTL(Overall)模型。从均值中可以看出，NB-MTL 模型均比 VSTG-MTL 模型和 NB-STL 模型提高了约 1 个百分点，且运行时间仅为 VSTG-MTL 模型的 1/2。同时，NB-MTL 模型在 ionosphere 和 sick 数据集上取得了与 spMTFL 模型基本相当的效果，但运行时间比 spMTFL 模型缩短了

约 1/3。另一方面，理论分析和实验结果均表明 NB-MTL 模型的分类结果均优于 NB-STL 模型，从而证明该 NB-MTL 模型具有更好的泛化性能。从表 4 可以进一步看出，在模型运行时间上，NB-MTL 模型的性能显著优于 VSTG-MTL 模型和 spMTFL 模型。综上可以得到，NB-MTL 模型在计算效率和分类效果上优于当前主流的多任务学习模型。

4.3 真实数据集及结果

为了进一步验证所提方法的可行性，本文采用多任务学习经典 School exam 数据集生成的分类数据集进行实验。School exam 数据集是由伦敦教育当局内部获得的回归数据集，它包括 1985-1987 年三年期间伦敦 139 所中学的 15362 名学生的考试成绩。该数据集有 139 个任务和 15362 个观察值，每个任务和观察值对应一个学校和一个学生的考试测验。每个观察值由 3 个连续变量和 23 个二元变量表示，分别为学校和学生专业属性等。本实验对 School exam 数据集进行预处理，并对样本类标签进行划分。当学生成绩大于 20 时为正样本，有 6984 条，占比 45.46%；当成绩小于等于 20 时为负样本，有 8378 条，占比 54.54%，正负样本之比约为 1:1。

本实验对原始 School exam 数据集进行预处理后，采用 5 折交叉验证法对不同模型进行测试并计算不同任务上最终分类结果的总 F1 值，实验结果如表 5 所示。

表 5 不同模型在真实数据集上的实验结果和运行时间

Table 5 Experimental Results and Running Time of Different Models on Real Dataset

School Dataset	VSTG-MTL	spMTFL	NB-STL	NB-MTL (Optimal)	NB-MTL (Average)
F1 值	0.6743	0.6701	0.6611	**0.6781**	0.6611
Time(s)	308.1290	173.0650	**0.7500**	56.5740	54.9000

从表 5 可以看出，NB-MTL(Optimal)模型达到了最佳的分类效果，优于其它模型。此外，NB-MTL(Optimal)模型虽然 F1 值仅高于 VSTG-MTL 模型 0.0038 和 spMTFL 模型 0.0080，但在实验过程中，NB-MTL(Optimal)模型的计算时间仅为 VSTG-MTL 模型的 1/5 和 spMTFL 模型的 1/3，时间开销大幅下降。实验结果表明，NB-MTL 用于提高数据的分类效果和模型的泛化性能是可行的。

5 结束语

本文首先分析现有多任务学习模型的不足，针对这些不足提出了一种新的多任务学习模型 NB-MTL。在第 3 节，本文对 NB-MTL 模型进行详细阐述，说明了 NB-MTL 模型两种的更新策略。最后，本文分别从理论分析和实验说明的角度论证了 NB-MTL 模型能够利用任务之间的相关性提高模型性能，同时具有较好的泛化性能。本文目前采用的朴素贝叶斯模型拥有较强的独立性假设，在后续工作中，可以松弛该假设，引入特征间的相关性，并对模型性能进行进一步的探索。

参考文献：

[1] 李焕,杜镇宇,张亮.网络安全威胁情报 TI 标准分析研究[J].信息工程大学学报,2018,19(6):762-768. DOI:10.3969/j.issn.1671-0673.2018.06.022.

[2] 马建阳,张宝鹏.基于多任务学习的多源数据分类研究[J].计算机应用研究,2018,35(11):3228-3231.

[3] RICH C. Multitask learning[J]. Machine Learning. 1997,28(7):41–75.

[4] ZHANG Y, YANG Q. A survey on multi-task learning[EB/OL]. https://arxiv.org/pdf/1707.08114.pdf.

[5] DIANE O, TERRAN L. Bayesian discovery of multiple bayesian networks via transfer learning[C]. International Conference on Data Mining. 2013: 577-586.

[6] ZHOU Y, WANG J, ZHU C, ZHANG WM. Multiple dags learning with non-negative matrix factorization[C]. Advanced Methodologies for Bayesian Networks, 2017: 81-92.
[7] ZHANG Y, YING W, QIANG Y. Learning to multitask[C]. Neural Information Processing Systems, 2018: 5776-5787.
[8] ZHOU Y, WANG PC. An ensemble learning approach for XSS attack detection with domain knowledge and threat intelligence[J]. Computers & Security 2019(82): 261-269.
[9] EVGENIOU T, PONTIL M. Regularized multi-task learning[C]. International Conference on Knowledge Discovery and Data Mining. 2004:109- 117.
[10] ARGYRIOU A, EVGENIOU T, PONTIL M. Multi-task feature learning[C]. Neural Information Processing Systems. 2006: 41-48.
[11] ANDREAS A, THEODOROS E, MASSIMILIANO P. Convex multi-task feature learning[J]. Machine Learning, 2008,73(3): 243-272.
[12] KANG ZL, GRAUMAN K, SHA F. Learning with whom to share in multi-task feature learning[C]. International Conference on Machine Learning. 2011: 521–528
[13] ABHISHEK K, HAL D. Learning task grouping and overlap in multi-task learning[C]. International Conference on Machine Learning. 2012: 1383–1390.
[14] JUN YJ, CHI HJ. Variable selection and task grouping for multi-task learning[C]. International Conference on Knowledge Discovery and Data Mining. 2018: 1589-1598.
[15] MURUGESAN K , CARBONELL J . Self-paced multitask learning with shared knowledge[C]. International Joint Conference on Artificial Intelligence, 2017: 2522-2528.
[16] DE C, CASSIO P, CORANI G, SCANAGATTA M, CUCCU M, ZAFFALON M. Learning extended tree augmented naïve structures[C]. International Journal of Approximate Reasoning, 2016,68:153-163.