基于特征和实例迁移的加权多任务聚类

张晓彤 张宪超 刘 晗

(大连理工大学软件学院 辽宁 大连 116620)

(大连理工大学辽宁省泛在网络与服务软件重点实验室 辽宁 大连 116620)

摘 要 传统聚类方法只对每个数据集单独进行聚类,但是有时单个数据集中的数据不足以挖掘一个良好的簇结构.在现实生活中,有很多数据集包含相同的类标签,因此存在多个相关的聚类任务.多任务聚类通过在相关任务之间迁移知识来提升每个任务的聚类性能,近些年来它获得越来越多的关注.一个好的多任务聚类算法要完成以下两方面工作:(1)它应该充分利用来自其它任务的知识;(2)它能够自动地评估任务相关性以避免负面迁移.然而,现有多任务聚类方法还不能很好地完成任意一方面的工作.本文提出一个基于特征和实例迁移的加权多任务聚类算法 MTCFIR.一方面,它在任务之间既迁移特征表示知识又迁移实例知识,要比大部分现有多任务聚类方法更充分地利用跨任务知识.另一方面,它自动地学习任务相关性来避免负面迁移,并且没有现有评估任务相关性的多任务聚类方法的限制条件.MTCFIR 执行以下三个步骤.首先,它利用边缘堆栈降噪自编码器在任务之间学习一个共有的特征表示.该步骤通过迁移特征表示知识来降低任务之间的分布差异,这是一致相似度矩阵学习的前提.其次,它通过在任务之间迁移实例知识来为每个任务学习一个一致相似度矩阵,并且通过对任务进行加权来决定不同任务对每个任务的一致相似度矩阵学习的贡献程度.该步骤可以避免在不太相关的任务之间强制迁移知识所带来的负面影响.最后,它在每个任务的一致相似度矩阵上执行对称非负矩阵分解来得到聚类结果.在真实数据集上的实验结果说明本文提出的方法比传统单任务聚类方法和现有多任务聚类方法具有更好的聚类效果,并且要比大部分多任务聚类方法高效.

关键词 多任务聚类;特征表示迁移;实例迁移;任务相关性学习;一致相似度矩阵学习

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2019.02614

Weighed Multi-Task Clustering by Feature and Instance Transfer

ZHANG Xiao-Tong ZHANG Xian-Chao LIU Han

(*School of Software Technology, Dalian University of Technology, Dalian, Liaoning* 116620)

(*Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian University of Technology, Dalian, Liaoning* 116620)

Abstract Traditional clustering methods cluster the samples in each data set individually by only using the knowledge within each data set, but sometimes the data samples in a single data set are not enough to discover a good cluster structure. There are many data sets which contain the same class labels in the real world, hence there exist many related clustering tasks. Multi-task clustering can transfer the relevant knowledge across the related tasks to improve the clustering performance of each task, which has received more and more attentions in recent years. A good multi-task clustering algorithm should accomplish both the following two aspects of work: (1) it should make full use of the useful knowledge from the other related tasks; (2) it can automatically assess the task relatedness among the tasks to avoid negative transfer. Nevertheless, existing multi-task clustering methods have not accomplished either of them well. This paper proposes a

收稿日期:2018-06-01;在线出版日期:2019-03-21.本课题得到国家自然科学基金(61632019)资助.**张晓彤**,博士,中国计算机学会(CCF)会员,主要研究方向为多任务聚类、多任务学习.E-mail:zxt.dut@hotmail.com.**张宪超**,博士,教授,主要研究领域为数据挖掘、机器学习.**刘 晗**,博士,主要研究方向为不确定数据挖掘、多任务聚类.

weighted multi-task clustering method based on feature and instance transfer, which is called as MTCFIR. On one hand, MTCFIR transfers both the knowledge of feature representation and instances across the related tasks, thus making better use of the cross-task knowledge than most existing multi-task clustering methods. On the other hand, MTCFIR automatically learns the task relatedness to avoid negative transfer, and it does not have the limitations of the existing multi-task clustering methods which can assess the task relatedness, e. g. , all the tasks should have the same cluster numbers, and the label marginal distribution in each task distributes evenly. There are three steps in the MTCFIR method. First, it learns a common feature representation among the related tasks with marginalized stacked denoising autoencoders (SDA). SDA can abstract a set of high-level features that indicate the generic concepts, learning these features for multi-task data is beneficial to extract commonality from the original features in different tasks. This step can reduce the distribution difference among the tasks by transferring the knowledge of feature representations, which is the premise of learning the consistent similarity matrix in the second step. Second, it learns a consistent similarity matrix for each task by transferring the instance knowledge across the related tasks, and determines the contribution degree of different tasks for learning the consistent similarity matrix of each task by weighting the tasks. This step can avoid the negative effects caused by enforcing the knowledge transfer among the tasks that are not too related, thus taking full advantage of the relevant instance knowledge among the tasks. Third, it performs the symmetric nonnegative matrix factorization (SNMF) on the consistent similarity matrix of each task to get the clustering results. SNMF is a clustering technique which can capture the cluster structure embedded into the similarity matrix. Experimental results on several real world data sets show that the proposed method MTCFIR performs much better than traditional single-task clustering methods and existing multi-task clustering methods, and is more efficient than existing multi-task clustering methods in general. The learned task relatedness shown in the experiments further verify the effectiveness of the proposed MTCFIR.

Keywords　multi-task clustering; feature representation transfers; instance transfer; task relatedness learning; consistent similarity matrix learning

1　引　言

传统的聚类算法可以对一个单独的数据集进行聚类，但是一个数据集中的信息可能不足以挖掘正确的簇结构. 多任务聚类以传统的聚类方法为手段，通过学习任务之间的相关知识并在任务之间迁移这些知识来提高每个任务的聚类性能. 一个好的多任务聚类算法应该能完成两方面的工作：(1) 充分利用其它任务中的知识；(2) 评估任务相关性来避免负面迁移问题[1]. 然而，现有的多任务聚类算法并不能很好地完成这两个工作.

早期的多任务聚类算法通常基于一个理想的假设，即所有的任务是完全相关的(所有任务共享相同的簇标签)，所以它们只能处理第一个工作. 在多任务聚类中，主要有三种方式来迁移相关知识[1]：特征表示迁移在相关任务之间学习一个共有的特征表示；实例迁移利用其它任务中的相关实例来帮助每个任务进行聚类；模型参数迁移为所有任务学习它们的共享模型参数或者模型超参数的先验分布. 早期的多任务聚类方法通常只能在任务之间迁移一种知识[2]. 最近，以 MTCTKI[2] 为代表的多任务聚类算法能够在任务之间同时迁移特征表示和实例知识，从而更充分地利用其它任务中的知识.

在现实生活中，任务通常是部分相关的(任务之间只共享一部分簇标签)，而强制地在部分相关任务之间迁移知识会降低任务的聚类性能. 在任务之间迁移知识导致任务聚类性能变差的现象被称为负面迁移[1]. 因此第二个工作对于多任务聚类算法是必不可少的. 目前有两个代表性的多任务聚类算法可以自动评估任务的相关性. (1) DMTRC[3] 通过高斯先验来学习任务相关性. 但是它基于一个很严格的

假设,即所有任务的聚类个数是相同的并且每个任务的簇标签边缘分布是均匀的;(2) SAMTC[4] 为每对任务学习一对可能相关的子任务,然后评估每对子任务之间的相关性.但是它将子任务外的数据直接丢弃,这可能会丢失一些在其它任务中潜在相关的有用信息.

在本文中,我们提出了一个基于特征和实例迁移的加权多任务聚类算法 MTCFIR,它不仅能够在任务之间同时迁移特征表示和实例知识,还能够自动地学习任务之间的相关性,从而避免负面迁移问题.MTCFIR 算法执行以下三个步骤:(1)共有特征表示学习.该步骤用边缘堆栈降噪自编码器方法 mSDA[5] 来为所有任务学习一个共有的特征表示.该步骤通过迁移特征表示知识来降低任务之间的分布差异,这是一致相似度矩阵学习的前提;(2)一致相似度矩阵学习.该步骤通过在任务之间迁移实例知识来为每个任务学习一个一致相似度矩阵.同时该步骤自动学习任务相关性来对每个任务进行加权,以决定其它任务对于当前任务的一致相似度矩阵学习的贡献程度;(3)对称非负矩阵分解.该步骤对每个任务的一致相似度矩阵进行对称非负矩阵分解[6],从而得到每个任务的聚类结果.在真实数据集上的实验结果验证了本文提出的 MTCFIR 算法相比于传统的单任务聚类算法和现有多任务聚类算法具有更好的聚类性能.

2 相关工作

2.1 多任务聚类

多任务聚类是一种无监督的多任务学习方法,它在近十年来获得了越来越多的关注.多任务聚类通过在相关任务之间迁移知识来提升每个任务的聚类性能.现有的多任务聚类算法有三种迁移知识的方式.

(1)特征表示迁移.该方式在任务之间学习一个共有的特征表示,这个共有特征表示会降低任务之间的分布差异.该方式基于相关任务之间通常会共享一些相同语义特征的观察.由于该共有特征表示是利用所有任务的特征知识一起学习出来的,因此不同任务的特征知识都将融入到该共有特征表示中.之后每个任务都会用该共有特征表示重新进行特征构造,所以每个任务都会利用到其它任务中的特征知识,即该共有特征表示起到在任务之间迁移特征知识的作用.LSSMTC 算法[7] 学习一个所有任务共享相同质心的子空间.LNKMTC 和 LSKMTC 算法[8] 学习一个所有任务分布相近且保留原始数据几何结构的核空间.MCDA 算法[9] 学习一个所有任务分布相近的共享子空间.ITCC 算法[10] 利用信息论联合聚类为每对任务学习一个特征关联矩阵.

(2)实例迁移.该方式利用其它任务中的相关实例帮助每个任务进行聚类.该方式基于相关任务之间会共享一些相同的簇标签,而不同任务之间具有相同簇标签的数据通常是相关的观察.MBC 算法[11] 和它的两个改进算法 S-MBC 和 S-MKC[12-13] 交替进行簇质心学习和任务间质心相关性学习.SMT-NMF 算法[14] 引入一个需要人工设置的任务间偏差来对不同任务之间的任意两个数据点的距离进行加权,但是该方法只能处理具有两个簇的任务.SAMTC 算法[4] 首先通过可用实例寻找步骤来为每对任务构造一对子任务,然后学习每对子任务之间的相关性,最后其它子任务中的数据参与到每个任务共享最近邻相似度矩阵的计算中.

(3)模型参数迁移.该方式为所有任务学习一个共享模型参数或者模型超参数的先验分布.该方式基于相关任务之间会共享一些相同的簇标签,而任务之间相同簇标签所对应的模型参数应该是相似的观察.DMTFC 和 DMTRC 算法[3] 分别学习特征相关性和任务相关性,这两种方法首先引入模型参数的高斯先验,然后通过计算高斯先验中的协方差矩阵来学习特征相关性和任务相关性.上述方法要求每个任务具有相同的簇个数,并且每个任务中的簇标签是均匀分布的,即每个任务中不同簇中的数据个数是相同的.MTSC[15] 为每个任务引入一个线性回归模型,然后在所有任务的模型参数上加入 $\ell_{2,p}$ 范数正则化,从而使模型参数只在任务之间的某部分特征上进行迁移.

由于基于实例迁移和模型参数迁移的方法都是在没有考虑任务分布差异的情况下,直接在原始空间中迁移实例或模型参数知识,因此这些方法更适合处理分布相近的任务.

最近的研究提出一种同时迁移特征表示和实例知识的 MTCTKI 算法[2].它首先采用最大平均差异分布度量方法来学习一个任务分布互相接近的共享子空间.然后在这个共享子空间中,对于每个任务中的两个数据点,该方法通过利用它们在其它任务中的共享最近邻来参与计算它们的相似度.最后,该方法在每个任务的共享最近邻相似度矩阵上执行谱聚类来得到最终的聚类结果.但是,MTCTKI 只考

虑任务完全相关的情况，因此它在处理部分相关任务时，会强制其它任务中的实例知识完全参与到每个任务的相似度矩阵学习中.

在上述多任务聚类算法中，大部分方法都是针对完全相关任务的，因此该类方法强制在任务之间迁移所有知识. 但是在现实生活中，任务通常是部分相关的，这时在任务之间强制迁移所有知识会导致负面迁移问题[1]. 一种避免负面迁移问题的手段是评估任务相关性，即通过任务相关性来控制任务之间知识的迁移量. 在多任务聚类的文献中，以 DMTRC[3] 和 SAMTC[4] 为代表的多任务聚类算法能够自动学习任务相关性，但是它们有一些限制条件. DMTRC 算法假设所有任务的聚类个数是相同的并且每个任务的各个簇具有相同的数据个数. SAMTC 算法直接丢弃了子任务外的数据，这可能会导致其它任务中潜在相关信息的丢失. 此外，这两个算法直接在原始空间中迁移模型参数或实例知识，没有考虑任务的分布差异，因此更适合于处理分布差异较小的任务.

根据上述分析，现有的多任务聚类方法还不能够同时做到既迁移多种知识，又自动学习任务之间的相关性. 为了解决这一问题，本文提出一个基于特征和实例迁移的加权多任务聚类算法 MTCFIR. 一方面，通过同时迁移特征和实例知识，MTCFIR 可以利用更多的任务间知识对每个任务进行聚类. 另一方面，通过学习任务之间的相关性，MTCFIR 可以只让一个任务中一定比例的实例知识迁移给另一个任务，从而避免负面迁移问题.

2.2　多任务学习

多任务学习（监督多任务学习）[16] 比无监督的多任务聚类研究得更加成熟. 多任务学习通过在任务之间迁移知识来提高所有任务的预测性能. 与多任务聚类方法类似，多任务学习方法也主要迁移三种类型的知识：特征表示、实例和模型参数.

最近的多任务学习文献中也提出一种同时迁移特征表示和模型参数的多任务学习方法[17]，该方法可以充分利用任务之间的特征表示和模型参数知识，来提高每个任务的预测性能.

在多任务学习中，有很多方法都是针对部分相关任务而被提出来的. 一种解决部分相关任务的方式是将任务划分成簇[18-19]，簇内的任务是相关的，不同簇中的任务是不相关的. 因此该类方法只对每个簇内部的任务迁移知识，而不同簇中的任务不迁移任何知识. 该类方法过于绝对地区分任务是相关或不相关的，这样的假设过于理想. 另一种解决部分相关任务的方式是评估任务之间的相关性. 一种评估任务相关性的方式是像 DMTRC 算法假设任务之间的模型参数共享高斯先验，然后通过学习协方差矩阵来学习任务之间的相关性[20-21]. 另一种评估任务相关性的方式是假设有一个任务关联矩阵，然后在监督多任务学习过程中自动学习该任务关联矩阵[22]. 从目前的监督多任务学习方法可以看出，评估任务相关性来处理部分相关任务是一种主流趋势. 因此，本文也采用此类方式来解决任务部分相关的情况.

3　MTCFIR 算法

3.1　多任务数据符号定义

给定 $\mathcal{T}$ 个聚类任务，第 t 个任务 $X^t=\{\boldsymbol{x}_1^t,\boldsymbol{x}_2^t,\cdots,\boldsymbol{x}_{n^t}^t\}\in\mathbb{R}^{d\times n^t}$，其中 n^t 是第 t 个任务的样本个数，d 是特征个数. 在多任务聚类领域，不同的任务会被预处理到一个相同的特征空间中[7]. 例如，对于文本任务，如果一个任务不包含其它任务中的某些单词特征，我们会对该任务的这些单词特征进行补零操作；对于图像任务，我们会将所有任务的图像缩放到相同的像素尺寸. 尽管这 $\mathcal{T}$ 个聚类任务被预处理到相同的 d 维特征空间中，但是这样简单的特征预处理方法并没有改变每个任务的分布特性，即这些任务的分布差异依然很大.

3.2　算法概览

MTCFIR 算法包含三个步骤，其总体算法流程如算法 1 所示.

算法 1.　MTCFIR.

输入：$\mathcal{T}$ 个任务 $\{X^t\}_{t=1}^{\mathcal{T}}$，所有任务的聚类个数 $\{k^t\}_{t=1}^{\mathcal{T}}$，任务内部最近邻个数 $\{l^t\}_{t=1}^{\mathcal{T}}$，共有特征表示学习层数 g，特征加入噪声的概率 p

输出：簇划分 $\{C^t\}_{t=1}^{\mathcal{T}}$.

1. 共有特征表示学习步骤（算法 2）.
2. 一致相似度矩阵学习步骤（算法 3）.
3. 对称非负矩阵分解聚类步骤（算法 4）.

图 1 展示了 MTCFIR 算法中三个步骤的衔接关系. 在共有特征表示学习步骤中，原始特征表示下的 $\mathcal{T}$ 个任务 $\{X^t\}_{t=1}^{\mathcal{T}}$ 转化为新特征表示下的 $\mathcal{T}$ 个任务 $\{Z^t\}_{t=1}^{\mathcal{T}}$. 在一致相似度矩阵学习步骤中，根据

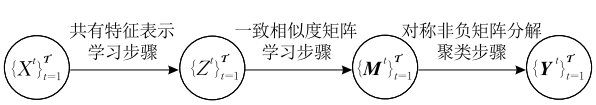

图 1　MTCFIR 算法的三个步骤衔接关系

$\{Z^t\}_{t=1}^{\mathcal{T}}$来为每个任务学习一个一致相似度矩阵$\{\boldsymbol{M}^t\}_{t=1}^{\mathcal{T}}$.在对称非负矩阵分解聚类步骤中,通过对$\{\boldsymbol{M}^t\}_{t=1}^{\mathcal{T}}$执行对称非负矩阵分解聚类来得到每个任务的簇指示矩阵$\{\boldsymbol{Y}^t\}_{t=1}^{\mathcal{T}}$.

MTCFIR 算法的三个步骤的职能如下:

(1)共有特征表示学习步骤会输出表示所有任务的新的特征集合,该特征集合包含两部分:每个任务的原始特征和所有任务的共有特征.每个任务的原始特征中包含任务独有特征,这些独有特征有利于对任务本身进行聚类.所有任务的共有特征是指任务共享的语义特征,它是利用边缘降噪自编码器根据所有$\mathcal{T}$个任务一起学习到的.这些共有特征可以降低任务的分布差异,这有利于从其它任务中学习到关联值较大的数据.本步骤输出的特征会重新表示所有任务的数据,进而用于下一步骤的一致相似度矩阵构造中.

(2)一致相似度矩阵学习步骤利用所有$\mathcal{T}$个任务为每个任务学习一个一致相似度矩阵.它基于的思想是一个任务中两个相似的数据点与其它数据点的关联值也很相似.由于每个任务的数据点都能为第t个任务学习一个相似度矩阵,因此对于第t个任务,我们会得到$\mathcal{T}$个相似度矩阵.但是因为每个任务应该具有一个确定的聚类结果,所以我们用一个一致相似度矩阵去表示这$\mathcal{T}$个相似度矩阵.此外,考虑到任务之间的关联性强弱,我们为每对任务施加一个权重,控制任务之间实例知识迁移的程度,这样有助于避免任务之间的负面迁移问题[1].本步骤输出的一致相似度矩阵将会用于下一步骤的聚类过程中.

(3)对称非负矩阵分解聚类步骤对每个任务的一致相似度矩阵进行对称非负矩阵分解聚类,从而得到每个任务的聚类结果.

3.3 共有特征表示学习

当任务分布差异较大时,我们通常找不到任务之间关联较大的数据,因此我们希望能够学习任务之间的共有特征,从而降低任务之间的分布差异.该步骤为下一步骤的一致相似度矩阵学习提供了前提条件.

考虑到相关任务通常会共享一些具有相同语义的特征,我们利用边缘堆栈降噪自编码器 mSDA 方法[5]来学习任务之间共有的语义特征,即共有特征表示.mSDA 是堆栈降噪自编码器 SDA[23]的改进方法.mSDA 和 SDA 是一类深度特征学习方法,它们都能够学习高层次的具有语义概念的特征[5],但是 mSDA 比 SDA 更高效.

在 mSDA 中,数据是通过随机将某些特征的值置零来进行加噪的[5].mSDA 的目标是学习一个特征映射矩阵$\boldsymbol{W}$,通过$\boldsymbol{W}$的特征映射,加噪后的数据能够恢复成原始的数据.这里$\boldsymbol{W}$起到降噪的作用.

由于不同任务中具有相同类标签的数据通常只是在某些特征上存在差异,因此 mSDA 的加噪降噪思想特别适用于为这些任务学习出共有的语义特征.具体地说,mSDA 对同一数据x进行多次随机加噪所得到的数据可以类比于不同任务中具有相同类标签的数据.对多次随机加噪的数据降噪为原始数据x可以类比于学习不同任务中具有相同类标签的数据的共有特征表示.

为了学习所有任务共有的特征表示,我们将所有任务的数据合并起来,然后利用 mSDA 学习一个特征映射矩阵$\boldsymbol{W}$,其具体计算过程如下:

令$\boldsymbol{X}=[X^1,\cdots,X^{\mathcal{T}}]\in\mathbb{R}^{d\times\tilde{n}}$,其中$\tilde{n}$是所有任务的数据个数.将一个常数特征加入到$\boldsymbol{X}$中,即$\boldsymbol{X}=[\boldsymbol{X};\boldsymbol{1}]\in\mathbb{R}^{(d+1)\times\tilde{n}}$,其中$\boldsymbol{1}\in\mathbb{R}^{1\times\tilde{n}}$,利用 mSDA 学习单层共有特征表示的优化目标为

$$\min_{\boldsymbol{W}}\frac{1}{2m\tilde{n}}\sum_{i=1}^{m}\left\|\boldsymbol{X}-\boldsymbol{W}\widetilde{X}_i\right\|_F^2 \tag{1}$$

其中,$\widetilde{X}_i$是$\boldsymbol{X}$的第i个加入噪声的版本,m是加入噪声的总次数.如果一个特征被加入噪声,那么该特征将被置为 0.

式(1)可以被简化为

$$\min_{\boldsymbol{W}}\left\|\bar{\boldsymbol{X}}-\boldsymbol{W}\hat{\boldsymbol{X}}\right\|_F^2 \tag{2}$$

其中,$\bar{\boldsymbol{X}}=[\boldsymbol{X},\cdots,\boldsymbol{X}]$是$\boldsymbol{X}$的$m$次重复版本,$\hat{\boldsymbol{X}}=[\widetilde{X}_1,\cdots,\widetilde{X}_m]$是$\bar{\boldsymbol{X}}$的加入噪声的版本.$\boldsymbol{W}=[\boldsymbol{W},\boldsymbol{b}]\in\mathbb{R}^{(d+1)\times(d+1)}$是一个特征映射矩阵,$\boldsymbol{b}\in\mathbb{R}^{(d+1)\times1}$是一个偏差项.

式(2)可以通过最小二乘方法[24]来解决,即

$$\boldsymbol{W}=PQ^{-1} \tag{3}$$

其中,$P=\bar{\boldsymbol{X}}\hat{\boldsymbol{X}}^{\mathrm{T}}$,$Q=\hat{\boldsymbol{X}}\hat{\boldsymbol{X}}^{\mathrm{T}}$.式(3)的结果依赖于数据点的哪些特征被随机地加入噪声.为了提高结果的鲁棒性,需要加入噪声的次数m应该尽可能高.根据弱大数定律,当$m\to+\infty$时,P和Q的期望$E[P]$和$E[Q]$以及$\boldsymbol{W}$的计算方式如下所示.

令每个原始特征加入噪声的概率为p,常数特征加入噪声的概率永远是 0,可得所有特征未被加入噪声的概率向量$\boldsymbol{q}=[1-p,\cdots,1-p,1]^{\mathrm{T}}\in\mathbb{R}^{(d+1)\times1}$,其中$q_i$是第$i$个特征未被加入噪声的概率.令$S=\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$,有

$$\boldsymbol{W}=E[P]E[Q]^{-1} \tag{4}$$

其中，

$$(E[P])_{ij}=S_{ij}q_j \tag{5}$$

$$(E[Q])_{ij}=\begin{cases}S_{ij}q_iq_j, & \text{如果 } i\neq j\\ S_{ij}q_i, & \text{如果 } i=j\end{cases} \tag{6}$$

在实践中，式(4)能够通过 MATLAB 中的右除 $E[P]/E[Q]$来计算，这可以避免代价较高的矩阵求逆过程[5].

在计算出 $\boldsymbol{W}$ 后，我们可以得到还原的数据 $W_{1:d,:}\boldsymbol{X}$，其中 $W_{1:d,:}$ 是 $\boldsymbol{W}$ 的前 d 行. 为了确保生成的特征是非线性的，我们将非线性编码函数 tanh 赋予给还原的数据，即单层共有特征表示下的数据为 $\tanh(W_{1:d,:}\boldsymbol{X})$.

对于多层共有特征表示学习，令第 i 层的输出数据为 h^i，第 i 层的特征映射矩阵为 $\boldsymbol{W}^i$，原始输入数据为 $\boldsymbol{h}^0=[X^1,\cdots,X^{\mathcal{T}}]$，层数为 g. 第 i 层的输出数据为 $\boldsymbol{h}^i=\tanh(W^i_{1:d,:}\bar{\boldsymbol{h}}^{i-1})$，其中 $\bar{\boldsymbol{h}}^{i-1}=[\boldsymbol{h}^{i-1};\boldsymbol{1}]$是加入常数特征的数据.

每个任务的原始特征既包含任务独有的特征，又隐含任务之间共有的语义特征，前者有利于聚类任务本身，后者有利于降低任务之间的分布差异. 由于多层共有特征表示学习输出的是任务之间共有的语义特征，因此为了保留任务独有的特征，我们最终得到的数据既包括数据的原始特征，又包括共有特征表示学习中各层输出的特征，即 $H=[\boldsymbol{h}^0;\cdots;\boldsymbol{h}^g]\in\mathbb{R}^{d(g+1)\times\bar{n}}$，这里的分号是对矩阵按行叠加. 根据所有任务的数据集 H，可以得到每个任务的数据集 $\{Z^t\}_{t=1}^{\mathcal{T}}$，其中 $Z^t=[\boldsymbol{z}_1^t,\cdots,\boldsymbol{z}_{n^t}^t]\in\mathbb{R}^{d(g+1)\times n^t}$.

共有特征表示学习步骤的算法流程如算法 2 所示.

算法 2.　共有特征表示学习步骤.

输入：$\mathcal{T}$ 个任务 $\{X^t\}_{t=1}^{\mathcal{T}}$，共有特征表示学习层数 g，特征加入噪声的概率 p

输出：新特征表示下的 $\mathcal{T}$ 个任务 $\{Z^t\}_{t=1}^{\mathcal{T}}$

1. 设置 $\boldsymbol{h}^0=[X^1,\cdots,X^{\mathcal{T}}]$.
2. FOR $i=1$ TO g DO
3. 　对 $\boldsymbol{h}^{i-1}$ 加入常数特征 $\bar{\boldsymbol{h}}^{i-1}=[\boldsymbol{h}^{i-1};\boldsymbol{1}]$.
4. 　根据式(4)计算第 i 层的特征映射矩阵 $\boldsymbol{W}^i$.
5. 计算第 i 层的共有特征表示下的数据 $\boldsymbol{h}^i=\tanh(W^i_{1:d,:}\bar{\boldsymbol{h}}^{i-1})$.
6. END FOR
7. 计算所有任务在原始特征和共有特征表示下的数据 $H=[\boldsymbol{h}^0;\cdots;\boldsymbol{h}^g]$，从而得到每个任务的数据集 $\{Z^t\}_{t=1}^{\mathcal{T}}$.

3.4　一致相似度矩阵学习

直觉上，对于一个任务中任意两个数据点，如果它们与其它数据点的关联值很接近，那么它们通常具有较高的相似度.

给定第 t 个任务的任意两个数据点 $\boldsymbol{z}_i^t$ 和 $\boldsymbol{z}_j^t$，以及它们和第 s 个任务 Z^s 中数据点的关联值 $V^{ts}_{i,:}$ 和 $V^{ts}_{j,:}$，利用第 s 个任务来为第 t 个任务学习相似度矩阵 $\boldsymbol{M}_s^t\in\mathbb{R}^{n^t\times n^t}$ 的优化目标为

$$\begin{aligned}&\min_{\boldsymbol{M}_s^t}\sum_{i=1}^{n^t}\sum_{j=1}^{n^t}\|V^{ts}_{i,:}-V^{ts}_{j,:}\|_2^2(\boldsymbol{M}_s^t)_{ij}+\beta\|\boldsymbol{M}_s^t\|_F^2\\ &\text{s.t.}\ \sum_{i=1}^{n^t}(\boldsymbol{M}_s^t)_{ij}=1,\ (\boldsymbol{M}_s^t)_{ij}\geq 0\end{aligned} \tag{7}$$

其中，$V^{ts}_{i,:}\in\mathbb{R}^{1\times n^s}$ 是 $\boldsymbol{V}^{ts}$ 的第 i 行. $\boldsymbol{V}^{ts}\in\mathbb{R}^{n^t\times n^s}$ 是任务 t 和任务 s 数据之间的关联值矩阵，它可以通过余弦相似度或者高斯核相似度等度量方法来计算. β 是一个正则化参数.

式(7)的第一项意味着 $V^{ts}_{i,:}$ 和 $V^{ts}_{j,:}$ 之间的欧几里得距离越小，它们的相似度 $(\boldsymbol{M}_s^t)_{ij}$ 越高. 式(7)的第二项 $\beta\|\boldsymbol{M}_s^t\|_F^2$ 可以防止 $\boldsymbol{M}_s^t$ 的每一列中，只有最近的 $V^{ts}_{i,:}$ 和 $V^{ts}_{j,:}$ 之间具有相似度 $(\boldsymbol{M}_s^t)_{ij}$ 等于 1，其余的相似度 $(\boldsymbol{M}_s^t)_{ij}$ 均为 0 的这种情况发生. 计算时约束 $\sum_{i=1}^{n^t}(\boldsymbol{M}_s^t)_{ij}=1$ 更有利于计算出 $\boldsymbol{M}_s^t$，约束 $(\boldsymbol{M}_s^t)_{ij}\geq 0$ 可以确保相似度 $(\boldsymbol{M}_s^t)_{ij}$ 的非负性.

根据式(7)，对于每个任务 t，$\mathcal{T}$ 个相似度矩阵 $\{\boldsymbol{M}_s^t\}_{s=1}^{\mathcal{T}}\in\mathbb{R}^{n^t\times n^t}$ 可以从 $\mathcal{T}$ 个任务中学习到. 由于每个任务 t 应该具有一个确定的相似度矩阵，从而获得一个确定的聚类结果，因此我们令这 $\mathcal{T}$ 个相似度矩阵 $\{\boldsymbol{M}_s^t\}_{s=1}^{\mathcal{T}}$ 趋向于同一个相似度矩阵 $\boldsymbol{M}^t\in\mathbb{R}^{n^t\times n^t}$. 这里 $\boldsymbol{M}^t$ 就是我们要学习的一致相似度矩阵，实例知识可以通过一致相似度矩阵 $\boldsymbol{M}^t$ 在任务间进行迁移.

考虑到任务之间相关性有强弱之分，我们在计算 $\boldsymbol{M}^t$ 时赋予不同任务的数据不同的权重. 因此任务 t 学习一致相似度矩阵 $\boldsymbol{M}^t$ 的优化目标为

$$\begin{aligned}&\min_{\boldsymbol{M}^t}\sum_{i=1}^{n^t}\sum_{j=1}^{n^t}\sum_{s=1}^{\mathcal{T}}\alpha_s^t\|V^{ts}_{i,:}-V^{ts}_{j,:}\|_2^2M^t_{ij}+\beta\|\boldsymbol{M}^t\|_F^2\\ &\text{s.t.}\ \sum_{i=1}^{n^t}M^t_{ij}=1, M^t_{ij}\geq 0(j=1,\cdots,n^t)\end{aligned} \tag{8}$$

其中，α_s^t 是任务 s 对于任务 t 的关联系数，该参数用于控制任务 s 中多少比例的实例知识来计算 $\boldsymbol{M}^t$.

式(8)包含变量 $\boldsymbol{M}^t$、参数 β 以及任务关联系数

α_s^t,其计算过程如下:

(1)求解 $\boldsymbol{M}^t$:

最小化式(8)来求解 $\boldsymbol{M}^t$ 的每一列 $M_{:,j}^t$($j=1,\cdots,n^t$)的优化目标为

$$\min_{M_{:,j}^t}\sum_{i=1}^{n^t}\sum_{s=1}^{\mathcal{T}}\alpha_s^t\left\|V_{i,:}^{ts}-V_{j,:}^{ts}\right\|_2^2 M_{ij}^t+\beta\left\|M_{:,j}^t\right\|_2^2$$
$$\text{s.t.}\ \sum_{i=1}^{n^t}M_{ij}^t=1,\ M_{ij}^t\geqslant 0 \tag{9}$$

其中,$M_{:,j}^t$ 是 $\boldsymbol{M}^t$ 的第 j 列. 式(9)可以通过二次规划来求解,但是因为二次规划对于较大的数据量的计算来说不够高效,因此本节采用一种更高效的方法来优化 $M_{:,j}^t$,其求解过程如下.

令 $A_{ij}=\sum_{s=1}^{\mathcal{T}}\alpha_s^t\left\|V_{i,:}^{ts}-V_{j,:}^{ts}\right\|_2^2$,$A_{:,j}$ 是 $\boldsymbol{A}$ 的第 j 列,式(9)可以被改写为

$$\min_{M_{:,j}^t}\frac{1}{2}\left\|M_{:,j}^t+\frac{1}{2\beta}A_{:,j}\right\|_2^2$$
$$\text{s.t.}\ \sum_{i=1}^{n^t}M_{ij}^t=1,\ M_{ij}^t\geqslant 0 \tag{10}$$

式(10)的拉格朗日函数为

$$L(M_{:,j}^t)=\frac{1}{2}\left\|M_{:,j}^t+\frac{1}{2\beta}A_{:,j}\right\|_2^2-\lambda(\mathbf{1}^{\mathrm{T}}M_{:,j}^t-1)-\boldsymbol{\mu}^{\mathrm{T}}M_{:,j}^t \tag{11}$$

其中,λ 是一个实数的拉格朗日乘子,$\boldsymbol{\mu}\in\mathbb{R}^{n^t\times 1}$ 是一个非负拉格朗日乘子,$\mathbf{1}\in\mathbb{R}^{n^t\times 1}$. 根据 KKT 条件 $\frac{\partial L(M_{:,j}^t)}{\partial M_{:,j}^t}=0$,有

$$M_{:,j}^t+\frac{1}{2\beta}A_{:,j}-\lambda\mathbf{1}-\boldsymbol{\mu}=0 \tag{12}$$

$M_{:,j}^t$ 的第 i 个元素为

$$M_{ij}^t=-\frac{1}{2\beta}A_{ij}+\lambda+\mu_i \tag{13}$$

其中,μ_i 是 $\boldsymbol{\mu}$ 的第 i 个元素.

由于一个稀疏相似度矩阵通常能够获得更好的聚类性能[25],所以 MTCFIR 只保留 $M_{:,j}^t$ 的前 l^t 个最大的元素,令其它元素和 M_{jj}^t 均为 0. 根据 KKT 条件 $\mu_i M_{ij}^t=0$,有

$$M_{ij}^t=\begin{cases}-\frac{1}{2\beta}A_{ij}+\lambda, & \text{如果 } \boldsymbol{x}_i^t\in N(\boldsymbol{x}_j^t)\\ 0, & \text{否则}\end{cases} \tag{14}$$

其中,$\mathcal{N}(\boldsymbol{x}_j^t)$ 是 $\boldsymbol{x}_j^t$ 的 l^t 个最近邻集合,这可以通过将 $A_{:,j}$ 升序排列来获得.

(2)求解 λ 和 β:

将式(14)代入到约束 $\sum_{i=1}^{n^t}M_{ij}^t=1$ 中,并定义 $A_{:,j}$ 的升序排列为 $B_{:,j}$,有

$$\sum_{i=2}^{l^t+1}\left(-\frac{1}{2\beta}B_{ij}+\lambda\right)=1 \tag{15}$$

则

$$\lambda=\frac{1}{l^t}+\frac{1}{2l^t\beta}\sum_{i=2}^{l^t+1}B_{ij} \tag{16}$$

根据式(13)和(14),当 $\boldsymbol{x}_i^t\in\mathcal{N}(\boldsymbol{x}_j^t)$ 时,有 $-\frac{1}{2\beta}A_{ij}+\lambda>0$;当 $\boldsymbol{x}_i^t\notin\mathcal{N}(\boldsymbol{x}_j^t)$ 时,由于 $M_{ij}^t=0$,$\mu\geq 0$,有 $-\frac{1}{2\beta}A_{ij}+\lambda\leqslant 0$,即

$$-\frac{1}{2\beta}B_{l^t+1,j}+\lambda>0,\ -\frac{1}{2\beta}B_{l^t+2,j}+\lambda\leqslant 0 \tag{17}$$

其中,$B_{l^t+1,j}$ 是 B 的第 l^t+1 行第 j 列元素. 将式(16)代入到式(17)中,有

$$\frac{1}{2}(l^tB_{l^t+1,j}-\sum_{i=2}^{l^t+1}B_{ij})<\beta\leqslant\frac{1}{2}(l^tB_{l^t+2,j}-\sum_{i=2}^{l^t+1}B_{ij}) \tag{18}$$

因此可得

$$\beta=\frac{1}{2}(l^tB_{l^t+2,j}-\sum_{i=2}^{l^t+1}B_{ij}) \tag{19}$$

(3)求解 α_s^t:

直觉上,一对任务越相关,它们具有越多的相关实例. 我们已经计算了任意两个任务之间的数据关联值矩阵 $\boldsymbol{V}^{ts}$,因此两个任务的相关性可以通过 $\boldsymbol{V}^{ts}$ 中具有较高关联值的元素比例来计算. 为了筛选较高关联值的元素,我们为每个任务设置了一个阈值 $\varepsilon^t=\text{median}(U_{l^t+1,:}^{tt})$,其中 $\text{median}(U_{l^t+1,:}^{tt})$ 是 $U_{l^t+1,:}^{tt}$ 的中位数,U^{tt} 是 V^{tt} 按列的降序排列. 我们设置这样的阈值主要是因为只有任务 t 中 l^t 个最近邻的关联值被认为比较高.

在为每个任务 t 设置阈值 ε^t 后,任务 t 和任务 s 之间的关联值矩阵 $\boldsymbol{V}^{ts}$ 中具有较高关联值的元素比例就是任务 s 相对于任务 t 的关联系数.

$$\alpha_s^t=\frac{|\boldsymbol{V}^{ts}\geqslant\varepsilon^t|}{n^tn^s} \tag{20}$$

一致相似度矩阵学习步骤的算法流程如算法 3 所示.

算法 3. 一致相似度矩阵学习步骤.

输入:新特征表示下的 $\mathcal{T}$ 个任务 $\{Z^t\}_{t=1}^{\mathcal{T}}$,任务内部最近邻个数 $\{l^t\}_{t=1}^{\mathcal{T}}$

输出：每个任务的一致相似度矩阵$\{\boldsymbol{M}^t\}_{t=1}^{\mathcal{T}}$

1. FOR $t=1$ TO $\mathcal{T}$ DO
2. FOR $s=1$ TO $\mathcal{T}$ DO
3. 根据$\{Z^t\}_{t=1}^{\mathcal{T}}$计算关联值矩阵$\boldsymbol{V}^{ts}$.
4. 根据式(20)计算任务s相对于任务t的关联系数α_s^t.
5. END FOR
6. 根据公式$A_{ij}=\sum_{s=1}^{\mathcal{T}}\alpha_s^t\|V_{i,:}^{ts}-V_{j,:}^{ts}\|_2^2$计算$\boldsymbol{A}$.
7. FOR $j=1$ TO n^t DO
8. 分别根据式(16)和(19)计算λ和β.
9. 根据式(14)计算$\boldsymbol{M}^t$的每一列$M_{:,j}^t$.
10. END FOR
11. END FOR

在算法3中，每个任务的一致相似度矩阵$\boldsymbol{M}^t$的计算式(14)包含3个变量：矩阵$\boldsymbol{A}$、参数λ和β.由于矩阵$\boldsymbol{A}$的计算需要利用任务间的关联系数α_s^t，所以α_s^t参与到$\boldsymbol{M}^t$的计算过程中，这控制了$\boldsymbol{M}^t$从其它任务中获取的实例知识量.这样，在对$\boldsymbol{M}^t$进行对称非负矩阵分解聚类时，就可以避免其它任务中的负面实例知识降低第t个任务的聚类性能.

3.5　对称非负矩阵分解聚类

在计算出任务t的一致相似度矩阵$\boldsymbol{M}^t$后，我们希望利用对称非负矩阵分解方法[6]将每个任务t划分为k^t个簇.对称非负矩阵分解是一种利用相似度矩阵来对数据进行聚类的方法，它能够挖掘嵌入到相似度矩阵中的簇结构，其优化目标为

$$\min_{\boldsymbol{Y}^t}\|\boldsymbol{M}^t-\boldsymbol{Y}^t(\boldsymbol{Y}^t)^{\mathrm{T}}\|_F^2\quad \text{s.t. } \boldsymbol{Y}^t\geqslant 0 \tag{21}$$

其中，$\boldsymbol{M}^t\in\mathbb{R}^{n^t\times n^t}$是任务$t$的一致相似度矩阵，$\boldsymbol{Y}^t\in\mathbb{R}^{n^t\times k^t}$是任务$t$的簇指示矩阵.

根据式(21)，可以看出在对称非负矩阵分解中，有$M_{ij}^t=\boldsymbol{Y}_{i,:}^t(\boldsymbol{Y}_{j,:}^t)^{\mathrm{T}}$.在理想情况下，如果任务$t$的两个数据点$\boldsymbol{x}_i^t$和$\boldsymbol{x}_j^t$属于同一个簇，那么应有$M_{ij}^t=1$，否则有$M_{ij}^t=0$.但是式(8)中的约束$\sum_{i=1}^{n^t}M_{ij}^t=1$使$M_{ij}^t$远小于1，这不利于获取真实的聚类结果.因此，我们将$\boldsymbol{M}^t$的每一列元素$M_{:,j}^t$同比例扩大，使$M_{:,j}^t$中最大的元素扩大为1，即$\max(M_{:,j}^t)=1$，其中$\max(M_{:,j}^t)$是$M_{:,j}^t$中最大的元素.最后，为了使$\boldsymbol{M}^t$对称，我们令$\boldsymbol{M}^t=(\boldsymbol{M}^t+(\boldsymbol{M}^t)^{\mathrm{T}})/2$.

在对每个任务t的一致相似度矩阵$\boldsymbol{M}^t$进行上述操作后，我们再通过对称非负矩阵分解即式(21)来对任务t进行聚类，其求解过程如下.

式(21)的拉格朗日函数为

$$L(\boldsymbol{Y}^t)=\|\boldsymbol{M}^t-\boldsymbol{Y}^t(\boldsymbol{Y}^t)^{\mathrm{T}}\|_F^2-tr(\Delta(\boldsymbol{Y}^t)^{\mathrm{T}}) \tag{22}$$

其中，$\Delta\in\mathbb{R}^{n^t\times k^t}$是拉格朗日乘子.

令$\frac{\partial L(\boldsymbol{Y}^t)}{\partial \boldsymbol{Y}^t}=0$，有

$$\Delta=-2\boldsymbol{M}^t\boldsymbol{Y}^t+2\boldsymbol{Y}^t(\boldsymbol{Y}^t)^{\mathrm{T}}\boldsymbol{Y}^t \tag{23}$$

根据KKT条件$\Delta_{ij}Y_{ij}^t=0$，有

$$(-\boldsymbol{M}^t\boldsymbol{Y}^t+\boldsymbol{Y}^t(\boldsymbol{Y}^t)^{\mathrm{T}}\boldsymbol{Y}^t)_{ij}Y_{ij}^t=0 \tag{24}$$

根据非负矩阵分解的乘法更新规则[6]，任务t中数据的簇指示矩阵$\boldsymbol{Y}^t$的计算公式为

$$Y_{ij}^t\leftarrow Y_{ij}^t\sqrt{\frac{[\boldsymbol{M}^t\boldsymbol{Y}^t]_{ij}}{[\boldsymbol{Y}^t(\boldsymbol{Y}^t)^{\mathrm{T}}\boldsymbol{Y}^t]_{ij}}} \tag{25}$$

在初始化$\boldsymbol{Y}^t$时，MTCFIR遵循传统的非负矩阵分解方法[26]，即用k均值聚类方法初始化$\boldsymbol{Y}^t$，然后设置$\boldsymbol{Y}^t=\boldsymbol{Y}^t+0.2$.这里采用$k$均值聚类方法初始化$\boldsymbol{Y}^t$是因为传统聚类方法计算的$\boldsymbol{Y}^t$要比随机初始化的$\boldsymbol{Y}^t$更好地指示相对正确的簇划分，因此这有利于后续迭代更新的$\boldsymbol{Y}^t$朝着指示正确簇划分的方向计算.但是因为k均值聚类方法得到的$\boldsymbol{Y}^t$中存在零元素，如果初始化的$Y_{ij}^t=0$，根据式(25)会有Y_{ij}^t永远是0，这样式(25)并没有起到迭代更新$\boldsymbol{Y}^t$的作用，所以我们设置$\boldsymbol{Y}^t=\boldsymbol{Y}^t+0.2$来避免$\boldsymbol{Y}^t$中存在零元素.

对称非负矩阵分解聚类步骤的算法流程如算法4所示.

算法4.　对称非负矩阵分解聚类步骤.

输入：每个任务的一致相似度矩阵$\{\boldsymbol{M}^t\}_{t=1}^{\mathcal{T}}$，所有任务的聚类个数$\{k^t\}_{t=1}^{\mathcal{T}}$

输出：簇划分$\{C^t\}_{t=1}^{\mathcal{T}}$.

1. FOR $t=1$ TO $\mathcal{T}$ DO
2. 根据式(25)计算簇指示矩阵$\boldsymbol{Y}^t$.
3. END FOR

3.6　时间复杂度分析

令n为每个任务的样本个数，d为特征个数，k为每个任务的簇个数，$\mathcal{T}$为任务个数，g为共有特征表示学习的层数，$\tilde{I}$为k均值聚类的迭代次数，$\hat{I}$为对称非负矩阵分解的迭代次数.计算共有特征表示下所有任务的数据集H的时间复杂度为$O(\mathcal{T}gd^2n)$.计算一致相似度矩阵$\boldsymbol{M}^t$的时间复杂度为$O(\mathcal{T}^2n^2gd)$.运行对称非负矩阵分解的时间复杂度为$O(\mathcal{T}\tilde{I}kgdn+\mathcal{T}\hat{I}n^2k)$.由于$k$、$\mathcal{T}$、$g$、$\tilde{I}$和$\hat{I}$通常比$n$和$d$小很多，MTCFIR整体的时间复杂度可以简化为$O(n^2d+d^2n)$.

4 实 验

4.1 数据集

现有提出的大部分多任务聚类方法都是基于很严格的假设,这些假设主要呈现在以下两个方面:不同任务具有相同的真实类标签、不同任务具有相同的簇个数.为了充分验证本文提出的 MTCFIR 方法在遵循或违反以上假设的多任务数据集上的聚类性能,本实验构造了三种常见的多任务数据集形式.

(1)任务是完全相关的,即不同任务具有相同的真实类标签且具有相同的簇个数.本实验用 WebKB4① 和 Handdigits② 表示这种形式.

WebKB4 包含来自 4 个大学计算机科学学院网站的网页,这 4 个大学分别是康奈尔大学、德克萨斯大学、华盛顿大学和威斯康星大学.每个大学的网页包含 4 个类标签:课程、职工、工程和学生.因此 WebKB4 具有 4 个任务,每个任务由一个大学的 4 个类标签下的网页构成.

Handdigits 包含来自两个手写体数字数据集 MNIST 和 USPS 的图片.这两个数据集都具有 10 个类标签,即 0 到 9 这 10 个阿拉伯数字.因此 Handdigits 具有 2 个任务,它们分别由 MNIST 和 USPS 中的 10 个类标签下的手写体数字图片构成.

(2)任务是部分相关的(即不同任务具有不完全相同的真实类标签),并且每个任务的簇个数不完全相同.本实验用 20NewsGroups③ 来表示这种形式.

20NewsGroups 是一个新闻文本数据集,它包含 6 个根类,本实验选取 4 个最密集的根类:Comp、Rec、Sci 和 Talk.20NewsGroups 的任务 1 由 Comp.graphics、Rec.auto 和 Sci.crypt 构成,任务 2 由 Comp.os.ms-win.misc、Rec.motocycle、Sci.electronics 和 Talk.politic.mideast 构成,任务 3 由 Comp.sys.ibm.pc.hw、Rec.sport.baseball 和 Sci.med 构成,任务 4 由 Comp.sys.mac.hw、Rec.sport.hockey、Sci.space 和 Talk.religion.misc 构成.

(3)任务是部分相关的(即不同任务具有不完全相同的真实类标签),并且所有任务的簇个数是相同的.本实验用 Reuters④ 来表示这种形式.

Reuters 是一个来自于路透社的新闻文本数据集,它包含 65 个子类下的新闻文档,本实验选取 4 个最密集的根类:经济指数、能源、食物和金属.Reuters 的任务 1 由国民生产总值、金和可可 3 个类构成,任务 2 由居民消费价格指数、天然气和钢铁 3 个类构成,任务 3 由工业生产指数、铜和咖啡 3 个类构成.

以上数据集的详细构造如表 1 所示.

表 1 数据集

数据集	任务	类标签(每类样本数)										维度
WebKB4	1	Cornell.course(44)			Cornell.faculty(34)		Cornell.project(20)			Cornell.student(127)		2500
	2	Texas.course(38)			Texas.faculty(46)		Texas.project(20)			Texas.student(146)		2500
	3	Washington.course(77)			Washington.faculty(31)		Washington.project(21)			Washington.student(126)		2500
	4	Wisconsin.course(85)			Wisconsin.faculty(42)		Wisconsin.project(25)			Wisconsin.student(154)		2500
Handdigits	1	0(509)	1(554)	2(488)	3(514)	4(476)	5(475)	6(473)	7(489)	8(520)	9(502)	256
	2	0(813)	1(685)	2(497)	3(449)	4(449)	5(378)	6(465)	7(436)	8(373)	9(455)	256
20NewsGroups	1	Comp.graphics(387)			Rec.auto(395)		Sci.crypt(395)					3000
	2	Comp.os.ms-win.misc(391)			Rec.motocycle(397)		Sci.electronics(393)			Talk.politic.mideast(376)		3000
	3	Comp.sys.ibm.pc.hw(392)			Rec.sport.baseball(396)		Sci.med(392)					3000
	4	Comp.sys.mac.hw (383)			Rec.sport.hockey(399)		Sci.space(392)			Talk.religion.misc(250)		3000
Reuters	1	Economic index.gnp(63)			Metal.gold(90)		Food.cocoa(53)					5000
	2	Economic index.cpi(60)			Energy.nat gas(33)		Metal.iron steel(37)					5000
	3	Economic index.ipi(36)			Metal.copper(44)		Food.coffee(110)					5000

4.2 对比算法

本实验将提出的 MTCFIR 方法与以下方法进行对比:

(1)传统单任务聚类方法:k 均值聚类(k-means)和对称非负矩阵分解(SNMF)[6].

(2)基于特征表示迁移的多任务聚类方法:共享子空间多任务聚类方法(LSSMTC)[7].

(3)基于实例迁移的多任务聚类方法:智能多任务布雷格曼聚类方法(S-MBC)[12-13]、智能多任务

① http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/

② http://www.cad.zju.edu.cn/home/dengcai/Data/MLData.html

③ http://qwone.com/~jason/20Newsgroups/

④ http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html

核聚类方法(S-MKC)[12-13]和自适应多任务聚类方法(SAMTC)[4].

(4)基于模型参数迁移的多任务聚类方法：判别多任务特征聚类方法(DMTFC)[3]和判别多任务关系聚类方法(DMTRC)[3].

(5)基于特征表示和实例迁移的多任务聚类方法：MTCTKI[2].

(6)MTCFIR 的变体方法：不进行共有特征表示学习的 MTCFIR 方法(MTCFIR-nF)、不进行一致相似度矩阵学习的 MTCFIR 方法(MTCFIR-nI)和不进行任务相关性学习的 MTCFIR 方法(MTCFIR-nR)，即设置 α_s^t 为 1.

4.3　参数设置

本实验利用多任务聚类领域最常用的网格搜索法[7]来确定算法的参数. 本实验设置 MTCFIR 的最近邻个数 l^t 在集合 $ceil\left(\lambda\times\frac{n^t}{k^t}\right)$ 中进行选取，其中 $\lambda=\{0.1,0.3,0.5,0.7,0.9\}$，$ceil(x)$ 表示比 x 大的最小整数. 层数 $g=3$，概率 p 在集合 $\{0.5,0.6,0.7,0.8,0.9\}$ 中进行选取. 本实验计算关联值矩阵 $\boldsymbol{V}^s$ 的相似度度量为余弦相似度. 对于 LSSMTC，参数 λ 搜索集合 $\{0.1,0.2,\cdots,0.9\}$，共享子空间维度搜索集合 $\{2,4,\cdots,10\}$. 对于 S-MBC 和 S-MKC，参数 λ 搜索集合 $\{0.1,0.2,\cdots,1\}$. S-MBC 中的布雷格曼散度为欧几里得距离. 对于 SAMTC，任务内最近邻个数 l_t^t 在集合 $ceil\left(\lambda_1\times\frac{n^t}{k^t}\right)$ 中进行选取，其中 $\lambda_1=\{0.1,0.3,0.5,0.7,0.9\}$，$ceil(x)$ 表示比 x 大的最小整数. 其它任务中最近邻个数 l_s^t 在集合 $ceil\left(\lambda_2\times\frac{\tilde{n}^s}{\tilde{k}^s}\right)$ 中进行选取，其中 $\lambda_2=\{0.1,0.3,0.5,0.7,0.9\}$，$\tilde{n}^s$ 和 $\tilde{k}^s$ 分别是源子任务中的样本个数和簇个数. 对于 DMTFC 和 DMTRC，λ_1 和 λ_2 都搜索集合 $\{2^{-10},2^{-8},\cdots,2^{-2}\}$. 对于 MTCTKI，参数 $\lambda=0.5$，任务内最近邻个数 l^t 在集合 $ceil\left(\mu\times\frac{n^t}{k^t}\right)$ 中进行选取，其中 $\mu=\{0.1,0.3,0.5,0.7,0.9\}$，共享子空间维度在集合 $\{10,20,\cdots,50\}$ 中进行选取. S-MKC、SAMTC 和 MTCTKI 中的高斯核带宽设为数据点的欧几里得距离的中位数.

4.4　聚类性能分析

本文参考现有的多任务聚类文献[7]，采用准确率(Acc)和正则化交互信息(NMI)来评价 MTCFIR 的聚类性能.

由于 DMTFC 和 DMTRC 是凸优化算法，本实验在给定参数范围内的每组参数值下运行它们 1 次，然后报告最佳参数值对应的聚类结果. 对于其它算法，本实验在给定参数范围内的每组参数值下运行它们 10 次，然后报告最佳参数值对应的平均聚类结果和标准差. 这里最佳参数值是指在参数搜索范围内，取得最佳聚类性能的参数值. 目前的多任务聚类方法是根据最佳参数下的聚类性能进行算法聚类性能比较的[7]. 由于 LSSMTC、DMTFC 和 DMTRC 只能处理任务中簇个数相同的情况，因此本实验只在 WebKB4、Handdigits 和 Reuters 上运行它们.

本实验将这些算法的聚类结果展示在表 2、表 3、表 4 和表 5 中. 表中给出的聚类性能 Acc 和 NMI 均省略了百分号，$Acc(1)$ 和 $NMI(1)$ 中括号内的数字代表任务号，即它们分别表示任务 1 的 Acc 和 NMI. “±”符号前面的数字代表平均聚类结果，“±”符号后面的数字代表标准差.

表 2　在 WebKB4 上的聚类结果

方法	$Acc(1)$	$NMI(1)$	$Acc(2)$	$NMI(2)$	$Acc(3)$	$NMI(3)$	$Acc(4)$	$NMI(4)$
k-means	64.00±0.00	18.49±0.01	58.00±5.64	12.11±1.11	51.57±2.67	6.74±3.65	58.27±0.80	20.04±3.24
SNMF	72.89±0.00	37.11±0.00	65.04±3.61	30.55±2.09	71.18±3.34	39.25±2.47	75.42±3.14	46.65±2.25
LSSMTC	63.38±5.18	28.16±4.17	64.36±4.46	23.30±6.97	61.02±5.41	27.74±3.61	65.88±10.16	37.78±6.53
S-MBC	57.91±8.43	24.95±2.80	63.84±7.37	26.97±2.23	57.80±5.38	26.72±3.36	70.85±6.56	39.38±5.45
S-MKC	46.98±2.15	19.81±4.10	45.08±4.51	22.82±4.29	49.25±5.85	23.72±7.21	52.94±3.65	31.38±3.12
SAMTC	71.16±4.34	38.33±4.24	67.48±2.88	35.41±5.74	63.80±5.62	35.78±6.11	72.29±1.46	46.11±2.18
DMTFC	75.11	38.31	68.80	36.80	62.35	30.00	71.24	49.38
DMTRC	40.89	13.56	45.60	16.96	60.01	46.55	55.88	36.91
MTCTKI	68.44±0.00	38.22±0.05	70.56±1.35	37.33±2.27	70.59±0.00	36.80±0.00	77.45±8.27	54.52±2.15
MTCFIR-nF	71.07±0.14	36.82±0.27	72.36±4.49	41.09±4.32	65.73±2.24	38.69±5.15	77.42±0.10	42.77±0.21
MTCFIR-nI	76.53±6.83	48.91±3.07	73.24±4.52	38.80±7.66	72.94±0.00	45.01±0.00	80.16±0.71	55.99±1.02
MTCFIR-nR	78.09±5.85	49.66±3.02	78.36±3.39	49.01±3.07	**74.51±0.00**	46.56±0.00	82.96±3.04	60.21±2.20
MTCFIR	**79.91±1.06**	**50.39±2.54**	**79.08±2.96**	**51.20±4.06**	**74.51±0.00**	**46.83±0.11**	**86.60±0.00**	**62.76±0.20**

表 3 在 Handdigits 上的聚类结果

方法	*Acc*(1)	*NMI*(1)	*Acc*(2)	*NMI*(2)
k-means	51.80±2.81	49.43±1.56	68.65±0.09	63.46±0.08
SNMF	51.14±0.64	49.24±0.20	68.69±0.06	63.44±0.07
LSSMTC	39.09±4.12	31.75±2.46	51.55±4.65	44.59±2.80
S-MBC	39.10±3.22	31.92±2.06	47.85±4.85	40.55±4.12
S-MKC	40.00±3.96	31.60±2.55	11.74±0.20	0.33±0.06
SAMTC	64.83±2.65	62.53±2.31	70.12±3.80	65.08±3.65
DMTFC	23.34	16.23	40.74	32.98
DMTRC	16.72	3.25	34.48	20.90
MTCTKI	62.90±2.34	61.42±0.51	69.23±3.15	64.33±1.52
MTCFIR-nF	60.22±4.53	63.21±0.95	67.24±2.03	63.77±0.74
MTCFIR-nI	51.38±1.89	49.25±0.35	66.92±0.12	61.53±0.13
MTCFIR-nR	59.25±2.65	62.94±0.24	62.70±0.05	57.23±0.05
MTCFIR	**65.18±1.10**	**64.51±2.54**	**70.16±0.74**	**65.25±0.09**

表 4 在 20NewsGroups 上的聚类结果

方法	*Acc*(1)	*NMI*(1)	*Acc*(2)	*NMI*(2)	*Acc*(3)	*NMI*(3)	*Acc*(4)	*NMI*(4)
k-means	33.90±0.00	2.90±0.00	27.26±0.58	7.00±1.58	33.93±0.11	2.89±0.09	28.15±0.14	2.23±0.40
SNMF	80.99±0.00	48.78±0.00	61.84±2.77	41.04±1.11	59.58±0.00	30.02±0.00	83.64±7.55	61.54±10.14
S-MBC	44.06±4.46	14.61±11.98	46.64±5.75	22.51±5.52	43.93±0.89	19.45±3.52	70.88±9.52	40.61±8.56
S-MKC	74.26±1.97	38.53±3.32	63.45±1.98	41.70±3.43	62.83±3.16	30.63±3.44	64.47±5.30	37.19±5.05
SAMTC	75.32±7.81	42.99±6.71	64.65±5.43	42.37±4.60	83.25±1.48	54.64±3.08	73.22±4.15	51.24±4.06
MTCTKI	84.11±0.00	54.55±0.00	74.64±3.57	52.99±0.35	87.54±0.00	61.24±0.00	83.85±0.00	64.17±0.00
MTCFIR-nF	73.07±0.00	33.17±0.00	64.57±0.35	38.75±0.51	60.93±0.03	35.04±0.00	66.29±12.66	38.59±13.22
MTCFIR-nI	86.47±0.21	59.88±0.44	73.80±9.22	56.89±3.54	95.14±0.27	80.07±0.76	80.16±4.68	65.08±3.23
MTCFIR-nR	85.89±0.73	56.54±1.84	69.23±10.71	53.92±3.61	95.11±0.13	80.05±0.42	77.83±12.40	62.59±4.75
MTCFIR	**88.31±0.29**	**62.31±0.65**	**79.97±9.92**	**60.43±4.13**	**95.51±0.13**	**81.25±0.39**	**86.85±1.05**	**67.01±1.45**

表 5 在 Reuters 上的聚类结果

方法	*Acc*(1)	*NMI*(1)	*Acc*(2)	*NMI*(2)	*Acc*(3)	*NMI*(3)
k-means	71.70±4.58	43.42±4.03	65.77±2.38	45.45±5.40	46.74±1.58	20.63±1.54
SNMF	97.57±0.00	89.49±0.00	90.69±5.74	74.72±11.13	91.79±2.45	73.82±4.71
LSSMTC	90.68±6.17	75.24±9.89	90.46±6.12	76.32±7.51	68.63±9.99	47.23±13.34
S-MBC	93.54±7.24	81.79±10.96	86.62±12.69	72.66±13.25	73.16±10.17	56.43±5.66
S-MKC	96.31±1.08	85.87±3.58	90.77±2.81	75.85±2.24	76.58±2.38	54.53±1.68
SAMTC	98.19±0.15	93.17±0.75	74.08±8.82	62.62±5.59	90.11±9.33	76.00±12.55
DMTFC	92.72	81.29	88.46	66.21	79.47	45.51
DMTRC	87.38	67.51	81.54	64.08	61.58	50.49
MTCTKI	98.06±0.00	91.19±0.00	69.23±0.00	62.93±0.00	93.68±0.00	76.08±0.00
MTCFIR-nF	98.06±0.00	92.09±0.00	89.23±12.16	74.26±8.82	93.68±0.00	76.08±0.00
MTCFIR-nI	95.73±0.20	84.15±0.59	91.38±6.20	76.25±9.53	91.05±0.00	68.06±0.00
MTCFIR-nR	96.60±0.00	86.54±0.00	74.38±14.24	56.07±5.79	86.32±0.00	**76.40±0.00**
MTCFIR	**98.54±0.00**	**94.63±0.00**	**95.38±0.00**	**83.46±0.00**	**94.21±0.00**	**76.40±0.00**

从实验结果中可以观察到以下现象：

(1) MTCFIR 比单任务聚类方法 *k*-means 和 SNMF 的聚类性能好，因为 MTCFIR 不仅利用到所有任务的特征和实例知识，还通过学习任务相关性来控制任务之间实例知识迁移的量，从而避免了负面迁移问题，而单任务聚类方法只利用到了自身任务中的知识.

(2) 多任务聚类方法 LSSMTC、S-MBC、S-MKC、SAMTC、DMTFC、DMTRC 和 MTCTKI 比 MTCFIR 的聚类性能差，其原因如下：

① LSSMTC 只通过学习一个共享子空间来迁移特征知识. 其次，它要求所有任务在共享子空间中具有相同的质心，这一要求并不适用于处理具有不同类标签的部分相关任务. 此外，它没有考虑学习任务的相关性，即使任务完全相关，不同任务中具有相同类标签的数据也有可能因为属性值差异过大而没有相关性.

② S-MBC 和 S-MKC 只通过学习任务之间质心的相关性来迁移实例知识. 此外，它要求不同任务共享一部分相同的数据[12-13]，但是测试的数据集并

不满足这一条件.

③ SAMTC 只通过共享最近邻相似度在任务之间迁移实例知识. 此外，它直接忽略了不在子任务中的数据，可能会丢失其它任务中的一些潜在有用信息.

④ DMTFC 只通过学习模型参数在特征维度上的协方差矩阵来迁移模型参数知识. 此外，因为它假设不同任务的模型参数共享相同的高斯先验，所以它更适合处理完全相关的任务.

⑤ DMTRC 只通过学习模型参数在任务维度上的协方差矩阵来迁移模型参数知识. 此外，它假设每个任务的簇标签分布是均匀的，即每个任务中不同类标签下的数据个数相同，但是测试的数据集并不满足这一条件.

⑥ S-MBC、S-MKC、SAMTC、DMTFC 和 DMTRC 没有考虑学习一个特征表示来降低任务之间的分布差异，而是直接在原始空间中迁移实例或模型参数知识，导致负面迁移问题. 这是因为不同任务的分布通常是不同的，致使任务在原始空间中相关性不大.

⑦ MTCTKI 虽然在任务之间同时迁移特征和实例知识，但是它没有考虑任务的相关性强弱. 即使任务完全相关，不同任务中具有相同类标签的数据也有可能因为属性值差异过大而没有相关性.

⑧ MTCFIR 通过学习共有特征表示来降低任务之间的分布差异，因此它可以处理分布差异较大的任务. 其次，它同时在任务之间迁移特征和实例知识，这更充分利用了来自其它任务中的知识. 此外，它在一致相似度矩阵学习步骤中加入了任务相关性学习，可以控制任务之间实例知识迁移的量，从而避免负面迁移问题.

(3) 由于上述原因，LSSMTC、S-MBC、S-MKC、SAMTC、DMTFC、DMTRC 和 MTCTKI 有时甚至表现地比单任务聚类方法的聚类性能差.

(4) MTCFIR-nF 在一些情况下比它的单任务聚类方法 SNMF 的聚类性能差，这说明在原始空间中迁移相关实例会产生负面迁移. 这是因为任务的分布存在差异，导致原始空间中任务相关性不大.

(5) MTCFIR 比 MTCFIR-nF 和 MTCFIR-nI 聚类性能好，因为 MTCFIR 同时利用了任务间的特征和实例知识，这确保了任务间相关知识的充分利用，而 MTCFIR-nF 和 MTCFIR-nI 分别只利用了实例知识和特征知识.

(6) MTCFIR-nR 比 MTCFIR 聚类性能差，这是因为不同任务中的数据并不是完全相关的，学习任务相关性可以控制任务之间实例知识迁移的量，从而避免负面迁移问题. 此外，由实验结果可以看到即使任务完全相关，例如 WebKB4 和 Handdigits，强制迁移所有实例知识也会导致负面迁移问题. 这说明完全相关任务中存在着一些不相关的噪声数据.

综上所述，MTCFIR 没有其它多任务聚类方法的限制条件，实验结果验证了 MTCFIR 的实用性和有效性.

4.5　参数分析

本节分析共有特征表示学习层数 g、近邻参数 λ 和噪声概率 p 对 MTCFIR 的聚类性能的影响.

图 2 和图 3 分别给出了 4 个数据集中共有特征表示学习层数 g 对 MTCFIR 的准确率 Acc 和正则化交互信息 NMI 的影响. 此处的 Acc 和 NMI 计算方式如下：首先固定层数 g；然后令 λ 搜索 $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ 且 p 搜索 $\{0.5, 0.6, 0.7, 0.8, 0.9\}$，得到每对参数 λ 和 p 下的所有任务的平均聚类性能；最后对这些聚类性能加和取平均.

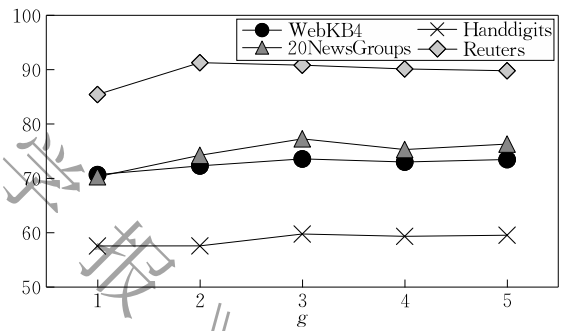

图 2　层数 g 对 MTCFIR 的 Acc 的影响

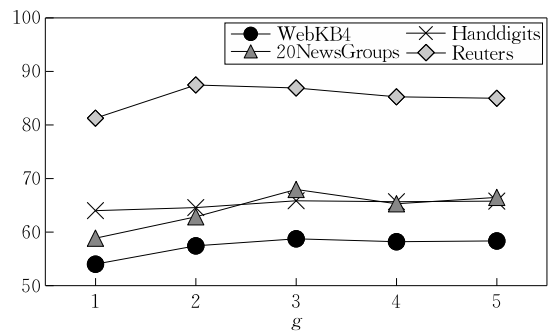

图 3　层数 g 对 MTCFIR 的 NMI 的影响

从图 2 和图 3 可以看出，MTCFIR 的聚类性能随层数 g 的增加呈现先急速上升再缓慢下降的趋势. 急速上升阶段表明共有特征表示学习步骤获得的共有语义特征降低了任务之间的分布差异，因此 MTCFIR 可以从其它任务中挖掘出更多的相关实例参与到一致相似度矩阵学习步骤中. 缓慢

下降阶段是因为过大的层数 g 会升高数据的维度,研究表明高维数据比较稀疏,不利于数据的聚类[27].

综上所述,当 $g=3$ 时,MTCFIR 在 4 个数据集上的聚类性能都相对较好.因此本实验将共有特征表示学习层数 g 统一设置为 3.

图 4、图 5、图 6 和图 7 分别给出了当 $g=3$ 时,4 个数据集中近邻参数 λ 和噪声概率 p 对 MTCFIR 的聚类性能的影响.图中每个方块代表 MTCFIR 在每个参数下所有任务的平均 Acc 和 NMI,颜色越浅的方块代表 MTCFIR 在对应参数下具有越好的聚类性能.

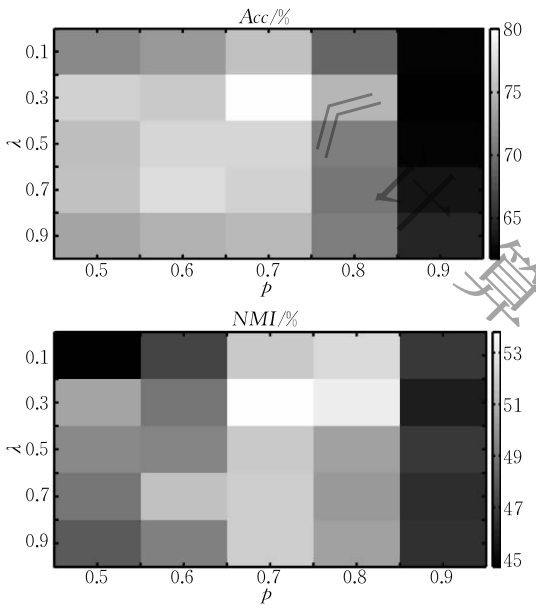

图 4 λ 和 p 对 MTCFIR 在 WebKB4 上的聚类性能影响

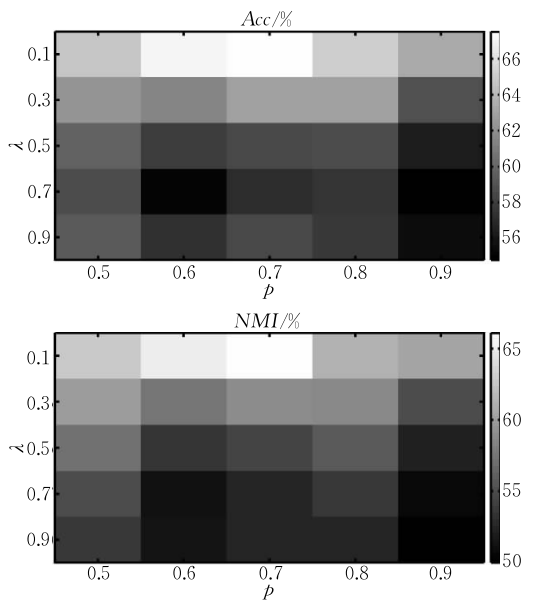

图 5 λ 和 p 对 MTCFIR 在 Handdigits 上的聚类性能影响

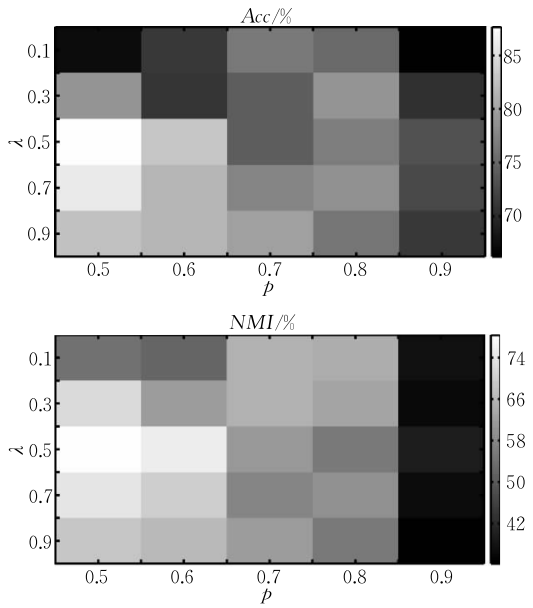

图 6 λ 和 p 对 MTCFIR 在 20NewsGroups 上的聚类性能影响

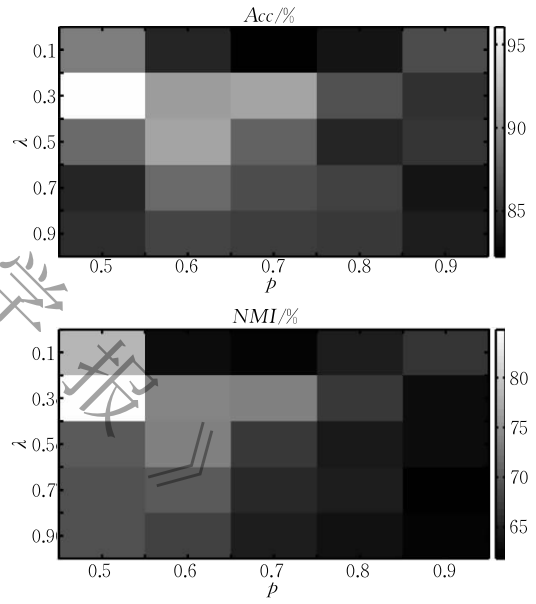

图 7 λ 和 p 对 MTCFIR 在 Reuters 上的聚类性能影响

从图 4、图 5、图 6 和图 7 中,我们可以观察到以下现象.

(1) 在噪声概率 $p=0.9$ 时,MTCFIR 的聚类性能很差.这说明过大的噪声概率并不利于 MTCFIR 获得好的聚类性能.

(2) 当 $\lambda\in\{0.1,0.3,0.5\}$ 和 $p\in\{0.5,0.6,0.7\}$ 时,MTCFIR 更容易获得比较好的聚类性能.

4.6 任务关联系数分析

首先我们分析一下本实验采用的 4 个数据集的理想任务关联系数.这里任务 s 相对于任务 t 的理想任务关联系数为任务 t 中与任务 s 共享类标签的数据比例乘以任务 s 中与任务 t 共享类标签的数据比例.

因此，对于完全相关任务 WebKB4 和 Handdigits，任意两个任务之间的相关系数均为 1. 对于部分相关任务 20NewsGroups 和 Reuters，它们的任务关联系数如表 6 和表 7 所示.

表 6　20NewsGroups 的理想任务关联系数

任务 t	任务 s			
	任务 1	任务 2	任务 3	任务 4
任务 1	1	0.76	1	0.82
任务 2	0.76	1	0.76	1
任务 3	1	0.76	1	0.82
任务 4	0.82	1	0.82	1

表 7　Reuters 的理想任务关联系数

任务 t	任务 s		
	任务 1	任务 2	任务 3
任务 1	1	0.55	1
任务 2	0.55	1	0.31
任务 3	1	0.31	1

下面我们给出 MTCFIR 算法根据式(20)计算出的 4 个实验数据集的任务关联系数，如表 8、表 9、表 10 和表 11 所示. 为了方便与理想任务关联系数比较，我们令 $\alpha_t^t=1$，然后将任务 s 相对于任务 t 的任务关联系数 $\alpha_s^t(s=1,\cdots,\mathcal{T})$ 同比例扩大. 注意这里同比例扩大任务关联系数不会改变一致相似度矩阵 $\boldsymbol{M}^t$ 的计算结果，因为式(14)和(16)中的分母 β 以及分子 A_{ij} 与 B_{ij} 都是同比例放大的.

表 8　MTCFIR 计算的 WebKB4 的任务关联系数

任务 t	任务 s			
	任务 1	任务 2	任务 3	任务 4
任务 1	1	0.39	0.30	0.37
任务 2	0.29	1	0.13	0.19
任务 3	0.53	0.37	1	0.40
任务 4	0.55	0.42	0.34	1

表 9　MTCFIR 计算的 Handdigits 的任务关联系数

任务 t	任务 s	
	任务 1	任务 2
任务 1	1	0.21
任务 2	0.96	1

表 10　MTCFIR 计算的 20NewsGroups 的任务关联系数

任务 t	任务 s			
	任务 1	任务 2	任务 3	任务 4
任务 1	1	0.53	0.36	0.35
任务 2	0.81	1	0.58	0.50
任务 3	0.56	0.57	1	0.60
任务 4	0.57	0.53	0.65	1

表 11　MTCFIR 计算的 Reuters 的任务关联系数

任务 t	任务 s		
	任务 1	任务 2	任务 3
任务 1	1	0.43	0.18
任务 2	0.24	1	0.29
任务 3	0.15	0.44	1

多任务数据集的理想任务关联系数是从任务间共享类标签的数据比例的角度来计算的. 而 MTCFIR 算法的任务关联系数是从任务间具有较高相似度的数据比例的角度来计算的. 从表 6 到表 11 可以看出，MTCFIR 计算的任务关联系数整体上要比理想任务关联系数小. 这是因为虽然有些任务间的数据共享相同的类标签，但是由于它们的属性值差异过大而具有很低的相似度. 总体上来说，MTCFIR 计算的任务关联系数要比理想任务关联系数更严格一些.

实际上，采用 MTCFIR 计算的任务关联系数要比采用理想任务关联系数更容易获得较好的聚类效果，我们可以从表 2 和表 3 中 MTCFIR-nR 和 MTCFIR 的聚类结果得到验证. 虽然 WebKB4 和 Handdigits 中的任务是完全相关的，但是采用理想任务关联系数的 MTCFIR-nR(α_s^t 均为 1)要比 MTCFIR 聚类性能差.

4.7　任务分布差异分析

为了验证共有特征表示学习步骤学习到的新特征可以降低任务之间的分布差异，本实验采用最大平均差异分布度量[28]来分别计算原始特征表示下和共有特征表示下学习步骤中新特征表示下的任务分布距离，如表 12 所示. 从表 12 中我们可以看出共有特征表示学习步骤学习到的新特征确实降低了任务间的分布差异.

表 12　原始任务和共有特征表示下的任务分布距离

数据集	原始特征表示下的任务分布距离	共有特征表示下的任务分布距离
WebKB4	0.2445	0.0576
Handdigits	0.5347	0.0660
20NewsGroups	0.0368	0.0092
Reuters	0.1443	0.0374

4.8　运行时间

本实验调研 MTCFIR 和其它多任务聚类方法的运行时间，如图 8 所示. 由于 LSSMTC、DMTFC 和 DMTRC 只能应用在簇个数相同的任务上，因此本实验只给出它们在 WebKB4、Handdigits 和 Reuters 数据集上的运行时间.

从图 8 中可以看出 S-MBC 是运行最快的方法. LSSMTC 的运行时间与特征个数的平方成正比，MTCTKI 和 MTCFIR 的运行时间与特征个数和样

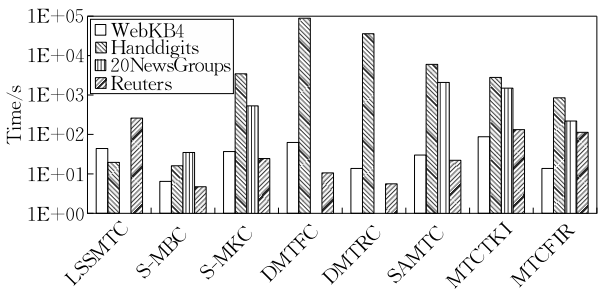

图 8 运行时间

本个数的平方成正比. S-MKC 和 SAMTC 的运行时间与样本个数的平方成正比. DMTFC 和 DMTRC 的运行时间与样本个数的立方成正比. 总体上，MTCFIR 在效率上和其它算法相比很有竞争力.

5 结 论

本文提出了一个基于特征和实例迁移的加权多任务聚类方法 MTCFIR，它不仅可以在任务之间同时迁移特征表示和实例知识，还可以自动学习任务关联性来避免负面迁移问题. 首先，MTCFIR 利用边缘堆栈降噪自编码器为所有任务学习一个共有特征表示. 然后，MTCFIR 利用任务之间的实例知识为每个任务学习一个一致相似度矩阵，同时它通过学习任务关联性来对任务进行加权，控制该一致相似度矩阵从其它任务中获取的实例知识量. 最后，MTCFIR 为每个任务的一致相似度矩阵进行对称非负矩阵分解聚类. 在多个真实数据集上的实验结果验证了 MTCFIR 要比单任务聚类算法和现有多任务聚类算法具有更好的聚类性能. 本文提出的 MTCFIR 算法依次执行特征表示迁移、实例迁移和任务聚类. 未来我们将提出一个将这三者整合的多任务聚类框架，从而通过交替迭代的优化方法来取得更好的聚类性能.

参 考 文 献

[1] Pan S J, Yang Q. A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 2010, 22(10): 1345-1359

[2] Zhang Xiaotong, Zhang Xianchao, Liu Han, Liu Xinyue. Multitask clustering through instances transfer. Neurocomputing, 2017, 251: 145-155

[3] Zhang Xiao-Lei. Convex discriminative multitask clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(1): 28-40

[4] Zhang Xianchao, Zhang Xiaotong, Liu Han. Self-adapted multi-task clustering//Proceedings of the 25h International Joint Conference on Artificial Intelligence. New York, USA, 2016: 2357-2363

[5] Chen Minin, Xu Zhixiang Eddie, Weinberger Kilian Q, Sha Fei. Marginalized denoising autoencoders for domain adaptation//Proceedings of the 29th International Conference on Machine Learning. Edinburgh, UK, 2012: 1627-1634

[6] Kuang D, Park H, Ding C H Q. Symmetric nonnegative matrix factorization for graph clustering//Proceedings of the 12th SIAM International Conference on Data Mining. Anaheim, USA, 2012: 106-117

[7] Gu Quanquan, Zhou Jie. Learning the shared subspace for multi-task clustering and transductive transfer classification //Proceedings of the 9th International Conference on Data Mining. Miami, USA, 2009: 159-168

[8] Gu Quanquan, Li Zhenhui, Han Jiawei. Learning a kernel for multi-task clustering//Proceedings of the 25th AAAI Conference on Artificial Intelligence. San Francisco, USA, 2011: 368-373

[9] Zhang Zhihao, Zhou Jie. Multi-task clustering via domain adaptation. Pattern Recognition, 2012, 45(1): 465-473

[10] Xie Saining, Lu Hongtao, He Yangcheng. Multi-task co-clustering via nonnegative matrix factorization//Proceedings of the 21st International Conference on Pattern Recognition. Tsukuba, Japan, 2012: 2954-2958

[11] Zhang Jianwen, Zhang Changshui. Multitask Bregman clustering//Proceedngs of the 24th AAAI Conference on Artificial Intelligence. Atlanta, USA, 2010: 655-660

[12] Zhang Xianchao, Zhang Xiaotong. Smart multi-task Bregman clustering and multi-task kernel clustering//Proceedngs of the 27th AAAI Conference on Artificial Intelligence. Bellevue, USA, 2013: 1034-1040

[13] Zhang Xianchao, Zhang Xiaotong, Liu Han. Smart multitask Bregman clustering and multitask kernel clustering. ACM Transactions on Knowledge Discovery and Data, 2015, 10(1): 8

[14] Al-Stouhi S, Reddy C K. Multi-task clustering using constrained symmetric non-negative matrix factorization//Proceedings of the 14th SIAM International Conference on Data Mining. Philadephia, USA, 2014: 785-793

[15] Yang Yang, Ma Zhigang, Yang Yi, et al. Multitask spectral clustering by exploring intertask correlation. IEEE Transactions on Cybernetics, 2015, 45(5): 1069-1080

[16] Zhang Yu, Yang Qiang. An overview of multi-task learning. National Science Review, 2018, 5: 30-43

[17] Li Ya, Tian Xinmei, Liu Tongliang, Tao Dacheng. Multi-task model and feature joint learning//Proceedings of the 24th International Joint Conference on Artificial Intelligence. Buenos Aires, Argentina, 2015: 3643-3649

[18] Thrun S, O'Sullivan J. Discovering structure in multiple learning tasks: The TC algorithm//Proceedings of the 13th

International Conference on Machine Learning. Bari, Italy, 1996: 489-497

[19] Bakker B, Heskes T. Task clustering and gating for Bayesian multitask learning. Journal of Machine Learning Research, 2003, 4: 83-99

[20] Zhang Yu, Yeung Dit-Yan. A convex formulation for learning task relationships in multi-task learning//Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence. Catalina Island, USA, 2010: 733-742

[21] Zhang Yu, Yeung Dit-Yan. A regularization approach to learning task relationships in multitask learning. ACM Transactions on Knowledge Discovery from Data, 2013, 8(3): 12

[22] Lee G, Yang E, Hwang S J. Asymmetric multi-task learning based on task relatedness and loss//Proceedings of the 33rd International Conference on Machine Learning. New York, USA, 2016: 230-238

[23] Vincent P, Larochelle H, Bengio Y, Manzagol P-A. Extracting and composing robust features with denoising autoencoders//Proceedings of the 25th International Conference on Machine Learning. Helsinki, Finland, 2008: 1096-1103

[24] Golub G H, Van Loan C F. Matrix Computations. 3rd Edition. Baltimore, USA: Johns Hopkins University Press, 1996

[25] Nie Feiping, Wang Xiaoqian, Jordan M I, Huang Heng. The constrained Laplacian rank algorithm for graph-based clustering//Proceedings of the 30th AAAI Conference on Artificial Intelligence. Phoenix, USA, 2016: 1969-1976

[26] Ding C H Q, Li Tao, Peng Wei, Park H. Orthogonal non-negative matrix t-factorizations for clustering//Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Philadelphia, USA, 2006: 126-135

[27] Kriegel H-P, Kröger P, Zimek A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Transactions on Knowledge Discovery from Data, 2009, 3(1): 1

[28] Gretton A, Borgwardt K M, Rasch M J, et al. A kernel method for the two-sample problem//Proceedings of the 20th Annual Conference on Neural Information Processing Systems. Vancouver, Canada, 2006: 513-520

ZHANG Xiao-Tong, Ph. D. Her current research interests include multi-task clustering and multi-task learning.

ZHANG Xian-Chao, Ph. D., professor. His current research interests include data mining and machine learning.

LIU Han, Ph. D. His current research interests include uncertain data mining and multi-task clustering.

Background

Multi-task clustering has received increasing attention in recent years. It improves the clustering performance of each task by transferring knowledge across related tasks. There are mainly two issues to be studied in multi-task clustering. One issue is which kind of knowledge to be transferred among the tasks. The other issue is how to assess the task relatedness to avoid negative transfer.

For the first issue, existing multi-task clustering methods usually transfer one kind of knowledges such as feature representation, instance or model parameter among the tasks. Only MTCTKI transfers the knowledge of both feature representation and instances among the tasks, which can take advantage of more related knowledge among the tasks than transferring only one kind of knowledge.

For the second issue, there are only two multi-task clustering methods which can automatically assess the task relatedness, but they have some limitations. (1) DMTRC learns the task relatedness through Gaussian prior. But it is based on a strict assumption that all the tasks have the same cluster number and the label marginal distribution in each task distributes evenly. (2) SAMTC learns a pair of possibly related subtasks for each pair of tasks, then assesses task relatedness of the subtasks.

But it directly discards the data points that are considered useless, which may miss some potentially useful information in the other tasks.

In this paper, we propose a weighted multi-task clustering by feature and instance transfer method called MTCFIR, which not only make full use of the related knowledge by transferring both the feature representation and instance

knowledge among the tasks, but also automatically learns the task relatedness to avoid negative transfer. MTCFIR executes the following three steps. (1) Common feature representation learning: it learns a common feature representation among the tasks with marginalized stacked denoising autoencoders. This step transfers the feature representation knowledge to reduce the distribution difference among the tasks, which is the premise of learning the consistent similarity matrix. (2) Consistent similarity matrix learning: it learns a consistent similarity matrix for each task by transferring the instance knowledge across the tasks. Meanwhile the task relatedness is automatically learned to determine the contribution degree of different tasks for learning the consistent similarity matrix. (3) Symmetric nonnegative matrix factorization: it performs a symmetric nonnegative matrix factorization on the consistent similarity matrix to get the clustering results of each task.

This research was supported by the National Natural Science Foundation of China (No. 61632019). Our group has studied multi-task clustering for many years and published a lot of papers about multi-task clustering. Our group mainly studied two issues in multi-task clustering: how to transfer knowledge among the tasks and how to deal with partially related tasks to avoid negative transfer.