Knowledge-Based Systems 000 (2016) 1–15

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Multi-bridge transfer learning

Xuegang Hu*, Jianhan Pan*, Peipei Li, Huizong Li, Wei He, Yuhong Zhang

*Hefei University of Technology, Hefei, 230009, China*

ARTICLE INFO

*Article history:*
Received 5 March 2015
Revised 8 October 2015
Accepted 12 January 2016
Available online xxx

*Keywords:*
Transfer learning
Non-negative matrix tri-factorization
Multi-bridge
Cross-domain classification

ABSTRACT

Transfer learning, which aims to exploit the knowledge in the source domains to promote the learning tasks in the target domains, has attracted extensive research interests recently. The general idea of the previous approaches is to model the shared structure in one latent space as the bridge across domains by reducing the distribution divergences. However, there exist some latent factors in the other latent spaces, which can also be utilized to draw the corresponding distributions closer for establishing the bridges. In this paper, we propose a novel transfer learning method, referred to as Multi-Bridge Transfer Learning (MBTL), to learn the distributions in the different latent spaces together. Therefore, more latent factors shared can be utilized to transfer knowledge. Additionally, an iterative algorithm with convergence guarantee based on non-negative matrix tri-factorization techniques is proposed to solve the optimization problem. Comprehensive experiments demonstrate that MBTL can significantly outperform state-of-the-art learning methods on the topic and sentiment classification tasks.



## 1. Introduction

Traditional machine learning classification algorithms implicitly assume that the training and test data are drawn from the same distribution. However, this assumption seldom holds in reality. To tackle the challenge of different data distributions, many transfer learning methods have been proposed recently for real-world applications, such as text classification [13], computational biology [11] and image classification [12]. The key idea of transfer learning or domain adaptation is to exploit the labeled examples in the source domain to model a better classifier for predicting the classes of the test examples in the target domain where exist less or no labeled examples. Some of these previous approaches show that features on raw words are not reliable for text classification in cross-domain learning. For example, when the documents which belong to the category of "computer" are drawn from the domains "hardware" and "software", the words in these documents indicating the concept of "computer technology" can be "keyboard", "CPU", "operating system", "programmer", and so on. However, the frequencies of these words may be different in different domains. In the domain of "software", high-frequency words are "operating system", "programmer", etc., while the words like "keyboard" and "CPU" are the high-frequency ones in the domain of "hardware". Since these original features can not be shared directly, only the high-level concept "computer technology", which is extracted from these words, can be utilized to distinguish the category of "computer" across domains. Therefore, the latent high-level concepts, which are related to feature clusters extracted on the raw features, are more appropriate for the text classification across domains than learning from the original features [7]. CoCC [3] learns the identical concept. MTrick [4] exploits the association between the homogenous concept and the example classes as the bridge across domains. DTL [5] models the shared concepts including the identical and homogenous concepts to establish the bridge. In addition, Tri-TL [6] and HIDC [7] exploit the distinct concept to training classifier besides the shared concepts.

These previous methods usually build one bridge across domains by constructing a transformed high-level feature space and reducing the corresponding distribution divergences. We represent such method as the single bridge transfer learning. The limitation of the single bridge approaches is two-fold. (1) A set of latent factors in one latent feature space is just a subset of all the latent factors. The assumption that all the shared factors only exist in one latent feature space cannot hold in reality and may ignore the latent factors existing in the other latent feature spaces, which may also help to model the shared structure across domains. (2) To transfer knowledge, these methods exploit the latent factors in one latent feature space to learn the corresponding distributions in this latent feature space. However, not all these latent factors can be utilized to draw the corresponding distributions in the latent feature space closer. Some of the latent factors which are also useful for knowledge transfer may represent the discrepancy between

* Corresponding author. Tel.: +86 13856988808 (Xuegang Hu). +86 13170017991 (Jianhan Pan).
*E-mail addresses:* jsjxhuxg@gmail.com (X. Hu), peter.jhpan@gmail.com (J. Pan).

http://dx.doi.org/10.1016/j.knosys.2016.01.016
0950-7051/

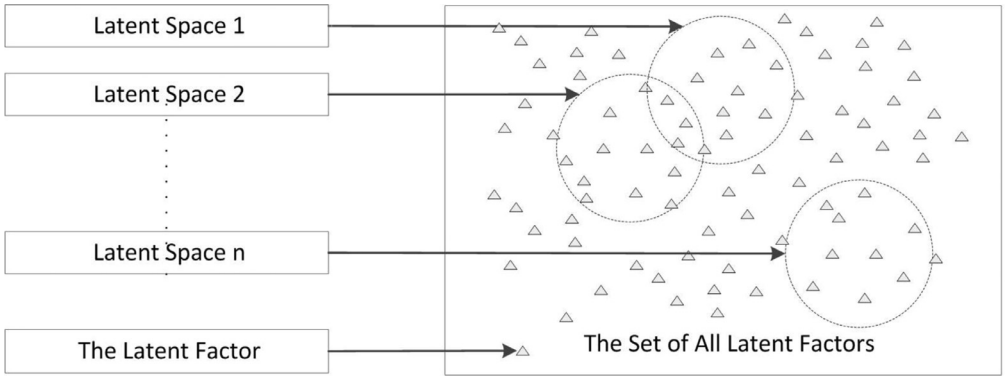



**Fig. 1.** Distributions of latent factors in the different latent feature spaces.

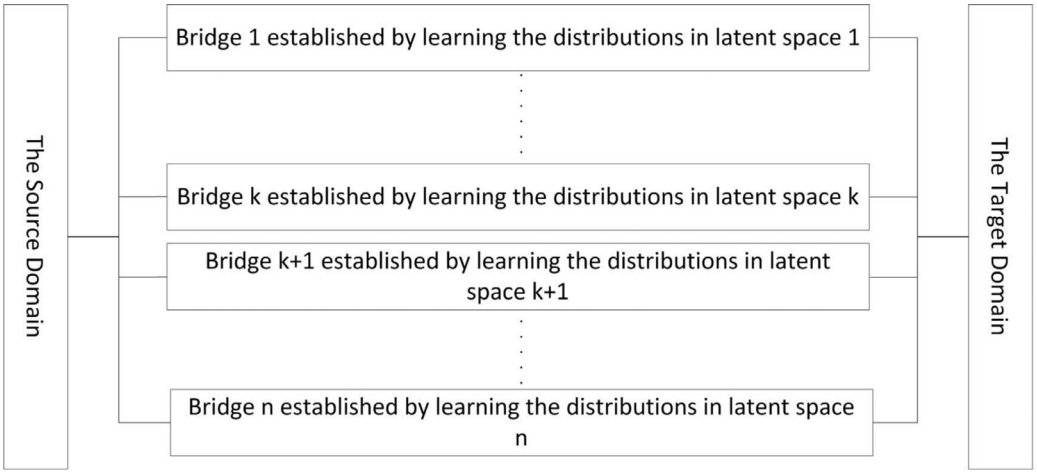



**Fig. 2.** Multiple bridges built by learning distributions in the different latent feature spaces.

the distributions across domains. In the worst case, when the divergences of the distribution in one latent space are so large and there exist some latent factors, which cannot be used to draw the corresponding distributions in the latent feature space closer, utilized to learn the distributions, these single bridge methods will happen negative transfer.

In this paper, we propose Multi-Bridge Transfer Learning (MBTL), a novel transfer learning method based on non-negative matrix tri-factorization (NMTF) techniques, which constructs multiple latent feature spaces, and learns the corresponding distributions in the different latent spaces simultaneously. The key idea of MBTL is as follows. Firstly, as shown in Fig. 1, by constructing multiple different latent feature spaces, more latent factors can be exploited to model the shared structure across domains. Secondly, since MBTL learns the distributions in the different latent spaces simultaneously to establish multiple bridges across domains as shown in Fig. 2, the latent factors in the different latent feature spaces can be used to reduce the distribution divergences in the different latent feature spaces respectively. Additionally, when Some of latent factors in one latent feature space represent the discrepancy between the distributions across domains in this latent space, they can also be utilized to reduce the distribution discrepancies in the other latent feature spaces.

The main contributions of MBTL are summarized as follows:

(1) Motivated by the observation that the latent factors usually exist in different latent feature spaces, we propose the MBTL method to utilize the latent factors to reduce the distribution divergences in different latent feature spaces simultaneously.
(2) To solve the proposed method MBTL, we present an iterative algorithm with convergence guarantee based on non-negative matrix tri-factorization techniques.
(3) We construct the systematic experiments to show the effectiveness of MBTL. In particular, all the compared topic-oriented methods happen negative transfer on data set 20-Newsgroups frequently and can not compete with the traditional machine learning method Logistic Regression (LR) on the sentiment tasks. Only MBTL seldom occurs negative transfer and obtains the best performance on all the tasks.

The rest of this paper is organized as follows. Section 2 introduces the related work. We review preliminary knowledge in Section 3. In Section 4, we describe the model of MBTL. Section 5 provides the experimental results. Finally, Section 6 concludes this paper.

## 2. Related works

In this section, we discuss some previous transfer learning methods. Recently, transfer learning techniques have been applied in many real-world applications. In [3–7], transfer learning techniques are proposed to learn text data across domains. In [11,28–30], transfer learning techniques are applied to biological fields. [12,31] are proposed for computer vision and image processing. In [39–41], transfer learning techniques are proposed to solve collaborative filtering problems. According to literature survey [1,27], most previous methods can be divided into five categories including the computational intelligence-based methods, the self-labeling methods, the parameter-based methods, the weighting-based methods and the feature representation-based methods.

Please cite this article as: X. Hu et al., Multi-bridge transfer learning, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.01.016

Computational intelligence-based transfer learning methods relate to various Technical fields including neural network, Bayes and fuzzy system. Swietojanski et al. [32] used restricted Boltzmann machine to pre-train deep neural network, and the outputs of the network are utilized as features for a hidden Markov model. Roy and Kaelbling [33], proposed an alternative method of transferring the naïve Bayes classifier. Behbood et al. [34,35] developed a fuzzy-based transductive transfer learning method for long term bank failure prediction, in which the distribution of data in the source domain differs from that in the target domain.

Self-labeling methods which are closely related to the Expectation Maximization (EM) algorithm include unlabeled target domain examples in the training process and initialize their labels and then iteratively refine the labels. Tan et al. [36] proposed Adapted naïve Bayes (ANB), a weighted transfer version of naïve Bayes Classifier. Dai et al. [37] proposed a novel transfer-learning algorithm for text classification based on an EM-based naïve Bayes classifiers.

Parameter-based approaches assume that the source and target tasks share some parameters or prior distributions of the hyper parameters of the models. Gao et al. [20] developed a dynamic model weighting approach for each test data according to the similarity between the local structure in the target domain and the classification model. Dredze et al. [38] proposed a new multi-domain online learning framework based on parameter combination from multiple classifiers for a new target domain.

Weighting-based approaches focus on a re-weighting strategy [19,25]. The general idea for these methods is to increase the weight of the data in source domain which is close to the data in the target domain; otherwise, decrease the weight. From the view of Instance weighting, Jiang and Zhai [25] proposed a general framework based on instance weighting to deal with NLP tasks. Dai et al. [19] extended boosting-style learning algorithm for cross-domain learning by the re-weighting strategy.

Our work belongs to the feature representation-based methods, which can be grouped into two subcategories further, including the feature selection-based and feature mapping-based approaches. The general strategy of feature selection-based approaches is to select the general features which are useful for domain adaptation from the raw features among different domains [3,21,22]. Dai et al. [3] proposed a coclustering-based method, which identified the word clusters among the source and target domains, by propagating the class information and knowledge from the source domain to the target domain. Jiang and Zhai [21] developed a two-step feature selection framework for knowledge transfer with the strategy that the features highly related to class labels should be assigned to large weights in the learnt model. Uguroglu and Carbonell [22] proposed an approach to identify variant and invariant features between two data sets for transfer learning. On the other side, feature mapping-based approaches, to which MBTL belongs, map the raw feature space to a latent high-level feature space, under a guidance of the principle that the source and the target domains are drawn from the same distribution [2,4–7,23,24]. Blitzer et al. [2] presented a correspondence Learning method, which first identifies the correspondence among features and then explores this correspondence for knowledge transfer. Zhuang et al. [4] proposed a method to exploit the associations between feature clusters and example clusters for cross-domain learning. Long et al. [5] proposed a method, Dual Transfer Learning (DTL), which utilized the duality between the marginal and conditional distributions. In [6], the proposed Triplex Transfer Learning (Tri-TL) based on nonnegative matrix tri-factorization (NMTF) analyzes the three kinds of concepts (identical, homogeneous and distinct concepts), then models them together. And the proposed method HIDC [7] is similar to Tri-TL but considering the distribution of domains. Pan et al. [23] proposed an algorithm to find out the latent feature space based on dimensionality reduction. Long et al. [24] proposed a method based on property preservation, which extracts shared latent factors between domains by preserving the important geometric structure properties of the original data.

The common strategy of these feature mapping-based approaches is to establish one bridge across domains for knowledge transfer by extracting one transformed latent space, in which the distributions of the source and target data are drawn close. However, these single bridge methods ignored that some latent factors, which may also help to model the shared structure, may represent the distribution divergencies across domains in the one latent space. Along this line, the Multi-Bridge Transfer Learning (MBTL) method is proposed. This model utilizes the latent factors to reduce the distribution divergences in different latent feature spaces simultaneously. Moreover, we demonstrate the effectiveness of MBTL compared with the existing approaches by extensive experimental evaluation.

**Table 1**
Notations and descriptions.

| Notations | Descriptions |
|---|---|
| $\mathcal{D}_r$ | Domain $r$ |
| $r$ | Domain index |
| $s$ | The number of source domains |
| $t$ | The number of target domains |
| $X_r$ | The feature-example co-occurrence matrix of $\mathcal{D}_r$ |
| $U$ | The matrix of feature clusters |
| $H$ | The matrix of the association between feature clusters and example classes |
| $V$ | The matrix of example classes |
| $m$ | The number of original features |
| $n_r$ | The number of examples in domain $\mathcal{D}_r$ |
| $c$ | The number of example classes |
| $k_a$ | The number of identical concepts |
| $k_b$ | The number of homogeneous concepts |
| $k$ | The number of high-level feature clusters (concepts) |
| $T$ | Transposition of matrix |

## 3. Preliminary knowledge

In this section, we first give some basic concepts and mathematical notations used in this paper, and, then introduce the nonnegative matrix tri-factorization (NMTF) model briefly.

### 3.1. Notations and basic concepts

In this paper, we represent data Matrix with uppercase, such as $X$ and $Y$, and denote the element at the $i$th row and $j$th column of matrix $X$ as $X_{[i,j]}$. Then, we use $1_m$ to represent a vector with $m$ rows and one column, and each element in this vector is equal to 1. Then, the set of real numbers and nonnegative real numbers are denoted as $R$ and $R_+$ respectively. Let $\mathcal{D} = (\mathcal{D}_1, \ldots, \mathcal{D}_s, \mathcal{D}_{s+1}, \ldots, \mathcal{D}_{s+t})$ be a family of data domains which has $s$ source domains and $t$ target domains. The first $s$ domains are source domains denoted as $\mathcal{D}_r = \{x_i^r, y_i^r\}|_{i=1}^{n_r} (1 \le r \le s)$, where $y_i^r$ is the label of the example $x_i^r$ , and the rest $t$ domains are target domains denoted as $\mathcal{D}_r = \{x_i^r\}|_{i=1}^{n_r} (s+1 \le r \le s+t)$ that the documents are unlabeled. The variables $r$ and $n_r$ represent the index of domains and the number of examples in the domain $\mathcal{D}_r$ respectively. For each domain $\mathcal{D}_r$ $(1 \le r \le s+t)$, we represent the feature-example co-occurrence matrix as $X_r \in R_+^{m \times n_r}$ under the assumption that all the elements of the matrix $X_r$ which involves $m$ features and $n_r$ examples are non-negative. In Table 1, we summarize the frequently-used notations in MBTL.

### 3.2. Non-negative matrix tri-factorization

Since the nonnegative matrix tri-factorization (NMTF) model has been widely used for text data classification [3–7,10,13,14], we briefly introduce NMTF, on which our model is based. In NMTF,

the matrix $X \in R^{m \times n}$ approximates a product of three nonnegative factors $U \in R^{m \times k}$, $H \in R^{k \times c}$, and $V \in R^{n \times c}$, such that $X \approx UHV^T$. Then we can obtain the basic formula as follows

$$X_{m\times n} = U_{m\times k} H_{k\times c} V^T_{n\times c} \tag{1}$$

where $X \in R^{m \times n}$ represents the feature-example matrix. $m$, $n$, $k$, $c$ represents the numbers of raw features, examples, high-level feature clusters, and example classes respectively. $U \in R^{m \times k}$ represents a feature-cluster matrix owning $k$ feature clusters and $m$ raw features which actually means a mapping that from raw features to high-level features. The high-level feature clusters can be seen to the probability distributions over $m$ raw features. $V \in R^{n \times c}$ represents an example-class matrix owning $c$ classes and $n$ examples which refers to the mapping that from examples to classes. The class clusters in $V$ can be described as the probability distributions over $n$ examples. $H \in R^{k \times c}$ represents a cluster-class matrix owning $k$ high-level feature clusters and $c$ classes which means the association between high-level feature clusters and examples classes. The class clusters in $H$ can be viewed as the probability distributions over $k$ high-level feature clusters.

Additionally, NMTF which can be solved by an iterative update algorithm is an optimization problem, and we show the formula as follows

$$\begin{aligned} &\min_{U,H,V\geq 0} \| X - UHV^T \|^2 \\ &s.t. \sum_{i=1}^{m} U_{[i,j]} = 1, \sum_{j=1}^{c} V_{[i,j]} = 1 \end{aligned} \tag{2}$$

To adapt transfer learning, NMTF is extended to multiple domains. The formula of the NMTF optimization problem can be rewritten as

$$\begin{aligned} &\min_{U_r H, V_r \geq 0} \sum_{r=1}^{s+t} ||X_r - U_r H V_r^T||^2 \\ &s.t. \sum_{i=1}^{m} U_{r[i,j]} = 1, \quad \sum_{j=1}^{c} V_{r[i,j]} = 1 \end{aligned} \tag{3}$$

Here, we also introduce some basic concepts about NMTF used in Section 4

**Definition 3.1** (Trace of matrix)**.** Given a data matrix $X \in R^{n \times n}$, the trace of X is computed as

$$\mathrm{Tr}(X) = \sum_{i=1}^{n} X_{[ii]} \tag{4}$$

In fact, we can also compute the trace of matrix square when the matrix is a square matrix.

**Definition 3.2** (Frobenius norm of matrix)**.** Given a data matrix $X \in R^{m \times n}$, the Frobenius norm of X is computed as

$$||X||^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} X^2_{[ij]} \tag{5}$$

Next, we list some conversion formulas of the trace of matrix.

**Property 3.1.** Given a matrix $X \in R^{m \times n}$, then

$$\mathrm{Tr}(X^T X) = \mathrm{Tr}(XX^T) \tag{6}$$

**Property 3.2.** Given two matrices $X$, $Y \in R^{m \times n}$, then

$$\mathrm{Tr}(a \cdot X + b \cdot Y) = a \cdot \mathrm{Tr}(X) + b \cdot \mathrm{Tr}(Y) \tag{7}$$

**Property 3.3.** Given a matrix $X \in R^{m \times n}$, then

$$||X||^2 = \mathrm{Tr}(X^T X) = \mathrm{Tr}(XX^T) \tag{8}$$

**Table 2**
categories of concepts in one latent feature space.

| Notation | Description |
|---|---|
| Identical concept | A identical concept has the same extension and the same intension across domains. |
| Homogeneous concept | A homogeneous concept has the different extension and the same intension across domains. |

## 4. Multi-bridge transfer learning

In this section, we first define the problem setting and formulate the MBTL model as an optimization problem. After that, we introduce the solution to the model and the proof of algorithm convergence.

### 4.1. Problem definition

We first analyze the high-level concepts (i.e., feature clusters and topics), and then lay out two kinds of explanations of a concept with different perspectives.

**Definition 4.1** (Concept extension)**.** The extension of a high-level concept $z$ is represented by a multinomial distribution $p(w|z)$ over raw features.

It indicates the degree of applicability of each word $w$ for that concept $z$ [7]. In this study, we utilize matrix $U$ to represent Concept Extension [6].

**Definition 4.2** (Concept intension)**.** The Intension of a high-level concept $z$ is represented by its association with each document class $y$, denoted by the conditional probability $p(z|y)$.

It indicates that when $p(z|y)$ is large, the concept $z$ is strongly related to the document class $y$ [7]. In this study, we utilize matrix $H$ to represent Concept Intension [6].

In one latent feature space, When the source and target domains have the same concept extension and the same concept intension, we denote it as **Identical Concept**. Similarly, When the source and target domains have the different concept extension and the same concept intension, we denote it as **Homogeneous Concept**. The description of these concepts in one latent feature space is shown in Table 2.

To fit different situations on the data distributions in one latent spaces, MBTL learns the identical concept and the homogeneous concept together. These concepts are more appropriate for the text classification across domains than learning from the original features. Since identical concepts are same in different domains, they can be directly utilized to construct the bridge for knowledge transfer as shared concepts. As for the homogenous concepts, it is quite often that different domains utilize different phrases to represent the same class [4]. For example, on the computer-related pages the terms describing the class of computer can be “hardware technology”, “software engineering”, and so on. Obviously, the different expression in different domains can refer to the same meaning. Along this line, we divide $U$ and $H$ into two parts respectively. Specifically, $U = [U^1_{m\times k_a}, U^2_{m\times k_b}](k_a + k_b = k)$, and $U^1_{m\times k_a}$ represents the feature clusters for the identical concept and $U^2_{m\times k_b}$ represents the feature clusters for the homogeneous concept. Accordingly, $H$ can be indicated as $H = \begin{bmatrix} H^1_{k_a\times c} \\ H^2_{k_b\times c} \end{bmatrix}$, and $H^1_{k_a\times c}$ represents the association between example classes and the identical concept and $H^2_{k_b\times c}$ represents the association between example classes and the homogeneous concept. Therefore, formula (1) can be rewritten

as

$$X_{m\times n} = U_{m\times k}H_{k\times c}V^T_{n\times c} = \left[U^1_{m\times k_a}, U^2_{m\times k_b}\right]\begin{bmatrix}H^1_{k_a\times c}\\ H^2_{k_b\times c}\end{bmatrix}V^T_{n\times c} \tag{9}$$

Then, the objective function is formulated as follows:

$$\mathcal{L} = \sum_{r=1}^{s+t} \| X_r - U_r H_r V_r^T \|^2 \tag{10}$$

where $X_r \in R_+^{m\times n^r}$, $U_r \in R_+^{m\times k}$, $H_r \in R_+^{k\times c}$ and $V_r^T \in R_+^{n^r\times c}$.

According to formula (9), the objective function can be rewritten as follows:

$$\begin{aligned}\mathcal{L} &= \sum_{r=1}^{s+t} ||X_r - U_r H_r V_r^T||^2 \\ &= \sum_{r=1}^{s+t} \| X_r - \left[U^1, U_r^2\right]\begin{bmatrix}H^1\\ H^2\end{bmatrix}V_r^T\|^2\end{aligned} \tag{11}$$

For Eq. (11), $U^1 \in R_+^{m\times k_a}$ which is identical in different domains represents the set of feature clusters for the identical concept. $U_r^2 \in R_+^{m\times k_b}$ which is different in different domains represents the set of feature clusters for the homogeneous concept. $H^1 \in R_+^{k_a\times c}$ represents the association between the identical concept and example classes. $H^2 \in R_+^{k_b\times c}$ represents the association between the homogeneous concept and example classes. And $V_r \in R_+^{n_r\times c}(s+1 \le r \le s+t)$ represents the classifier used in target domain.

To utilize more latent factors for training a more effective classification model, we first construct multiple different latent spaces by running a clustering algorithm with different parameters respectively, and then simultaneously learn the corresponding marginal distributions and conditional distributions in the different latent feature spaces as the bridges across domains. In this paper, to describe the effectiveness of MBTL in brief, we set the number of latent spaces to three. Then we can obtain the corresponding feature clusters related to learning the distributions in the different latent spaces, and these feature clusters which are represented as $U^1$ and $U_r^2$ can be divided into three parts according to these different latent feature spaces respectively, such that $U^1 = [U^1_{LatentSpace1}, U^1_{LatentSpace2}, U^1_{LatentSpace3}]$, where $U^1 \in R_+^{m\times k_a}$, $U^1_{LatentSpace1} \in R_+^{m\times k_a^1}$, $U^1_{LatentSpace2} \in R_+^{m\times k_a^2}$, $U^1_{LatentSpace3} \in R_+^{m\times k_a^3}$ and $k_a^1 + k_a^2 + k_a^3 = k_a$. $U_r^2 = [U^2_{r(LatentSpace1)}, U^2_{r(LatentSpace2)}, U^2_{r(LatentSpace3)}]$, where $U_r^2 \in R_+^{m\times k_b}$, $U^2_{r(LatentSpace1)} \in R_+^{m\times k_b^1}$, $U^2_{r(LatentSpace2)} \in R_+^{m\times k_b^2}$, $U^2_{r(LatentSpace3)} \in R_+^{m\times k_b^3}$ and $k_b^1 + k_b^2 + k_b^3 = k_b$. Similarly, $H^1$ and $H^2$, which are represented as the association between example classes and the identical concepts, and the association between example classes and the homogeneous concepts respectively, can also be divided into three parts according to the different latent feature spaces respectively, such that $H^1 = [H^1_{LatentSpace1}, H^1_{LatentSpace2}, H^1_{LatentSpace3}]$, where $H^1 \in R_+^{k_a\times c}$, $H^1_{LatentSpace1} \in R_+^{k_a^1\times c}$, $H^1_{LatentSpace2} \in R_+^{k_a^2\times c}$, $H^1_{LatentSpace3} \in R_+^{k_a^3\times c}$ and $k_a^1 + k_a^2 + k_a^3 = k_a$. $H^2 = [H^2_{LatentSpace1}, H^2_{LatentSpace2}, H^2_{LatentSpace3}]$, where $H^2 \in R_+^{k_b\times c}$, $H^2_{LatentSpace1} \in R_+^{k_b^1\times c}$, $H^2_{LatentSpace2} \in R_+^{k_b^2\times c}$, $H^2_{LatentSpace3} \in R_+^{k_b^3\times c}$ and $k_b^1 + k_b^2 + k_b^3 = k_b$. On the other hand, we model these distributions in different latent spaces together. Then we can rewrite the Eq. (11) as follows

$$\begin{aligned}\mathcal{L} &= \sum_{r=1}^{s+t} ||X_r - U_r H_r V_r^T||^2 \\ &= \sum_{r=1}^{s+t} \| X_r - \left[U^1_{LatentSpace1}, U^1_{LatentSpace2}, U^1_{LatentSpace3}, U^2_{r(LatentSpace1)}, U^2_{r(LatentSpace2)}, U^2_{r(LatentSpace3)}\right]\\ &\quad\begin{bmatrix}H^1_{LatentSpace1}\\ H^1_{LatentSpace2}\\ H^1_{LatentSpace3}\\ H^2_{LatentSpace1}\\ H^2_{LatentSpace2}\\ H^2_{LatentSpace3}\end{bmatrix}V_r^T\|^2\end{aligned} \tag{12}$$

To quantify the relationships among the original features, the feature clusters and the example classes, we model the constraint conditions for $U^1_{LatentSpace1}$, $U^1_{LatentSpace2}$, $U^1_{LatentSpace3}$, $U^2_{r(LatentSpace1)}$, $U^2_{r(LatentSpace2)}$, $U^2_{r(LatentSpace3)}$, $H^1_{LatentSpace1}$, $H^1_{LatentSpace2}$, $H^1_{LatentSpace3}$, $H^2_{LatentSpace1}$, $H^2_{LatentSpace2}$, $H^2_{LatentSpace3}$ and $V_r$ simultaneously, and educe the optimization problem as follows:

$$\begin{aligned}&\min_{U_r,H_r,V_r} \mathcal{L}\\ &s.t. \sum_{j=1}^{k_a^1} U^1_{(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{k_a^2} U^1_{(LatentSpace2)[i,j]} = 1,\\ &\sum_{j=1}^{k_a^3} U^1_{(LatentSpace3)[i,j]} = 1,\\ &\sum_{j=1}^{k_b^1} U^2_{r(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{k_b^2} U^2_{r(LatentSpace2)[i,j]} = 1,\\ &\sum_{j=1}^{k_b^3} U^2_{r(LatentSpace3)[i,j]} = 1,\\ &\sum_{j=1}^{c} H^1_{(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{c} H^1_{(LatentSpace2)[i,j]} = 1,\\ &\sum_{j=1}^{c} H^1_{(LatentSpace3)[i,j]} = 1,\\ &\sum_{j=1}^{c} H^2_{(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{c} H^2_{(LatentSpace2)[i,j]} = 1,\\ &\sum_{j=1}^{c} H^2_{(LatentSpace3)[i,j]} = 1, \quad \sum_{j=1}^{c} V_{r[i,j]} = 1.\end{aligned} \tag{13}$$

Here, these constraint conditions represent the feature clusters distribution of the original features, the classes distribution of the feature clusters, and the classes distribution of the examples respectively.

### 4.2. MBTL solution

To solve the method MBTL, we first give the analysis of the objective function, and, then derive the updating rules. Additionally an iterative algorithm is proposed to solve the optimization problem. The formulation of the objective function is

$$\mathcal{L} = \sum_{r=1}^{s+t} \| X_r - [U^1_{LatentSpace1}, U^1_{LatentSpace2}, U^1_{LatentSpace3}, U^2_{r(LatentSpace1)}, U^2_{r(LatentSpace2)}, U^2_{r(LatentSpace3)}]$$

$$\left[\begin{matrix} H^1_{LatentSpace1} \\ H^1_{LatentSpace2} \\ H^1_{LatentSpace3} \\ H^2_{LatentSpace1} \\ H^2_{LatentSpace2} \\ H^2_{LatentSpace3} \end{matrix}\right] V_r^T \|^2 \tag{14}$$

$$= \sum_{r=1}^{s+t} tr(X_r^T X_r - 2 \cdot X_r^T [U^1_{LatentSpace1}, U^1_{LatentSpace2}, U^1_{LatentSpace3},$$

$$U^2_{r(LatentSpace1)}, U^2_{r(LatentSpace2)}, U^2_{r(LatentSpace3)}] \left[\begin{matrix} H^1_{LatentSpace1} \\ H^1_{LatentSpace2} \\ H^1_{LatentSpace3} \\ H^2_{LatentSpace1} \\ H^2_{LatentSpace2} \\ H^2_{LatentSpace3} \end{matrix}\right] V_r^T$$

$$+ V_r \left[\begin{matrix} H^1_{LatentSpace1} \\ H^1_{LatentSpace2} \\ H^1_{LatentSpace3} \\ H^2_{LatentSpace1} \\ H^2_{LatentSpace2} \\ H^2_{LatentSpace3} \end{matrix}\right]^T [U^1_{LatentSpace1}, U^1_{LatentSpace2}, U^1_{LatentSpace3},$$

$$U^2_{r(LatentSpace1)}, U^2_{r(LatentSpace2)}, U^2_{r(LatentSpace3)}]^T$$

$$[U^1_{LatentSpace1}, U^1_{LatentSpace2}, U^1_{LatentSpace3},$$

$$U^2_{r(LatentSpace1)}, U^2_{r(LatentSpace2)}, U^2_{r(LatentSpace3)}] \left[\begin{matrix} H^1_{LatentSpace1} \\ H^1_{LatentSpace2} \\ H^1_{LatentSpace3} \\ H^2_{LatentSpace1} \\ H^2_{LatentSpace2} \\ H^2_{LatentSpace3} \end{matrix}\right] V_r^T)$$

$$= \sum_{r=1}^{s+t} tr\big(X_r^T X_r - 2 \cdot X_r^T A_r - 2 \cdot X_r^T B_r - 2 \cdot X_r^T C_r - 2 \cdot X_r^T D_r - 2 \cdot X_r^T E_r$$
$$- 2 \cdot X_r^T F_r + A_r^T A_r + B_r^T B_r + C_r^T C_r + D_r^T D_r + E_r^T E_r + F_r^T F_r$$
$$+ 2 \cdot A_r^T B_r + 2 \cdot A_r^T C_r + 2 \cdot A_r^T D_r + 2 \cdot A_r^T E_r + 2 \cdot A_r^T F_r$$
$$+ 2 \cdot B_r^T C_r + 2 \cdot B_r^T D_r + 2 \cdot B_r^T E_r + 2 \cdot B_r^T F_r + 2 \cdot C_r^T D_r$$
$$+ 2 \cdot C_r^T E_r + 2 \cdot C_r^T F_r + 2 \cdot D_r^T E_r + 2 \cdot D_r^T F_r + 2 \cdot E_r^T F_r\big)$$

$$s.t. \sum_{j=1}^{k_a^1} U^1_{(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{k_a^2} U^1_{(LatentSpace2)[i,j]} = 1,$$

$$\sum_{j=1}^{k_a^3} U^1_{(LatentSpace3)[i,j]} = 1,$$

$$\sum_{j=1}^{k_b^1} U^2_{r(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{k_b^2} U^2_{r(LatentSpace2)[i,j]} = 1,$$

$$\sum_{j=1}^{k_b^3} U^2_{r(LatentSpace3)[i,j]} = 1,$$

$$\sum_{j=1}^{c} H^1_{(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{c} H^1_{(LatentSpace2)[i,j]} = 1,$$

$$\sum_{j=1}^{c} H^1_{(LatentSpace3)[i,j]} = 1,$$

$$\sum_{j=1}^{c} H^2_{(LatentSpace1)[i,j]} = 1, \quad \sum_{j=1}^{c} H^2_{(LatentSpace2)[i,j]} = 1,$$

$$\sum_{j=1}^{c} H^2_{(LatentSpace3)[i,j]} = 1, \quad \sum_{j=1}^{c} V_{r[i,j]} = 1.$$

where $A_r = U^1_{LatentSpace1} \quad H^1_{LatentSpace1} V_r^T$, $A_r^T = V_r H^{1^T}_{LatentSpace1} \quad U^{1^T}_{LatentSpace1}$, $B_r = U^1_{LatentSpace2} \quad H^1_{LatentSpace2} V_r^T$, $B_r^T = V_r H^{1^T}_{LatentSpace2} \quad U^{1^T}_{LatentSpace2}$, $C_r = U^1_{LatentSpace3} \quad H^1_{LatentSpace3} V_r^T$, $C_r^T = V_r H^{1^T}_{LatentSpace3} \quad U^{1^T}_{LatentSpace3}$, $D_r = U^2_{r(LatentSpace1)} \quad H^2_{LatentSpace1} V_r^T$, $D_r^T = V_r H^{2^T}_{LatentSpace1} \quad U^{2^T}_{r(LatentSpace1)}$, $E_r = U^2_{r(LatentSpace2)} \quad H^2_{LatentSpace2} V_r^T$, $E_r^T = V_r H^{2^T}_{LatentSpace2} \quad U^{2^T}_{r(LatentSpace2)}$, $F_r = U^2_{r(LatentSpace3)} \quad H^2_{LatentSpace3} V_r^T$ and $F_r^T = V_r H^{2^T}_{LatentSpace3} \quad U^{2^T}_{r(LatentSpace3)}$. The partial differentials of $\mathcal{L}$ are as follows:

$$\frac{\partial \mathcal{L}}{\partial U^1_{LatentSpace1}} = \sum_{r=1}^{s+t} -2 \cdot X_r V_r H^{1^T}_{LatentSpace1} + 2 \cdot A_r V_r H^{1^T}_{LatentSpace1}$$
$$+ 2 \cdot B_r V_r H^{1^T}_{LatentSpace1}$$
$$+ 2 \cdot C_r V_r H^{1^T}_{LatentSpace1} + 2 \cdot D_r V_r H^{1^T}_{LatentSpace1}$$
$$+ 2 \cdot E_r V_r H^{1^T}_{LatentSpace1} + 2 \cdot F_r V_r H^{1^T}_{LatentSpace1} \tag{15}$$

$$\frac{\partial \mathcal{L}}{\partial U^1_{LatentSpace2}} = \sum_{r=1}^{s+t} -2 \cdot X_r V_r H^{1^T}_{LatentSpace2} + 2 \cdot A_r V_r H^{1^T}_{LatentSpace2}$$
$$+ 2 \cdot B_r V_r H^{1^T}_{LatentSpace2}$$
$$+ 2 \cdot C_r V_r H^{1^T}_{LatentSpace2} + 2 \cdot D_r V_r H^{1^T}_{LatentSpace2}$$
$$+ 2 \cdot E_r V_r H^{1^T}_{LatentSpace2} + 2 \cdot F_r V_r H^{1^T}_{LatentSpace2} \tag{16}$$

$$\frac{\partial \mathcal{L}}{\partial U^1_{LatentSpace3}} = \sum_{r=1}^{s+t} -2 \cdot X_r V_r H^{1^T}_{LatentSpace3} + 2 \cdot A_r V_r H^{1^T}_{LatentSpace3}$$
$$+ 2 \cdot B_r V_r H^{1^T}_{LatentSpace3}$$
$$+ 2 \cdot C_r V_r H^{1^T}_{LatentSpace3} + 2 \cdot D_r V_r H^{1^T}_{LatentSpace3}$$
$$+ 2 \cdot E_r V_r H^{1^T}_{LatentSpace3} + 2 \cdot F_r V_r H^{1^T}_{LatentSpace3} \tag{17}$$

$$\frac{\partial \mathcal{L}}{\partial U^2_{r(LatentSpace1)}} = -2 \cdot X_r V_r H^{2^T}_{LatentSpace1} + 2 \cdot A_r V_r H^{2^T}_{LatentSpace1}$$
$$+ 2 \cdot B_r V_r H^{2^T}_{LatentSpace1}$$
$$+ 2 \cdot C_r V_r H^{2^T}_{LatentSpace1} + 2 \cdot D_r V_r H^{2^T}_{LatentSpace1}$$
$$+ 2 \cdot E_r V_r H^{2^T}_{LatentSpace1} + 2 \cdot F_r V_r H^{2^T}_{LatentSpace1} \tag{18}$$

$$\frac{\partial \mathcal{L}}{\partial U^2_{r(LatentSpace2)}} = -2 \cdot X_r V_r H^{2^T}_{LatentSpace2} + 2 \cdot A_r V_r H^{2^T}_{LatentSpace2}$$
$$+ 2 \cdot B_r V_r H^{2^T}_{LatentSpace2}$$
$$+ 2 \cdot C_r V_r H^{2^T}_{LatentSpace2} + 2 \cdot D_r V_r H^{2^T}_{LatentSpace2}$$
$$+ 2 \cdot E_r V_r H^{2^T}_{LatentSpace2} + 2 \cdot F_r V_r H^{2^T}_{LatentSpace2} \tag{19}$$

$$\frac{\partial \mathcal{L}}{\partial U^2_{r(LatentSpace3)}} = -2 \cdot X_r V_r H^{2^T}_{LatentSpace3} + 2 \cdot A_r V_r H^{2^T}_{LatentSpace3}$$
$$+ 2 \cdot B_r V_r H^{2^T}_{LatentSpace3}$$
$$+ 2 \cdot C_r V_r H^{2^T}_{LatentSpace3} + 2 \cdot D_r V_r H^{2^T}_{LatentSpace3}$$
$$+ 2 \cdot E_r V_r H^{2^T}_{LatentSpace3} + 2 \cdot F_r V_r H^{2^T}_{LatentSpace3} \tag{20}$$

$$\frac{\partial \mathcal{L}}{\partial H^1_{LatentSpace1}} = \sum_{r=1}^{s+t} -2 \cdot U^{1^T}_{LatentSpace1} X_r V_r + 2 \cdot U^{1^T}_{LatentSpace1} A_r V_r$$
$$+ 2 \cdot U^{1^T}_{LatentSpace1} B_r V_r$$

$$+ 2 \cdot U^{1^T}_{LatentSpace1} C_r V_r + 2 \cdot U^{1^T}_{LatentSpace1} D_r V_r + 2 \cdot U^{1^T}_{LatentSpace1} E_r V_r + 2 \cdot U^{1^T}_{LatentSpace1} F_r V_r \quad (21)$$

$$\frac{\partial \mathcal{L}}{\partial H^1_{LatentSpace2}} = \sum_{r=1}^{s+t} -2 \cdot U^{1^T}_{LatentSpace2} X_r V_r + 2 \cdot U^{1^T}_{LatentSpace2} A_r V_r + 2 \cdot U^{1^T}_{LatentSpace2} B_r V_r + 2 \cdot U^{1^T}_{LatentSpace2} C_r V_r + 2 \cdot U^{1^T}_{LatentSpace2} D_r V_r + 2 \cdot U^{1^T}_{LatentSpace2} E_r V_r + 2 \cdot U^{1^T}_{LatentSpace2} F_r V_r \quad (22)$$

$$\frac{\partial \mathcal{L}}{\partial H^1_{LatentSpace3}} = \sum_{r=1}^{s+t} -2 \cdot U^{1^T}_{LatentSpace3} X_r V_r + 2 \cdot U^{1^T}_{LatentSpace3} A_r V_r + 2 \cdot U^{1^T}_{LatentSpace3} B_r V_r + 2 \cdot U^{1^T}_{LatentSpace3} C_r V_r + 2 \cdot U^{1^T}_{LatentSpace3} D_r V_r + 2 \cdot U^{1^T}_{LatentSpace3} E_r V_r + 2 \cdot U^{1^T}_{LatentSpace3} F_r V_r \quad (23)$$

$$\frac{\partial \mathcal{L}}{\partial H^2_{LatentSpace1}} = \sum_{r=1}^{s+t} -2 \cdot U^{2^T}_{r(LatentSpace1)} X_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace1)} A_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace1)} B_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace1)} C_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace1)} D_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace1)} E_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace1)} F_r V_r \quad (24)$$

$$\frac{\partial \mathcal{L}}{\partial H^2_{LatentSpace2}} = \sum_{r=1}^{s+t} -2 \cdot U^{2^T}_{r(LatentSpace2)} X_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace2)} A_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace2)} B_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace2)} C_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace2)} D_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace2)} E_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace2)} F_r V_r \quad (25)$$

$$\frac{\partial \mathcal{L}}{\partial H^2_{LatentSpace3}} = \sum_{r=1}^{s+t} -2 \cdot U^{2^T}_{r(LatentSpace3)} X_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace3)} A_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace3)} B_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace3)} C_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace3)} D_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace3)} E_r V_r + 2 \cdot U^{2^T}_{r(LatentSpace3)} F_r V_r \quad (26)$$

$$\frac{\partial \mathcal{L}}{\partial V_r} = -2 \cdot X_r^T U_r H_r + 2 \cdot V_r H_r^T U_r^T U_r H_r \quad (27)$$

Since the objective function is not concave, it is very difficult to get a global solution by using the non-linear optimization technique. Then we propose an alternately iterative algorithm with convergence guarantee to obtain a local optimal solution. These variables are updated as:

$$U^1_{(LatentSpace1)[i,j]} \leftarrow U^1_{(LatentSpace1)[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} X_r V_r H^{1^T}_{LatentSpace1} \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} A_r V_r H^{1^T}_{LatentSpace1} + B_r V_r H^{1^T}_{LatentSpace1} + C_r V_r H^{1^T}_{LatentSpace1} + D_r V_r H^{1^T}_{LatentSpace1} + E_r V_r H^{1^T}_{LatentSpace1} + F_r V_r H^{1^T}_{LatentSpace1} \right]_{[i,j]} \right)^{1/2} \quad (28)$$

$$U^1_{(LatentSpace2)[i,j]} \leftarrow U^1_{(LatentSpace2)[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} X_r V_r H^{1^T}_{LatentSpace2} \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} A_r V_r H^{1^T}_{LatentSpace2} + B_r V_r H^{1^T}_{LatentSpace2} + C_r V_r H^{1^T}_{LatentSpace2} + D_r V_r H^{1^T}_{LatentSpace2} + E_r V_r H^{1^T}_{LatentSpace2} + F_r V_r H^{1^T}_{LatentSpace2} \right]_{[i,j]} \right)^{1/2} \quad (29)$$

$$U^1_{(LatentSpace3)[i,j]} \leftarrow U^1_{(LatentSpace3)[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} X_r V_r H^{1^T}_{LatentSpace3} \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} A_r V_r H^{1^T}_{LatentSpace3} + B_r V_r H^{1^T}_{LatentSpace3} + C_r V_r H^{1^T}_{LatentSpace3} + D_r V_r H^{1^T}_{LatentSpace3} + E_r V_r H^{1^T}_{LatentSpace3} + F_r V_r H^{1^T}_{LatentSpace3} \right]_{[i,j]} \right)^{1/2} \quad (30)$$

$$U^2_{r(LatentSpace1)[i,j]} \leftarrow U^2_{r(LatentSpace1)[i,j]} \cdot \left( \left[ X_r V_r H^{2^T}_{LatentSpace1} \right]_{[i,j]} \Big/ \left[ A_r V_r H^{2^T}_{LatentSpace1} + B_r V_r H^{2^T}_{LatentSpace1} + C_r V_r H^{2^T}_{LatentSpace1} + D_r V_r H^{2^T}_{LatentSpace1} + E_r V_r H^{2^T}_{LatentSpace1} + F_r V_r H^{2^T}_{LatentSpace1} \right]_{[i,j]} \right)^{1/2} \quad (31)$$

$$U^2_{r(LatentSpace2)[i,j]} \leftarrow U^2_{r(LatentSpace2)[i,j]} \cdot \left( \left[ X_r V_r H^{2^T}_{LatentSpace2} \right]_{[i,j]} \Big/ \left[ A_r V_r H^{2^T}_{LatentSpace2} + B_r V_r H^{2^T}_{LatentSpace2} + C_r V_r H^{2^T}_{LatentSpace2} + D_r V_r H^{2^T}_{LatentSpace2} + E_r V_r H^{2^T}_{LatentSpace2} + F_r V_r H^{2^T}_{LatentSpace2} \right]_{[i,j]} \right)^{1/2} \quad (32)$$

$$U^2_{r(LatentSpace3)[i,j]} \leftarrow U^2_{r(LatentSpace3)[i,j]} \cdot \left( \left[ X_r V_r H^{2^T}_{LatentSpace3} \right]_{[i,j]} \Big/ \left[ A_r V_r H^{2^T}_{LatentSpace3} + B_r V_r H^{2^T}_{LatentSpace3} + C_r V_r H^{2^T}_{LatentSpace3} + D_r V_r H^{2^T}_{LatentSpace3} + E_r V_r H^{2^T}_{LatentSpace3} + F_r V_r H^{2^T}_{LatentSpace3} \right]_{[i,j]} \right)^{1/2} \quad (33)$$

$$H^1_{LatentSpace1[i,j]} \leftarrow H^1_{LatentSpace1[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} U^{1^T}_{LatentSpace1} X_r V_r \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} U^{1^T}_{LatentSpace1} A_r V_r + U^{1^T}_{LatentSpace1} B_r V_r + U^{1^T}_{LatentSpace1} C_r V_r + U^{1^T}_{LatentSpace1} D_r V_r + U^{1^T}_{LatentSpace1} E_r V_r + U^{1^T}_{LatentSpace1} F_r V_r \right]_{[i,j]} \right)^{1/2} \quad (34)$$

$$H^1_{LatentSpace2[i,j]} \leftarrow H^1_{LatentSpace2[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} U^{1^T}_{LatentSpace2} X_r V_r \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} U^{1^T}_{LatentSpace2} A_r V_r + U^{1^T}_{LatentSpace2} B_r V_r + U^{1^T}_{LatentSpace2} C_r V_r + U^{1^T}_{LatentSpace2} D_r V_r + U^{1^T}_{LatentSpace2} E_r V_r + U^{1^T}_{LatentSpace2} F_r V_r \right]_{[i,j]} \right)^{1/2} \quad (35)$$

$$H^1_{LatentSpace3[i,j]} \leftarrow H^1_{LatentSpace3[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} U^{1^T}_{LatentSpace3} X_r V_r \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} U^{1^T}_{LatentSpace3} A_r V_r + U^{1^T}_{LatentSpace3} B_r V_r + U^{1^T}_{LatentSpace3} C_r V_r + U^{1^T}_{LatentSpace3} D_r V_r + U^{1^T}_{LatentSpace3} E_r V_r + U^{1^T}_{LatentSpace3} F_r V_r \right]_{[i,j]} \right)^{1/2} \quad (36)$$

$$H^2_{LatentSpace1[i,j]} \leftarrow H^2_{LatentSpace1[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} U^{2^T}_{r(LatentSpace1)} X_r V_r \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} U^{2^T}_{r(LatentSpace1)} A_r V_r + U^{2^T}_{r(LatentSpace1)} B_r V_r + U^{2^T}_{r(LatentSpace1)} C_r V + U^{2^T}_{r(LatentSpace1)} D_r V_r + U^{2^T}_{r(LatentSpace1)} E_r V_r + U^{2^T}_{r(LatentSpace1)} F_r V_r \right]_{[i,j]} \right)^{1/2} \tag{37}$$

$$H^2_{LatentSpace2[i,j]} \leftarrow H^2_{LatentSpace2[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} U^{2^T}_{r(LatentSpace2)} X_r V_r \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} U^{2^T}_{r(LatentSpace2)} A_r V_r + U^{2^T}_{r(LatentSpace2)} B_r V_r + U^{2^T}_{r(LatentSpace2)} C_r V + U^{2^T}_{r(LatentSpace2)} D_r V_r + U^{2^T}_{r(LatentSpace2)} E_r V_r + U^{2^T}_{r(LatentSpace2)} F_r V_r \right]_{[i,j]} \right)^{1/2} \tag{38}$$

$$H^2_{LatentSpace3[i,j]} \leftarrow H^2_{LatentSpace3[i,j]} \cdot \left( \left[ \sum_{r=1}^{s+t} U^{2^T}_{r(LatentSpace3)} X_r V_r \right]_{[i,j]} \Big/ \left[ \sum_{r=1}^{s+t} U^{2^T}_{r(LatentSpace3)} A_r V_r + U^{2^T}_{r(LatentSpace3)} B_r V_r + U^{2^T}_{r(LatentSpace3)} C_r V + U^{2^T}_{r(LatentSpace3)} D_r V_r + U^{2^T}_{r(LatentSpace3)} E_r V_r + U^{2^T}_{r(LatentSpace3)} F_r V_r \right]_{[i,j]} \right)^{1/2} \tag{39}$$

$$V_{r[i,j]} \leftarrow V_{r[i,j]} \cdot \left( \left[ X_r{}^T U_r H_r \right]_{[i,j]} \Big/ \left[ V_r H_r{}^T U_r{}^T U_r H_r \right]_{[i,j]} \right)^{1/2} \tag{40}$$

In each iteration, we calculate all the variables under the updating rules and use Eq. (41) to normalize $U^1_{LatentSpace1}$, $U^1_{LatentSpace2}$, $U^1_{LatentSpace3}$, $U^2_{r(LatentSpace1)}$, $U^2_{r(LatentSpace2)}$, $U^2_{r(LatentSpace3)}$, $H^1_{LatentSpace1}$, $H^1_{LatentSpace2}$, $H^1_{LatentSpace3}$, $H^2_{LatentSpace1}$, $H^2_{LatentSpace2}$, $H^2_{LatentSpace3}$ and $V_r$.

$$U^1_{(LatentSpace1)[i,j]} \leftarrow \frac{U^1_{(LatentSpace1)[i,j]}}{\sum_{j=1}^{k^1_a} U^1_{(LatentSpace1)[i,j]}},$$

$$U^1_{(LatentSpace2)[i,j]} \leftarrow \frac{U^1_{(LatentSpace2)[i,j]}}{\sum_{j=1}^{k^2_a} U^1_{(LatentSpace2)[i,j]}},$$

$$U^1_{(LatentSpace3)[i,j]} \leftarrow \frac{U^1_{(LatentSpace3)[i,j]}}{\sum_{j=1}^{k^3_a} U^1_{(LatentSpace3)[i,j]}},$$

$$U^2_{r(LatentSpace1)[i,j]} \leftarrow \frac{U^2_{r(LatentSpace1)[i,j]}}{\sum_{j=1}^{k^1_b} U^2_{r(LatentSpace1)[i,j]}},$$

$$U^2_{r(LatentSpace2)[i,j]} \leftarrow \frac{U^2_{r(LatentSpace2)[i,j]}}{\sum_{j=1}^{k^2_b} U^2_{r(LatentSpace2)[i,j]}},$$

$$U^2_{r(LatentSpace3)[i,j]} \leftarrow \frac{U^2_{r(LatentSpace3)[i,j]}}{\sum_{j=1}^{k^3_b} U^2_{r(LatentSpace3)[i,j]}},$$

$$H^1_{(LatentSpace1)[i,j]} \leftarrow \frac{H^1_{(LatentSpace1)[i,j]}}{\sum_{j=1}^{c} H^1_{(LatentSpace1)[i,j]}},$$

$$H^1_{(LatentSpace2)[i,j]} \leftarrow \frac{H^1_{(LatentSpace2)[i,j]}}{\sum_{j=1}^{c} H^1_{(LatentSpace2)[i,j]}},$$

$$H^1_{(LatentSpace3)[i,j]} \leftarrow \frac{H^1_{(LatentSpace3)[i,j]}}{\sum_{j=1}^{c} H^1_{(LatentSpace3)[i,j]}},$$

$$H^2_{(LatentSpace1)[i,j]} \leftarrow \frac{H^2_{(LatentSpace1)[i,j]}}{\sum_{j=1}^{c} H^2_{(LatentSpace1)[i,j]}},$$

$$H^2_{(LatentSpace2)[i,j]} \leftarrow \frac{H^2_{(LatentSpace2)[i,j]}}{\sum_{j=1}^{c} H^2_{(LatentSpace2)[i,j]}},$$

$$H^2_{(LatentSpace3)[i,j]} \leftarrow \frac{H^2_{(LatentSpace3)[i,j]}}{\sum_{j=1}^{c} H^2_{(LatentSpace3)[i,j]}},$$

$$V_{r[i,j]} \leftarrow \frac{V_{r[i,j]}}{\sum_{j=1}^{c} V_{r[i,j]}} \tag{41}$$

Based on the above formulas, we propose an iterative algorithm and describe it in Algorithm 1. In this algorithm, $V_r(1 \le r \le s)$ is initialized by the true label information, and

**Algorithm 1:** MBTL: Multi-Bridge Transfer Learning Algorithm.

**Input**: $\{X_r\}_{r=1}^{s+t}$, $\{V_r\}_{r=1}^{s}$, parameters $k^1_a$,$k^1_b$,$k^2_a$,$k^2_b$,$k^3_a$,$k^3_b$ and the number of iterations *maxIter*.

**Output**: $U^1_{LatentSpace1}$, $U^1_{LatentSpace2}$, $U^1_{LatentSpace3}$, $U^2_{r(LatentSpace1)}$, $U^2_{r(LatentSpace2)}$, $U^2_{r(LatentSpace3)}$, $H^1_{LatentSpace1}$, $H^1_{LatentSpace2}$, $H^1_{LatentSpace3}$, $H^2_{LatentSpace1}$, $H^2_{LatentSpace2}$, $H^2_{LatentSpace3}$ $(1 \le r \le s+t)$ and $V_r(1+s \le r \le s+t)$.

1 Normalize the data matrices by $X_{r[i,j]} \leftarrow X_{r[i,j]} / \sum_{i=1}^{m} X_{r[i,j]}$, $(1 \le r \le s+t)$;
2 The initializations of $U^{1(0)}_{LatentSpace1}$, $U^{1(0)}_{LatentSpace2}$, $U^{1(0)}_{LatentSpace3}$, $U^{2(0)}_{r(LatentSpace1)}$, $U^{2(0)}_{r(LatentSpace2)}$ and $U^{2(0)}_{r(LatentSpace3)}$ are detailed in section 4.2, $H^{1(0)}_{LatentSpace1}$, $H^{1(0)}_{LatentSpace2}$, $H^{1(0)}_{LatentSpace3}$, $H^{2(0)}_{LatentSpace1}$, $H^{2(0)}_{LatentSpace2}$ and $H^{2(0)}_{LatentSpace3}$ are randomly assigned, and $V^{(0)}_r(1+s \le r \le s+t)$ is initialized by Logistic Regression;
3 **for** $k \leftarrow 1$ *to maxIter* **do**
4 Update $U^{1(k)}_{LatentSpace1}$, $U^{1(k)}_{LatentSpace2}$ and $U^{1(k)}_{LatentSpace3}$, by Eq. (28) 0, Eq. (29) and Eq. (30) respectively;
5 **for** $r \leftarrow 1$ *to s+t* **do**
6 Update $U^{2(k)}_{r(LatentSpace1)}$, $U^{2(k)}_{r(LatentSpace2)}$ and $U^{2(k)}_{r(LatentSpace3)}$ by Eq. (31), Eq. (32) and Eq. (33) respectively;
7 **end**
8 Update $H^{1(k)}_{LatentSpace1}$, $H^{1(k)}_{LatentSpace2}$ and $H^{1(k)}_{LatentSpace3}$ by Eq. (34), Eq. (35) and Eq. (36) respectively;
9 **for** $r \leftarrow 1$ *to s+t* **do**
10 Update $H^{2(k)}_{LatentSpace1}$, $H^{2(k)}_{LatentSpace2}$ and $H^{2(k)}_{LatentSpace3}$ by Eq. (37), Eq. (38) and Eq. (39) respectively;
11 **end**
12 **for** $r \leftarrow s+1$ *to s+t* **do**
13 Update $V^{(k)}_r$ by (40);
14 **end**
15 Normalize $U^{1(k)}_{LatentSpace1}$, $U^{1(k)}_{LatentSpace2}$, $U^{1(k)}_{LatentSpace3}$, $U^{2(k)}_{r(LatentSpace1)}$, $U^{2(k)}_{r(LatentSpace2)}$, $U^{2(k)}_{r(LatentSpace3)}$, $H^{1(k)}_{LatentSpace1}$, $H^{1(k)}_{LatentSpace2}$, $H^{1(k)}_{LatentSpace3}$, $H^{2(k)}_{LatentSpace1}$, $H^{2(k)}_{LatentSpace2}$, $H^{2(k)}_{LatentSpace3}$ and $V^{(k)}_r$ by (41);
16 Output $U^{1(k)}_{LatentSpace1}$, $U^{1(k)}_{LatentSpace2}$, $U^{1(k)}_{LatentSpace3}$, $U^{2(k)}_{r(LatentSpace1)}$, $U^{2(k)}_{r(LatentSpace2)}$, $U^{2(k)}_{r(LatentSpace3)}$, $H^{1(k)}_{LatentSpace1}$, $H^{1(k)}_{LatentSpace2}$, $H^{1(k)}_{LatentSpace3}$, $H^{2(k)}_{LatentSpace1}$, $H^{2(k)}_{LatentSpace2}$, $H^{2(k)}_{LatentSpace3}$ and $V^{(k)}_r$.
17 **end**

**Table 3**
Descriptions of the number of concepts.

| | The number of identical concepts | The number of homogeneous concepts | Total number of concepts |
|---|---|---|---|
| The number of concepts in latent space 1 | $k_a^1$ | $k_b^1$ | $k_1$ |
| The number of concepts in latent space 2 | $k_a^2$ | $k_b^2$ | $k_2$ |
| The number of concepts in latent space 3 | $k_a^3$ | $k_b^3$ | $k_3$ |
| Total number of concepts | $k_a$ | $k_b$ | $k$ |

$V_r(1+s \le r \le s+t)$ is initialized by Logistic Regression (LR) [8] which is trained on the source domain to make the algorithm converge faster. We normalize the data matrices such that $X_r^T 1_m = 1_n$. $U^1_{LatentSpace1}$, $U^1_{LatentSpace2}$, $U^1_{LatentSpace3}$, $U^2_{r(LatentSpace1)}$, $U^2_{r(LatentSpace2)}$ and $U^2_{r(LatentSpace3)}$ are initialized with the feature clusters which are obtained by implemented PLSA [9]. According to the number setting of the feature clusters showed in Table 3, we can obtain the feature information $W_1 \in \mathrm{R}_+^{m\times(k_a^1+k_b^1)}$, $W_2 \in \mathrm{R}_+^{m\times(k_a^2+k_b^2)}$ and $W_3 \in \mathrm{R}_+^{m\times(k_a^3+k_b^3)}$ in the corresponding latent spaces through conducting PLSA on the data from the source and target domains. Then we divide $W_1$, $W_2$ and $W_3$ into two parts respectively, such as $W_1 = [W_a^1, W_b^1]$ $(W_a^1 \in R_+^{m\times k_a^1}, W_b^1 \in R_+^{m\times k_b^1})$, $W_2 = [W_a^2, W_b^2]$ $(W_a^2 \in R_+^{m\times k_a^2}, W_b^2 \in R_+^{m\times k_b^2})$ and $W_3 = [W_a^3, W_b^3]$ $(W_a^3 \in R_+^{m\times k_a^3}, W_b^3 \in R_+^{m\times k_b^3})$. $U^1_{LatentSpace1}$, $U^1_{LatentSpace2}$, $U^1_{LatentSpace3}$, $U^2_{r(LatentSpace1)}$, $U^2_{r(LatentSpace2)}$ and $U^2_{r(LatentSpace3)}$ are initialized as $W_a^1$, $W_a^2$, $W_a^3$, $W_b^1$, $W_b^2$ and $W_b^3$ respectively.

### 4.3. Theoretical analysis

To prove the convergence of updating rules (28)–(40), which were derived following the theory of constraint optimization, we first formulate the Lagrange function of the objective function with constraint as follows:

$$\begin{aligned}\mathcal{L} = &\sum_{\mathrm{r}=1}^{s+t} \| X_{\mathrm{r}} - U_r H_r V_r^T \| + \sum_{r=1}^{s+t} tr(\Gamma_r(U_r 1_k - 1_m)(U_r 1_k - 1_m)^T) \\ &+ \sum_{r=1}^{s+t} tr(\lambda_r(H_r 1_c - 1_k)(H_r 1_c - 1_k)^T) \\ &+ \sum_{r=1}^{s+t} tr(\Lambda_r(V_r 1_c - 1_{n_r})(V_r 1_c - 1_{n_r})^T) \end{aligned} \tag{42}$$

where $\Gamma_r \in \mathrm{R}_+^{\mathrm{m}\times\mathrm{m}}$, $\lambda_r \in \mathrm{R}_+^{\mathrm{k}\times\mathrm{k}}$ and $\Lambda_r \in \mathrm{R}_+^{\mathrm{n_r}\times\mathrm{n_r}}$ are diagonal matrixes and represent the Lagrange multipliers.

Since the different variables are similar in the derivation process of the updating rule, we only show the detailed derivation for $V_r$ without lose generality. Therefore, Eq. (42) becomes

$$\begin{aligned}\mathcal{L}(V_r) = &\sum_{r=1}^{s+t} tr\left(-2\cdot X_r^T U_r H_r V_r^T + V_r H_r^T U_r^T U_r H_r V_r^T\right) \\ &+ \sum_{r=1}^{s+t} tr(\Lambda_r(V_r 1_c - 1_{n_r})(V_r 1_c - 1_{n_r})^T) \end{aligned} \tag{43}$$

Then the differential is

$$\frac{\partial \mathcal{L}}{\partial V_r} = -2\cdot X_r^T U_r H_r + 2\cdot V_r H_r^T U_r^T U_r H_r + 2\Lambda_r V_r 1_c 1_c^{\mathrm{T}} - 2\Lambda_r 1_{n_r} 1_c^{\mathrm{T}} \tag{44}$$

And the update formula is as follows:

$$V_{r[i,j]} \leftarrow V_{r[i,j]} \cdot \sqrt{\frac{\left[X_r^T U_r H_r\right]_{[i,j]} + \Lambda_r 1_{n_r} 1_c^{\mathrm{T}}}{\left[V_r H_r^T U_r^T U_r H_r\right]_{[i,j]} + \Lambda_r V_r 1_c 1_c^{\mathrm{T}}}} \tag{45}$$

**Lemma 4.1.** *Eq. (43) is monotone decreasing while using the updating rule (45).*

To prove Lemma 4.1, we describe the definitions of auxiliary function [15] as follow:

**Definition 4.3.** $\mathcal{J}(Y, \tilde{Y})$ is an auxiliary function of $\mathcal{L}(Y)$ if it satisfies

$$\mathcal{J}(Z, \tilde{Z}) \ge \mathcal{L}(Z) \text{ and } \mathcal{J}(Z, Z) = \mathcal{L}(Z) \tag{46}$$

for any $Z$, $\tilde{Z}$.

**Definition 4.4.** $\mathcal{J}(Y, \tilde{Y})$ is an auxiliary function of $\mathcal{L}(Y)$ if it satisfies the following formula:

$$\mathrm{Z}^{(t+1)} = \mathrm{argmin}_Z \mathcal{J}(Z, \mathrm{Z}^{(t)}) \tag{47}$$

Through these definitions, we derive Eq. (48) as follow:

$$\begin{aligned}\mathcal{L}(Z^{(t)}) &= \mathcal{J}(\mathrm{Z}^{(t)}, \mathrm{Z}^{(t)}) \ge \mathcal{J}(\mathrm{Z}^{(t+1)}, \mathrm{Z}^{(t)}) \ge \mathcal{J}(\mathrm{Z}^{(t+1)}, \mathrm{Z}^{(t+1)}) \\ &= \mathcal{L}(Z^{(t+1)}) \end{aligned} \tag{48}$$

From Eq. (48), we can find that the function $\mathcal{L}(Z)$ is monotone decreasing while using the updating rule of $Z$ which is derived by minimizing the auxiliary function $\mathcal{J}(Z, \tilde{Z})$. Then the auxiliary function for $\mathcal{L}(V_r)$ is constructed as

$$\begin{aligned}\mathcal{J}(V_r, \tilde{V}_r) = &\sum_{i,j} \left(\tilde{V}_r H_r^T U_r^T U_r H_r + \Lambda_r \tilde{V}_r 1_c 1_c^{\mathrm{T}}\right)_{[i,j]} \frac{(V_r)^2_{[i,j]}}{(\tilde{V}_r)_{[i,j]}} \\ &- 2\sum_{i,j} \left(X_r^T U_r H_r + \Lambda_r 1_{n_r} 1_c^{\mathrm{T}}\right)_{[i,j]} (\tilde{V}_r)_{[i,j]} \\ &\times \left(1 + \log \frac{(V_r)_{[i,j]}}{(\tilde{V}_r)_{[i,j]}}\right) \end{aligned} \tag{49}$$

Obviously, Equality $\mathcal{L}(V_r) = \mathcal{J}(V_r, \tilde{V}_r)$ holds when $V_r = \tilde{V}_r$. We can also verify $\mathcal{J}(Z, \tilde{Z}) \ge \mathcal{L}(Z)$ and the Hessian matrix $\nabla\nabla_{V_r}\mathcal{J}(V_r, \tilde{V}_r) \ge 0$ by the similar method in [16]. Then we minimize $\mathcal{J}(V_r, \tilde{V}_r)$ when the variable $\tilde{V}_r$ is fixed. The differential of $\mathcal{J}(V_r, \tilde{V}_r)$ is

$$\begin{aligned}\frac{\partial \mathcal{J}(V_r, \tilde{V}_r)}{\partial V_{r[i,j]}} = &2\sum_{i,j} (\tilde{V}_r H_r^T U_r^T U_r H_r + \Lambda_r \tilde{V}_r 1_c 1_c^{\mathrm{T}})_{[i,j]} \frac{(V_r)_{[i,j]}}{(\tilde{V}_r)_{[i,j]}} \\ &- 2\sum_{i,j} (X_r^T U_r H_r + \Lambda_r 1_{n_r} 1_c^{\mathrm{T}})_{[i,j]} \frac{(\tilde{V}_r)_{[i,j]}}{(V_r)_{[i,j]}} \end{aligned} \tag{50}$$

Let $\frac{\partial \mathcal{J}(V_r, \tilde{V}_r)}{\partial V_{r[i,j]}} = 0$, we can obtain the updating rule (45). Therefore, the updating rule (45) decreases the value of $\mathcal{L}(V_r)$, then Lemma 4.1 holds.

The only problem left is to calculate the Lagrange multipliers $\Lambda$. To satisfy the constraint conditions, we can utilize an iterative normalization technique from [14]. Specifically, we normalize $V_r$ by Eq. (41) in each iteration so that $V_r 1_c = 1_{n_r}$. Then we obtain the equation $\Lambda_r 1_{n_r} 1_c^{\mathrm{T}} = \Lambda_r V_r 1_c 1_c^{\mathrm{T}}$ which depends on $\Lambda_r$ only. Therefore, the influence of Eqs. (40) and(41) can be approximately updating rule of (45) without influencing convergence. Then we adopt the updating Eq. (40) for $V_r$ by omitting the equal items which only depend on $\Lambda_r$.

**Theorem 4.1.** *The objective function in (13) will not increase in Algorithm 1 at each iteration.*

Following the similar method as shown above, we could verify the convergence of the update rules for the rest variables respectively. Thus, Algorithm 1 will not increase (13), and Theorem 4.1

**Table 4**
The top categories and their subcategories.

| Top categories | Subcategories |
|---|---|
| | *comp.graphics, sys.mac.hardware* |
| *comp* | *comp.sys.ibm.pc.hardware* |
| | *comp.os.ms-windows.misc* |
| *rec* | *rec.autos, motorcycles* |
| | *rec.sport.baseball, hockey* |
| *sci* | *sci.crypt, med, electronics, space* |
| *talk* | *talk.politics.guns, Mideast, misc* |
| | *talk.religion.misc* |

holds. Since the objective function is lower bounded by zero, the convergence of Algorithm 1 is proved.

### 4.4. Computational complexity of the iterative algorithm

In this section, we analyze the computational complexity of the iterative algorithm. For each round of iteration in Algorithm 1, the computational complexity of (28) to calculate $U^1_{LatentSpace1}$ is $\mathcal{O}(\sum_{r=1}^{s+t} 13mn_rc + 7mck^1_a + mkc + mk^1_a)$. Since $c \ll k$ and $k \ll n$, the computational complexity of (28) can be rewritten as $\mathcal{O}(\sum_{r=1}^{s+t} mn_rc)$. Similarly, the computational complexity of (29)–(40) are $\mathcal{O}(\sum_{r=1}^{s+t} mn_rc)$, $\mathcal{O}(\sum_{r=1}^{s+t} mn_rc)$, $\mathcal{O}(mn_rc)$, $\mathcal{O}(mn_rc)$, $\mathcal{O}(mn_rc)$, $\mathcal{O}(\sum_{r=1}^{s+t} mk^1_an_r)$, $\mathcal{O}(\sum_{r=1}^{s+t} mk^2_an_r)$, $\mathcal{O}(\sum_{r=1}^{s+t} mk^3_an_r)$, $\mathcal{O}(\sum_{r=1}^{s+t} mk^1_bn_r)$, $\mathcal{O}(\sum_{r=1}^{s+t} mk^2_bn_r)$, $\mathcal{O}(\sum_{r=1}^{s+t} mk^3_bn_r)$ and $\mathcal{O}(mkn_r)$ respectively. Then, the maximal computational intensity in each round of iteration is $\mathcal{O}(\sum_{r=1}^{s+t} mkn_r)$. In summary, the computational complexity of Algorithm 1 is $\mathcal{O}(\sum_{r=1}^{s+t} maxIter \cdot mkn_r)$.

## 5. Experimental evaluation

In this section, we select 20-Newsgroups and Sentiment Data as benchmark datasets, then compare MBTL with other popular supervised, semi-supervised, and domain adaptation approaches.

### 5.1. Data preparation

Firstly, we demonstrate the effectiveness of MBTL on topic classification tasks with 20-Newsgroups dataset. Then we produce multi-source sentiment tasks to show that MBTL which is topic-oriented classification algorithm can also deal with the sentiment classification tasks.

**20-Newsgroups**[1] includes approximately 20,000 newsgroup examples evenly distributed in 20 different news-groups [1,18–20]. Some of newsgroups are similar and can be classified into a top category, e.g., the four subcategories *talk.politics.guns, talk.politics.Mideast, talk.politics. misc* and *talk.religion.misc* belong to the top category *talk*. The details of these top categories and subcategories are demonstrated in Table 4. We select two top categories *rec* and *sci* as positive class and negative class respectively. To produce the source domain, we randomly choose two subcategories from *rec* and *sci* respectively. The target domain is constructed similarly. Thus, $144(P^2_4 \times P^2_4)$ tasks for the data set *rec vs. sci* are produced in this way. Similarly, we also construct 720 tasks for the data set *rec vs. talk, sci vs. talk, comp vs. rec, comp vs. sci* and *comp vs. talk* respectively. Then, totally 864 topic classification tasks are generated.

**Sentiment data**[2] includes the positive and negative reviews from *books, dvd, electronics* and *kitchen* domains. To show the adaptability of MBTL on multi-source sentiment classification tasks, we generate the tasks with two source domains selected randomly from the four domains, and one target domain selected from the rest two domains. Then, 12 multi-source tasks are generated. To demonstrate MBTL can deal with tasks that have less examples, we only select 400 positive and 400 negative examples for the source domain, and 200 positive and 200 negative examples for the target domain.

### 5.2. Experimental setting

**Compared algorithms:** We compare MBTL with several state-of-the-art methods in the experiments: (1) Supervised method, including Logistic Regression (LR). It uses the source domain data to train, and utilizes the target domain data to test. (2) Semi-supervised method, including Non-negative Matrix Tri-Factorization (NMTF). It is trained on both of the source and the target domain data. (3) Transfer learning methods, including SFA [26], Dual Transfer Learning (DTL) [5], Tri-TL [6] and HIDC [7].

**Parameter setting:** Since it is extremely difficult to formalize the latent factors and quantify the relationships between the latent factors and the latent feature spaces, our method can not automatically tune the optimal number of the high-level latent feature spaces. Therefore, we evaluate MBTL on our data sets by empirically searching the parameter space for the optimal parameter setting, and show the details in Section 5.5. Additionally, we set $k^1_a = 9$, $k^1_b = 9$, $k^2_a = 10$, $k^2_b = 10$, $k^3_a = 11$, $k^3_b = 11$, and maxIter $= 200$. The baseline method LR is implemented by Matlab[3], NMTF is given by [17]. The parameters of SFA, DTL, Tri-TL and HIDC are set as the default ones in their papers.

We utilize the classification accuracy which is widely used as the evaluation metric,

$$Accuracy = \frac{|\{d : d \in D \wedge f(d) = y(d)\}|}{n}$$

where $y(d)$ is the true label of example $d$ , $f(d)$ is the label predicted by the classification model and $n$ is the number of the examples. To check out the classification results comprehensively, we also utilize a common used evaluation metric $F_1 - Measure$.

$$F_1 - Measure = (NF_1 + PF_1)/2$$

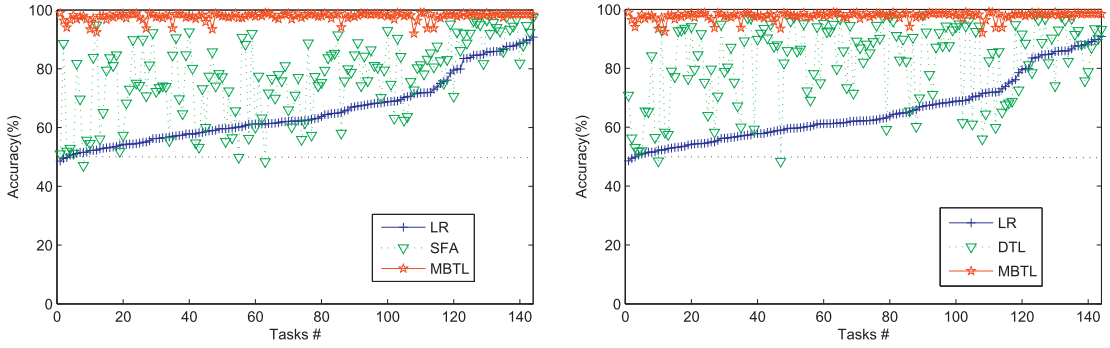
| | Predicted positive pairs | Predicted negative pairs |
|---|---|---|
| Positive pairs | a | b |
| Negative pairs | c | d |

where $NF_1(F_1$ on negative extractions$) = (2 \cdot NP \cdot NR)/(NP + NR)$, $PF_1(F_1$ on positive extractions$) = (2 \cdot PP \cdot PR)/(PP + PR)$, $NR$(recall on negative extractions$) = d/(d + c)$, $NP$(precision on negative extractions$) = d/(d + b)$, $PR$(recall on positive extractions$) = a/(a + b)$, $PP$(precision on positive extractions$) = a/(a + c)$. Additionally, we also adapt AUC to estimate the performance of MBTL on the sentiment tasks.

### 5.3. Experimental results

In this section, We compare our MBTL with LR, NMTF, SFA, DTL, Tri-TL and HIDC on the topic classification tasks and the multi-source sentiment classification tasks. Then we show the results in Fig. 3, and Tables 5 and 6. In Table 5 and 6, "Num of NT" represents the number of negative transfer. In Fig. 3, we sort the tasks on the data set *rec vs. sci* according to the ascending order

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/.
[2] http://www.cs.jhu.edu/mdredze/datasets/sentiment/.
[3] http://www.kyb.tuebingen.mpg.de/bs/people/pgehler/code/index.html.




(a) MBTL vs. LR, SFA

(b) MBTL vs. LR, DTL

(c) MBTL vs. LR, Tri-TL

(d) MBTL vs. LR, HIDC

**Fig. 3.** The performance comparison among LR, SFA, DTL, Tri-TL, HIDC and MBTL on the data set *rec vs. sci*.

**Table 5**
Average performances (%) on 864 topic classification tasks(10 repeated experiments).

| | | LR | NMTF | SFA | DTL | Tri-TL | HIDC | MBTL |
|---|---|---|---|---|---|---|---|---|
| rec vs. sci | Accuracies | 65.57 | 70.86 | 77.11 | 82.23 | 94.65 ± 0.02 | 95.25 ± 0.02 | **97.79 ± 0.01** |
| | $F_1$-Measure | 63.45 | 64.47 | 76.41 | 80.94 | 94.37 ± 0.01 | 95.21 ± 0.02 | **97.76 ± 0.01** |
| | Num of NT | – | – | 25 | 21 | 1 | 1 | **0** |
| sci vs. talk | Accuracies | 70.88 | 72.39 | 77.86 | 84.77 | 89.55 ± 0.03 | 93.58 ± 0.01 | **95.22 ± 0.01** |
| | $F_1$-Measure | 70.03 | 67.87 | 76.65 | 83.81 | 88.76 ± 0.02 | 93.46 ± 0.01 | **95.16 ± 0.00** |
| | Num of NT | – | – | 24 | 19 | 6 | **2** | **2** |
| rec vs. talk | Accuracies | 72.17 | 81.15 | 83.45 | 92.77 | 95.13 ± 0.01 | 95.54 ± 0.00 | **97.96 ± 0.00** |
| | $F_1$-Measure | 71.21 | 78.49 | 81.83 | 92.4 | 95.07 ± 0.00 | 95.42 ± 0.00 | **97.91 ± 0.00** |
| | Num of NT | – | – | 7 | 3 | **0** | **0** | **0** |
| comp vs. rec | Accuracies | 79.25 | 83.17 | 84.87 | 91.78 | 95.62 ± 0.02 | 97.23 ± 0.01 | **97.99 ± 0.01** |
| | $F_1$-Measure | 77.86 | 79.99 | 81.13 | 91.24 | 95.57 ± 0.01 | 97.19 ± 0.01 | **97.96 ± 0.01** |
| | Num of NT | – | – | 32 | 20 | 2 | **0** | **0** |
| comp vs. sci | Accuracies | 68.72 | 67.59 | 76.79 | 82.23 | 88.01 ± 0.05 | 90.79 ± 0.05 | **92.48 ± 0.07** |
| | $F_1$-Measure | 67.42 | 60.18 | 74.23 | 80.4 | 87.64 ± 0.04 | 90.73 ± 0.05 | **92.39 ± 0.06** |
| | Num of NT | – | – | 33 | 24 | 2 | 1 | **0** |
| comp vs. talk | Accuracies | 92.04 | 83.12 | 92.78 | 92.84 | 91.95 ± 0.04 | 95.5 ± 0.02 | **98.29 ± 0.01** |
| | $F_1$-Measure | 91.8 | 81.19 | 92.45 | 92.67 | 91.89 ± 0.03 | 95.4 ± 0.02 | **98.24 ± 0.01** |
| | Num of NT | – | – | 44 | 48 | 66 | 28 | **0** |

**Table 6**
Average performances (%) on 12 multi-source sentiment tasks(10 repeated experiments).

| | LR | NMTF | SFA | DTL | Tri-TL | HIDC | MBTL |
|---|---|---|---|---|---|---|---|
| Accuracies | 74.23 | 58.75 | 75.21 | 72.0 | 58.27 ± 0.07 | 71.31 ± 0.02 | **75.88 ± 0.05** |
| $F_1$-Measure | 74.14 | 58.52 | 75.12 | 71.94 | 57.26 ± 0.06 | 71.26 ± 0.02 | **75.75 ± 0.04** |
| AUC | 81.06 | 67.5 | **82.72** | 78.5 | 68.91 ± 0.9 | 77.23 ± 0.02 | 81.52 ± 0.08 |
| Num of NT | – | – | **3** | 9 | 12 | 10 | **3** |

**Table 7**
Parameter settings of MBTL variants.

| | The number of identical high-level concepts | The number of homogeneous high-level concepts |
|---|---|---|
| MBTL-space1 | 9 | 9 |
| MBTL-space2 | 10 | 10 |
| MBTL-space3 | 11 | 11 |
| MBTL4 | 8/9/10/11 | 8/9/10/11 |

of the performance of LR and utilize a horizontal line to distinguish whether occurring negative transfer. Negative transfer happens when the source domain data and task contribute to the reduced performance of learning in the target domain [1]. In this paper, we consider that a transfer learning method happens negative transfer on a classification task when the accuracy is lower than the probability of random selection or lower than the accuracy of the traditional machine learning method LR. Specifically, since we only focus on binary classification in the experiments, when the accuracy is lower than 50%, which is below the horizontal line in Fig. 3, or lower than the accuracy of LR, it means that negative transfer happens.

(1) *Comparison on the topic classification tasks:* For the tasks on the data set 20-Newsgroups, we can observe from the results in Table 5, that MBTL obtains the best average performance. Additionally, we can find that all the compared transfer algorithms occur negative transfer frequently. And MBTL happens negative transfer only two times on this data set. The reason that MBTL obtains satisfying performances is two-fold. Firstly, more useful latent factors in different latent spaces are modeled together to enhance the transfer capability. Secondly, by constructing multiple latent spaces and learning the corresponding distributions, MBTL can utilize some special latent factors, which represent the discrepancy of the distributions in the corresponding latent space, to reduce the divergence of the distributions in the other latent spaces. In particular, when the distribution is dominated by these latent factors and the distribution divergences among domains are so large, MBTL may avoid negative transfer. In addition, DTL is better than LR and NMTF. This shows that traditional machine learning methods may fail in transfer learning tasks. On the other hand, Tri-TL and HIDC outperform DTL, the reason may be that learning more distributions can fit different situations on the data distributions.

(2) *Comparison on the sentiment classification tasks:* For the multi-source sentiment classification tasks, we can observe that the performance of MBTL is better than all the topic-oriented methods. In addition, DTL is better than HIDC, HIDC outperforms Tri-TL and NMTF, and LR outperforms all the compared topic-oriented methods. From the results in Table 6, we find that all the compared topic-oriented transfer methods which can deal with topic classification tasks fail in these tasks. Only MBTL, which is also the topic-oriented algorithm, outperforms the traditional machine learning method LR and seldom happens negative transfer.
Although MBTL is topic-oriented classification algorithm, to further validate the adaptability of it, we compare MBTL with SFA which is classic sentiment algorithms. From the results in Table 6, we can find that MBTL obtains satisfactory performance.

### 5.4. Effectiveness of multi-bridge transfer learning

In this section, to validate the effectiveness of MBTL, we construct three single bridge learning methods, which are variants of MBTL. Actually, MBTL-space1, MBTL-space2 and MBTL-space3 represent the models trained in the three different latent spaces respectively. Moreover, we also construct a variant of MBTL (MBTL4), which learns the distributions in four latent spaces together. The parameter settings of these algorithms are shown in Table 7. And we show the results in Table 8. Then, we conduct a Mann–Whitney U Test using the accuracies on each tasks and set statistical significance to 95% (i.e., p-value = 0.05). Table 9 shows the Mann–Whitney U Test results in the form of win/tie/loss on all the tasks including 864 topic classification tasks and 12 multi-source sentiment tasks.

From the results in Tables 8 and 9, firstly, we can find that the multi-bridge methods(MBTL, MBTL4) outperform these single bridge learning methods on all the tasks. It means that the strategy of multi-bridge can improve the classification performance effectively. Secondly, we can find that the performance of MBTL is close to MBTL4 on all the tasks. Since the computational complexity is proportional to the number of high-level concepts, we set the number of the latent feature spaces to three for a more comprehensive effectiveness.

**Table 8**
Average performances (%) comparison between MBTL and the variants.

| | | MBTL-space1 | MBTL-space2 | MBTL-space3 | MBTL | MBTL4 |
|---|---|---|---|---|---|---|
| rec vs. sci | Accuracies | 96.96 ± 0.02 | 97.09 ± 0.02 | 96.47 ± 0.01 | **97.79 ± 0.01** | 97.36 ± 0.01 |
| | $F_1$-Measure | 96.93 ± 0.01 | 97.06 ± 0.02 | 96.44 ± 0.01 | **97.76 ± 0.01** | 97.33 ± 0.00 |
| | Num of NT | 1 | 1 | 2 | **0** | 1 |
| sci vs. talk | Accuracies | 94.58 ± 0.01 | 95.20 ± 0.01 | 95.10 ± 0.02 | 95.22 ± 0.01 | **95.37 ± 0.02** |
| | $F_1$-Measure | 94.53 ± 0.01 | 95.15 ± 0.01 | 95.05 ± 0.02 | 95.16 ± 0.00 | **95.3 ± 0.01** |
| | Num of NT | 4 | 3 | 3 | 2 | **0** |
| rec vs. talk | Accuracies | 96.34 ± 0.00 | 96.41 ± 0.01 | 96.42 ± 0.01 | **97.96 ± 0.00** | 96.67 ± 0.00 |
| | $F_1$-Measure | 96.28 ± 0.00 | 96.35 ± 0.00 | 96.36 ± 0.00 | **97.91 ± 0.00** | 96.6 ± 0.00 |
| | Num of NT | **0** | **0** | **0** | **0** | **0** |
| comp vs. rec | Accuracies | 97.52 ± 0.01 | 97.19 ± 0.02 | 97.4 ± 0.01 | 97.99 ± 0.01 | **98.17 ± 0.01** |
| | $F_1$-Measure | 97.49 ± 0.00 | 97.19 ± 0.02 | 97.37 ± 0.01 | 97.96 ± 0.01 | **98.14 ± 0.01** |
| | Num of NT | 1 | 2 | 1 | **0** | **0** |
| comp vs. sci | Accuracies | 91.7 ± 0.05 | 91.52 ± 0.05 | 91.64 ± 0.06 | 92.48 ± 0.07 | **93.7 ± 0.08** |
| | $F_1$-Measure | 91.59 ± 0.05 | 91.42 ± 0.04 | 91.53 ± 0.05 | 92.39 ± 0.06 | **93.67 ± 0.07** |
| | Num of NT | **0** | 1 | **0** | **0** | **0** |
| comp vs. talk | Accuracies | 97.52 ± 0.01 | 97.36 ± 0.01 | 97.32 ± 0.00 | **98.29 ± 0.01** | 98.26 ± 0.01 |
| | $F_1$-Measure | 97.44 ± 0.00 | 97.23 ± 0.01 | 97.24 ± 0.00 | **98.24 ± 0.01** | 98.21 ± 0.00 |
| | Num of NT | 7 | 9 | 10 | **0** | **0** |
| Multi-Source Sentiment Tasks | Accuracies | 72.23 ± 0.05 | 70.79 ± 0.07 | 71.71 ± 0.06 | 75.87 ± 0.05 | **77.45 ± 0.03** |
| | $F_1$-Measure | 72.16 ± 0.04 | 70.65 ± 0.07 | 71.62 ± 0.05 | 75.75 ± 0.04 | **77.36 ± 0.03** |
| | Num of NT | 9 | 10 | 10 | 3 | **2** |

Please cite this article as: X. Hu et al., Multi-bridge transfer learning, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.01.016

**Table 9**
Mann–Whitney U test results (p-Value = 0.05).

| | | MBTL-space1 | MBTL-space2 | MBTL-space3 |
|---|---|---|---|---|
| rec vs. sci | MBTL | 129/10/5 | 131/10/3 | 127/11/6 |
| | MBTL4 | 130/7/7 | 127/8/9 | 125/13/6 |
| sci vs. talk | MBTL | 109/15/20 | 97/29/18 | 101/26/17 |
| | MBTL4 | 134/5/5 | 133/10/1 | 135/7/2 |
| rec vs. talk | MBTL | 124/18/2 | 118/21/5 | 119/22/3 |
| | MBTL4 | 88/20/36 | 90/15/39 | 88/13/43 |
| comp vs. rec | MBTL | 104/23/17 | 101/22/21 | 102/16/26 |
| | MBTL4 | 85/32/27 | 71/43/30 | 78/31/35 |
| comp vs. sci | MBTL | 133/10/1 | 127/13/4 | 130/8/6 |
| | MBTL4 | 84/13/47 | 86/12/46 | 93/8/43 |
| comp vs. talk | MBTL | 73/29/42 | 71/27/46 | 70/30/44 |
| | MBTL4 | 98/14/32 | 101/11/32 | 104/11/29 |
| Total ratios | MBTL | 77.8%/12.2%/10.0% | 74.7%/14.1%/11.2% | 75.1%/13.1%/ 11.8% |
| | MBTL4 | 71.7%/10.5%/17.8% | 70.4%/11.4%/18.2% | 72.1%/9.6%/18.3% |
| Multi-source | MBTL | 10/0/2 | 9/1/2 | 9/0/3 |
| Sentiment tasks | MBTL4 | 10/1/1 | 10/0/2 | 11/0/1 |
| Total ratios | MBTL | 83.4%/ 00.0 %/ 16.6% | 75.0%/ 8.3%/ 16.7 % | 75.0 %/ 0.0%/ 25.0 % |
| | MBTL4 | 83.4%/ 8.3 %/ 8.3% | 83.4%/ 00.0 %/ 16.6% | 91.7 %/ 00.0%/ 8.3% |

**Table 10**
The parameter influence on performance (%) of algorithm MBTL.

| Sampling ID | $k_a^1$ | $k_b^1$ | $k_a^2$ | $k_b^2$ | $k_a^3$ | $k_b^3$ | Problem ID | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | 9 | 7 | 14 | 11 | 8 | 9 | 96.92 | 97.68 | 98.23 | 97.77 | 98.38 | 98.23 | 98.79 | 98.69 | 98.59 |
| 2 | 10 | 13 | 12 | 8 | 10 | 12 | 96.46 | 97.83 | 98.18 | 97.62 | 98.53 | 97.82 | 98.74 | 98.39 | 98.44 |
| 3 | 14 | 15 | 13 | 7 | 9 | 11 | 96.56 | 97.93 | 98.23 | 98.03 | 98.43 | 98.03 | 98.84 | 98.34 | 98.49 |
| 4 | 12 | 14 | 6 | 10 | 7 | 13 | 96.56 | 97.37 | 98.08 | 97.72 | 98.33 | 97.87 | 98.59 | 98.39 | 98.39 |
| 5 | 15 | 8 | 8 | 12 | 5 | 7 | 96.31 | 97.68 | 97.88 | 97.82 | 98.53 | 97.72 | 98.59 | 98.34 | 98.54 |
| 6 | 8 | 6 | 10 | 7 | 9 | 13 | 96.66 | 97.88 | 98.08 | 97.87 | 98.43 | 97.93 | 98.84 | 98.44 | 98.39 |
| 7 | 11 | 9 | 7 | 8 | 12 | 14 | 96.66 | 97.68 | 98.03 | 97.92 | 98.48 | 97.87 | 98.59 | 98.54 | 98.49 |
| 8 | 5 | 8 | 14 | 12 | 11 | 13 | 96.56 | 97.63 | 98.23 | 97.92 | 98.33 | 98.03 | 98.84 | 98.49 | 98.59 |
| 9 | 7 | 11 | 9 | 8 | 13 | 10 | 96.06 | 97.73 | 98.08 | 97.92 | 98.28 | 97.93 | 98.84 | 98.59 | 98.39 |
| Mean | | | | | | | 96.53 | 97.71 | 98.11 | 97.85 | 98.42 | 97.94 | 98.74 | 98.47 | 98.47 |
| Variance | | | | | | | 0.058 | 0.027 | 0.014 | 0.015 | 0.008 | 0.021 | 0.014 | 0.015 | 0.007 |
| This paper | 9 | 9 | 10 | 10 | 11 | 11 | 95.90 | 97.53 | 98.33 | 97.57 | 98.64 | 97.87 | 98.79 | 98.44 | 98.34 |

**Table 11**
Running time of MBTL and other compared methods (s)

| | LR | NMTF | SFA | DTL | Tri-TL | HIDC | MBTL | MBTL4 |
|---|---|---|---|---|---|---|---|---|
| Topic classification task | 11.4 | 14.1 | 26.7 | 21.1 | 45.2 | 771.3 | 75.6 | 107.9 |
| Sentiment classification task | 17.9 | 24.5 | 34.3 | 32.3 | 95.1 | 1153.1 | 127.4 | 169.1 |

### 5.5. Parameter sensitivity

In this section, we will investigate the parameter sensitivity of MBTL with six parameters, $k_a^1$,$k_a^2$,$k_a^3$, $k_b^1$,$k_b^2$,$k_b^3$, which represent the numbers of identical and homogeneous concepts in three latent spaces respectively, the detail description is shown in Table 3. To prove that MBTL is stable when the parameters vary in a widely range, we randomly select 9 combinations of parameter when $k_a^1 \in [5, 15]$, $k_a^2 \in [5, 15]$, $k_a^3 \in [5, 15]$, $k_b^1 \in [5, 15]$, $k_b^2 \in [5, 15]$ and $k_b^3 \in [5, 15]$, then investigate them on 9 randomly selected tasks on the data set *rec vs. sci*. From the results shown in Table 10, we can observe that the average accuracy of 9 parameter combinations on each selected task is almost the same as the one using the default parameters, and the variance is very small. Therefore, MBTL is generally not sensitive to the parameters which are selected from the predefined bounds.

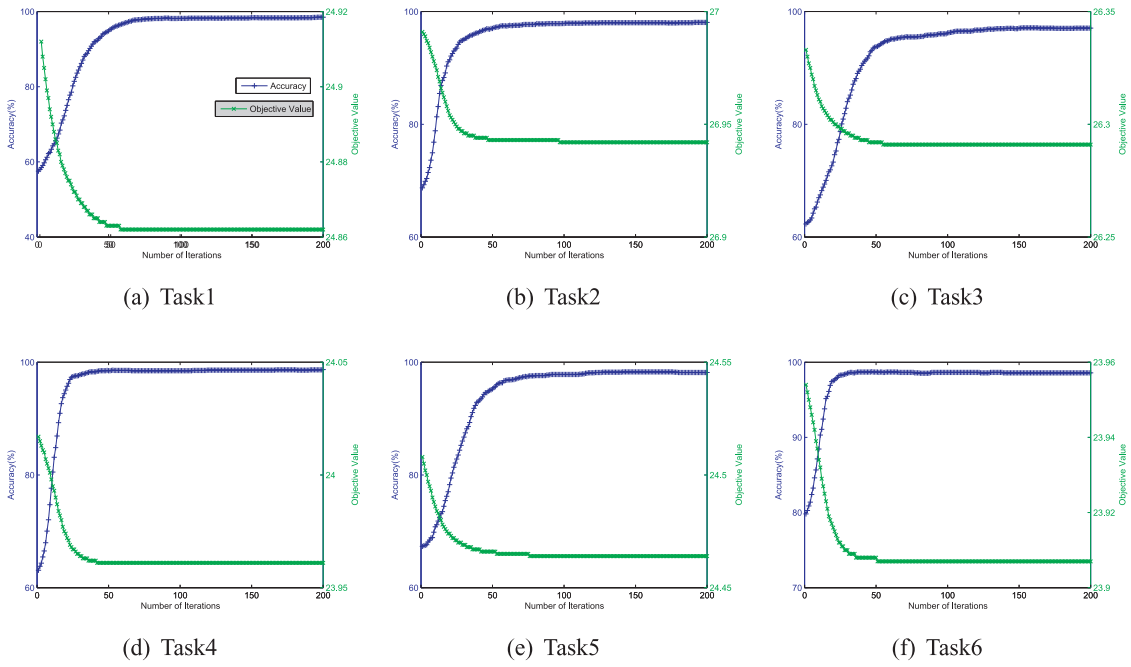
### 5.6. Running time

We empirically check the running time of all the methods on two tasks, which are selected randomly from topic classification tasks and sentiment tasks respectively. Experimental results are shown in Table 11. We can find that the running time of MBTL is within acceptable limits[4]. Additionally, from the results in Tables 8 and 9, we can see that the performances of MBTL and MBTL4 are approximate. But the running time of MBTL4 is more than MBTL. The reason is that MBTL4 constructs more latent spaces and contains more feature clusters.

### 5.7. Algorithm convergence

In this section, we check the convergence of MBTL on 6 tasks on the data set *rec vs. sci* chose randomly. In Fig. 4, the left and right *y*-axis indicate the prediction accuracy and the objective value in Eq. (13) respectively, and the *x*-axis indicates the number of iterations. In Fig. 4, we can observe that the prediction accuracy of MBTL increases with more iterations and the objective value decreases with more iterations conversely, and both of them converge within 200 iterations.

[4] The configuration of computing platform: Intel Core i5-3470s CPU 2.9 GHz, RAM 8.0 GB.




(a) Task1 (b) Task2 (c) Task3

(d) Task4 (e) Task5 (f) Task6

**Fig. 4.** The performance of MBTL and objective value vs. the number of iterations.

## 6. Conclusion

In this paper, we systemically analyze the effectiveness of MBTL, and proposed a general cross-domain learning model based on a non-negative matrix tri-factorization technology. This model constructs multiple latent spaces and learns the corresponding distributions to establish multiple bridges for knowledge transfer. Then an effective algorithm is proposed to derive the solution to the optimization problem. Finally, we conduct comprehensive experiments to show that MBTL outperforms all the comparison methods.

## Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant No. 61305063,61273292) and the Specialized Research Fund for the Doctoral Program of Higher Education under grant 20130111110011. The authors would like to thank the anonymous reviewers for their valuable and constructive comments.

## References

[1] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.
[2] J. Blitzer, R. McDonald, F. Pereira, Domain adaptation with structural correspondence learning, in: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2006, pp. 120–128.
[3] W.Y. Dai, G.R. Xue, Q. Yang, Y. Yu, Co-clustering based classification for out-of-domain documents, in: Proceedings of the 13th ACM SIGKDD, 2007, pp. 210–219.
[4] F.Z. Zhuang, P. Luo, H. Xiong, Q. He, Y.H. Xiong, Z.Z. Shi, Exploiting associations between word clusters and document classes for cross-domain text categorization, in: Proceedings of the 10th SIAM SDM, 2010, pp. 13–24.
[5] M. Long, J. Wang, G. Ding, W. Cheng, X. Zhang, W. Wang, Dual transfer learning, in: Proceedings of the 12th SIAM SDM, 2012, pp. 540–551.
[6] F.Z. Zhuang, P. Luo, C.Y. Du, Q. He, Z.Z. Shi, Triplex transfer learning: Exploiting both shared and distinct concepts for text classification, in: Proceedings of the 5th Web Search and Data Mining, 2013, pp. 425–434.
[7] F.Z. Zhuang, P. Luo, P.F. Yin, Q. He, Z.Z. Shi, Concept learning for cross-domain text classification: A general probabilistic framework, in: Proceedings of the 23th IJCAI, 2013, pp. 1960–1966.
[8] D. Hosmer, S. Lemeshow, Applied Logistic Regression, John Wiley, 2004.
[9] T. Hofmann, Unsupervised learning by probabilistic latent semantic analysis, J. Mach. Learn. 42 (1–2) (2001) 177–196.
[10] T. Li, V. Sindhwani, C. Ding, Y. Zhang, Bridging domains with words: Opinion analysis with matrix tri-factorizations, in: Proceedings of the 2010 SIAM International Conference on Data Mining (SDM), 2010, pp. 293–302.
[11] Q. Liu, A.J. Mackey, D.S. Roos, F.C.N. Pereira, Evigan: a hidden variable model for integrating gene evidence for eukaryotic gene prediction, Bioinformatics 24 (5) (2008) 597–605.
[12] Y. Zhu, Y. Chen, Z. Lu, S.J. Pan, G.R. Xue, Y. Yu, Q. Yang, Heterogeneous transfer learning for image classification, in: Proceedings of the 25th AAAI, 2011.
[13] F.Z. Zhuang, P. Luo, Z. Shen, Q. He, Y. Xiong, Z.Z. Shi, H. Xiong, Collaborative dual-PLSA: mining distinction and commonality across multiple domains for text classification, in: Proceedings of the 19th ACM CIKM, 2010, pp. 359–368.
[14] H. Wang, H. Huang, F. Nie, C. Ding, Cross-language web page classification via dual knowledge transfer using nonnegative matrix tri-factorization, in: Proceedings of the 34th ACM SIGIR, 2011, pp. 933–942.
[15] D.D. Lee, H.S. Seung, Algorithms for non-negative matrix factorization, in: Proceedings of Advances in neural information processing systems(NIPS), 2000, pp. 556–562.
[16] C. Ding, T. Li, W. Peng, H. Park, Orthogonal nonnegative matrix tri-factorizations for clustering, in: Proceedings of the 12th ACM SIGKDD, 2006, pp. 126–135.
[17] Z. Chen, W.X. Zhang, Domain adaptation with topic correspondence learning, in: Proceedings of the 23th IJCAI, 2013, pp. 1280–1286.
[18] D. Zhang, J. He, Y. Liu, L. Si, R.D. Lawrence, Multi-view transfer learning with a large margin approach, in: Proceedings of the 17th ACM SIGKDD, 2011, pp. 1208–1216.
[19] W.Y. Dai, Q. Yang, G.R. Xue, Y. Yu, Boosting for transfer learning, in: Proceedings of the 24th ICML, 2007, pp. 193–200.
[20] J. Gao, W. Fan, J. Jiang, J.W. Han, Knowledge transfer via multiple model local structure mapping, in: Proceedings of the 14th ACM SIGKDD, 2008, pp. 283–291.
[21] J. Jiang, C.X. Zhai, A two-stage approach to domain adaptation for statistical classifiers, in: Proceedings of the 16th ACM CIKM, 2007, pp. 401–410.
[22] S. Uguroglu, J. Carbonell, Feature selection for transfer learning, in: Machine Learning and Knowledge Discovery in Databases, 2011, pp. 430–442.
[23] S.J. Pan, J.T. Kwok, Q. Yang, Transfer learning via dimensionality reduction, in: Proceedings of the 23rd AAAI, 2008, pp. 677–682.
[24] M. Long, J. Wang, G. Ding, D. Shen, Q. Yang, Transfer learning with graph co-regularization, in: Proceedings of the 26th AAAI, 2012.
[25] J. Jiang, C.X. Zhai, Instance weighting for domain adaptation in nlp, in: Proceedings of the 45th ACL, 2007, pp. 264–271.
[26] S.J. Pan, X. Ni, J.T. Sun, Q. Yang, Z. Chen, Cross-domain sentiment classification via spectral feature alignment, in: Proceedings of the 19th international conference on World wide web, 2010, pp. 751–760.
[27] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, G. Zhang, Transfer learning using computational intelligence: A survey, Knowl. Based Syst. 80 (2015) 14–23.
[28] R. Caruana, Multitask learning, Mach. Learn. 28 (1) (1997) 41–75.
[29] D.L. Silver, R.E. Mercer, The task rehearsal method of life-long learning: overcoming impoverished data, Adv. Artif. Intell. (2002) 90–101.
[30] D.L. Silver, R.E. Mercer, Sequential inductive transfer for coronary artery disease diagnosis, in: Proceedings of the International Joint Conference on Neural Networks (IJCNN), Orlando, USA, 2007, pp. 2635–2641.

[31] S. Chopra, S. Balakrishnan, R. Gopalan, DLID: deep learning for domain adaptation by interpolating between domains, in: Proceedings of the ICML Workshop on Challenges in Representation Learning, vol. 2:5, 2013.
[32] P. Swietojanski, A. Ghoshal, S. Renals, Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR, in: Proceedings of the IEEE Workshop on Spoken Language Technology (SLT), Miami, USA, 2012.
[33] D.M. Roy, L.P. Kaelbling, Efficient Bayesian task-level transfer learning, in: Proceedings of the International Joint Conference on Artificial Intelligence, Hyderabad, India, 2007.
[34] V. Behbood, J. Lu, G. Zhang, Long term bank failure prediction using fuzzy refinement-based transductive transfer learning, in: Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ), Taipei, Taiwan, 2011.
[35] V. Behbood, J. Lu, G. Zhang, Fuzzy bridged refinement domain adaptation: long-term bank failure prediction, Int. J. Comput. Intell. Appl. 12 (01) (2013).
[36] S. Tan, X. Cheng, Y. Wang, H. Xu, Adapting naive bayes to domain adaptation for sentiment analysis, in: Proceedings of the 31st European Conference on Advances in Information Retrieval, Toulouse, France, 2009.
[37] W. Dai, G. Xue, Q. Yang, Y. Yu, Transferring naive Bayes classifiers for text classification, in: Proceedings of the 22nd National Conference on Artificial Intelligence, Vancouver, Canada, 2007.
[38] M. Dredze, A. Kulesza, K. Crammer, Multi-domain learning by confidence-weighted parameter combination, Mach. Learn. 79 (1) (2010) 123–149.
[39] J. Shi, M. Long, Q. Liu, G. Ding, J. Wang, Twin bridge transfer learning for sparse collaborative filtering, Adv. Knowl. Disc. Data Mining (2013) 496–507.
[40] B. Li, Q. Yang, X. Xue, Transfer learning for collaborative filtering via a Rating-Matrix generative model, in: Proceedings of the 26th International Conference on Machine Learning, 2009.
[41] B. Li, Q. Yang, X. Xue, Can movies and books collaborate? cross-domain collaborative filtering for sparsity reduction, in: Proceedings of the 21st International Joint Conference on Artificial Intelligence, 2009.

Please cite this article as: X. Hu et al., Multi-bridge transfer learning, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.01.016