Probabilistic Latent Variable Models for Unsupervised Many-to-Many Object Matching

Tomoharu Iwata[a,*], Tsutomu Hirao[a], Naonori Ueda[a]

[a] *NTT Communication Science Laboratories*

Abstract

Object matching is an important task for finding the correspondence between objects in different domains, such as documents in different languages and users in different databases. In this paper, we propose probabilistic latent variable models that offer many-to-many matching without correspondence information or similarity measures between different domains. The proposed model assumes that there is an infinite number of latent vectors that are shared by all domains, and that each object is generated from one of the latent vectors and a domain-specific projection. By inferring the latent vector used for generating each object, objects in different domains are clustered according to the vectors that they share. Thus, we can realize matching between groups of objects in different domains in an unsupervised manner. We give learning procedures of the proposed model based on a stochastic EM algorithm. We also derive learning procedures in a semi-supervised setting, where correspondence information for some objects are given. The effectiveness of the proposed models is demonstrated by experiments on synthetic and real data sets.

Keywords: latent variable model, mixture model, object matching

*Corresponding author

Email addresses: iwata.tomoharu@lab.ntt.co.jp (Tomoharu Iwata), hirao.tsutomu@lab.ntt.co.jp (Tsutomu Hirao), ueda.naonori@lab.ntt.co.jp (Naonori Ueda)

1. Introduction

Object matching is an important task for finding the correspondence between objects in different domains. Examples of object matching include matching an image with an annotation [1], an English word with a French word [2], and user identification in different databases for recommendation [3]. Most object matching methods require similarity measures between objects in the different domains, or paired data that contain correspondence information. When a similarity measure is given, we can match objects by finding pairs of objects that maximize the sum of the similarities. When correspondence information is given, we can obtain a mapping function from one domain to another by using supervised learning methods, and then we can calculate the similarities between objects in different domains.

However, similarity measures and correspondences might not be available. Defining similarities and generating correspondences incur considerable cost and time, and they are sometimes unobtainable because of the need to preserve privacy. For example, dictionaries between some languages might not exist, and different online stores cannot share user identification. For these situations, unsupervised object matching methods have been proposed; they include kernelized sorting [4], least squares object matching [5], matching canonical correlation analysis [6], and its Bayesian extension [7, 8]. These methods find one-to-one matches. However, matching is not necessarily one-to-one in some applications. For example, when matching English and German documents, multiple English documents with the similar topic could correspond to multiple German documents. In image annotation, related annotations 'tree', 'wood' and 'forest' can be attached to multiple images that look similar to each other. Other limitations of these methods are that the number of domains is limited to two, and the numbers of objects in the different domains must be the same. There can be more than two domains in some applications, for example matching multilingual documents such as English, French and German, and the number of documents for each language can be different.

In this paper, we propose a probabilistic latent variable model for finding correspondence between object clusters in multiple domains without correspondence information. We assume that objects in different domains share a hidden structure, which is represented by an infinite number of latent vectors that are shared by all domains. Each object is generated from one of the latent vectors and a domain-specific linear projection. The latent vectors used for generating objects are unknown. By assigning a latent vector to each object, we can allocate objects in different domains to common clusters, and find many-to-many matches. The number of clusters is automatically inferred from the given data by using a Dirichlet process prior. The proposed model can handle more than two domains with different numbers of objects. We infer the proposed model using a stochastic EM algorithm. The proposed model can ignore arbitrary linear transformations for different domains by inferring the domain-specific linear projection, and can find cluster matching in different domains, where similarity cannot be calculated directly.

The proposed model assumes a Gaussian distribution for each observed variable, and its mean is determined by a latent vector and a linear projection matrix. It is an extension of probabilistic principle component analysis (PCA) [9] and factor analysis (FA) [10], which are representative probabilistic latent variable models. With probabilistic PCA and FA, each object is associated with a latent vector. On the other hand, with the proposed model, the latent vector that is assigned to each object is hidden. When the number of domains is one, and every object is assigned to a cluster that is different from those of other objects, the proposed model corresponds to probabilistic principle component analysis.

The proposed model can be also used in a semi-supervised setting, where correspondence information for some objects is given [4, 11]. The information assists matching by incorporating a condition stating that the cluster assignments of the corresponding objects must be the same. We derive learning procedures for the semi-supervised setting by modifying the learning procedures for unsupervised setting.

This paper is an extended version of [12]. We newly proposed the inference procedure for a semi-supervised setting, and added derivations and experiments. The remainder of this paper is organized as follows. In Section 2, we review related work. In Section 3, we formulate the proposed model and describe its efficient learning procedures. We also present the learning procedures for a semi-supervised setting and for missing data. In Section 4, we demonstrate the effectiveness of the proposed models with experiments on synthetic and real data sets. Finally, we present concluding remarks and a discussion of future work in Section 5.

2. Related work

2.1. Unsupervised object matching

Unsupervised object matching is a task that involves finding the correspondence between objects in different domains without correspondence information. For example, kernelized sorting [4] finds the correspondence by permuting a set to maximize the dependence between two domains where the Hilbert Schmidt Independence Criterion (HSIC) is used as the dependence measure. Kernelized sorting requires the two domains have the same number of objects. Convex kernelized sorting [13] is a convex formulation of kernelized sorting. Matching canonical correlation analysis (MCCA) [6] is another unsupervised object matching method based on a probabilistic model, where bilingual translation lexicons are learned from two monolingual corpora. MCCA simultaneously finds latent variables that represent correspondences and latent vectors so that the latent vectors of corresponding objects exhibit the maximum correlation. [14] also proposed a method for unsupervised object matching that is related to MCCA. These methods assume one-to-one matching of objects. On the other hand, the proposed model can find many-to-many matching, and is applicable to objects in more than two domains. Bayesian solution for MCCA (BMCCA) has been proposed [7, 8]. BMCCA assumes that latent vectors are generated from a Gaussian distribution, and finds one-to-one matching by inferring a permutation

matrix. In contrast, the proposed model assumes that latent vectors are generated from an infinite Gaussian mixture model [15], and finds many-to-many matching by inferring cluster assignments.

Manifold alignment is related to the proposed model because they both find latent vectors of multiple sets in a joint latent space. The unsupervised manifold alignment method [16] finds latent vectors of different domains in a joint latent space in an unsupervised manner. The method first identifies all possible matches for each example by leveraging its local geometry, and then finds an embedding in the latent space. The method requires permutations of the order of the factorial of the size of neighborhoods to match the local geometry. Note that the method does not explicitly find correspondences.

There has been some work on improving the learning performance of the classification task by using labeled objects in different domains without correspondence information. For example, multiple outlook mapping (MOMAP) [17] improves the performance by matching the moments of the empirical distributions for each class of two domains. [18] proposed a transfer learning method that improves the learning performance by embedding both source and target domains in a joint latent space when a limited number of target objects are labeled. These methods require labeled objects. On the other hand, the proposed method does not require any labeled objects.

Consensus clustering [19] tries to find a single clustering from a number of different clusterings. In the case that a clustering is obtained using data in each domain, consensus clustering can find many-to-many matching. However, consensus clustering requires that objects are aligned across domains.

2.2. Latent variable models

The proposed model, as well as MCCA [6], can be considered as an unsupervised version of probabilistic canonical correlation analysis (CCA) [20]. Probabilistic CCA finds dependences between objects in two domains by projecting objects into a latent space. It requires correspondence information between different domains since it takes a set of paired objects as input. On the other hand,

the proposed model can find dependences in an unsupervised manner without correspondence information by taking a set of objects for each domain as input. CCA is being successfully used for a wide variety of applications, such as multi-label prediction [21, 22], information retrieval [23], and image annotation [24]. The proposed model can be used for these applications when supervised data are unavailable.

Mixtures of latent variable models have been also proposed, such as mixtures of probabilistic principal component analyzers [25] and mixtures of factor analyzer [26]. These mixture models assume that each object has its own latent vector, and different objects can be generated from different projection matrices, where the data are obtained from a domain. On the other hand, the proposed model assumes that different objects can share a latent vector, and all objects from a domain are generated from a domain-specific projection matrix. Thus, the proposed model can find many-to-many matching by assigning the same latent vectors to different objects from multiple domain data.

The proposed model can be seen as a generalization of the infinite Gaussian mixture model [15]. When the dimensionality of the latent space is the same as that of the observed space and the projection matrix is the identity matrix for all domains, the proposed model corresponds to the infinite Gaussian mixture model.

Recently, there have been proposed a number of deep learning methods for obtaining latent representations of multimodal data, such as multimodal deep learning [27], multimodal deep Boltzmann machine [28] and deep canonical correlation analysis [29]. These methods utilize deep architectures to model complex nonlinear transformations between latent representations and multimodal observations. However, these methods require correspondence information to learn the transformations, and is unapplicable in an unsupervised setting. We can use deep learning to obtain latent representations for each domain in preprocessing, and apply the proposed method to the latent representations for unsupervised object matching.

Table 1: Notation.

Symbol	Description
D	number of domains
N_d	number of objects in the dth domain
M_d	dimensionality of observed features in the dth domain
K	dimensionality of a latent vector
J	number of clusters (latent vectors) to which objects are assigned
$\boldsymbol{x}_{dn}$	observation of the nth object in the dth domain, $\boldsymbol{x}_{dn} \in \mathbb{R}^{M_d}$
$\boldsymbol{z}_j$	latent vector for the jth cluster, $\boldsymbol{z}_j \in \mathbb{R}^K$
$\boldsymbol{W}_d$	projection matrix for the dth domain, $\boldsymbol{W}_d \in \mathbb{R}^{M_d \times K}$
θ_j	mixture weight for the jth cluster, $\theta_j \geq 0$, $\sum_{j=1}^{\infty} \theta_j = 1$

3. Proposed method

3.1. Model

Suppose that we are given objects in D domains $\boldsymbol{X} = \{\boldsymbol{X}_d\}_{d=1}^{D}$, where $\boldsymbol{X}_d = \{\boldsymbol{x}_{dn}\}_{n=1}^{N_d}$ is a set of objects in the dth domain, and $\boldsymbol{x}_{dn} \in \mathbb{R}^{M_d}$ is the feature vector of the nth object in the dth domain. Our notation is summarized in Table 1. Note that we are unaware of any correspondence between objects in different domains. The number of objects N_d and the dimensionality M_d for each domain can be different from those of other domains. The task is to match groups of objects across multiple domains in an unsupervised manner. Figure 1 shows an example of the input (top) and output (bottom) for the proposed model with two domains, where the numbers of objects in the first and second domains are 50 and 60, and the numbers of observation features are 70 and 50, respectively. The color shows the value of the observation feature. In this case, the proposed model found five clusters, and the object indices are permutated so that matched objects are aligned.

The model proposed for this task is a probabilistic latent variable model. The proposed model assumes that 1) there is a potentially infinite number of clusters, and 2)each cluster j has a latent vector $\boldsymbol{z}_j \in \mathbb{R}^K$ in a K-dimensional

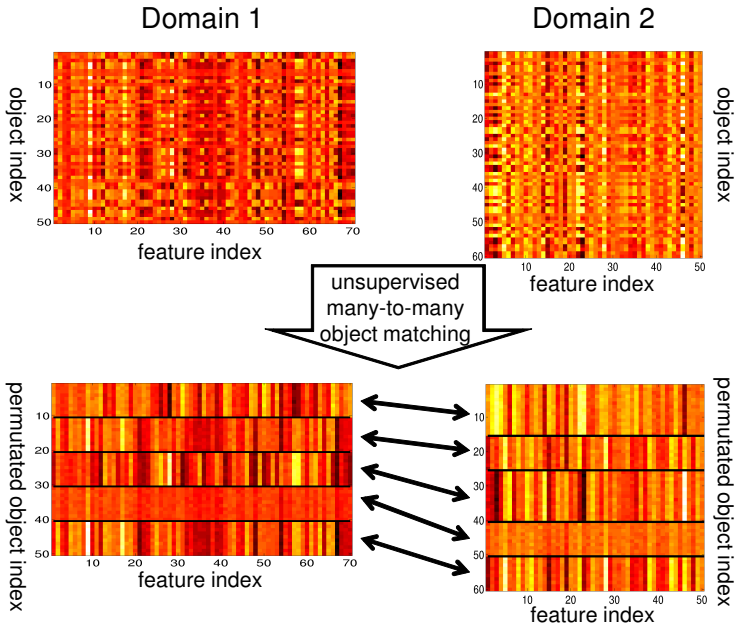


Figure 1: Example of the input (top) and output (bottom) for the proposed model with two domains. The vertical axis represents object indices, and the horizontal axis represents observation feature indices for each matrix. The object indices of the output are permutated so that matched objects are aligned.

latent space. Each object $\boldsymbol{x}_{dn}$ in the dth domain is generated depending on domain-specific projection matrix $\boldsymbol{W}_d \in \mathbb{R}^{M_d \times K}$ and latent vector $\boldsymbol{z}_{s_{dn}}$ that is selected from a set of latent vectors $\boldsymbol{Z} = \{\boldsymbol{z}_j\}_{j=1}^{\infty}$. Here, $s_{dn} \in \{1, \cdots, \infty\}$ is the latent cluster assignment of object $\boldsymbol{x}_{dn}$. Objects that use the same latent vector, or that have the same cluster assignment, are considered to match. Figure 2 shows the relationship between latent vectors and objects in two domains, where arrows that indicate the corresponding latent vectors for each object are hidden.

To be precise, the proposed model is an infinite mixture model, where the probability of object $\boldsymbol{x}_{dn}$ is given by

$$p(\boldsymbol{x}_{dn}|\boldsymbol{Z}, \boldsymbol{W}, \boldsymbol{\theta}) = \sum_{j=1}^{\infty} \theta_j \mathcal{N}(\boldsymbol{x}_{dn}|\boldsymbol{W}_d \boldsymbol{z}_j, \alpha^{-1}\boldsymbol{I}), \tag{1}$$

where $\boldsymbol{W} = \{\boldsymbol{W}_d\}_{d=1}^{D}$ is a set of projection matrices, $\boldsymbol{\theta} = \{\theta_j\}_{j=1}^{\infty}$ is a set of

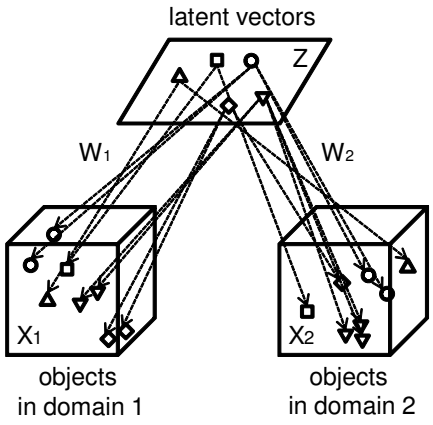


Figure 2: Relationship between latent vectors and objects in two domains.

mixture weights, θ_j represents the probability that the jth cluster is chosen, α is a precision parameter, and $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes a multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. In the proposed model, a set of latent vectors $\boldsymbol{Z}$ is shared among multiple domains, but projection matrix $\boldsymbol{W}_d$ depends on the domain. The benefit of the proposed model is as follows:

- By sharing the latent vectors, we can assign objects in different domains to common clusters, and find matchings between clusters.
- By employing domain-specific projection matrices, we can handle multiple domains with different dimensionalities by adjusting the size of the matrices, and different statistical properties, such as means and covariances, by inferring the values of the matrices from the given data.
- Given latent vectors, an arbitrary number of objects can be generated for each domain independently. Therefore, we can handle domains with different numbers of objects.

Specifically, the proposed model generates objects in multiple domains $\boldsymbol{X}$ according to the following process,

1. Draw mixture weights $\boldsymbol{\theta} \sim \mathrm{Stick}(\gamma)$

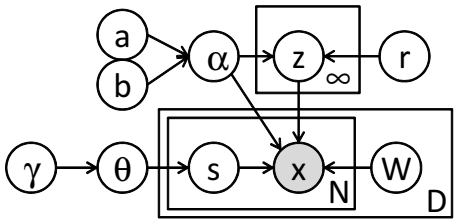


Figure 3: Graphical model representation of the proposed model.

2. Draw a precision parameter $\alpha \sim \mathcal{G}(a, b)$
3. For each cluster: $j = 1, \ldots, \infty$
   (a) Draw a latent vector $\boldsymbol{z}_j \sim \mathcal{N}(\mathbf{0}, (\alpha r)^{-1}\boldsymbol{I})$
4. For each domain: $d = 1, \ldots, D$
   (a) For each object: $n = 1, \ldots, N_d$
      i. Draw a cluster assignment $s_{dn} \sim \text{Categorical}(\boldsymbol{\theta})$
      ii. Draw an observation vector $\boldsymbol{x}_{dn} \sim \mathcal{N}(\boldsymbol{W}_d \boldsymbol{z}_{s_{dn}}, \alpha^{-1}\boldsymbol{I})$

Here, r is a parameter for controlling the precision of latent vectors, and Stick(γ) is the stick-breaking process [30] that generates mixture weights for a Dirichlet process with concentration parameter γ. By using a Dirichlet process, we can automatically find the number of clusters from the given data. Discrete($\cdot$) represents a categorical distribution. $\mathcal{G}(a, b)$ represents a Gamma distribution with parameters a and b. We assume Gauss-Gamma distribution $p(\boldsymbol{Z}, \alpha | a, b, r) = \mathcal{N}(\boldsymbol{Z}|\mathbf{0}, (\alpha r)^{-1}\boldsymbol{I})\mathcal{G}(\alpha|a, b)$ for the prior of latent vectors $\boldsymbol{Z}$ and α because it is a conjugate prior for a Gaussian $p(\boldsymbol{x}_{dn}|\boldsymbol{z}_j, \alpha, \boldsymbol{W}) = \mathcal{N}(\boldsymbol{x}_{dn}|\boldsymbol{W}_d\boldsymbol{z}_j, \alpha^{-1}\boldsymbol{I})$ and it enables us to analytically integrate out the latent vectors as shown in Appendix A. Figure 3 shows a graphical model representation of the proposed model, where shaded and unshaded nodes indicate observed and latent variables, respectively.

The joint probability of data $\boldsymbol{X}$ and cluster assignments $\boldsymbol{S} = \{\{s_{dn}\}_{n=1}^{N_d}\}_{d=1}^{D}$ is given by

$$p(\boldsymbol{X}, \boldsymbol{S}|\boldsymbol{W}, a, b, r, \gamma) = p(\boldsymbol{S}|\gamma)p(\boldsymbol{X}|\boldsymbol{S}, \boldsymbol{W}, a, b, r). \tag{2}$$

By marginalizing out mixture weights $\boldsymbol{\theta}$, the first factor is calculated by

$$p(\boldsymbol{S}|\gamma) = \frac{\gamma^J \prod_{j=1}^{J}(N_{\cdot j}-1)!}{\gamma(\gamma+1)\cdots(\gamma+N-1)}, \tag{3}$$

where $N = \sum_{d=1}^{D} N_d$ is the total number of objects, $N_{\cdot j}$ represents the number of objects assigned to cluster j, and J is the number of clusters for which $N_{\cdot j} > 0$. By marginalizing out latent vectors $\boldsymbol{Z}$ and precision parameter α, the second factor of (2) is calculated by

$$p(\boldsymbol{X}|\boldsymbol{S},\boldsymbol{W},a,b,r) = (2\pi)^{-\frac{\sum_d M_d N_d}{2}} r^{\frac{KJ}{2}} \frac{b^a}{b'^{a'}} \frac{\Gamma(a')}{\Gamma(a)} \prod_{j=1}^{J} |\boldsymbol{C}_j|^{\frac{1}{2}}. \tag{4}$$

Here,

$$a' = a + \frac{\sum_{d=1}^{D} M_d N_d}{2}, \tag{5}$$

$$b' = b + \frac{1}{2}\sum_{d=1}^{D}\sum_{n=1}^{N_d} \boldsymbol{x}_{dn}^\top \boldsymbol{x}_{dn} - \frac{1}{2}\sum_{j=1}^{J} \boldsymbol{\mu}_j^\top \boldsymbol{C}_j^{-1} \boldsymbol{\mu}_j, \tag{6}$$

$$\boldsymbol{\mu}_j = \boldsymbol{C}_j \sum_{d=1}^{D} \boldsymbol{W}_d^\top \sum_{n:s_{dn}=j} \boldsymbol{x}_{dn}, \tag{7}$$

$$\boldsymbol{C}_j^{-1} = \sum_{d=1}^{D} N_{dj} \boldsymbol{W}_d^\top \boldsymbol{W}_d + r\boldsymbol{I}, \tag{8}$$

where N_{dj} is the number of objects assigned to cluster j in domain d. The posterior for the precision parameter α is given by

$$p(\alpha|\boldsymbol{X},\boldsymbol{S},\boldsymbol{W},a,b) = \mathcal{G}(a',b'), \tag{9}$$

and the posterior for the latent vector $\boldsymbol{z}_j$ is given by

$$p(\boldsymbol{z}_j|\alpha,\boldsymbol{X},\boldsymbol{S},\boldsymbol{W},r) = \mathcal{N}(\boldsymbol{\mu}_j, \alpha^{-1}\boldsymbol{C}_j). \tag{10}$$

See Appendix B for the derivation.

3.2. Learning

We describe the learning procedures for the proposed model based on a stochastic EM algorithm, in which collapsed Gibbs sampling of cluster assignments $\boldsymbol{S}$ and the maximum joint likelihood estimation of projection matrices $\boldsymbol{W}$ are alternately iterated while marginalizing out latent vectors $\boldsymbol{Z}$, precision parameter α, and cluster proportions $\boldsymbol{\theta}$. By collapsing, or marginalizing out, the variables, the time-consuming step of drawing the variables is skipped, and the sample autocorrelations are usually reduced [31].

In the E-step, given the current state of all but one latent cluster assignments, s_{dn}, a new value for s_{dn} is sampled from the following probability,

$$p(s_{dn}=j|\boldsymbol{X},\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r,\gamma) \propto \frac{p(s_{dn}=j,\boldsymbol{S}_{\backslash dn}|\gamma)}{p(\boldsymbol{S}_{\backslash dn}|\gamma)} \cdot \frac{p(\boldsymbol{X}|s_{dn}=j,\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r)}{p(\boldsymbol{X}_{\backslash dn}|\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r)}, \tag{11}$$

where $\backslash dn$ represents a value or set excluding the nth object in the dth domain, and we use the fact that $p(s_{dn}=j|\boldsymbol{X},\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r,\gamma)$ does not depend on $p(\boldsymbol{x}_{dn}|\boldsymbol{W},a,b,r)$. See Appendix C for the derivation. The first factor is given by

$$\frac{p(s_{dn}=j,\boldsymbol{S}_{\backslash dn}|\gamma)}{p(\boldsymbol{S}_{\backslash dn}|\gamma)} = \begin{cases} \frac{N_{\cdot j\backslash dn}}{N-1+\gamma} & \text{for an existing cluster, } j\in\{1,\cdots,J\} \\ \frac{\gamma}{N-1+\gamma} & \text{for a new cluster, } j=J+1, \end{cases} \tag{12}$$

using (3). By using (4), the second factor is given by

$$\begin{aligned} &\frac{p(\boldsymbol{X}|s_{dn}=j,\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r)}{p(\boldsymbol{X}_{\backslash dn}|\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r)} \\ &= (2\pi)^{-\frac{M_d}{2}} r^{\frac{1}{2}I(j>J_{\backslash dn})} \frac{b_{\backslash dn}'^{a'_{\backslash dn}}}{b_{s_{dn}=j}'^{a'_{s_{dn}=j}}} \frac{\Gamma(a'_{s_{dn}=j})}{\Gamma(a'_{\backslash dn})} \frac{|\boldsymbol{C}_{j,s_{dn}=j}|^{\frac{1}{2}}}{|\boldsymbol{C}_{j\backslash dn}|^{\frac{1}{2}}}, \end{aligned} \tag{13}$$

where subscript $s_{dn}=j$ indicates the value when object $\boldsymbol{x}_{dn}$ is assigned to cluster j as follows,

$$a'_{s_{dn}=j} = a', \tag{14}$$

$$b'_{s_{dn}=j} = b'_{\backslash dn} + \frac{1}{2}\boldsymbol{x}_{dn}^\top \boldsymbol{x}_{dn} + \frac{1}{2}\boldsymbol{\mu}_{j\backslash dn}^\top \boldsymbol{C}_{j\backslash dn}^{-1}\boldsymbol{\mu}_{j\backslash dn} - \frac{1}{2}\boldsymbol{\mu}_{j,s_{dn}=j}^\top \boldsymbol{C}_{j,s_{dn}=j}^{-1}\boldsymbol{\mu}_{j,s_{dn}=j}, \tag{15}$$

$$\boldsymbol{\mu}_{j,s_{dn}=j} = \boldsymbol{C}_{j,s_{dn}=j}(\boldsymbol{W}_d^\top \boldsymbol{x}_{dn} + \boldsymbol{C}_{j\backslash dn}^{-1}\boldsymbol{\mu}_{j\backslash dn}), \tag{16}$$

$$\boldsymbol{C}_{j,s_{dn}=j}^{-1} = \boldsymbol{W}_d^\top \boldsymbol{W}_d + \boldsymbol{C}_{j\backslash dn}^{-1}, \tag{17}$$

and $I(\cdot)$ is used to denote the indicator function, i.e. $I(A) = 1$ if A is true, $I(A) = 0$ otherwise.

In the M-step, the projection matrices $\boldsymbol{W}$ are estimated by maximizing the logarithm of the joint likelihood (2). We maximize it by using a gradient-based numerical optimization method such as the quasi-Newton method [32]. The gradient of the joint log likelihood is calculated by

$$\begin{aligned}&\frac{\partial \log p(\boldsymbol{X}, \boldsymbol{S}|\boldsymbol{W}, a, b, r, \gamma)}{\partial \boldsymbol{W}_d} \\ &= -\boldsymbol{W}_d \sum_{j=1}^{J} N_{dj}\boldsymbol{C}_j - \frac{a'}{b'}\sum_{j=1}^{J}\Big(N_{dj}\boldsymbol{W}_d\boldsymbol{\mu}_j\boldsymbol{\mu}_j^\top - \sum_{n:s_{dn}=j}\boldsymbol{x}_{dn}\boldsymbol{\mu}_j^\top\Big).\end{aligned} \tag{18}$$

We can obtain the projection matrices that maximize the joint likelihood analytically as follows,

$$\boldsymbol{W} = \left(\sum_{j=1}^{J} N_{dj}\boldsymbol{C}_j + \frac{a'}{b'}N_{dj}\boldsymbol{\mu}_j\boldsymbol{\mu}_j^\top\right)^{-1} \frac{a'}{b'}\sum_{j=1}^{J}\sum_{n:s_{dn}=j}\boldsymbol{x}_{dn}\boldsymbol{\mu}_j^\top. \tag{19}$$

In our experiments, we used gradient-based updates since it found better local optimum solutions by updating parameters gradually.

Algorithm 1 shows the procedures for inferring the proposed model based on the stochastic EM algorithm. Here, T is the number of iterations for the stochastic EM algorithm. For the input, we give the initial number of clusters J. The cluster assignments $\boldsymbol{S}$ are initialized by randomly selecting an integer from $\{1, \cdots, J\}$. The projection matrices $\boldsymbol{W}$ can be initialized by Gaussian with zero mean and small variance. By iterating the E- and M-steps, we can obtain an estimate of the cluster assignments and projection matrices. In all of the experiments, we used the following hyperparameters: $a = b = r = \gamma = 1$.

Algorithm 1 Learning procedures for the proposed model.

Input: multiple domain data sets $\boldsymbol{X}$, initial number of clusters J, hyperparameters a, b, r, γ, iteration number T

Output: cluster assignments $\boldsymbol{S}$, projection matrices $\boldsymbol{W}$

initialize $\boldsymbol{S}$ and $\boldsymbol{W}$
for $t = 1, \cdots, T$ **do**
  //E-step
  for $d = 1, \cdots, D$ **do**
    for $n = 1, \cdots, N_d$ **do**
      sample s_{dn} using probability $p(j|\boldsymbol{X}, \boldsymbol{S}_{\backslash dn}, \boldsymbol{W}, a, b, r, \gamma)$ (11) from $j = 1, \cdots, J+1$
      if $s_{dn} = J+1$ **then**
        update the number of clusters $J \leftarrow J+1$
      end if
    end for
  end for
  //M-step
  for $d = 1, \cdots, D$ **do**
    update $\boldsymbol{W}_d$ using a numerical optimization method using (18)
  end for
end for

We can use cross-validation to select an appropriate dimensionality for latent space K. H-fold cross-validation is as follows,

1. The elements of observation matrices $\boldsymbol{X}$ are partitioned into H subsets.
2. For each of H experiments, $H-1$ subsets are used for inferring the model while the remaining one is assumed missing. The test likelihood of the missing part is calculated by using the inferred model.
3. We select the K value that performed the best in terms of the average test likelihood over H experiments.

The learning procedure for missing data is described in Section 3.4.

When we use a regular EM algorithm instead of the stochastic EM algorithm, latent vectors are not integrated out, and therefore need to be estimated. We can use Gibbs sampling for inferring projection matrices. However, we learned them by maximizing the likelihood since priors, which require additional hyper-parameters, are not necessary.

3.3. Semi-supervised setting

In some applications, correspondence information on a few pairs of objects might be available. This correspondence information helps to improve matching performance. Here, we describe learning procedures for the semi-supervised setting, where correspondence information between some objects in different domains is given, based on a stochastic EM algorithm. The task of finding correspondence in a semi-supervised setting is also called informed sorting [4] or seed alignment [11]. Let $i = (d, n)$ and $i' = (d', n')$ be a pair of objects for which one-to-one correspondence is given. Their latent cluster assignments must be the same because we know that they match. Thus, in the E-step, we sample the cluster assignments for both of them, i and i', simultaneously so that they have the same cluster assignments. New values for the cluster assignments, s_i and $s_{i'}$, are sampled from the following probability,

$$\begin{aligned} &p(s_i = s_{i'} = j | \boldsymbol{X}, \boldsymbol{S}_{\backslash ii'}, \boldsymbol{W}, a, b, r, \gamma) \\ &= \frac{p(s_i = s_{i'} = j, \boldsymbol{S}_{\backslash ii'} | \gamma)}{p(\boldsymbol{S}_{\backslash ii'} | \gamma)} \cdot \frac{p(\boldsymbol{X} | s_i = s_{i'} = j, \boldsymbol{S}_{\backslash ii'}, \boldsymbol{W}, a, b, r)}{p(\boldsymbol{X}_{\backslash ii'} | \boldsymbol{S}_{\backslash ii'}, \boldsymbol{W}, a, b, r)} \end{aligned} \tag{20}$$

where $\backslash ii'$ represents a value or set excluding objects i and i'. The first factor of the left-hand side is calculated by

$$\frac{p(s_i = s_{i'} = j, \boldsymbol{S}_{\backslash ii'} | \gamma)}{p(\boldsymbol{S}_{\backslash ii'} | \gamma)} = \begin{cases} \frac{N_{\cdot j \backslash ii'}(N_{\cdot j \backslash ii'} + 1)}{(N-2+\gamma)(N-1+\gamma)} & \text{for an existing cluster, } j \in \{1, \cdots, J\} \\ \frac{\gamma}{(N-2+\gamma)(N-1+\gamma)} & \text{for a new cluster, } j = J+1, \end{cases} \tag{21}$$

using (3), and the second factor is given by

$$\frac{p(\boldsymbol{X}|s_i = s_{i'} = j, \boldsymbol{S}_{\setminus ii'}, \boldsymbol{W}, a, b, r)}{p(\boldsymbol{X}_{\setminus ii'}|\boldsymbol{S}_{\setminus ii'}, \boldsymbol{W}, a, b, r)}$$
$$= (2\pi)^{-\frac{M_d + M_{d'}}{2}} r^{\frac{1}{2} I(j > J_{\setminus ii'})} \frac{b_{\setminus ii'}'^{a'_{\setminus ii'}}}{b_{s_{ii'}=j}'^{a'_{s_{ii'}=j}}} \frac{\Gamma(a'_{s_{ii'}=j})}{\Gamma(a'_{\setminus ii'})} \frac{|\boldsymbol{C}_{j, s_{ii'}=j}|^{\frac{1}{2}}}{|\boldsymbol{C}_{j \setminus ii'}|^{\frac{1}{2}}}, \tag{22}$$

using (3) and (4). When we are given correspondence information for more than two objects, we can derive a sampling probability with the same manner using (4). For objects without correspondence information, the sampling probability of the latent cluster assignment is the same as in the unsupervised setting (11). The M-step is the same as that in the unsupervised setting, and we can update projection matrices using gradient-based optimization with (18).

3.4. Missing data

Since the proposed model is a probabilistic generative model, it can handle missing data, where some observation features are missing. Let $\boldsymbol{h}_{dn} = (h_{dnm})_{m=1}^{M_d}$ be a vector indicating observed indices, where $h_{dnm} = 1$ if x_{dnm} is observed, $h_{dnm} = 0$ otherwise, and M_{dn} is the number of observed values for object $\boldsymbol{x}_{dn}$. The posterior parameters are calculated as follows,

$$a' = a + \frac{\sum_{d=1}^{D} \sum_{n=1}^{N_d} M_{dn}}{2}, \tag{23}$$

$$b' = b + \frac{1}{2} \sum_{d=1}^{D} \sum_{n=1}^{N_d} (\boldsymbol{h}_{dn} \circ \boldsymbol{x}_{dn})^\top (\boldsymbol{h}_{dn} \circ \boldsymbol{x}_{dn}) - \frac{1}{2} \sum_{j=1}^{J} \boldsymbol{\mu}_j^\top \boldsymbol{C}_j^{-1} \boldsymbol{\mu}_j, \tag{24}$$

$$\boldsymbol{\mu}_j = \boldsymbol{C}_j \sum_{d=1}^{D} \boldsymbol{W}_d^\top \sum_{n: s_{dn}=j} \boldsymbol{h}_{dn} \circ \boldsymbol{x}_{dn}, \tag{25}$$

$$\boldsymbol{C}_j^{-1} = \sum_{d=1}^{D} \boldsymbol{W}_d^\top \Big(\sum_{n: s_{dn}=j} \mathrm{diag}(\boldsymbol{h}_{dn}) \Big) \boldsymbol{W}_d + r\boldsymbol{I}, \tag{26}$$

where $\circ$ represents the Hadamard product, or element-wise product, and $\mathrm{diag}(\boldsymbol{h}_{dn})$ returns a diagonal matrix whose diagonal elements are $h_{dn1}, \ldots, h_{dnM_d}$. For

learning with missing data, we use (23) – (26) instead of (5) – (8) for calculating the sampling probability (11) in the E-step. In the M-step, the following gradient is used for optimizing projection matrices,

$$\frac{\partial \log p(\boldsymbol{X}, \boldsymbol{S}|\boldsymbol{W}, a, b, r, \gamma)}{\partial \boldsymbol{W}_d} = -\boldsymbol{W}_d \sum_{j=1}^{J} N_{dj} \boldsymbol{C}_j - \frac{a'}{b'} \sum_{j=1}^{J} \Big(N_{dj} \boldsymbol{W}_d \boldsymbol{\mu}_j \boldsymbol{\mu}_j^\top - \sum_{n:s_{dn}=j} (\boldsymbol{h}_{dn} \circ \boldsymbol{x}_{dn}) \boldsymbol{\mu}_j^\top \Big), \quad (27)$$

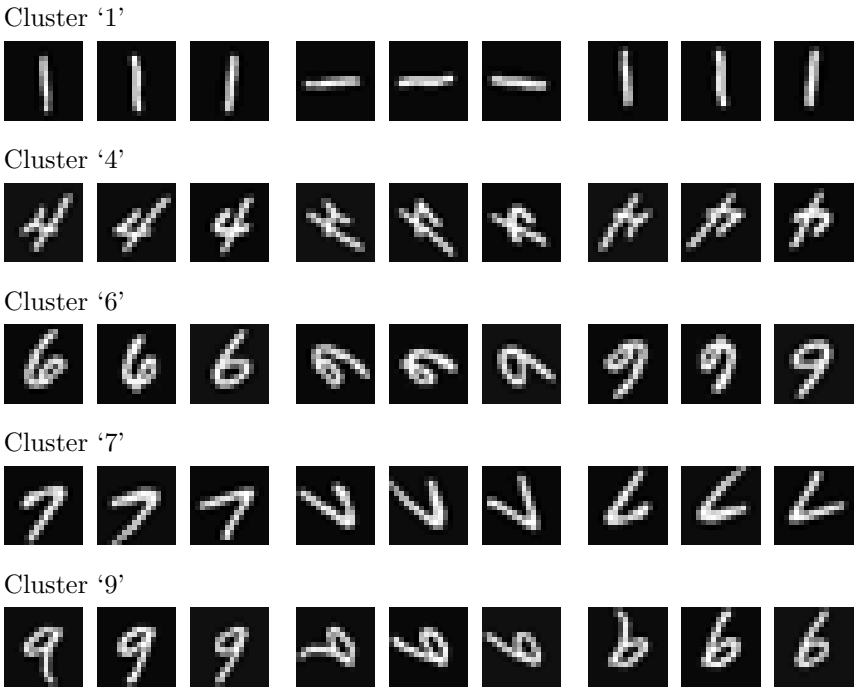
instead of (18).

4. Experiments

4.1. Matchin rotated handwritten digits

First, we demonstrate the proposed model described in Section 3 using a toy data set with three domains, which is created using handwritten digits from the MNIST database [33]. The first domain contains original handwritten digits, where each image is downsampled to 16×16 pixels. We synthesize objects for the second and third domains by rotating handwritten digits by 90 and 180 degrees, clockwise, respectively. Thus, we obtain three-domain objects that share a latent space. The number of objects in each domain is 200 for all domains. We would like to match the rotated digits in different domains without information about rotation or correspondence.

Figure 4 shows some examples of the clusters discovered by the proposed model with $K = 5$. The proposed model successfully clustered objects, the same digit with rotation. In each domain, similar objects are clustered. Across different domains, dissimilar objects can be clustered. For example, objects in cluster '6' in the first domain are more dissimilar to those in the third domain than objects in cluster '9' in the first domain. By using domain-dependent projection matrices, the proposed model finds clusters of objects across different domains that have similar latent features and that might have different observation features. When we use standard clustering methods such as k-means and Gaussian mixtures for the data that are constructed by combining data in all domains, '6' is never clustered with '6' with 180 degree rotation.



(a) Domain 1 (b) Domain 2 (c) Domain 3

Figure 4: Examples of clusters discovered by the proposed model with rotated handwritten data sets with three domains. Each row represents a cluster, and three objects are shown for each domain.

We can map the observation features to different domains through the latent space by using the inferred projection matrices. Observation feature vector $\boldsymbol{x}$ in domain d can be mapped into the latent space by $\hat{\boldsymbol{z}} = (\boldsymbol{W}_d^\top \boldsymbol{W}_d)^{-1} \boldsymbol{W}_d^\top \boldsymbol{x}$, where we used the pseudo-inverse of projection matrix $\boldsymbol{W}_d$. The latent vector $\boldsymbol{z}$ can be mapped into domain d' by $\hat{\boldsymbol{x}}_{d'} = \boldsymbol{W}_{d'} \hat{\boldsymbol{z}}$. Thus, the projection matrix that maps from domain d to domain d' is $\boldsymbol{W}_{d'} (\boldsymbol{W}_d^\top \boldsymbol{W}_d)^{-1} \boldsymbol{W}_d^\top$. Figure 5 shows the images mapped to the three domains by the projection matrices. The original image comes from the first domain. The mapped images to second and third domains are rotated images of the first domain by 90 and 180 degrees clockwise, respectively, which are consistent with the procedure used to synthesize the data set. This result indicates that we can infer the projection matrices properly,

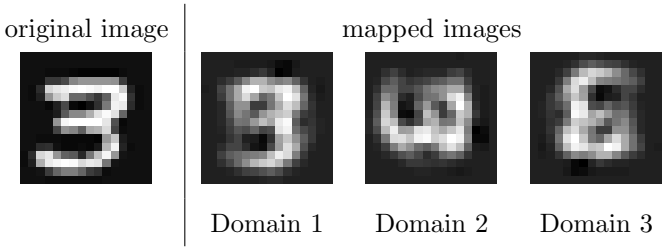


Figure 5: Mapping images to different domains. The left figure shows the original image, and the right figures show images mapped to the first, second and third domains.

and that the proposed model can map to different domains in an unsupervised fashion by using the shared latent space.

4.2. Data

Next, we evaluated the proposed model quantitatively by using three synthetic and five real-world data sets. The statistics of the seven data sets are shown in Table 2. There are two domains for all data sets. Synth3, Synth5, and Synth10 are synthetic data sets with different true dimensionalities of the latent space $K^* = 3, 5$ and 10, respectively. We generated the synthetic data sets using the following procedure. First, we sampled latent vectors $\boldsymbol{z}_j$ for $j = 1, \cdots, J^*$ from a K^*-dimensional normal distribution with mean $\mathbf{0}$ and covariance $\boldsymbol{I}$. Next, we generated projection matrices $\boldsymbol{W}_d$ for $d = 1, 2$, where each element is drawn from a normal distribution with mean 0 and variance 1. Finally, we generated N/J^* objects for each cluster j using a normal distribution with mean $\boldsymbol{W}_d \boldsymbol{z}_j$ and covariance $\alpha^{-1}\boldsymbol{I}$, and obtained N objects in total for each domain $d = 1, 2$.

Iris, Glass, Wine and MNIST, the real-world data sets, were obtained from LIBSVM multi-class data sets [34], and generated objects in two domains by randomly splitting the features into two parts for each data set as [4, 13] did for their experiments. Because there is no overlapping feature, we cannot calculate similarities between objects in different domains. Iris, Glass, Wine and MNIST were artificially splitted into two domains. On the other hand, Wiki data was a real-world two domain data set, which consisted of Wikipedia documents written in English and German. For each language, we sampled 150 documents

Table 2: Statistics of the data sets: the number of objects N, the dimensionality of the objects M_d, and the true number of clusters J^*, and the true dimensionality of latent space K^*.

	N_1/N_2	M_1	M_2	J^*	K^*
Synth3	200	50	50	5	3
Synth5	200	50	50	5	5
Synth10	200	50	50	5	10
Iris	150	2	2	3	N/A
Glass	214	4	5	7	N/A
Wine	178	6	7	3	N/A
MNIST	200	392	392	10	N/A
Wiki	150	1000	1000	5	N/A

that were categorized in 'Nobel laureates in Physics', 'Nobel laureates in Chemistry', 'American basketball players', 'American composers', and 'English footballers'. We selected these categories since since they contained enough number of documents, they were related to each other ('Nobel laureates in Physics' and 'Nobel laureates in Chemistry', 'American basketball players' and 'American composers', 'American basketball players' and 'English footballers'), and their topics seemed to be interpretable. We used 1,000 most frequently occurring words as features for each language to reduce computational time, where the frequencies were computed by all documents that were categorized in the five categories. With the real-world data sets, we used class labels as true clusters.

4.3. Comparing methods

For the proposed method, we used the dimensionality of the latent space, $K = 5$, for all data sets. To alleviate the local optimum problem, we ran the learning procedure five times with different initial conditions, and selected the result that achieved the highest likelihood. The number of iterations was 100. For comparison, we used k-means (KM), convex kernelized sorting (CKS) [13], and their combinations (KM-CKS and CKS-KM) as described below. The KM method is widely used for clustering. Although KM is not a many-to-many

matching method, we included KM as a baseline method to evaluate the case when only objects in the same domain are clustered. The CKS method is an unsupervised object matching method. It directly finds correspondence between objects, and does not cluster objects in the same domain. We used the CKS code provided by the authors[1], where we used their default setting and the maximum number of iterations was 10,000. With KM-CKS, first we discovered clusters by applying KM to each domain individually, and then found the correspondence between clusters in two domains by using CKS. We used the mean vector of each cluster as the input for cluster matching by CKS. With CKS-KM, after matching objects using CKS, we combined matched objects in two domains into a vector, and estimated clusters using KM. We employed CKS for comparison since it achieves higher performance than kernelized sorting and matching canonical correlation analysis [13, 11]. With KM, KM-CKS and CKS-KM, we used the number of clusters estimated by the proposed model. For comparison with object matching based methods (CKS, CKS-KM), we used data sets that had the same numbers of objects in the two domains $N_1 = N_2$ for each data set.

4.4. Evaluation measurment

For the evaluation measurement, we used the adjusted Rand index [35], which quantifies the similarity between inferred clusters and true clusters. It takes a value from -1 to 1; 0 represents random clustering. A higher value indicates better clustering performance. The adjusted Rand index becomes high when object pairs that belong to one true cluster are assigned to one inferred cluster simultaneously, and when object pairs that belong to different true clusters are assigned to different inferred clusters. Here, we use object pairs across all domains including the same domain. Even if no objects are correctly matched across different domains, the adjusted Rand index can be positive if clusters are correctly found within each domain. If objects are matched, they are considered assigned to the same cluster. Therefore, we can calculate the

[1] `http://astro.temple.edu/~tua95067/CKS_code.zip`

Table 3: Average adjusted Rand index and its standard deviation. Values in bold typeface are significantly better from those in normal typeface as indicated by a paired t-test.

	Proposed	KM	KM-CKS	CKS	CKS-KM
Synth3	$\mathbf{0.875} \pm 0.101$	0.525 ± 0.014	0.589 ± 0.117	0.014 ± 0.005	0.699 ± 0.135
Synth5	$\mathbf{0.893} \pm 0.126$	0.548 ± 0.029	0.583 ± 0.198	0.006 ± 0.007	0.571 ± 0.182
Synth10	$\mathbf{0.827} \pm 0.145$	0.556 ± 0.026	0.553 ± 0.165	0.009 ± 0.006	0.678 ± 0.170
Iris	$\mathbf{0.383} \pm 0.189$	0.224 ± 0.091	0.254 ± 0.154	0.003 ± 0.002	0.207 ± 0.089
Glass	$\mathbf{0.160} \pm 0.020$	0.050 ± 0.008	0.052 ± 0.011	0.001 ± 0.001	0.047 ± 0.010
Wine	$\mathbf{0.222} \pm 0.111$	0.125 ± 0.025	0.142 ± 0.046	0.001 ± 0.001	0.107 ± 0.038
MNIST	$\mathbf{0.085} \pm 0.016$	0.030 ± 0.007	0.037 ± 0.008	0.008 ± 0.005	0.041 ± 0.016
Wiki	$\mathbf{0.222} \pm 0.048$	0.152 ± 0.024	$\mathbf{0.199} \pm 0.062$	0.013 ± 0.003	$\mathbf{0.207} \pm 0.061$

adjusted Rand index of CKS although it is likely to be low because all objects in the same domain are assigned to different clusters. The adjusted Rand index for KM is calculated by assuming that any clusters across different domains do not match. The adjusted Rand index measures how well objects with the same label in different domains are assigned to the same cluster as well as measuring the clustering performance within each domain. For the real data sets, we assume that the category label of each object is its true cluster assignment.

4.5. Results

Table 3 shows the adjusted Rand index for the seven data sets, which were averaged over ten experiments for each data set. For all data sets, the proposed model achieved the highest adjusted Rand index. This result indicates that the proposed model can infer matching clusters by assuming a shared latent space. KM-CKS achieved higher performance than KM by matching clusters in a post-processing step. With KM-CKS, since clusters are inferred individually for each domain, the estimated clusters might be different in different domains. On the other hand, since the proposed model infers clusters in all domains simultaneously, it more successfully found shared clusters than KM-CKS as shown by its higher adjusted Rand index. The adjusted Rand index obtained with the CKS method was low, because it does not cluster objects. By clustering the result

of CKS, CKS-KM improved the cluster matching performance. However, it did not outperform the proposed model, because errors that accumulate in the object matching process by CKS cannot be corrected in the clustering process with k-means. The better performance of the proposed model with fixed hyperparameters $a = b = r = \gamma = 1$ for all data sets indicates that the performance is not sensitive to hyperparameter setting.

The performance with the synthetic data by the proposed model was better than that with the real-world data since the synthetic data are generated by the generative process of the proposed model. Because the generative process, where observations are generated by linear projections of latent vectors, is simple compared with the real-world data, the performance with the synthetic data by the baselines was also better than the performance with the real-world data. With the Wiki data, although the proposed method achieved the best adjusted Rand index, it was not significantly better than KM-CKS and CKS-KM. This might be because the word frequency feature does not match with Gaussian noise which is the assumption of the proposed method. Another reason would be that linear projections is not appropriate for the Wiki data.

Figure 6 shows the adjusted Rand index for the Synth5 data set achieved by the proposed model with different latent dimensionalities. The value was highest when the latent dimensionality of the model was the same as the true latent dimensionality $K = K^* = 5$. The proposed model with $K \neq K^*$ also performed better than the other methods. This result indicates that the proposed model is robust to the latent dimensionality setting.

Figure 7 shows the adjusted Rand index for the Synth5 data set with different numbers of domains, D. The proposed model can handle data with more than two domains. For kernelized sorting based methods (KM-CKS, CKS-KM and CKS), we used one of the domains as a pivot for handling more than two domains. In particular, with these methods, we found matching across multiple domains by matching clusters/objects between the Dth domain and each of the other $D - 1$ domains, and then combined the results. In general, the adjusted Rand index decreases as the number of domains increases, since the

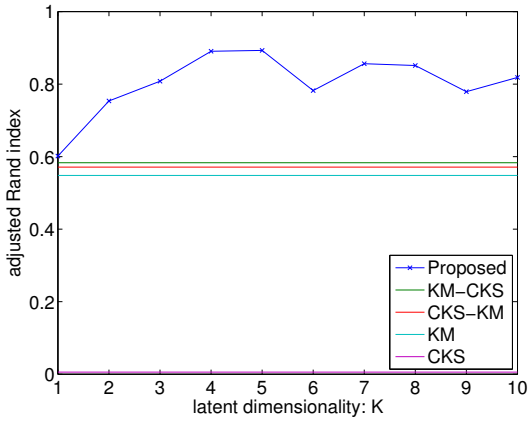


Figure 6: Adjusted Rand index achieved by the proposed model with different latent dimensionalities K for the Synth5 data set whose true latent dimensionality is $K^* = 5$.

number of possible combinations of cluster matching increases. However, the proposed method consistently achieved the highest performance regardless of domain number.

The proposed model assumes that cluster proportions are shared by different domains. We evaluated the performance of the proposed model with synthetic data, in which cluster proportions were different across two domains. Each clusters contains 50 objects except for cluster1 of domain1, where we changed the number of objects from 10 to 100. The number of true clusters was $J^* = 5$, the latent dimensionality was $K^* = 5$, and the dimensionality of the objects was $M_1 = M_2 = 50$. Figure 8 shows the adjusted Rand index. Note that CKS and CKS-KM are unapplicable since they require that different domains contain the same number of objects. The proposed model achieved the best performance even with data with different cluster proportions. When the number of objects in cluster1 of domain1 was 50, the adjusted Rand index of the proposed model was highest, since the common cluster proportions were the same across different domains with this data set.

The computational time of the proposed model for learning the Synth5 data set was 22 seconds with 100 learning iterations when using a computer with

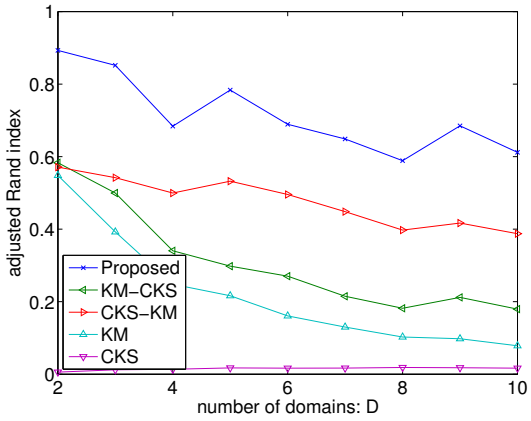


Figure 7: Adjusted Rand index for the Synth5 data set with different numbers of domains, D.

3.20GHz CPU. The computational complexity of each E-step is $O(NJK^3)$, where N is the total number of objects, J is the number of clusters, and K is the latent dimensionality. It increases linearly with the number of objects N because we need to perform sampling for each object. Cluster assignment is sampled from $J+1$ clusters, in which we calculate the probability for each of the $J+1$ clusters. Therefore, the complexity increases linearly with the number of clusters J. The complexity of calculating each sampling probability is dominated by the $O(K^3)$ computation of inverse $\boldsymbol{C}_j^{-1}$. Figure 9 shows the experimental computation time of the proposed model including E- and M-steps. The computation time linearly increases with the number of objects as shown in Figure 9(a). The dimensionality of the observation and latent spaces, and the true number of clusters do not strongly impact the computation time (Figure 9(b,c,d)). Because this experiment used small latent dimensionality, it might not alter the total computation time. The computation time increases superlinearly with the number of domains (Figure 9(e)). The time increases because the number of parameters in the projection matrices increases linearly with the number of domains, and the total number of objects is also increased.

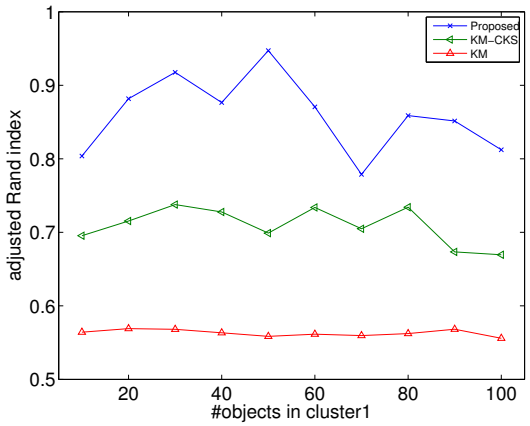
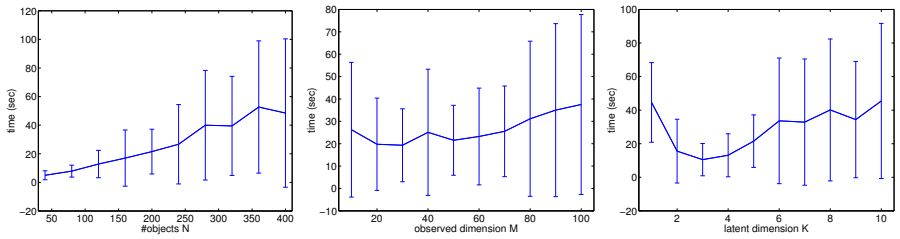


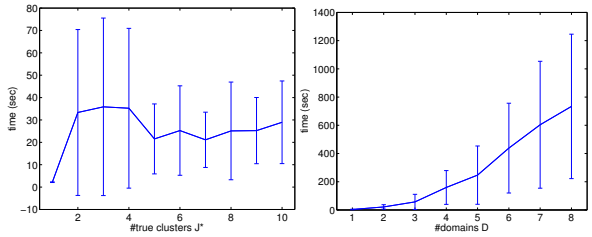
Figure 8: Adjusted Rand index for the synthetic data with different cluster proportions across different domains.

4.6. Semi-supervised setting

Figure 10 shows the average adjusted Rand index achieved by the proposed model with semi-supervised setting over ten experiments, where different objects were labeled for each experiment. Here, we used the seven data sets shown in Table 2. For each data set, we randomly selected pairs of objects to be labeled according to the labeled object rate, which is the horizontal axis of Figure 10, and correspondence information to the different domain for the selected objects were attached. The performance increases as the rate of labeled objects increases. With some data sets (e.g. Synth5, Synth10 and Wine), the addition of just a few labeled objects drastically increases the adjusted Rand index. This result indicates that these data sets can be appropriately modeled by the linear projection of latent vectors, and the proposed model can utilize correspondence information for a small number of objects for finding many-to-many matching. Because the true latent dimensionality of the Synth5 data was the same with the model used in this experiment, the adjusted Rand index became one with additional labeled data. With the Synth3 data, the performance was not improved well since the proposed model overfit to the data due to its high latent dimensionality.



(a) #objects N (b) observed dimension M (c) latent dimension K



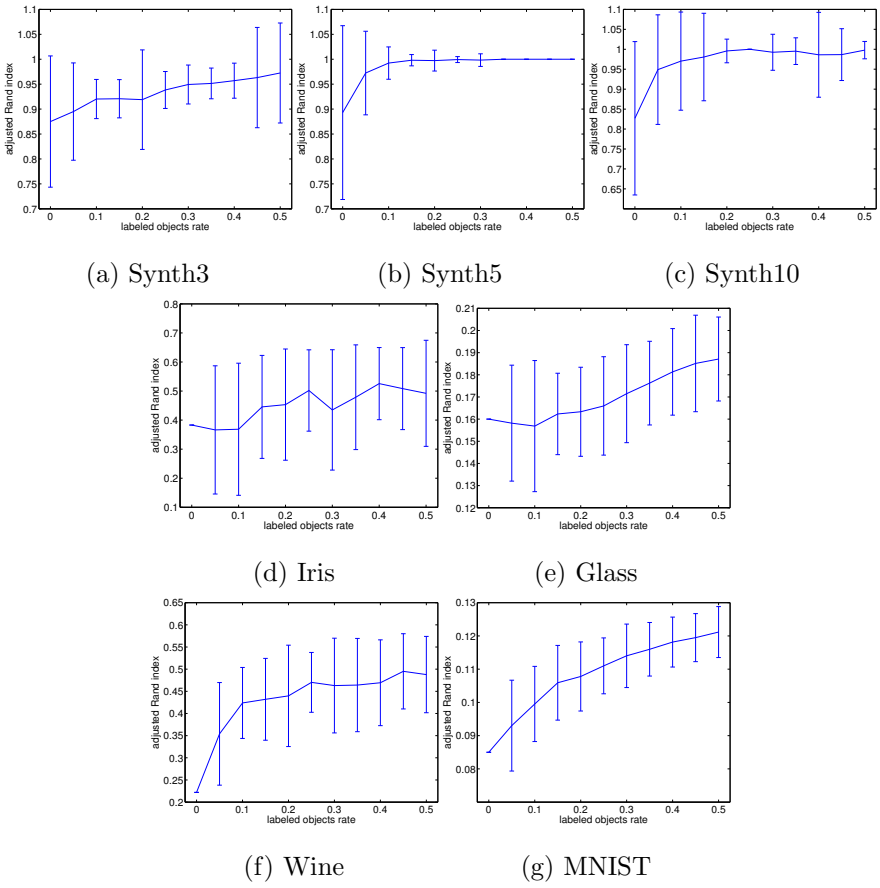
(d) #true clusters J^* (e) #domains D

Figure 9: Computation time in seconds with different settings. The original data set is Synth5, where the number of objects in each domain $N_d = 200$, the number of features in a domain $M_d = 50$, the latent dimensionality $K = 5$, the true number of clusters $J^* = 5$ and the number of domains $D = 2$. Each figure shows the results with data sets when just one of the settings is changed. The plot shows the average computational time over 100 experiments, and the bar shows the standard deviation.

5. Conclusion

We proposed a generative model approach for finding many-to-many matching based on probabilistic latent variable models. In experiments, we confirmed that the proposed model can perform much better than conventional methods based on object matching, clustering and their combinations. Advantages of the proposed model over the existing methods are that it can find many-to-many matching, and can handle multiple domains with different numbers of objects with no prior knowledge. Because the proposed approach uses probabilistic generative models, we can extend it in a probabilistically principled manner, and use it, for example, to handle missing data, integration with other probabilistic models, and generalization to exponential family distributions.

Although our results have been encouraging as a first step towards unsuper-



(a) Synth3 (b) Synth5 (c) Synth10

(d) Iris (e) Glass

(f) Wine (g) MNIST

Figure 10: Average adjusted Rand index and its standard deviation achieved by the proposed model in a semi-supervised setting with different rates of labeled objects.

vised many-to-many object matching, we must extend our approach in a number of directions. First, we would like to extend the proposed model to other types of data, such as time series and graph data. Second, we can relax the assumption that the observations are linear with respect to their latent vectors by using nonlinear matrix factorization techniques [36]. Third, we need some techniques for automatically inferring the latent dimensionality, such as automatic relevance determination [37]. Finally, we would like to use the proposed method for other applications, which include image annotation [1], cross domain recommendation [3], multi-lingual corpus analysis [38, 39], machine translation [6],

and bioinformatics [40].

Appendix A. Derivation of (4)

We give the derivation of the likelihood (4), in which latent vectors $\boldsymbol{Z}$ and precision parameter α are analytically integrated out.

$$
\begin{aligned}
&p(\boldsymbol{X}|\boldsymbol{S},\boldsymbol{W},a,b,r) \\
&= \int\int \prod_{d=1}^{D}\prod_{n=1}^{N_d} \mathcal{N}(\boldsymbol{x}_{dn}|\boldsymbol{W}_d\boldsymbol{z}_{s_{dn}},\alpha^{-1}\boldsymbol{I})\mathcal{G}(\alpha|a,b)\prod_{j=1}^{J}\mathcal{N}(\boldsymbol{z}_j|\boldsymbol{0},(ar)^{-1}\boldsymbol{I})d\boldsymbol{Z}d\alpha \\
&= \int\int \prod_{d=1}^{D}\prod_{n=1}^{N_d} \left(\frac{\alpha}{2\pi}\right)^{M_d/2}\exp(-\frac{\alpha}{2}\parallel \boldsymbol{x}_{dn}-\boldsymbol{W}_d\boldsymbol{z}_{s_{dn}}\parallel^2)\prod_{j=1}^{J}\left(\frac{\alpha r}{2\pi}\right)^{K/2}\exp(-\frac{\alpha r}{2}\parallel \boldsymbol{z}_j\parallel^2) \\
&\times \frac{1}{\Gamma(a)}b^a\alpha^{a-1}\exp(-b\alpha)d\boldsymbol{Z}d\alpha \\
&= \frac{b^a}{\Gamma(a)}\int\int \left(\frac{\alpha}{2\pi}\right)^{\sum_d M_dN_d/2}\left(\frac{\alpha r}{2\pi}\right)^{KJ/2}\exp\left(-\frac{\alpha}{2}\left[\sum_{j=1}^{J}(\boldsymbol{z}_j-\boldsymbol{\mu}_j)^\top\boldsymbol{C}_j(\boldsymbol{z}_j-\boldsymbol{\mu}_j)\right]\right)d\boldsymbol{Z} \\
&\times \exp\left(-\alpha\left[\frac{1}{2}\sum_{d=1}^{D}\sum_{n=1}^{N_d}\boldsymbol{x}_{dn}^\top\boldsymbol{x}_{dn}-\frac{1}{2}\sum_{j=1}^{J}\boldsymbol{\mu}_j^\top\boldsymbol{C}_j\boldsymbol{\mu}_j+b\right]\right)\alpha^{a-1}d\alpha \\
&= (2\pi)^{-\frac{\sum_d M_dN_d}{2}}r^{\frac{KJ}{2}}\frac{b^a}{\Gamma(a)}\prod_{j=1}^{J}|\boldsymbol{C}_j|^{\frac{1}{2}}\int\exp(-b'\alpha)\alpha^{a'-1}d\alpha \\
&= (2\pi)^{-\frac{\sum_d M_dN_d}{2}}r^{\frac{KJ}{2}}\frac{b^a}{b'^{a'}}\frac{\Gamma(a')}{\Gamma(a)}\prod_{j=1}^{J}|\boldsymbol{C}_j|^{\frac{1}{2}}. \qquad \text{(A.1)}
\end{aligned}
$$

In the third equation, factors related to $\boldsymbol{Z}$ are grouped together. In the fourth equation, we integrated out $\boldsymbol{Z}$ using

$$\int\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top\boldsymbol{C}(\boldsymbol{x}-\boldsymbol{\mu})\right)d\boldsymbol{x} = (2\pi)^{M/2}|\boldsymbol{C}|^{\frac{1}{2}}, \qquad \text{(A.2)}$$

which is the normalization constant of M-dimensional Gaussian distribution. Similarly, α is integrated out by using the following the normalization constant of Gamma distribution in the sixth equation

$$\int\alpha^{a-1}\exp(-b\alpha)d\alpha = \frac{\Gamma(a)}{b^a}. \qquad \text{(A.3)}$$

Appendix B. Derivation of (9) and (10)

We give here the derivation of posteriors of α (9) and $\boldsymbol{z}_j$ (10).

$$
\begin{aligned}
&p(\alpha|\boldsymbol{X},\boldsymbol{S},\boldsymbol{W},a,b)\prod_{j=1}^{J}p(\boldsymbol{z}_j|\alpha,\boldsymbol{X},\boldsymbol{S},\boldsymbol{W},r)\\
&\propto p(\boldsymbol{X}|\alpha,\boldsymbol{Z},\boldsymbol{S},\boldsymbol{W},a,b,r)p(\alpha|a,b)\prod_{j=1}^{J}p(\boldsymbol{z}_j|\alpha,r)\\
&=\prod_{d=1}^{D}\prod_{n=1}^{N_d}\mathcal{N}(\boldsymbol{x}_{dn}|\boldsymbol{W}_d\boldsymbol{z}_{s_{dn}},\alpha^{-1}\boldsymbol{I})\mathcal{G}(\alpha|a,b)\prod_{j=1}^{J}\mathcal{N}(\boldsymbol{z}_j|\boldsymbol{0},(ar)^{-1}\boldsymbol{I})\\
&=\prod_{d=1}^{D}\prod_{n=1}^{N_d}\Big(\frac{\alpha}{2\pi}\Big)^{M_d/2}\exp(-\frac{\alpha}{2}\parallel\boldsymbol{x}_{dn}-\boldsymbol{W}_d\boldsymbol{z}_{s_{dn}}\parallel^2)\frac{1}{\Gamma(a)}b^a\alpha^{a-1}\exp(-b\alpha)\\
&\times\prod_{j=1}^{J}\Big(\frac{\alpha r}{2\pi}\Big)^{K/2}\exp(-\frac{\alpha r}{2}\parallel\boldsymbol{z}_j\parallel^2)\\
&\propto\alpha^{a'-1}\exp(-b'\alpha)\prod_{j=1}^{J}|\boldsymbol{C}_j|^{-\frac{1}{2}}\exp\Big(-\frac{\alpha}{2}(\boldsymbol{z}_j-\boldsymbol{\mu}_j)^\top\boldsymbol{C}_j(\boldsymbol{z}_j-\boldsymbol{\mu}_j)\Big)\\
&\propto\mathcal{G}(a',b')\prod_{j=1}^{J}\mathcal{N}(\boldsymbol{\mu}_j,\alpha^{-1}\boldsymbol{C}_j),
\end{aligned}
\tag{B.1}
$$

where we used Bayes' rule.

Appendix C. Derivation of (11)

We give here the derivation of the E-step (11) for learning based on a stochastic EM algorithm.

$$
\begin{aligned}
&p(s_{dn}=j|\boldsymbol{X},\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r,\gamma)\\
&=\frac{p(s_{dn}=j,\boldsymbol{S}_{\backslash dn},\boldsymbol{X}|\boldsymbol{W},a,b,r,\gamma)}{p(\boldsymbol{S}_{\backslash dn},\boldsymbol{x}_{dn},\boldsymbol{X}_{\backslash dn}|\boldsymbol{W},a,b,r,\gamma)}\\
&=\frac{p(s_{dn}=j,\boldsymbol{S}_{\backslash dn}|\gamma)p(\boldsymbol{X}|s_{dn}=j,\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r,\gamma)}{p(\boldsymbol{S}_{\backslash dn}|\gamma)p(\boldsymbol{X}_{\backslash dn}|\boldsymbol{S}_{\backslash dn}\boldsymbol{W},a,b,r,\gamma)p(\boldsymbol{x}_{dn}|\boldsymbol{W},a,b,r)}\\
&\propto\frac{p(s_{dn}=j,\boldsymbol{S}_{\backslash dn}|\gamma)}{p(\boldsymbol{S}_{\backslash dn}|\gamma)}\cdot\frac{p(\boldsymbol{X}|s_{dn}=j,\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r)}{p(\boldsymbol{X}_{\backslash dn}|\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r)},
\end{aligned}
\tag{C.1}
$$

where $p(\boldsymbol{x}_{dn}|\boldsymbol{W},a,b,r)$ does not depend on $p(s_{dn}=j|\boldsymbol{X},\boldsymbol{S}_{\backslash dn},\boldsymbol{W},a,b,r,\gamma)$.

References

[1] R. Socher, L. Fei-Fei, Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2010, pp. 966–973.

[2] A. Tripathi, A. Klami, S. Virpioja, Bilingual sentence matching using kernel CCA, in: MLSP '10: Proceedings of the 2010 IEEE International Workshop on Machine Learning for Signal Processing, 2010, pp. 130–135.

[3] B. Li, Q. Yang, X. Xue, Transfer learning for collaborative filtering via a rating-matrix generative model, in: Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, 2009, pp. 617–624.

[4] N. Quadrianto, A. J. Smola, L. Song, T. Tuytelaars, Kernelized sorting, IEEE Trans. on Pattern Analysis and Machine Intelligence 32 (10) (2010) 1809–1821.

[5] M. Yamada, M. Sugiyama, Cross-domain object matching with model selection, in: In Proceedings of Fourteenth International Conference on Articial Intelligence and Statistics, AISTATS '11, 2011, pp. 807–815.

[6] A. Haghighi, P. Liang, T. Berg-Kirkpatrick, D. Klein, Learning bilingual lexicons from monolingual corpora, in: Proceedings of ACL-08: HLT, 2008, pp. 771–779.

[7] A. Klami, Variational Bayesian matching, in: Proceedings of Asian Conference on Machine Learning, 2012, pp. 205–220.

[8] A. Klami, Bayesian object matching, Machine learning 92 (2-3) (2013) 225–250.

[9] M. E. Tipping, C. M. Bishop, Probabilistic principal component analysis, Journal of the Royal Statistical Society, Series B 61 (1999) 611–622.

[10] B. S. Everitt, An introduction to latent variable models, Chapman and Hall London, 1984.

[11] J. Jagarlamudi, S. Juarez, H. Daumé III, Kernelized sorting for natural language processing, in: AAAI '10: Proceedings of the 24th AAAI Conference on Artificial Intelligence, 2010.

[12] T. Iwata, T. Hirao, N. Ueda, Unsupervised cluster matching via probabilistic latent variable models, in: Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, AAAI, 2013.

[13] N. Djuric, M. Grbovic, S. Vucetic, Convex kernelized sorting, in: Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[14] A. Tripathi, A. Klami, M. Orešič, S. Kaski, Matching samples of multiple views, Data Min. Knowl. Discov. 23 (2011) 300–321.

[15] C. Rasmussen, The infinite Gaussian mixture model, Advances in neural information processing systems 12 (5.2) (2000) 2.

[16] C. Wang, S. Mahadevan, Manifold alignment without correspondence, in: Proceedings of the 21st International Joint Conference on Artifical Intelligence, IJCAI'09, 2009, pp. 1273–1278.

[17] M. Harel, S. Mannor, Learning from multiple outlooks, in: L. Getoor, T. Scheffer (Eds.), Proceedings of the 28th International Conference on Machine Learning, ICML '11, 2011, pp. 401–408.

[18] X. Shi, Q. Liu, W. Fan, P. S. Yu, R. Zhu, Transfer learning on heterogenous feature spaces via spectral transformation, in: Proceedings of the IEEE International Conference on Data Mining, ICDM '10, 2010, pp. 1049–1054.

[19] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, The Journal of Machine Learning Research 3 (2003) 583–617.

[20] F. R. Bach, M. I. Jordan, A probabilistic interpretation of canonical correlation analysis, Tech. Rep. 688, Department of Statistics, University of California, Berkeley (2005).

[21] P. Rai, H. Daumé III, Multi-label prediction via sparse infinite CCA, in: Advances in Neural Information Processing Systems, Vol. 22, 2009, pp. 1518–1526.

[22] L. Sun, S. Ji, J. Ye, Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (2011) 194–200.

[23] D. R. Hardoon, S. Szedmak, J. Shawe-Taylor, Canonical correlation analysis: an overview with application to learning methods, Neural Computation 16 (12) (2004) 2639–2664.

[24] A. Kimura, H. Kameoka, M. Sugiyama, T. Nakano, E. Maeda, H. Sakano, K. Ishiguro, SemiCCA: Efficient semi-supervised learning of canonical correlations, in: Proceedings of IAPR International Conference on Pattern Recognition, ICPR '10, 2010, pp. 2933–2936.

[25] M. Tipping, C. Bishop, Mixtures of probabilistic principal component analyzers, Neural Computation 11 (2) (1999) 443–482.

[26] Z. Ghahramani, G. E. Hinton, et al., The EM algorithm for mixtures of factor analyzers, Tech. rep., Technical Report CRG-TR-96-1, University of Toronto (1996).

[27] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, A. Y. Ng, Multimodal deep learning, in: Proceedings of the 28th International Conference on Machine Learning, 2011, pp. 689–696.

[28] N. Srivastava, R. Salakhutdinov, Multimodal learning with deep Boltzmann machines, in: Advances in Neural Information Processing Systems, 2012, pp. 2222–2230.

[29] G. Andrew, R. Arora, J. Bilmes, K. Livescu, Deep canonical correlation analysis, in: Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 1247–1255.

[30] J. Sethuraman, A constructive definition of Dirichlet priors, Statistica Sinica 4 (1994) 639–650.

[31] J. S. Liu, The collapsed Gibbs sampler in bayesian computations with applications to a gene regulation problem, Journal of the American Statistical Association 89 (427) (1994) 958–966.

[32] J. Nocedal, Updating quasi-Newton matrices with limited storage, Mathematics of computation 35 (151) (1980) 773–782.

[33] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[34] C. Chang, C. Lin, LIBSVM: a library for support vector machines, ACM Transactions on Intelligent Systems and Technology (TIST) 2 (3) (2011) 27.

[35] L. Hubert, P. Arabie, Comparing partitions, Journal of Classification 2 (1) (1985) 193–218.

[36] N. D. Lawrence, R. Urtasun, Non-linear matrix factorization with Gaussian processes, in: Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, 2009, pp. 601–608.

[37] D. J. MacKay, Bayesian interpolation, Neural computation 4 (3) (1992) 415–447.

[38] J. Boyd-Graber, D. Blei, Multilingual topic models for unaligned text, in: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, AUAI Press, 2009, pp. 75–82.

[39] T. Iwata, D. Mochihashi, H. Sawada, Learning common grammar from multilingual corpus, in: Proceedings of 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 184–188.

[40] C. Wang, S. Mahadevan, Manifold alignment using Procrustes analysis, in: ICML '08: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 1120–1127.