Cognitive Computation (2018) 10:1075–1084
https://doi.org/10.1007/s12559-018-9565-x

Knowledge Base Completion by Variational Bayesian Neural Tensor Decomposition

Lirong He[1] · Bin Liu[1] · Guangxi Li[1] · Yongpan Sheng[1] · Yafang Wang[2] · Zenglin Xu[1]

Received: 5 February 2018 / Accepted: 22 May 2018 / Published online: 26 June 2018

Abstract

Knowledge base completion is an important research problem in knowledge bases, which play important roles in question answering, information retrieval, and other applications. A number of relational learning algorithms have been proposed to solve this problem. However, despite their success in modeling the entity relations, they are not well founded in a Bayesian manner and thus are hard to model the prior information of the entity and relation factors. Furthermore, they under-represent the interaction between entity and relation factors. In order to avoid these disadvantages, we provide a neural-inspired approach, namely Bayesian Neural Tensor Decomposition approach for knowledge base completion based on the Stochastic Gradient Variational Bayesian framework. We employ a multivariate Bernoulli likelihood function to represent the existence of facts in knowledge graphs. We further employ a Multi-layered Perceptrons to represent more complex interactions between the latent *subject*, *predicate*, and *object* factors. The SGVB framework can enable us to make efficient approximate variational inference for the proposed nonlinear probabilistic tensor decomposition by a novel local reparameterization trick. This way avoids the need of expensive iterative inference schemes such as MCMC and does not make any over-simplified assumptions about the posterior distributions, in contrary to the common variational inference. In order to evaluate the proposed model, we have conducted experiments on real-world knowledge bases, i.e., FreeBase and WordNet. Experimental results have indicated the promising performance of the proposed method.

Keywords Knowledge base completion · Variational Bayesian · Neural networks

Introduction

Knowledge graphs or more generally knowledge bases provide semantically structured information that can be interpreted by computers [19]. Such representative knowledge graphs include DBPedia [1], the YAGO project [24], Freebase [3], NELL, and Google Knowledge Graphs [7]. Knowledge graphs, representing facts in the form of binary relationships, in particular the (*subject*, *predicate*, *object*) triples, where *subject* and *object* are entities and *predicate* denotes the relation between them, can be very useful in a number of intelligent systems including question answering, query expansion, and organizing structured information in search engines, e.g., [22] proposes a method to enrich SenticNet (a commonsense knowledge base of concepts) for domain-specific sentiment analysis; [26] employs common sense knowledge as a new linguistic context in handwritten Chinese text recognition and the results show that it yields improvements in recognition performance.

Knowledge graphs are usually incomplete, which are also referred to as the open-world assumption—existing triples encode known true relationships while non-existing triples remain to be unexplored. The problem of completing knowledge graphs based on existing facts can usually be addressed by multi-relational learning [19] or tensor factorization [30, 32–36]. The goal is to predict missing facts which is true with the supervision of the existing triples

✉ Zenglin Xu
zlxu@uestc.edu.cn

[1] SMILE Lab, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, China

[2] Ant Financial Service Co., Hangzhou, Zhejiang, China

as discussed in a number of methods shown in literature [7, 9, 16, 21, 23, 27, 28]. Among these methods, in particular, RESCAL [21] explains triples via pairwise interactions of latent entity features; Neural Tensor Network (NTN) [23] imposes more complex interactions between entity pairs, leading to a combined method between traditional MLPs and bilinear factor models. Probabilistic Belief Embedding (PBE) [9] estimates the probability of each candidate belief to learn the distributed representations for entities and relations, as well as the words in relation mentions simultaneously.

Despite the successful applications of these methods in knowledge base completion, they are not good at modeling the uncertainty of the output triples and incorporating prior knowledge in knowledge graphs, since they are not well founded in a Bayesian manner. However, methods based on Bayesian learning can explicitly model the uncertainty of the output triples and learn the interpretable representations of entities and relations. Another property of these methods is that they did not explicitly model the correlation or dependency between the predicate vectors, which may underestimate the deep correlations of predicates in knowledge graphs. In order to avoid these disadvantages, we provide a Bayesian Neural Tensor Decomposition approach for knowledge base completion, which is based on the recently proposed Stochastic Gradient Variational Bayesian (SGVB) framework [12]. In the generative model, as shown in Fig. 1, in order to describe the entity pair Y_{ijk} (e.g., *(tail, gecko)* holds the relationship *part of*), we introduce latent factors $\{\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k\}$ (e.g., denoting the latent representation of *(tail, part of, gecko)*). To explore the underlying complex mechanism of generating such triples, we can employ sufficiently complex functions, such as Multi-Layered Perceptrons (MLPs) or Convolutional Neural Networks (CNNs). The output of the neural network, X_{ijk}, can then be directly linked to Y_{ijk} with a Bernoulli distribution.

The proposed Bayesian neural tensor decomposition enjoys the following properties:

- Flexible link functions to represent various data types. In order to embody the existence of a triple in knowledge graphs, we employ a multivariate Bernoulli likelihood function to model the existence of facts in knowledge graphs. Alternatively, we can use Gaussian likelihood functions to describe confidence levels of facts being true in certain knowledge graphs.
- Deep modeling on the complex entity/predicate interactions. It can represent more complex interactions between the latent *subject*, *predicate*, and *object* factors, by taking arbitrary encoder functions, e.g., MLPs or CNNs. In particular, in this paper, we propose to use Multi-layered Perceptron (MLP) to model the interactions between the latent factors due to its convenience.
- Efficient and effective approximate inference. The SGVB framework can enable us to make efficient approximate variational inference for the proposed nonlinear probabilistic tensor decomposition whose latent factors have intractable posterior distributions. This is achieved by a novel reparameterization trick to the variational lower bound, which yields a simple and differential unbiased estimator of the lower bound; the so-obtained SGVB estimator can be used for efficient approximate posterior inference [12]. Then we can employ standard stochastic gradient ascent to optimize over the parameters. This way avoids the need of expensive iterative inference schemes such as MCMC and does not make any over-simplified assumptions about the posterior distributions, in contrary to the common variational inference.

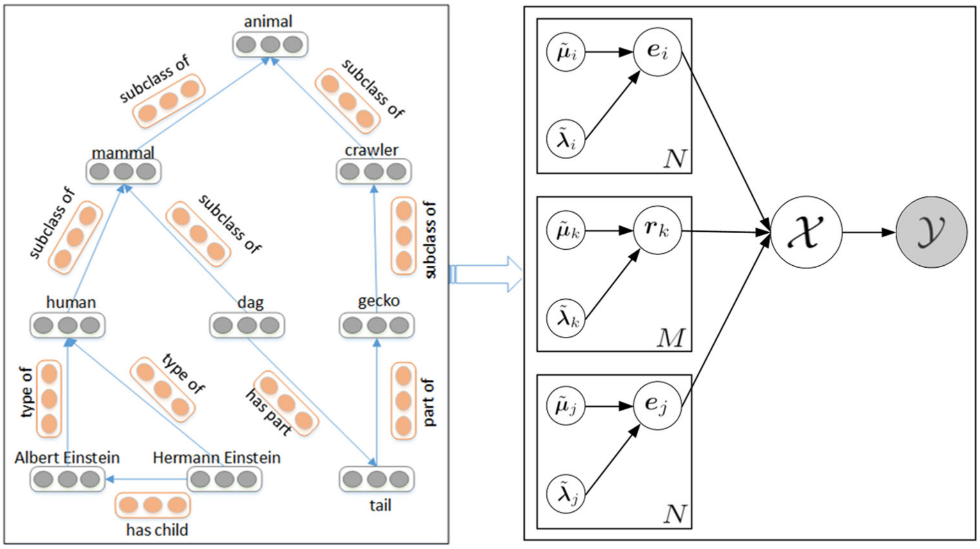


Fig. 1 Overview of our model which learns vector representations for entities and relations in a knowledge base. Entities and relations take Gaussian priors. Each triple, such as *(tail, part of, gecko)*, is given as input to a neural network. Latent variable $\mathcal{X}$ is the output of the neural network. The model predicts if the entity pair *(tail, gecko)* holds the relationship *part of* ($\mathcal{Y}$)

In order to evaluate the proposed model, we have conducted the experiment on real-world knowledge bases, i.e., WordNet and FreeBase. Experimental results have indicated the promising performance of the proposed method.

Related Work

Methods for knowledge graph completion usually follow two lines [19]. The first line of work is based on matrix factorization [10, 11, 17, 31] and tensor factorization [6, 15, 29, 30], which formulates the knowledge base completion problem as a multi-relational representation learning problem. More specifically, this kind of method encodes entities and relations in a low-dimensional latent space by utilizing Bayesian clustering methods [25, 37], energy-based models [4, 5, 16, 23], and so on. The second line of work tries to mine observed patterns in the graphs. For example, path ranking algorithm [13, 14] has been adopted to predict the links in multi-relational knowledge graphs by extending the random walks of bounded lengths.

In this paper, we focus on the first line of approach, which models the interaction schemes of (*subjects*, *predicates*, *objects*) by latent factors. In the following, we review standard and state-of-the-art methods, including TransE [5], TransR [16], RESCAL [21], Probabilistic Belief Embedding (PBE) [9], and Neural Tensor Network (NTN) [23].

TransE Model The TransE model is an energy-based model for learning low-dimensional vector embeddings of entities and relations. For a given triple (h, ℓ, t), It regards the relation ℓ as the translation from h to t, such that $\boldsymbol{h} + \boldsymbol{\ell} \approx \boldsymbol{t}$. In other words, if the relation holds, the difference measured by $f_l(h, t) = \|\boldsymbol{h} + \boldsymbol{\ell} - \boldsymbol{t}\|_2^2$ should be small. Naturally, the embedding can be learned via minimizing the margin-based loss function:

$$\mathcal{L} = \sum_{(h,\ell,t)\in S} \sum_{(h',\ell,t')\in S'_{(h,\ell,t)}} [\gamma + f_l(h, t) - f_l(h', t')]_+,$$

where $[x]_+ = \max(0, x)$ denotes the hinge loss, $\gamma > 0$, and the constructed corrupted triple set is defined as follows:

$$S'_{(h,\ell,t)} = \{(h', \ell, t)|h' \in E\} \cup \{(h, \ell, t')|t' \in E\}.$$

Here, the corrupted triplet set consists of training triplets with either its head or tail replaced by an randomly chosen entity from the existing triple set.

TransR Model Since different relations may reflect various properties and aspects of entities, it may be inappropriate for the TransE model to manipulate both entities and relations into the same semantic space. To address this issue, the TransR model introduces a projection matrix to map entities and relations in the same space. First, for each triple(h, r, t), entities $\boldsymbol{h}_r, \boldsymbol{t}_r \in \mathbb{R}^k$ are projected into an r-dimensional relation space via the projection matrix $\boldsymbol{M}_r \in \mathbb{R}^{k\times d}$, i.e., $\boldsymbol{h}_r = \boldsymbol{h}\boldsymbol{M}_r, \boldsymbol{t}_r = \boldsymbol{t}\boldsymbol{M}_r$, and then solve the approximation problem to make $\boldsymbol{h}_r + \boldsymbol{r} \approx \boldsymbol{t}_r$.

RESCAL RESCAL is a relational latent feature model, which represents the knowledge base as a tensor with each tensor entry denoting a triple. It explains the score of a tensor entry x_{ijk} as follows:

$$f_{ijk}^{RESCAL} = \boldsymbol{e}_i^\top \mathbf{W}^k \boldsymbol{e}_j = \sum_{a=1}^{N_e} \sum_{b=1}^{N_e} \boldsymbol{W}_{ab}^k \boldsymbol{e}_{ia} \boldsymbol{e}_{jb},$$

where $\boldsymbol{W}^k \in \mathbb{R}^{N_e\times N_e}$ is a weight matrix whose entries $\boldsymbol{W}_{ab}^k$ specify how much the ath latent feature and bth latent feature interact in the kth relation, and the weighted sum over entities $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$ represents how the latent features affect the score of x_{ijk} for the kth relation.

Probabilistic Belief Embedding (PBE) The PBE model is an embedding model which estimates the probability of each candidate belief (h, r, t, m), i.e., $Pr(h, r, t, m)$. Here, m denotes the words in relation mentions extracted from free texts.

It is assumed that $Pr(h, r, t, m)$ is computed by $Pr(h|r, t)$, $Pr(t|h, r)$, and $Pr(r|h, t, m)$, where $Pr(h|r, t)$ denotes the conditional probability of inferring the head entity h given the relation r and the tail entity t, $Pr(t|h, r)$ is considered as the conditional probability of reasoning the tail entity t given the relation r and the head entity h, and $Pr(r|h, t, m)$ represents the conditional probability of predicting the relation r given the head entity h, the tail entity t and the relation mentions m. Then, $Pr(h, r, t, m)$ is defined as:

$$Pr(h, r, t, m) = \sqrt[3]{Pr(h|r, t)Pr(t|h, r)Pr(r|h, t, m)}.$$

Meanwhile, based on the assumption that two entities (h, t) are independent of the relation mention m, $Pr(r|h, t, m)$ is decomposed into the following form,

$$Pr(r|h, t, m) = \sqrt{Pr(r|h, t)Pr(r|m)}.$$

Neural Tensor Network Model The Neural Tensor Network (NTN) model is originally proposed to describe whether the entities follow in a certain relation in common sense reasoning. It employs a standard linear neural network layer to represent the interaction between entities, and uses a bilinear tensor layer to directly build the connection between the latent relation factor with the entity factors. In detail, the score function for a triple (h, r, t) in this model is given as follows:

$$f_r(h,t) = \boldsymbol{\mu}_r^\top g(\boldsymbol{h}^\top \boldsymbol{M}_r \boldsymbol{t} + \boldsymbol{M}_{r,1}\boldsymbol{h} + \boldsymbol{M}_{r,2}\boldsymbol{t} + \boldsymbol{b}_r),$$

where $\boldsymbol{\mu}_r$ is a relation-specific linear layer, $g(\cdot)$ is the tanh function, $\boldsymbol{M}_r \in \mathbb{R}^{d\times d\times k}$ is a 3-way tensor, and $\boldsymbol{M}_{r,1}, \boldsymbol{M}_{r,2} \in \mathbb{R}^{k\times d}$ are weight matrices, $\boldsymbol{b}_r$ denotes the bias vector.

Among these methods, the most similar to ours is NTN. However, it is unable to model uncertainty of relations between two entities and incapable to incorporate prior knowledge on entity pairs or relations. In contrast, our method mainly focuses on modeling the uncertainty of predicting the relations.

Bayesian Neural Tensor Decomposition

A knowledge base can be represented as a tensor $\mathcal{Y} \in \mathbb{R}^{N\times N\times M}$ where each triple $(\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k)$ is represented as a tensor element, where $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$ denote latent variables corresponding to the i-th entity and the j-th entity respectively, $\boldsymbol{r}_k$ denotes the latent relationship between $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$, and $i, j \in \{1, \ldots, N\}, k \in \{1, \ldots, M\}$. A tensor element $\mathcal{Y}_{ijk} = 1$ denotes the fact that there exists the triple $(\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k)$. Otherwise, for non-existing triples, the element is set to zero. We use $\mathcal{U} = \{\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k\}_{\forall i,j,k}$ to denote the latent variables to be estimated.

In order to fully take the advantages of the complex interactions between the entities and relations, in this paper, we propose a Bayesian Neural Tensor Decomposition model for knowledge base completion. Since the value of a tensor element is binary, we postulate a likelihood function for tensor $\mathcal{Y}$ conditioned on a latent tensor $\mathcal{X}$ as follows,

$$p(\mathcal{Y}|\mathcal{X}) = \prod_{i=1}^{N}\prod_{j=1}^{N}\prod_{k=1}^{M} \mathrm{Ber}(y_{ijk}|\sigma(x_{ijk};\alpha)), \quad (1)$$

where $\mathrm{Ber}(y_{ijk}|\sigma(x_{ijk};\alpha))$ is a Bernoulli distribution with the mean of $\sigma(x_{ijk};\alpha)$, and $\sigma(u;\alpha) = 1/(1+\exp^{-\alpha u})$ is the sigmoid function. The Bernoulli likelihood model would imply a logistic loss function [20]. In this paper, we model the latent variable x_{ijk} as the output of multi-layered perceptions (MLPs) whose inputs are the latent triple $(\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k)$, i.e.,

$$x_{ijk} = \boldsymbol{w}^\top f\left(\boldsymbol{h}_{ijk}^{A} + \boldsymbol{h}_{ijk}^{B_1} + \boldsymbol{h}_{ijk}^{B_2} + \boldsymbol{h}_{ijk}^{B_3} + \boldsymbol{b}\right) + b_0, \quad (2)$$

$$\boldsymbol{h}_{ijk}^{A} = \boldsymbol{A}[\boldsymbol{e}_i; \boldsymbol{e}_j; \boldsymbol{r}_k], \quad (3)$$

$$\boldsymbol{h}_{ijk}^{B_1} = \left[\boldsymbol{e}_i^\top \boldsymbol{B}_1^{[1]}\boldsymbol{e}_j, \ldots, \boldsymbol{e}_i^\top \boldsymbol{B}_1^{[K]}\boldsymbol{e}_j\right]^\top, \quad (4)$$

$$\boldsymbol{h}_{ijk}^{B_2} = \left[\boldsymbol{e}_i^\top \boldsymbol{B}_2^{[1]}\boldsymbol{r}_k, \ldots, \boldsymbol{e}_i^\top \boldsymbol{B}_2^{[K]}\boldsymbol{r}_k\right]^\top, \quad (5)$$

$$\boldsymbol{h}_{ijk}^{B_3} = \left[\boldsymbol{e}_j^\top \boldsymbol{B}_3^{[1]}\boldsymbol{r}_k, \ldots, \boldsymbol{e}_j^\top \boldsymbol{B}_3^{[K]}\boldsymbol{r}_k\right]^\top, \quad (6)$$

where $f(\cdot)$ is the element-wise tanh activation function, $\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k \in \mathbb{R}^{d\times 1}, \boldsymbol{b} \in \mathbb{R}^{K\times 1}, \boldsymbol{w} \in \mathbb{R}^{K\times 1}, \boldsymbol{A} \in \mathbb{R}^{K\times 3d}, \boldsymbol{B}_1^{[1:K]}, \boldsymbol{B}_2^{[1:K]}, \boldsymbol{B}_3^{[1:K]} \in \mathbb{R}^{d\times d\times K}$. The network architecture is shown in Fig. 2 omitting the biases $\{\boldsymbol{b}, b_0\}$ for simplicity. We denote the parameter set $\boldsymbol{W} = \{\boldsymbol{A}, \boldsymbol{B}_1^{[1:K]}, \boldsymbol{B}_2^{[1:K]}, \boldsymbol{B}_3^{[1:K]}, \boldsymbol{b}\}$ as the weights and biases parameters of the MLP.

As shown from the above model, the pairwise interaction between (subject, object, and predicate) has been incorporated, which provides more abundant semantics than traditional models. Indeed, the proposed model can be seen as extensions of some previous factor-analysis based methods, such as RESCAL [21] and Neural Tensor Network (NTN) [23]. In particular, if we remove $\boldsymbol{h}_{ijk}^{B_2}$ and $\boldsymbol{h}_{ijk}^{B_3}$ terms, our model is similar to NTN, and if we remove $\boldsymbol{h}_{ijk}^{A}$, $\boldsymbol{h}_{ijk}^{B_2}$ and $\boldsymbol{h}_{ijk}^{B_3}$ terms, our model degenerates to RESCAL. Another advantage of employing pairwise interaction tensor over the

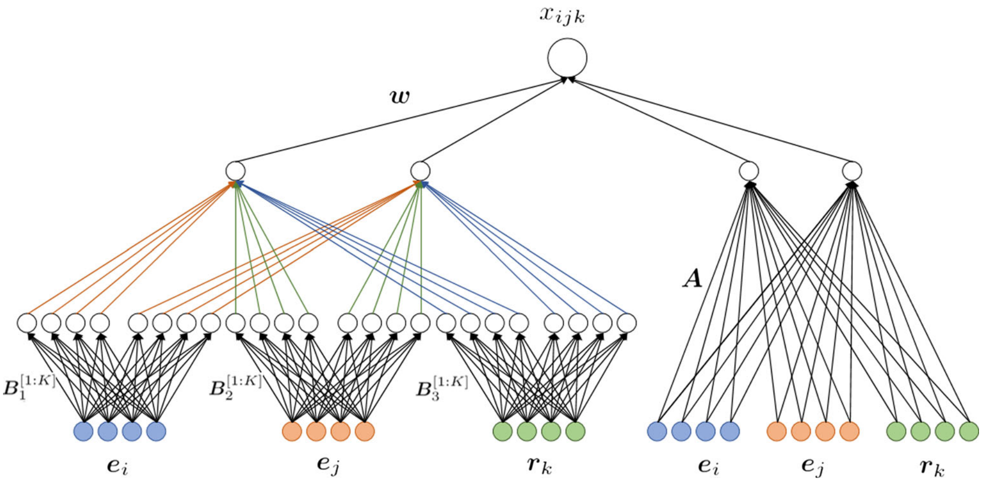


Fig. 2 The network architecture of modeling the latent variable x_{ijk} as the output of multi-layered perceptions (MLPs) whose inputs are the latent triple $(\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k)$

latent factors is its efficiency in computation. It can maintain the same level of computation complexity as NTN.

By introducing appropriate prior distributions over $\mathcal{U}$, we obtain the joint distribution as:

$$p(\mathcal{Y}, \mathcal{X}, \mathcal{U}|\Theta) = p(\mathcal{Y}|\mathcal{X})p(\mathcal{X}|\mathcal{U}, \Theta)p(\mathcal{U}). \tag{7}$$

The inference problem in Eq. 7 is intractable. However, we can obtain approximate solutions resorting to either variational inference or Markov Chain Monte Carlo(MCMC) [2]. In variational inference, to make the inference more efficient, we have to assign an appropriate prior in order to conjugate posterior distributions. The choices for certain priors are usually limited and thus lead to underestimated inference. Although MCMC can lead to more accurate estimation on posterior distributions, they are usually cost expensive. Recently, The AEVB (Auto-Encoding Variational Bayes) [12] method simplifies those complex inference and provides a more powerful way of modeling the relationships among observed variables, latent variables, and unknown parameters by introducing a Stochastic Gradient Variational Bayes Estimator.

Model Inference

Before presenting the inference procedure of Bayesian Neural Tensor Decomposition model, we first review the recently proposed Stochastic Gradient Variational Bayes method.

Stochastic Gradient Variational Bayes

In variational Inference, for a given individual example x and its parameter θ, its marginal distribution can be estimated by introducing a latent variable z, as shown in the following,

$$\log p(x|\theta) = KL(q(z|x, \phi)||p(z|x, \theta)) + \mathcal{L}(\theta, \phi|x), \tag{8}$$

where $KL(\cdot||\cdot)$ denotes the KL divergence between two distributions, θ and ϕ denote the parameters for the posterior and variational distributions, respectively. Here, the first term measures the KL divergence between the approximate posterior distribution and the real posterior distribution, and the second term is called the Evidence Lower Bound (ELBO).

Since the first term is always non-negative, maximizing the marginal distribution is equivalent to maximizing the second term $\mathcal{L}(\theta, \phi|x)$. In this subsection, a conditional independence assumption is introduced for the observed variable x given the corresponding latent variable z. Making use of the product rule of probabilities, the ELBO can be further rewritten as,

$$\begin{aligned}\log p(x|\theta) \geq \mathcal{L}(\theta, \phi|x) &= \mathbb{E}_{q(z|x,\phi)}[\log p(x, z|\theta) - \log q(z|x, \phi)]\\ &= \mathbb{E}_{q(z|x,\phi)}[\log p(x|z, \theta)]\\ &\quad -KL(q(z|x, \phi)||p(z|\theta)),\end{aligned} \tag{9}$$

where $p(x, z|\theta)$ denotes the conditional likelihood function, $\mathbb{E}_{q(z|x,\phi)}[\cdot]$ denotes the posterior expectation of the likelihood function with respect to latent variable z, whose distribution is described as the sought approximate variable posterior $q(z|x, \phi)$ and $KL(q||p)$ denotes the KL divergence between the posterior distribution $q(\cdot)$ and the prior distribution $p(\cdot)$.

In most scenarios (with non-conjugate setting), the expectation term $\mathbb{E}_{q(z|x,\phi)}[\log p(x|z, \theta)]$ in Eq. 9 is intractable. What's worse, it is hard to derive the gradient of the lower bound with respect to its posterior parameters directly. The SGVB in [12] addresses this problem by reparameterizing the latent variable $z \sim q(z|x, \phi)$ in the expectation term using a differentiable transformation $g_\phi(\epsilon)$ of an additional random noise variable ϵ,

$$z = g_\phi(\epsilon), \quad \epsilon \sim p(\epsilon). \tag{10}$$

By employing this reparameterization, we can reformulate (9) as follows:

$$\mathcal{L}(\theta, \phi|x) = -KL(q(z|x, \phi)||p(z|\theta)) + \frac{1}{L}\sum_{l=1}^{L} \log p(x|z^{(l)}, \theta), \tag{11}$$

where

$$z^{(l)} = g_\phi(\epsilon^{(l)}) \quad with \quad \epsilon^{(l)} \sim p(\epsilon). \tag{12}$$

From Eqs. 11 and 12, we can see that the main difference between SGVB estimator and a naive Monte Carlo estimator is that the drawn samples of the latent variable z, employed to estimate the intractable posterior expectation $\mathbb{E}_{q(z|x,\phi)}[\log p(x|z, \theta)]$, are considered as a transformation function g_ϕ of the parameters ϕ at present. Here, the sample size is denoted by L. Under normal circumstances, we need to select a large L (for example, $L = 20$), but experiments confirmed that we can even select to $L = 1$ as the number of drawn samples L per data point used by SGVB, as long as the minibatch size M is sufficiently large.

It is necessary to know that the reparameterization (11) of the variational lower bound also can be constructed by an estimator based on minibatches. Specifically, if we suppose the minibatch $X^{(M)} = \{x_i\}_{i=1}^{M}$ contains M randomly drawn data points from the whole dataset X, which comprises N data points, then we have

$$\mathcal{L}(\theta, \phi|X) \approx \frac{N}{M}\mathcal{L}(\theta, \phi|X^{(M)}). \tag{13}$$

We can obtain the update rules for θ, ϕ using stochastic optimization methods such as stochastic gradient descent (SGD) or Adagrad [8], to generate an efficient parameter optimization algorithm.

The architecture shown in Eq. 11 can be regarded as an Auto-Encoding Variational Bayesian (AEVB) problem. For each observed variable x, we try to encode it with the corresponding latent variable $z \sim q(z|x, \phi)$. By updating $q(z|x, \phi)$, the reconstruction loss can be measured using the second term $\frac{1}{L}\sum_{l=1}^{L} \log p(x|z^{(l)}, \theta)$, and the regularization can be calculated from the KL divergence term.

SGVB for Bayesian Neural Tensor Decomposition

To perform variational inference for our Bayesian Neural Tensor Decomposition model, appropriate prior distributions over the latent entity $\boldsymbol{e}_i$ and relationship $\boldsymbol{r}_k$ variables have to be imposed. The Gaussian distribution is useful due to the central limit theorem, therefore, we employ Gaussian priors as follows,

$$p(\boldsymbol{e}_i|\boldsymbol{\mu}_E, \boldsymbol{\lambda}_E) = \mathcal{N}(\boldsymbol{e}_i|\boldsymbol{\mu}_E, \mathrm{diag}(\boldsymbol{\lambda}_E^{-1})), \tag{14}$$

$$p(\boldsymbol{r}_k|\boldsymbol{\mu}_R, \boldsymbol{\lambda}_R) = \mathcal{N}(\boldsymbol{r}_k|\boldsymbol{\mu}_R, \mathrm{diag}(\boldsymbol{\lambda}_R^{-1})), \tag{15}$$

where $\mathrm{diag}(\boldsymbol{\lambda})$ denotes a diagonal matrix and its elements are the entries of the vector $\boldsymbol{\lambda}$. In practice, the $\{\boldsymbol{\mu}_E, \boldsymbol{\mu}_R\}$ are assigned with $\mathbf{0}$, and the precision $\{\boldsymbol{\lambda}_E, \boldsymbol{\lambda}_R\}$ are set as $\boldsymbol{I}$.

To perform Bayesian inference with AEVB for our model, we assume that the variational posterior distributions for latent variable $\boldsymbol{e}_i, \boldsymbol{r}_k$ are Gaussian distributions which are usually used in the natural and social sciences to represent real-valued random variables whose distributions are not known, as follows,

$$q(\boldsymbol{e}_i|\tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\lambda}}_i) = \mathcal{N}(\boldsymbol{e}_i|\tilde{\boldsymbol{\mu}}_i, \mathrm{diag}(\tilde{\boldsymbol{\lambda}}_i^{-1})), \tag{16}$$

$$q(\boldsymbol{r}_k|\tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\lambda}}_k) = \mathcal{N}(\boldsymbol{r}_k|\tilde{\boldsymbol{\mu}}_k, \mathrm{diag}(\tilde{\boldsymbol{\lambda}}_k^{-1})). \tag{17}$$

The posterior distributions are governed by the hyper-parameters denoted by $\phi = \{\tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\lambda}}_i, \tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\lambda}}_k\}$, which will be updated in the optimization process. Please note that, in contrast to the conventional SGVB inference, our model processes the posterior distributions over the latent feature vectors $\boldsymbol{e}_i$ and $\boldsymbol{r}_k$ that are not given by observed data.

The variational lower bound approximation in Eq. 11 of our model can be written as the following form by combining (1)–(6) and (14)–(17),

$$\begin{aligned}\mathcal{L}(\Theta, \Phi|\mathcal{Y}) = & -\sum_{i=1}^{N} KL[q(\boldsymbol{e}_i|\tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\lambda}}_i)||p(\boldsymbol{e}_i|\boldsymbol{\mu}_E, \boldsymbol{\lambda}_E)] \\ & -\sum_{j=1}^{N} KL[q(\boldsymbol{e}_j|\tilde{\boldsymbol{\mu}}_j, \tilde{\boldsymbol{\lambda}}_j)||p(\boldsymbol{e}_j|\boldsymbol{\mu}_E, \boldsymbol{\lambda}_E)] \\ & -\sum_{k=1}^{M} KL[q(\boldsymbol{r}_k|\tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\lambda}}_k)||p(\boldsymbol{r}_k|\boldsymbol{\mu}_R, \boldsymbol{\lambda}_R)] \\ & +\sum_{l=1}^{L}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{M} \frac{I_{ijk}}{L} \log Ber(y_{ijk}|\sigma^{(l)}(x_{ijk}; \alpha)),\end{aligned} \tag{18}$$

where

$$\begin{aligned}\boldsymbol{e}_i^{(l)} &= \tilde{\boldsymbol{\mu}}_i + diag(\tilde{\boldsymbol{\lambda}}_i^{-1\backslash 2})\boldsymbol{\epsilon}_i^{(l)}, \\ \boldsymbol{e}_j^{(l)} &= \tilde{\boldsymbol{\mu}}_j + diag(\tilde{\boldsymbol{\lambda}}_j^{-1\backslash 2})\boldsymbol{\epsilon}_j^{(l)}, \\ \boldsymbol{r}_k^{(l)} &= \tilde{\boldsymbol{\mu}}_k + diag(\tilde{\boldsymbol{\lambda}}_k^{-1\backslash 2})\boldsymbol{\epsilon}_k^{(l)},\end{aligned} \tag{19}$$

and

$$\boldsymbol{\epsilon}_i^{(l)}, \boldsymbol{\epsilon}_j^{(l)}, \boldsymbol{\epsilon}_k^{(l)} \sim \mathcal{N}(0, I), \tag{20}$$

where $\Theta = \{\boldsymbol{w}, \boldsymbol{A}, \boldsymbol{B}_1^{[1:K]}, \boldsymbol{B}_2^{[1:K]}, \boldsymbol{B}_3^{[1:K]}, \boldsymbol{b}, b_0\}$, $\Phi = \{\tilde{\boldsymbol{\mu}}_i, \tilde{\boldsymbol{\lambda}}_i, \tilde{\boldsymbol{\mu}}_j, \tilde{\boldsymbol{\lambda}}_j, \tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\lambda}}_k\}$, and $\boldsymbol{\mu}_E, \boldsymbol{\mu}_R$ are set as $\mathbf{0}, \boldsymbol{\lambda}_E, \boldsymbol{\lambda}_R$ are set as $\mathbf{1}$ (all one vector) in present paper. By replacing $\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{r}_k$ with differentiable transformations as in Eq. 19, we can optimize the lower bound in Eq. 18 with stochastic gradient ascent. The pseudo code in Algorithm 1 summarizes the details of the optimization process.

Algorithm 1 Our proposed model

```
Require: Training set $D, d, K, L$.
Initialize parameters, e.g., $\Theta, \Phi$
repeat
    $\mathcal{Y}^{(M)}$ ← Randomly select M datapoints (drawn from training set $D$)
    $\epsilon$ ← Random sampling of noise distribution $p(\epsilon)$
    $g \leftarrow \nabla_{\Theta,\Phi}\mathcal{L}(\Theta, \Phi|\mathcal{Y}^{(M)})$
    $\Theta, \Phi$ ← Update parameters by using SGD
until Meet the convergence criterion of parameters $\{\Theta, \Phi\}$
return $\Theta, \Phi$
```

Experiments

Our experiments are designed to predict the truth of facts in the form of $(subject, predict, object)$ of our Bayesian

Neural Tensor Decomposition model on the available real-world datasets: WordNet and Freebase. All the experiments are run on a 12-core server with 2.60GHz Intel(R) Xeon(R) E5-2630 processor and 64GB of RAM.

We evaluate our model with four baseline models:

- TransE [5]: TransE is a relatively simple model that only thinks about the distance between entities, and it does not refer to other information, such as the interaction of entities.
- TransR [16]: TransR is a development of TransE. In this model, calculation of the distance between two entities is established on the basis of them in the same space. TransR and TransE have the same shortcomings.
- RESCAL [21]: RESCAL is a method that uses a three-way tensor factorization model that takes into account the inherent structure of multi-relational data, which is able to perform collective learning.
- NTN [23]: NTN considers the effect of the linear representation and the tensor interaction between two entities. This model doesn't involve interaction between each relationship, and each relationship has its own set of parameters.

Datasets We evaluate our method with two datasets used in NTN [23], including WordNet [18] and Freebase [3]. WordNet is a combination of English dictinary and thesaurus. Words are grouped into sets of synonyms. Relations include {has_instance, type_of, member_meronym, member_holonym, part_of, has_part, subordinate_instance_of, domain_region, synset_domain_topic, similar_to, domain_topic}. See example in Fig. 1 (Hermann_Einstein, has_child, Albert_Einstein). For Freebase, it only contains relational triples from People domain with 13 relations. Relations contain {gender, nationality, profession, place_of_death, place_of_birth, location, institution, cause_of_death, religion, parents, children, ethnicity, spouse}. These two datasets have been divided into training sets, test sets and verification sets. Statistics of the two datasets are summarized in Table 1. These datasets also filters out trivial test triplets, the ones also occurs on the training dataset in a different relation or order. For instance, (e_1, similar_to, e_2) is removed from the test dataset if (e_2, similar_to, e_1) is on the training data.

During the experiments we need to artificially generate an equivalent amount of negative samples for an adversarial balance, since the training sets of two datasets only contain positive instances. In our experiments, we built the negative triplets following the same procedure used in NTN. Specifically, the negative triples on the datasets are constructed by randomly switching entities from correct triples. For example, given a true triple (David_Beckham, nationality, United_Kingdom), the negative example is (David_Beckham, nationality, United_States). The metric used in our experiments is the accuracy that calculated by the correct predicted number of triples. All the methods use the same training, validation and test datasets.

Parameter Settings The hyperparameter settings of baselines are the default. For convenience purposes, we use η, γ, d, B to denote the learning rate, margin, embedding dimension and batch size respectively. For TransE, $\eta = 0.01, \gamma = 2.0, d = 20, B = 120$ on Wordnet; $\eta = 0.001, \gamma = 2.0, d = 100, B = 30$ on Freebase. For TransR, $\eta = 0.001, \gamma = 4.0, d = 20, B = 120$ on Wordnet; $\eta = 0.0001, \gamma = 2.0, d = 100, B = 480$ on Freebase. For NTN, the regularization parameter is set to 0.0001; the dimensionality of the hidden vector is set to 100 and the number of slices is set to 3. For RESCAL, $d = 100$, the regularization parameter is set to 0 for scalability reasons.

In contrast, for our Bayesian Neural Tensor Decomposition model, we search the learning rate η in {0.001, 0.005, 0.01, 0,1}, the number of drawn samples L in {1, 10, 20, 30}, the hyperparameter of the sigmoid function α in {0.001, 0.01, 0,1}, the dimensionality of the latent feature vectors d in {10, 20, 50, 100}, and the dimensionality of the output of multiple-layer perception (MLP) K in {2, 4, 6}. The optimal configurations of our model are: $\eta = 0.01, L = 20, \alpha = 0.01, d = 20, K = 4$ on Wordnet; $\eta = 0.001, L = 20, \alpha = 0.01, d = 20, K = 4$ on Freebase. During the experiments, we find that the results of the experiments are not sensitive to the dimensionality of the latent feature vectors, and the dimensionality is not the bigger the better. so we set up the relatively low dimensions. At the same time, it could have an excellent effect, and the computing complexity is decreased. All these hyperparameters are determined by cross-validation.

Experimental Results Since the TransE is the simplest model, which contains the least information, it gets the worst result. TransR model can outperform TransE model because of the utilization of projection matrix. Rescal model has a better accuracy than TransR model. Furthermore, since NTN thinks about the interaction between entities, it

Table 1 The statistics for WordNet and Freebase

Dataset	# Relation	# Entity	# Train	# Validation	# Test
WordNet	11	38,696	112,581	2609	10,544
Freebase	13	75,043	316,232	5908	23,733

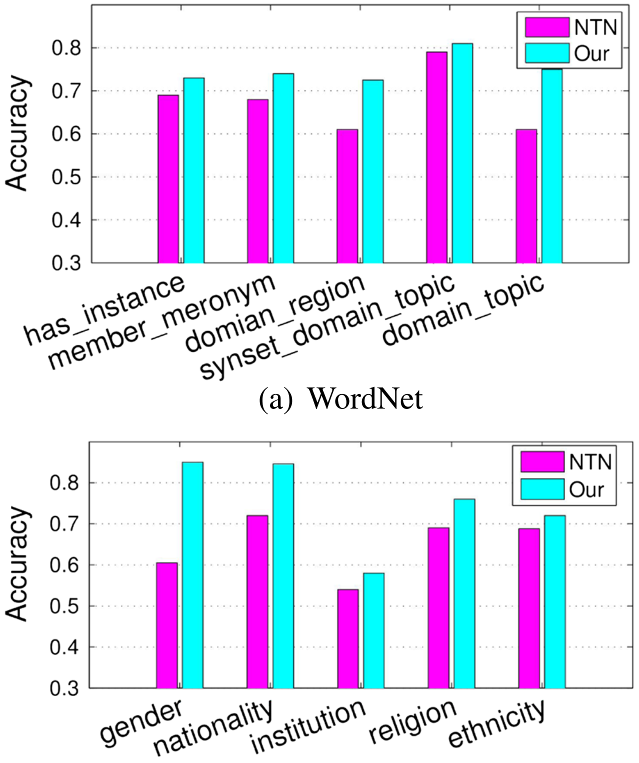
Table 2 Comparison of accuracy of different methods on two datasets

Dataset	WordNet	Freebase	Average
TransE	0.682	0.610	0.646
TransR	0.700	0.643	0.672
RESCAL	0.719	0.652	0.685
NTN	0.724	0.664	0.694
Ours	**0.741**	**0.752**	**0.747**

Results of our method are shown in bold

can have a better accuracy than TransR model. Lastly, our approach is the extension of NTN by taking into account the cross tensor interaction between entities and relations, accordingly have a higher performance than NTN on both datasets. Table 2 shows the average accuracy for all models. It can be seen that our model achieves 74.1% on WordNet dataset and 75.2% on Freebase dataset. It outperforms all the other baselines, which is about 5% higher than NTN on average.

In order to study the significance of the improvement of our model over NTN, we concretely compare the two methods in each relation. We select five relations from each dataset to illustrate the results of the comparison between our model and NTN in Fig. 3. It can be observed that our model outperforms NTN in all relations on both datasets.



(a) WordNet

(b) Freebase

Fig. 3 Comparison of accuracy of different relations of our model and NTN on two datasets

Since our model embraces the prior knowledge of entities' latent features, it could improve the prediction by the posterior of latent features in the Bayesian framework. Specifically, on WordNet, we compare five kinds of relations with NTN, the accuracy varies from 73% (has_instance) to 81% (synset_domain_topic). In terms of the domian_topic, our model wins NTN with a margin of 15%.

On Freebase, similarly, the accuracy varies from 58% (institution) to 85% (gender). Although the performance on different relations varies, our proposed model achieves generally higher accuracy than NTN. The high precision about gender and nationality is probably caused by the facts that the number of triples about them is more than the number of triples about other relations on the training data, and the entities related to them change within a small ranges.

Conclusions and Future Work

In this paper we have presented a Bayesian approach to Neural Tensor Decomposition for knowledge base completion. By encoding the interaction scheme between the decomposed latent factors of (*subject*, *predicate*, object) triples with multi-layered perceptrons or any other arbitrary functions, we can model the deep correlations or dependence between the latent factors. Furthermore, by taking advantages of the Stochastic Gradient Variational Bayes framework, we can make efficient approximate variational inference for the proposed nonlinear probabilistic tensor decomposition by a novel local reparameterization trick. Experimental results on two real-world knowledge base datasets have indicated that the proposed method can achieve very promising results in predicting missing triples compared with state-of-the-art knowledge comparison methods.

For the future work, we plan to study online learning and parallel computing to improve the scalability of Bayesian Neural Tensor Decomposition. We also plan to explore other neural networks, e.g., Convolutional Neural Networks, to model the complex interactions of *subject*, *predicate*, and *object* factors.

Funding Information All the authors except Yafang Wang were supported by the Natural Science Foundation of China (No. 61572111), 1000-Talent Startup Funds (Nos. G05QNQR004, A1098531023601041) and a Fundamental Research Fund for the Central Universities of China (No. ZYGX2016Z003).

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed Consent Informed consent was not required as no humans or animals were involved.

References

1. Auer S, Bizer C, Kobilarov G, Lehmann J, Ives Z. DBpedia: A nucleus for a web of open data. In: Proceedings ISWC; 2007. p. 11–15.
2. Bishop CM, Nasrabadi NM. Pattern recognition and machine learning. Springer, 2006. p. 461–462.
3. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J. Freebase: a collaboratively created graph database for structuring human knowledge. In: ACM'S special interest group on management of data conference; 2008. p. 1247–1250.
4. Bordes A, Glorot X, Weston J, Bengio Y. A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation. Mach Learn. 2014;94(2):233–259.
5. Bordes A, Usunier N, García-Durán A, Weston J, Yakhnenko O. Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems; 2013. pp. 2787–2795.
6. Chen S, Lyu MR, King I, Xu Z. Exact and stable recovery of pairwise interaction tensors. In: Advances in neural information processing systems 26: 27th annual conference on neural information processing systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe; 2013. pp. 1691–1699.
7. Dong X, Gabrilovich E, Heitz G, Horn W, Lao N, Murphy K, Strohmann T, Sun S, Zhang W. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: ACM SIGKDD international conference on knowledge discovery and data mining; 2014. p. 601–610.
8. Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res. 2011;12(Jul):2121–2159.
9. Fan M, Zhou Q, Abel A, Zheng TF, Grishman R. Probabilistic belief embedding for large-scale knowledge population. Cogn Comput. 2016;8(6):1087–1102.
10. Huang S, Wang H, Li T, Li T, Xu Z. Robust graph regularized nonnegative matrix factorization for clustering. Data Min Knowl Discov. 2018;32(2):483–503. https://doi.org/10.1007/s10618-017-0543-9.
11. Huang S, Xu Z, Lv J. Adaptive local structure learning for document co-clustering. Knowl-Based Syst. 2018;148:74–84. https://doi.org/10.1016/j.knosys.2018.02.020.
12. Kingma DP, Welling M. Auto-encoding variational bayes. arXiv:1312.6114. 2013.
13. Lao N, Cohen WW. Relational retrieval using a combination of path-constrained random walks. Mach Learn. 2010;81(1):53–67.
14. Lao N, Mitchell T, Cohen WW. Random walk inference and learning in a large scale knowledge base. In: Conference on empirical methods in natural language processing, EMNLP 2011, john mcintyre conference centre, edinburgh, uk, a meeting of sigdat, a special interest group of the ACL; 2012. p. 529–539.
15. Li G, Xu Z, Wang L, Ye J, King I, Lyu MR. Simple and efficient parallelization for probabilistic temporal tensor factorization. In: 2017 international joint conference on neural networks, IJCNN 2017, anchorage; 2017, p. 1–8.
16. Lin Y, Liu Z, Zhu X, Zhu X, Zhu X. Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI conference on artificial intelligence; 2015. p. 2181–2187.
17. Liu B, Li Y, Xu Z. Manifold regularized matrix completion for multi-label learning with ADMM. Neural Netw. 2018;101:57–67. https://doi.org/10.1016/j.neunet.2018.01.011.
18. Miller GA. Wordnet: a lexical database for english. Commun Acm. 1995;38(11):39–41.
19. Nickel M, Murphy K, Tresp V, Gabrilovich E. A review of relational machine learning for knowledge graphs. Proc IEEE. 2016;104(1):11–33.
20. Nickel M, Tresp V. Logistic tensor factorization for multi-relational data. arXiv:1306.2084. 2013.
21. Nickel M, Tresp V, Kriegel HP. A three-way model for collective learning on multi-relational data. In: International conference on international conference on machine learning; 2011, vol. 11. p. 809–816.
22. Ofek N, Poria S, Rokach L, Cambria E, Hussain A, Shabtai A. Unsupervised commonsense knowledge enrichment for domain-specific sentiment analysis. Cogn Comput. 2016;8(3):467–477.
23. Socher R, Chen D, Manning CD, Ng AY. Reasoning with neural tensor networks for knowledge base completion. In: Advances in neural information processing systems; 2013. p. 926–934.
24. Suchanek FM, Kasneci G, Weikum G. Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on World Wide Web. ACM; 2007. p. 697–706.
25. Sutskever I, Salakhutdinov R, Tenenbaum JB. Modelling relational data using bayesian clustered tensor factorization. In: Advances in neural information processing systems; 2009. p. 1821–1828.
26. Wang QF, Cambria E, Liu CL, Hussain A. Common sense knowledge for handwritten Chinese text recognition. Cogn Comput. 2013;5(2):234–242.
27. Wang Z, Zhang J, Feng J, Chen Z. Knowledge graph embedding by translating on hyperplanes. In: The association for the advance of artificial intelligence; 2014, vol. 14. p. 1112–1119.
28. Weston J, Bordes A, Yakhnenko O, Usunier N. Connecting language and knowledge bases with embedding models for relation extraction. In: Conference on empirical methods in natural language processing; 2013. p. 1366–1371.
29. Xu Z, Yan F, Qi Y. Infinite tucker decomposition: Nonparametric bayesian models for multiway data analysis. In: Proceedings of the 29th international conference on machine learning, ICML 2012. Edinburgh; 2012.
30. Xu Z, Yan F, Qi Y. Bayesian nonparametric models for multiway data analysis. IEEE Trans Pattern Anal Mach Intell. 2015;37(2):475–487.
31. Yang X, Huang K, Zhang R, Hussain A. Learning latent features with infinite non-negative binary matrix tri-factorization. IEEE Trans Emerg Topics Comput Intell. 2018;2(3). https://doi.org/10.1109/TETCI.2018.2806934.
32. Zhe S, Qi Y, Park Y, Xu Z, Molloy I, Chari S. Dintucker: Scaling up gaussian process models on large multidimensional arrays. In: Proceedings of the thirtieth AAAI conference on artificial intelligence. Phoenix; 2016. p. 2386–2392.
33. Zhe S, Xu Z, Chu X, Qi Y, Park Y. Scalable nonparametric multiway data analysis. In: Proceedings of the eighteenth

international conference on artificial intelligence and statistics, AISTATS 2015, San Diego; 2015.

34. Zhe S, Zhang K, Wang P, Lee K, Xu Z, Qi Y, Ghahramani Z. Distributed flexible nonlinear tensor factorization. In: Advances in neural information processing systems 29, Barcelona; 2016. p. 920–928.
35. Zhong G, Cheriet M. Large margin low rank tensor analysis. Neural Comput. 2014;26(4):761–780.
36. Zhong G, Cheriet M. Tensor representation learning based image patch analysis for text identification and recognition. Pattern Recogn. 2015;48(4):1211–1224.
37. Zhu J. Max-margin nonparametric latent feature models for link prediction. In: Proceedings of the 29th international coference on international conference on machine learning. Omnipress; 2012. p. 1179–1186.