An Effective and Efficient Entity Alignment Decoding Algorithm via Third-Order Tensor Isomorphism

Xin Mao[1*], Meirong Ma[2], Hao Yuan[2], Jianchao Zhu[2],
Zongyu Wang[3], Rui Xie[3], Wei Wu[3], Man Lan[1*]
[1]School of Computer Science and Technology, East China Normal University
[2]Transsion Group, [3]Meituan Group
xmao@stu.ecnu.edu.cn,mlan@cs.ecnu.edu.cn
{meirong.ma,hao.yuan,jianchao.zhu}@transsion.com
{wangzongyu02,rui.xie,wuwei30}@meituan.com

Abstract

Entity alignment (EA) aims to find the equivalent entity pairs between KGs, which is a crucial step for integrating multi-source KGs. For a long time, most researchers have regarded EA as a pure graph representation learning task and focused on improving graph encoders while paying little attention to the decoding process. In this paper, we propose an effective and efficient EA **D**ecoding **A**lgorithm via **T**hird-order **T**ensor **I**somorphism (DATTI). Specifically, we derive two sets of isomorphism equations: (1) Adjacency tensor isomorphism equations and (2) Gramian tensor isomorphism equations. By combining these equations, DATTI could effectively utilize the adjacency and inner correlation isomorphisms of KGs to enhance the decoding process of EA. Extensive experiments on public datasets indicate that our decoding algorithm can deliver significant performance improvements even on the most advanced EA methods, while the extra required time is less than 3 seconds.

1 Introduction

Knowledge graphs (KGs) illustrate the relations between real-world entities—e.g., objects, situations, or concepts—and usually are stored in the form of triples $(subject, relation, object)$. Over recent years, a large number of KGs have been constructed to provide structural knowledge to facilitate downstream applications, such as recommendation systems (Cao et al., 2019) and question-answering systems (Zhao et al., 2020).

Most KGs are independently extracted from different languages or domains. Thus, these KGs usually hold unique information individually but also have some shared parts. Integrating these cross-lingual / domain KGs could provide a broader view for users, especially for the minority language users who usually suffer from lacking language resources. As shown in Figure 1, entity alignment (EA) aims to find the equivalent entity pairs between KGs, which is a crucial step for integrating KGs.

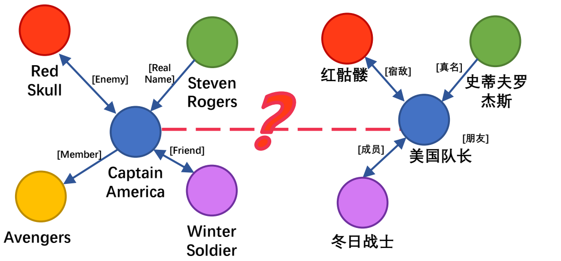

Figure 1: An example of cross-lingual entity alignment.

Existing EA methods are built on the same core premise: equivalent entity pairs between KGs have similar neighborhood structures (i.e., isomorphism). Therefore, most existing EA methods (Wang et al., 2018; Sun et al., 2020b; Mao et al., 2020) could be abstracted into the same architecture (as shown in Figure 2): encoding the structural information of KGs into a low-dimensional vector space by Siamese graph encoders and then mapping equivalent entity pairs into the proximate space by alignment loss functions.

For a long time, most researchers have regarded EA as a graph representation learning task and focused on improving graph encoders. Starting from the simplest graph encoder TransE (Bordes et al., 2013), the newest graph encoding methods are successively introduced into EA and achieve decent improvements. For example, GCN-align (Wang et al., 2018) first proposed to use graph convolutional networks (GCN) (Kipf and Welling, 2017) to encode KGs. RSN (Guo et al., 2019) introduces recurrent neural networks (RNN) (Graves et al., 2008) and biased random walk to exploit the long-term relational dependencies existing in KGs. Dual-AMN (Mao et al., 2021a) proposes the proxy-matching layer and normalized hard samples mining loss to speed up the training process.

In stark contrast to the efforts on graph encoders, few researchers focus on improving EA decoding

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 5888 - 5898
May 22-27, 2022 ©2022 Association for Computational Linguistics

algorithms (Sun et al., 2020c), which have been proved to significantly improve performance and reliability in other fields, such as dependency parsing (Zmigrod et al., 2020) and machine translation (He et al., 2021). Earlier EA studies (Wang et al., 2018; Sun et al., 2017) simply calculate the similarities of each pair of entities and select the closest one as the alignment result. This naive strategy results in one entity may be aligned to multiple entities simultaneously, which violates the one-to-one constraint of EA [1]. Thus, some recent studies (Xu et al., 2020; Zhu et al., 2021) propose the global alignment strategy, i.e., regarding the decoding process as a one-to-one assignment problem that could be solved by the Hungarian algorithm (Kuhn, 1955). Overall, these studies just use existing decoding algorithms without further exploration of KGs' characteristics. Similar to graph encoders, we argue that a good EA decoding algorithm should also be capable of exploiting the structural information of KGs.

In this paper, we propose an effective and efficient EA **D**ecoding **A**lgorithm via **T**hird-order **T**ensor **I**somorphism (DATTI). Different from recent studies (Fey et al., 2020; Mao et al., 2021b) that regard EA as a matrix (second-order tensor) isomorphism problem, we express the isomorphism of KGs in the form of third-order tensors, which could completely describe the structural information of KGs. Specifically, we derive two sets of tensor isomorphism equations: (1) Adjacency tensor isomorphism equations and (2) Gramian tensor isomorphism equations. By combining these equations, DATTI could effectively utilize the adjacency and inner correlation isomorphisms of KGs to enhance the decoding process of EA, thus significantly improving the performance. Besides, the introduction of third-order tensors will inevitably lead to a quadratic increase in space-time complexity. Therefore, we adopt the randomized truncated singular value decomposition algorithm (RTSVD) (Sarlós, 2006) and Sinkhorn operator (Sinkhorn, 1964) to improve efficiency.

To comprehensively evaluate our proposed method, we apply DATTI to three advanced EA methods with different kinds of graph encoders. Experimental results on two widely used public datasets show that DATTI can deliver significant performance improvements (3.9% on *Hits@1* and 3.2% on *MRR*) even on the most advanced EA methods. Furthermore, our decoding algorithm is highly efficient. The decoding time is less than 3 seconds, which is almost negligible compared to the time consumption of the training process. The main contributions are summarized as follows:

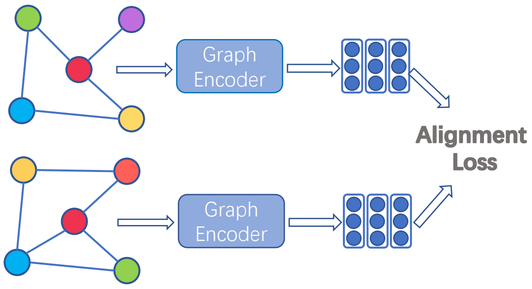

Figure 2: The architecture of existing EA methods.

- We propose an effective and efficient EA Decoding Algorithm via Third-order Tensor Isomorphism (DATTI), which consists of two sets of tensor isomorphism equations: (1) Adjacency tensor isomorphism equations and (2) Gramian tensor isomorphism equations.
- Extensive experiments on public datasets indicate that our decoding algorithm can deliver significant performance improvements even applied to the SOTA method, while the extra required time is less than 3 seconds.

2 Task Definition

A KG could be defined as $G = (E, R, T)$, where E,R, and T represent the entity set, relation set, and triple set, respectively. Given a source graph $G_s = (E_s, R_s, T_s)$ and a target graph $G_t = (E_t, R_t, T_t)$, the goal of EA is to explore the one-to-one entity correspondences $\boldsymbol{P}_e$ between KGs.

3 Related Work

3.1 Encoders and Enhancement

The core premise of EA methods is that equivalent entity pairs between KGs have similar neighborhood structures. As shown in Figure 2, most of them could be summarized into two steps: (1) Using KG embedding methods (e.g., TransE, GCN, and GAT (Velickovic et al., 2018)) to encode entities and relations into low-dimensional embeddings. (2) Mapping these embeddings into a unified vector space through pre-aligned entity pairs and alignment loss functions. To organize existing EA methods clearly, we categorize them based on the encoders and enhancement strategies in Table 1.

[1] Most KGs usually have removed the duplicated entities within the same KG.

Encoders and Losses. There are mainly two kinds of Encoders: *Trans* represents TransE (Bordes et al., 2013) and subsequent derivative algorithms. These methods assume that entity and relation embeddings follow the equation $h + r \approx t$. Because of the easy implementation, the *Trans* encoders are widely used in early EA methods. More recently, Graph Neural Networks (GNN) gradually became the mainstream encoder because of their powerful modeling capability on graph structures. Inspired by language models, RSN proposes a biased random walk sampling strategy and uses RNN to encode the sampled sequences. As for alignment losses, the vast majority of EA methods (Wang et al., 2018; Wu et al., 2019; Mao et al., 2020) adopt contrastive losses, e.g., Triplet loss (Schroff et al., 2015). These loss functions share one core idea, attracting positive entity pairs and repulsing negative entity pairs.

Enhancement. Due to the lack of labeled data, several methods (Sun et al., 2018; Mao et al., 2020) adopt iterative strategies to produce semi-supervised aligned entity pairs. Despite significant performance improvements, the time consumption of these methods increases several times more. Some methods (Xu et al., 2019; Yang et al., 2019) introduce textual information (e.g., entity name embeddings) as the initial features of GNN to provide a multi-aspect view. However, literal information is not always available in real applications. For example, there will be privacy risks when using user-generated content. Therefore, we will separately discuss these textual-based methods in the experiment section.

As mentioned in Section 1, some studies (Xu et al., 2020; Wu et al., 2019) regard the decoding process as a one-to-one assignment problem. The assignment problem is a fundamental combinatorial optimization problem. An intuitive instance is to assign N jobs for N workers. The assignment problem is to find a one-to-one assignment plan so that the total profit is maximum. Formally, it is equivalent to maximizing the following equation:

$$\underset{\boldsymbol{P} \in \mathbb{P}_N}{arg\ max}\ \langle \boldsymbol{P}, \boldsymbol{X} \rangle_F \tag{1}$$

$\boldsymbol{X} \in \mathbb{R}^{N \times N}$ is the profit matrix. $\boldsymbol{P}$ is a permutation matrix denoting the assignment plan. There are exactly one entry of 1 in each row and each column in $\boldsymbol{P}$ while 0s elsewhere. $\mathbb{P}_N$ represents the set of all N-dimensional permutation matrices. Here, $\langle \cdot \rangle_F$ represents the Frobenius inner product.

Method	Encoder	Enhancement
JAPE (Sun et al., 2017)	Trans	✗
GCN-Align (Wang et al., 2018)	GNN	✗
OTEA (Pei et al., 2019)	Trans	✗
RSN (Guo et al., 2019)	RNN	✗
BootEA (Sun et al., 2018)	Trans	Semi
TransEdge(Sun et al., 2020a)	Trans	Semi
MRAEA (Mao et al., 2020)	GNN	Semi
Dual-AMN (Mao et al., 2021a)	GNN	Semi
GM-Align (Xu et al., 2019)	GNN	Entity Name
RDGCN (Wu et al., 2019)	GNN	Entity Name
DGMC (Fey et al., 2020)	GNN	Entity Name
AttrGNN (Liu et al., 2020)	GNN	Entity Name
CREA (Xu et al., 2020)	GNN	Hungarian
RAGA (Zhu et al., 2021)	GNN	Hungarian

Table 1: Categorization of some popular EA methods.

4 The Proposed Method

In the following, we describe our proposed decoding algorithm (DATTI), which consists of two sets of tensor isomorphism equations: (1) Adjacency tensor isomorphism equations and (2) Gramian tensor isomorphism equations. Furthermore, we adopt the randomized truncated singular value decomposition (RTSVD) algorithm and the Sinkhorn operator to speed up the decoding process.

4.1 Adjacency Isomorphism

Some recent studies (Fey et al., 2020; Mao et al., 2021b) regard EA as a matrix isomorphism problem. These methods assume that the adjacency matrices $\boldsymbol{A}_s \in \mathbb{R}^{|E_s| \times |E_s|}$ of source graph G_s and $\boldsymbol{A}_t \in \mathbb{R}^{|E_t| \times |E_t|}$ of target graph G_t are isomorphic, i.e., $\boldsymbol{A}_s$ could be transformed into $\boldsymbol{A}_t$ according to the entity correspondence matrix $\boldsymbol{P}_e$:

$$\boldsymbol{P}_e \boldsymbol{A}_s \boldsymbol{P}_e^\top = \boldsymbol{A}_t \tag{2}$$

$\boldsymbol{P}_{e_{[i,j]}} = 1$ indicates that e_i and e_j are equivalent. However, matrices (second-order tensors) cannot fully describe the adjacency information of KGs, which is stored in the form of triples. Therefore, we use third-order tensors to express KGs to avoid the information missing from using matrices. Let $\boldsymbol{\mathcal{A}_s} \in \mathbb{R}^{|E_s| \times |R_s| \times |E_s|}$ and $\boldsymbol{\mathcal{A}_t} \in \mathbb{R}^{|E_t| \times |R_t| \times |E_t|}$ be the adjacency tensors of G_s and G_t. $\boldsymbol{\mathcal{A}}_{[h,r,t]} = 1$ indicates that the triple (h, r, t) is in the KG. The matrix isomorphism Equation (2) could be generalized into the third-order form as follows:

$$\boldsymbol{\mathcal{A}_s} \times_1 \mathbf{P_e} \times_2 \mathbf{P_r} \times_3 \mathbf{P_e} = \boldsymbol{\mathcal{A}_t} \tag{3}$$

where $\boldsymbol{P}_r$ represents the one-to-one relation correspondence matrix between G_s and G_t and $\times_k$ represents the k-mode tensor-matrix product.

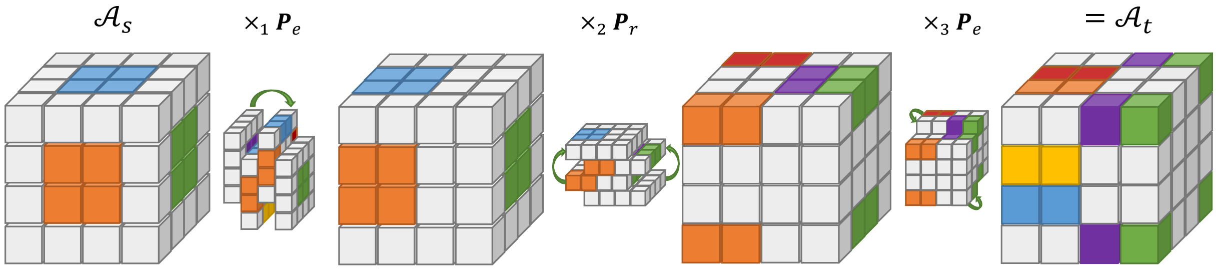
$\mathcal{A}_s$
$\times_1 \boldsymbol{P}_e$
$\times_2 \boldsymbol{P}_r$
$\times_3 \boldsymbol{P}_e$
$= \mathcal{A}_t$

Figure 3: The illustration of tensor-matrix product and isomorphic adjacency tensors.

As illustrated in Figure 3, Equation (3) can be interpreted as successively reordering the tensor along three axes. Since the number of triples $|T|$ is usually much less than $|E| \times |R| \times |E|$, $\boldsymbol{\mathcal{A}_s}$ and $\boldsymbol{\mathcal{A}_t}$ are extremely sparse. Unfortunately, existing tensor computing frameworks (e.g., Numpy (Harris et al., 2020) and Tensorflow (Abadi et al., 2015)) can only provide few and limited operators for third-order sparse tensors. Therefore, we have to re-transform Equation (3) into the matrix form:

$$\begin{aligned} & \boldsymbol{\mathcal{A}_s} \times_1 \mathbf{P_e} \times_2 \mathbf{P_r} \times_3 \mathbf{P_e} = \boldsymbol{\mathcal{A}_t} \\ \iff \quad & \boldsymbol{P}_e \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})} (\boldsymbol{P}_e \otimes \boldsymbol{P}_r)^\top = \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})} \\ & \boldsymbol{P}_r \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{2})} (\boldsymbol{P}_e \otimes \boldsymbol{P}_e)^\top = \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{2})} \\ & \boldsymbol{P}_e \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{3})} (\boldsymbol{P}_r \otimes \boldsymbol{P}_e)^\top = \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{3})} \end{aligned} \tag{4}$$

here $\otimes$ represents the Kronecker product, $\boldsymbol{P}_e \otimes \boldsymbol{P}_r \in \mathbb{P}^{(|E|\cdot|R|)\times(|E|\cdot|R|)}$. $\boldsymbol{\mathcal{A}}^{(\boldsymbol{k})}$ represents the mode-k unfolding matrix of the tensor $\boldsymbol{\mathcal{A}}$, e.g., $\boldsymbol{\mathcal{A}}^{(\mathbf{1})} = [\boldsymbol{\mathcal{A}}_{[:,:,0]} \| \boldsymbol{\mathcal{A}}_{[:,:,1]} \| ... \| \boldsymbol{\mathcal{A}}_{[:,:,|E|]}] \in \mathbb{R}^{|E|\times(|E|\cdot|R|)}$, where $\|$ is the concatenate operation. When $\boldsymbol{\mathcal{A}_s}$ and $\boldsymbol{\mathcal{A}_t}$ are second-order adjacency tensors, the above equations degrade to Equation (2):

$$\begin{aligned} & \boldsymbol{\mathcal{A}_s} \times_1 \mathbf{P_e} \times_2 \mathbf{P_e} = \boldsymbol{\mathcal{A}_t} \\ \iff \quad & \boldsymbol{P}_e \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})} \boldsymbol{P}_e^\top = \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})} \end{aligned} \tag{5}$$

4.2 Gramian Isomorphism

Gramian matrix $G(\boldsymbol{A}) = \boldsymbol{A}\boldsymbol{A}^\top$ reflects the inner correlations between each vector of matrix $\boldsymbol{A}$. If we regard $\boldsymbol{A}$ as random variables, $G(\boldsymbol{A})$ is equivalent to the uncentered covariance matrix. When $\boldsymbol{A}_s$ and $\boldsymbol{A}_t$ are isomorphic, their Gramian matrices $\boldsymbol{A}_s\boldsymbol{A}_s^\top$ and $\boldsymbol{A}_t\boldsymbol{A}_t^\top$ are isomorphic too:

$$\boldsymbol{A}_t\boldsymbol{A}_t^\top = (\boldsymbol{P}_e\boldsymbol{A}_s\boldsymbol{P}_e^\top)(\boldsymbol{P}_e\boldsymbol{A}_s\boldsymbol{P}_e^\top)^\top = \boldsymbol{P}_e\boldsymbol{A}_s\boldsymbol{A}_s^\top\boldsymbol{P}_e^\top \tag{6}$$

Similar to adjacency matrices, the Gramian matrix isomorphism equation could also be generalized into the third-order form:

$$\begin{aligned} \boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})}) \boldsymbol{P}_e^\top &= G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})}) \\ \boldsymbol{P}_r G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{2})}) \boldsymbol{P}_r^\top &= G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{2})}) \\ \boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{3})}) \boldsymbol{P}_e^\top &= G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{3})}) \end{aligned} \tag{7}$$

Furthermore, it is easy to prove that the following equations hold for arbitrary depth $l \in \mathbb{N}$:

$$\begin{aligned} \boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{P}_e^\top &= G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})})^l \\ \boldsymbol{P}_r G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{2})})^l \boldsymbol{P}_r^\top &= G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{2})})^l \\ \boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{3})})^l \boldsymbol{P}_e^\top &= G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{3})})^l \end{aligned} \tag{8}$$

4.3 Decoding via Isomorphism

Although we have derived two sets of isomorphic equations, neither of them could be solved directly. These equations are equivalent to the quadratic or cubic assignment problem (Yan et al., 2016), which has been proved to be NP-hard (Lawler, 1963). Fortunately, these isomorphic equations could be used to enhance the decoding process.

Let $\boldsymbol{H}_s^e \in \mathbb{R}^{|E_s|\times d^e}$ and $\boldsymbol{H}_s^r \in \mathbb{R}^{|R_s|\times d^r}$ represent the entity and relation embeddings of G_s. $\boldsymbol{H}_t^e \in \mathbb{R}^{|E_t|\times d^e}$ and $\boldsymbol{H}_t^r \in \mathbb{R}^{|R_t|\times d^r}$ represent the embeddings of G_t. Assume that these embeddings have been approximately aligned by EA methods:

$$\begin{aligned} \boldsymbol{P}_e\boldsymbol{H}_s^e &\approx \boldsymbol{H}_t^e \\ \boldsymbol{P}_r\boldsymbol{H}_s^r &\approx \boldsymbol{H}_t^r \end{aligned} \tag{9}$$

As mentioned in Section 1, some recent studies (Xu et al., 2020; Sun et al., 2020c) regard the decoding process of $\boldsymbol{P}_e$ as an assignment problem:

$$\begin{aligned} & \underset{\boldsymbol{P}_e \in \mathbb{P}_{|E|}}{arg\ min} \|\boldsymbol{P}_e\boldsymbol{H}_s^e - \boldsymbol{H}_t^{e^\top}\|_F^2 \\ \iff \quad & \underset{\boldsymbol{P}_e \in \mathbb{P}_{|E|}}{arg\ max} \left\langle \boldsymbol{P}_e, \boldsymbol{H}_s^e\boldsymbol{H}_t^{e^\top} \right\rangle_F \end{aligned} \tag{10}$$

Since this simple decoding strategy does not utilize the structural information of KGs, we propose to introduce the adjacency and Gramian isomorphism equations into the decoding process. By combining Equations (4), (8), and (9), the connection between the 8-tuple $\{\boldsymbol{\mathcal{A}_s}, \boldsymbol{\mathcal{A}_t}, \boldsymbol{H}_s^e, \boldsymbol{H}_t^e, \boldsymbol{H}_s^r, \boldsymbol{H}_t^r, \boldsymbol{P}_e, \boldsymbol{P}_r\}$ could be described as follows, for arbitrary depth $l \in \mathbb{N}$:

$$\boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})}(\boldsymbol{H}_s^e \otimes \boldsymbol{H}_s^r) \approx G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})}(\boldsymbol{H}_t^e \otimes \boldsymbol{H}_t^r) \tag{11}$$

$$\boldsymbol{P}_r G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{2})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{2})}(\boldsymbol{H}_s^e \otimes \boldsymbol{H}_s^e) \approx G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{2})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{2})}(\boldsymbol{H}_t^e \otimes \boldsymbol{H}_t^e) \tag{12}$$

$$\boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{3})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{3})}(\boldsymbol{H}_s^r \otimes \boldsymbol{H}_s^e) \approx G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{3})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{3})}(\boldsymbol{H}_t^r \otimes \boldsymbol{H}_t^e) \tag{13}$$

Detailed proof is listed in Appendix A. Although it looks complex, the above equations essentially have the same form as Equation (9). Take Equation (11) as an example, let $\hat{\boldsymbol{H}}_s^l = G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})}(\boldsymbol{H}_s^e \otimes \boldsymbol{H}_s^r)$ and $\hat{\boldsymbol{H}}_t^l = G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})}(\boldsymbol{H}_t^e \otimes \boldsymbol{H}_t^r)$, Equation (11) can be simplified into as follows:

$$\boldsymbol{P}_e \hat{\boldsymbol{H}}_s^l \approx \hat{\boldsymbol{H}}_t^l \tag{14}$$

Therefore, $\boldsymbol{P}_e$ could also be solved by maximizing the equation $\underset{\boldsymbol{P}_e \in \mathbb{P}_{|E|}}{arg\ max} \left\langle \boldsymbol{P}_e, \hat{\boldsymbol{H}}_s^l \hat{\boldsymbol{H}}_t^{l\top} \right\rangle_F$. Theoretically, for arbitrarily depth $l \in \mathbb{N}$, the result of $\boldsymbol{P}_e$ should be the same. However, the above equations are based on the ideal isomorphic situation. In practice, $\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}$ and $\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}$ can not always be strictly isomorphic. In order to reduce the impact of noise existing in practice, $\boldsymbol{P}_e$ should be fit for various l:

$$\begin{aligned} &\sum_{l=0}^{L} \underset{\boldsymbol{P}_e \in \mathbb{P}_{|E|}}{arg\ max} \left\langle \boldsymbol{P}_e, \hat{\boldsymbol{H}}_s^l \hat{\boldsymbol{H}}_t^{l\top} \right\rangle_F \\ \Longleftrightarrow \quad &\underset{\boldsymbol{P}_e \in \mathbb{P}_{|E|}}{arg\ max} \left\langle \boldsymbol{P}_e, \sum_{l=0}^{L} \hat{\boldsymbol{H}}_s^l \hat{\boldsymbol{H}}_t^{l\top} \right\rangle_F \end{aligned} \tag{15}$$

By Equation (15), we successfully integrate the adjacency and Gramian isomorphism equations into the decoding process of EA. Similar to the above, Equation (12) could obtain the relation alignment result $\boldsymbol{P}_r$. Because Equation (13) is equivalent to Equation (11), it only needs to solve either of them to obtain the entity alignment result $\boldsymbol{P}_e$. It is noted that entity scales $|E_s|$ and $|E_t|$ are usually inconsistent in practice, which is called the unbalanced assignment problem. Assuming that $|E_s|>|E_t|$, a naive solution is to pad the profit matrix with zeros such that its shape becomes $\mathbb{R}^{|E_s|\times|E_s|}$.

4.4 Reducing the Complexity

Randomized truncated SVD. The introduction of third-order tensors enables DATTI to fully describe the structural information of KGs. However, there is no such thing as a free lunch. The space-time complexity also increases quadratically. The main bottleneck is to compute $\hat{\boldsymbol{H}}_s^l \in \mathbb{R}^{|E_s|\times(d^e \cdot d^r)}$ and $\hat{\boldsymbol{H}}_t^l \in \mathbb{R}^{|E_t|\times(d^e \cdot d^r)}$. Even with the sparse optimization trick, the complexity is still up to $O(ld^r d^e|T|)$, which is much worse than most GNN encoders $O(l(d^e + d^r)|T|)$ (Mao et al., 2020).

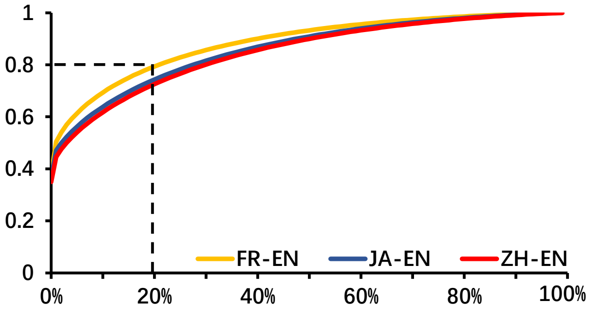

Figure 4: The singular value distribution of $\hat{\boldsymbol{H}}_s^l$ obtained by TransEdge on DBP15K.The abscissa represents the top $k\%$ singular values, and the ordinate represents the proportion of these singular values in total.

In Figure 4, we list the singular value distribution of $\hat{\boldsymbol{H}}_s^l$ obtained by TransEdge (Sun et al., 2020a) on DBP15K. Interestingly, the distribution is highly concentrated in the top 20%, which means the contained information of $\hat{\boldsymbol{H}}_s^l$ is sparse and compressible. By dropping the smaller singular values of $\hat{\boldsymbol{H}}_s^l$ and $\hat{\boldsymbol{H}}_t^l$, the space-time complexity could be significantly reduced. This paper adopts randomized truncated SVD (Sarlós, 2006) to decompose matrices approximately and only retains the top $\phi\%$ of the singular values of $\hat{\boldsymbol{H}}_s^l$ and $\hat{\boldsymbol{H}}_t^l$.

Sinkhorn operator. The first and most well-known solving algorithm for the assignment problem is the Hungarian algorithm (Kuhn, 1955), which is based on improving a matching along the augmenting paths. The time complexity of the original Hungarian algorithm is $O(n^4)$. Then, Jonker and Volgenant (1987) improve the algorithm to achieve an $O(n^3)$ running time.

Besides the Hungarian algorithm, the assignment problem could also be regarded as a special case of the optimal transport (OT) problem. Based on the Sinkhorn operator (Sinkhorn, 1964), Cuturi (2013) proposes a fast and completely parallelizable algorithm for OT problem:

$$\begin{aligned} S^0(\boldsymbol{X}) &= exp(\boldsymbol{X}), \\ S^k(\boldsymbol{X}) &= \mathcal{N}_c(\mathcal{N}_r(S^{k-1}(\boldsymbol{X}))), \\ \text{Sinkhorn}(\boldsymbol{X}) &= \lim_{k\to\infty} S^k(\boldsymbol{X}). \end{aligned} \tag{16}$$

where $\mathcal{N}_r(\boldsymbol{X}){=}\boldsymbol{X}\oslash(\boldsymbol{X}\mathbf{1}_N\mathbf{1}_N^T)$ and $\mathcal{N}_c{=}\boldsymbol{X}\oslash(\mathbf{1}_N\mathbf{1}_N^T\boldsymbol{X})$ are the row and column-wise normalization operators of a matrix, $\oslash$ represents the element-wise division, and $\mathbf{1}_N$ is a column vector of ones.

Datasets		$\|E\|$	$\|R\|$	$\|T\|$
$\text{DBP}_{\text{ZH-EN}}$	Chinese	19,388	1,701	70,414
	English	19,572	1,323	95,142
$\text{DBP}_{\text{JA-EN}}$	Japanese	19,814	1,299	77,214
	English	19,780	1,153	93,484
$\text{DBP}_{\text{FR-EN}}$	French	19,661	903	105,998
	English	19,993	1,208	115,722
$\text{SRPRS}_{\text{FR-EN}}$	French	15,000	177	33,532
	English	15,000	221	36,508
$\text{SRPRS}_{\text{DE-EN}}$	German	15,000	120	37,377
	English	15,000	222	38,363

Table 2: Statistical data of DBP15K and SRPRS.

Then, Mena et al. (2018) further prove that the Sinkhorn operation could also solve the assignment problem as a special case of OT problem:

$$\begin{aligned} & \underset{\boldsymbol{P}\in\mathbb{P}_N}{arg\ max}\ \langle \boldsymbol{P}, \boldsymbol{X}\rangle_F \\ = & \lim_{\tau\to 0^+} \text{Sinkhorn}(\boldsymbol{X}/\tau) \end{aligned} \tag{17}$$

The time complexity of the Sinkhorn operator is $O(kn^2)$. According to our experimental results, a small k is enough to achieve decent performance. Compared with the Hungarian algorithm, the Sinkhorn operation is much more efficient. Therefore, this paper adopts the Sinkhorn operator to solve Equation (15).

5 Experiments

Our experiments are conducted on a PC with a GeForce GTX 3090 GPU and a Ryzen ThreadRipper 3970X CPU. The code and datasets are available in Github [2].

5.1 Datasets

To comprehensively evaluate the proposed decoding algorithm, we experiment with two widely used public datasets: (1) **DBP15K** (Sun et al., 2017) consists of three cross-lingual subsets from multilingual DBpedia. Each subset contains $15,000$ entity pairs. (2) **SRPRS** (Guo et al., 2019). Each subset also contains $15,000$ entity pairs but with much fewer triples compared to DBP15K. The statistics of these datasets are summarized in Table 2. To be consistent with previous studies (Wang et al., 2018; Sun et al., 2018), we randomly split 30% of the pre-aligned entity pairs for training and development while using the remaining 70% for testing. All the results are the average of five independent runs.

5.2 Baselines

To ensure the universality, we evaluate DATTI on three advanced EA methods with different types of graph encoders: Dual-AMN (Mao et al., 2021a) is the SOTA of GNN-based methods; TransEdge (Sun et al., 2020a) is the SOTA of Trans-based methods; RSN (Guo et al., 2019) is the only EA method using RNN as the encoder. Furthermore, we choose the Hungarian algorithm (*Hun.*) as the decoding baseline, proven to be effective by recent EA methods (Xu et al., 2020; Zhu et al., 2021).

5.3 Settings

Metrics. Following convention, we use *Hits@k* and *Mean Reciprocal Rank* (MRR) as the evaluation metrics. The $Hits@k$ score is calculated by measuring the proportion of correct pairs in the top-k. In particular, *Hits@1* equals accuracy.

Hyper-parameter. For TransEdge, we retain the top ϕ=20% of the singular values of $\hat{\boldsymbol{H}}_s^l$ and $\hat{\boldsymbol{H}}_t^l$. Since the output dimensions of Dual-AMN (d^e=768, d^r=128) and RSN (d^e=d^r=256) are much larger than TransEdge (d^e=d^r=75), we only set the retaining ratio ϕ=2%. Other hyper-parameters keep the same for all datasets and methods: iterations k=15; temperature τ=0.02; max depth L=3.

5.4 Main Experiments

We list the main experimental results in Table 3. Among these three EA methods, Dual-AMN beats other baselines by more than 5.5% on *Hits@1* and 4.2% on *MRR*, which indicates the advantages of GNN encoders. On RSN and TransEdge, the Hungarian algorithm shows decent performance improvements on *Hits@1* by at least 3.2%. In contrast, the Hungarian does not positively affect Dual-AMN, probably due to the bi-directional nearest iterative strategy of Dual-AMN that has included the core idea of the Hungarian algorithm.

Our proposed DATTI consistently achieves the best performances on all datasets and baselines. On DBP15K, DATTI delivers performance gains by at least 2.8% on *Hits@1* and 3.2% on *MRR*. Especially for the SOTA method Dual-AMN, DATTI further raises the performance ceiling of EA by more than 3.9% on *Hits@1*. On SRPRS, DATTI could significantly improve the performances of RSN and TransEdge. But for Dual-AMN, the improvements are much less. One possible explanation is that SRPRS removes too many triples, resulting in a lower performance ceiling.

[2] https://github.com/MaoXinn/DATTI

Method	DBP$_{\text{ZH-EN}}$			DBP$_{\text{JA-EN}}$			DBP$_{\text{FR-EN}}$			SRPRS$_{\text{FR-EN}}$			SRPRS$_{\text{DE-EN}}$		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
RSN	0.607	0.829	0.685	0.591	0.815	0.670	0.632	0.864	0.713	0.351	0.638	0.447	0.511	0.744	0.590
+ Hun.	0.661	-	-	0.633	-	-	0.693	-	-	0.374	-	-	0.538	-	-
+ DATTI	0.721	0.903	0.785	0.686	0.895	0.759	0.720	0.918	0.790	0.407	0.694	0.502	0.559	0.782	0.637
(Imp.%)	9.1%	8.9%	14.6%	8.4%	9.8%	13.3%	3.9%	6.3%	10.8%	8.8%	8.8%	12.3%	3.9%	5.1%	8.0%
TransEdge	0.762	0.921	0.818	0.746	0.929	0.811	0.769	0.940	0.830	0.403	0.675	0.492	0.556	0.753	0.633
+Hun.	0.787	-	-	0.771	-	-	0.796	-	-	0.427	-	-	0.574	-	-
+DATTI	0.814	0.947	0.863	0.804	0.957	0.861	0.818	0.965	0.873	0.441	0.707	0.521	0.593	0.782	0.673
(Imp.%)	3.4%	2.8%	5.5%	4.3%	3.0%	6.2%	2.8%	2.7%	5.2%	3.3%	4.7%	5.9%	3.5%	3.8%	6.3%
Dual-AMN	0.804	0.937	0.853	0.803	0.947	0.856	0.834	0.962	0.881	0.483	0.755	0.573	0.612	0.819	0.683
+Hun.	0.801	-	-	0.803	-	-	0.839	-	-	0.483	-	-	0.611	-	-
+DATTI	**0.835**	**0.953**	**0.880**	**0.836**	**0.969**	**0.884**	**0.873**	**0.979**	**0.913**	**0.495**	**0.760**	**0.583**	**0.623**	**0.822**	**0.691**
(Imp.%)	3.9%	1.7%	3.2%	4.1%	2.3%	3.3%	4.7%	1.8%	3.6%	2.5%	0.6%	1.7%	1.8%	0.4%	1.2%

Table 3: Main experimental results on DBP15K and SRPRS. All the results and initial embeddings are obtained by their official code with default hyper-parameters. Imp.% represents the percentage increase of DATTI compared to the suboptimal result. Since the Hungarian algorithm only outputs one aligned entity pair for each entity, instead of a rank list, we can only report *Hits@1*. All improvements are statistically significant with p<0.01 on paired t-test.

Method	DBP15K		SRPRS	
	Train	DATTI	Train	DATTI
RSN	3,659	2.4	1,279	1.7
TransEdge	1,625	1.3	907	1.2
Dual-AMN	177	3.3	163	2.6

Table 4: Time costs (second) on DBP15K and SRPRS.

5.5 Auxiliary Experiments

To explore the behavior of our proposed decoding algorithm in different situations, we design the following experiments:

Time Efficiency. By adopting RTSVD and the Sinkhorn operator, our proposed decoding algorithm acquires high efficiency. Table 4 lists the time costs of the training and decoding process (DATTI) of three EA methods on DBP15K and SRPRS. DATTI only requires 3 seconds to obtain the result at most, which is negligible even compared to the training process of the fastest method Dual-AMN.

Adjacency and Gramian Isomorphism. The core contribution of DATTI is to introduce the adjacency and Gramian isomorphism equations into the EA decoding process. To demonstrate their effectiveness, we independently add each of them on Dual-AMN. As shown in Table 5, both could slightly improve the performance (less than 1.6% on *Hits@1*). Interestingly, the performance gain brought by their combination is greater than the sum of their independent gains, which means these two kinds of isomorphism equations could capture non-overlapping information.

Method	DBP$_{\text{ZH-EN}}$		DBP$_{\text{JA-EN}}$		DBP$_{\text{FR-EN}}$	
	Hits@1	MRR	Hits@1	MRR	Hits@1	MRR
Dual-AMN	0.804	0.853	0.803	0.856	0.834	0.881
+Adj.	0.820	0.866	0.818	0.868	0.859	0.902
+Gram.	0.809	0.857	0.812	0.863	0.848	0.895
+DATTI	0.835	0.880	0.836	0.884	0.873	0.913

Table 5: Ablation studies on DBP15K.

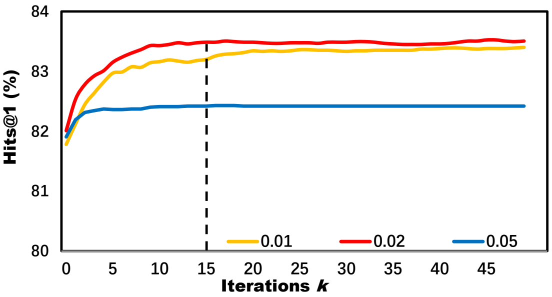

Figure 5: *Hits@1* on DBP$_{\text{ZH-EN}}$ with different τ.

Iterations k and Temperature τ. The τ in the Sinkhorn operator is used to make distribution closer to one-hot, which is similar to the τ in the softmax operator. We set τ from 0.01 to 0.05 and report the corresponding performance curves of DATTI (Dual-AMN) on DBP$_{\text{ZH-EN}}$ in Figure 5. If we choose an appropriate value, the Sinkhorn operator will converge quickly to the optimal solution. Although τ theoretically needs to be close to zero, an over small τ will make the algorithm unstable because of the error of big floating-point numbers. In contrast, an over large τ will lead the algorithm to fail to converge.

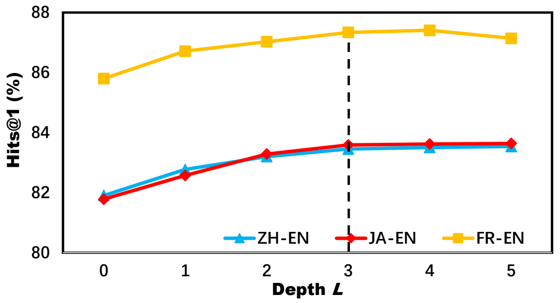

Figure 6: *Hits@1* on DBP15K with different depths L.

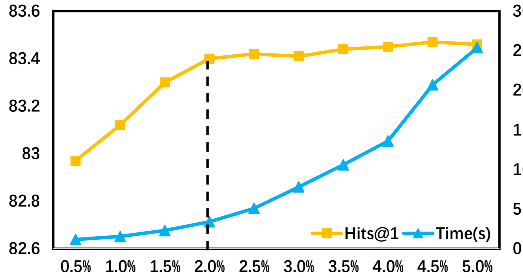

Figure 7: *Hits@1* and time cost (second) on DBP$_{\text{ZH-EN}}$ with different retaining ratios ϕ.

Method	DBP$_{\text{ZH-EN}}$		DBP$_{\text{JA-EN}}$		DBP$_{\text{FR-EN}}$	
	Hits@1	Hits@10	Hits@1	Hits@10	Hits@1	Hits@10
GM-Align	0.679	0.785	0.740	0.872	0.894	0.952
RDGCN	0.697	0.842	0.763	0.763	0.873	0.957
DGMC	0.801	0.875	0.848	0.897	0.933	0.960
AtrrGNN	0.796	0.929	0.783	0.920	0.919	0.979
CREA	0.736	-	0.792	-	0.924	-
RAGA	0.873	-	0.909	-	0.966	-
Init-Emb	0.625	0.756	0.680	0.807	0.848	0.919
+Hun.	0.667	-	0.728	-	0.893	-
+DATTI	**0.890**	**0.958**	**0.921**	**0.971**	**0.979**	**0.995**
(Imp.%)	1.9%	3.1%	1.3%	5.5%	1.3%	1.6%

Table 6: Performances of textual-based EA methods. The results of baselines are collected from the origin papers. Init-Emb represents only using the cosine similarity between the averaged name embeddings.

Depth L. Figure 6 lists the performances of DATTI (Dual-AMN) with different max depths L. In particular, $L = 0$ is equivalent to only using adjacency isomorphism equations to decode $\boldsymbol{P}_e$. When the depth L is less than 3, each additional layer could deliver significant performance improvements on all subsets of DBP15K. When stacking more layers, the performance gains become negligible or even degrade, which indicates that over-smoothing (Kipf and Welling, 2017) also exists in DATTI.

Retaining ratio ϕ. To reduce the space-time complexity of DATTI, we only retain the top $\phi\%$ of the singular values of $\hat{\boldsymbol{H}}_s^l$ and $\hat{\boldsymbol{H}}_t^l$. In Figure 7, we report the *Hits@1* and time cost of DATTI (Dual-AMN) on DBP$_{\text{ZH-EN}}$ with different retaining ratios ϕ. From the observation, when the retaining ratio exceeds 2%, the growth of *Hits@1* becomes very slow, while the time cost still keeps quadratic growing. Therefore, ϕ=2% is the sweet spot between performance and efficiency in this situation. In practice, the retaining ratio ϕ could be adjusted according to computing resources and data scales.

5.6 Unsupervised Entity Alignment

So far, all the experiments are based on pure structural-based EA methods. As mentioned in Section 3.1, some methods (Xu et al., 2020; Wu et al., 2019) introduce textual information (e.g., entity name) to provide a multi-aspect view. Specifically, these methods first use machine translation systems or cross-lingual word embeddings to map entity and relation names into a unified semantic space and then average the pre-trained word embeddings to construct the initial features for entities and relations. In our opinion, since the initial features of entity $\boldsymbol{H}^e$ and relation $\boldsymbol{H}^r$ have been pre-mapped, these textual-based EA methods are more like decoding algorithms to eliminate the translation noise. In this situation, DATTI could also play a similar role even without any pre-aligned entity pairs.

To make fair comparisons with these textural-based EA methods, we use the same entity name translations and pre-trained word embeddings provided by Xu et al. (2019). For DATTI, we retain the top 10% of the singular values of $\hat{\boldsymbol{H}}_s^l$ and $\hat{\boldsymbol{H}}_t^l$, while keeping other hyper-parameters the same. Table 6 lists the performances of DATTI and six baselines on DBP15K. Surprisingly, unsupervised DATTI outperforms all the supervised competitors, improves the performance on *Hits@1* by more than 1.3%. Besides showing the powerful competitiveness of DATTI, this result also indicates that existing textural-based EA methods have considerable redundancy. When the initial features have been pre-mapped, complex neural networks and pre-aligned entity pairs may not be necessary.

6 Conclusion

In this paper, we propose an effective and efficient EA decoding algorithm via third-order tensor isomorphism (DATTI). Extensive experiments on public datasets indicate that our decoding algorithm can deliver significant performance improvements even on the most advanced EA methods, while the extra required time is less than 3 seconds.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. 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, Nevada, United States*, pages 2787–2795.

Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 151–161.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. 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, Nevada, United States*, pages 2292–2300.

Matthias Fey, Jan Eric Lenssen, Christopher Morris, Jonathan Masci, and Nils M. Kriege. 2020. Deep graph matching consensus. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. 2008. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):855–868.

Lingbing Guo, Zequn Sun, and Wei Hu. 2019. Learning to exploit long-term relational dependencies in knowledge graphs. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 2505–2514.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature*, 585(7825):357–362.

Hao He, Qian Wang, Zhipeng Yu, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2021. Synchronous interactive decoding for multilingual neural machine translation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12981–12988. AAAI Press.

Roy Jonker and A. Volgenant. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Eugene L Lawler. 1963. The quadratic assignment problem. *Management science*, 9(4):586–599.

Zhiyuan Liu, Yixin Cao, Liangming Pan, Juanzi Li, and Tat-Seng Chua. 2020. Exploring and evaluating attributes, values, and structures for entity alignment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6355–6364. Association for Computational Linguistics.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2021a. Boosting the speed of entity alignment 10 ×: Dual attention matching network with normalized hard sample mining. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 821–832. ACM / IW3C2.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2021b. From alignment to assignment: Frustratingly simple unsupervised entity alignment. *CoRR*, abs/2109.02363.

Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020. MRAEA: an efficient and robust entity alignment approach for cross-lingual knowledge graph. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 420–428.

Gonzalo E. Mena, David Belanger, Scott W. Linderman, and Jasper Snoek. 2018. Learning latent permutations with gumbel-sinkhorn networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Shichao Pei, Lu Yu, and Xiangliang Zhang. 2019. Improving cross-lingual entity alignment via optimal transport. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3231–3237.

Tamás Sarlós. 2006. Improved approximation algorithms for large matrices via random projections. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 143–152. IEEE Computer Society.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823. IEEE Computer Society.

Richard Sinkhorn. 1964. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879.

Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, volume 10587 of *Lecture Notes in Computer Science*, pages 628–644. Springer.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4396–4402.

Zequn Sun, JiaCheng Huang, Wei Hu, Muchao Chen, Lingbing Guo, and Yuzhong Qu. 2020a. Transedge: Translating relation-contextualized embeddings for knowledge graphs. *CoRR*, abs/2004.13579.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020b. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 222–229.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020c. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proc. VLDB Endow.*, 13(11):2326–2340.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. 2018. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 349–357.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019. Relation-aware entity alignment for heterogeneous knowledge graphs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5278–5284.

Kun Xu, Linfeng Song, Yansong Feng, Yan Song, and Dong Yu. 2020. Coordinated reasoning for cross-lingual knowledge graph alignment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9354–9361.

Kun Xu, Liwei Wang, Mo Yu, Yansong Feng, Yan Song, Zhiguo Wang, and Dong Yu. 2019. Cross-lingual knowledge graph alignment via graph matching neural network. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3156–3161.

Junchi Yan, Xu-Cheng Yin, Weiyao Lin, Cheng Deng, Hongyuan Zha, and Xiaokang Yang. 2016. A short survey of recent advances in graph matching. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, ICMR 2016, New York, New York, USA, June 6-9, 2016*, pages 167–174. ACM.

Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. 2019. Aligning cross-lingual entities with multi-aspect information. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4430–4440.

Chen Zhao, Chenyan Xiong, Xin Qian, and Jordan L. Boyd-Graber. 2020. Complex factoid question answering with a free-text knowledge graph. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 1205–1216.

Renbo Zhu, Meng Ma, and Ping Wang. 2021. RAGA: relation-aware graph attention networks for global entity alignment. In *Advances in Knowledge Discovery and Data Mining - 25th Pacific-Asia Conference, PAKDD 2021, Virtual Event, May 11-14, 2021, Proceedings, Part I*, volume 12712 of *Lecture Notes in Computer Science*, pages 501–513. Springer.

Ran Zmigrod, Tim Vieira, and Ryan Cotterell. 2020. Please mind the root: Decoding arborescences for dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4809–4819. Association for Computational Linguistics.

A Appendix

Proof: To prove Equation (11), we combine the first sub-equations of Equation (4) and (8):

$$\begin{cases} \boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{P}_e^\top = G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})})^l \\ \boldsymbol{P}_e \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})} (\boldsymbol{P}_e \otimes \boldsymbol{P}_r)^\top = \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})} \end{cases}$$

Because $\boldsymbol{P}_e^\top \boldsymbol{P}_e = \boldsymbol{E}$, thus:

$$\boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})} (\boldsymbol{P}_e \otimes \boldsymbol{P}_r)^\top = G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})}$$

According to Equation (9), we could obtain:

$$\boldsymbol{P}_e \boldsymbol{H}_s^e \otimes \boldsymbol{P}_r \boldsymbol{H}_s^r \approx \boldsymbol{H}_t^e \otimes \boldsymbol{H}_t^r \tag{18}$$

Finally, because of $(\boldsymbol{P}_e \otimes \boldsymbol{P}_r)^\top (\boldsymbol{P}_e \boldsymbol{H}_s^e \otimes \boldsymbol{P}_r \boldsymbol{H}_s^r) = \boldsymbol{P}_e^\top \boldsymbol{P}_e \boldsymbol{H}_s^e \otimes \boldsymbol{P}_r^\top \boldsymbol{P}_r \boldsymbol{H}_s^r = \boldsymbol{H}_s^e \otimes \boldsymbol{H}_s^r$, Equation (11) is proved as follows:

$$\begin{aligned} &\boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})} (\boldsymbol{P}_e \otimes \boldsymbol{P}_r)^\top (\boldsymbol{P}_e \boldsymbol{H}_s^e \otimes \boldsymbol{P}_r \boldsymbol{H}_s^r) \\ &\qquad = \boldsymbol{P}_e G(\boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{s}}^{(\mathbf{1})} (\boldsymbol{H}_s^e \otimes \boldsymbol{H}_s^r) \\ &\qquad \approx G(\boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})})^l \boldsymbol{\mathcal{A}}_{\boldsymbol{t}}^{(\mathbf{1})} (\boldsymbol{H}_t^e \otimes \boldsymbol{H}_t^r) \end{aligned}$$

Furthermore, Equations (12) and (13) could also be proved in similar way. □