Rethinking Smoothness for Fast and Adaptable Entity Alignment Decoding

Anonymous ACL submission

Abstract

Entity alignment (EA) is crucial for integrating multi-source knowledge graphs (KGs), aiming to identify equivalent entities across different graphs. However, most existing EA decoding methods rely on both entity and relation embeddings, limiting their generalizability and efficiency, especially in GNN-based models. To address these challenges, we propose Triple Feature Propagation (TFP), an adaptable and fast EA decoding framework that only utilizes entity embeddings. TFP reconstructs KG representation by maximizing the smoothness of entity embeddings. The discretized smoothness-maximization process yields the explicit Euler solution of TFP. We also generalize multi-view matrices: *entity-to-entity*, *entity-to-relation*, *relation-to-entity*, and *relation-to-triple*, to capture structural diversity. Extensive experiments on public datasets demonstrate that TFP is fast and adaptable to various encoders, achieving comparable results to state-of-the-art methods in under 6 seconds, and surpassing them in many cases.

1 Introduction

Knowledge Graphs (KGs) are pivotal in diverse Natural Language Processing applications (Yu and Yang, 2023; Wang et al., 2023; Chakrabarti, 2022), serving as crucial repositories of structured knowledge. However, their utility is often hampered by incomplete coverage. To integrate multi-source KGs, which often contain overlapping facts and complementary information, Entity Alignment (EA) seeks to identify equivalent entities in distinct KGs. This typically consists of an encoding phase, where entity and relation embeddings are generated, and a decoding phase, crucial for establishing alignments (as shown in Fig.1 (1)) (Ji et al., 2022).

Recent advancements in EA encoders fall into two categories: translation-based models, such as TransE (Bordes et al., 2013) and its variants (Chen et al., 2017; Guo et al., 2019), and Graph Neural Network (GNN)-based models, like GCN-Align (Wang et al., 2018) and Dual-AMN (Mao et al., 2021a). Translation-based models treat relation embeddings as translation vectors between entities, while GNN-based models aggregate neighboring embeddings to generate entity representations.

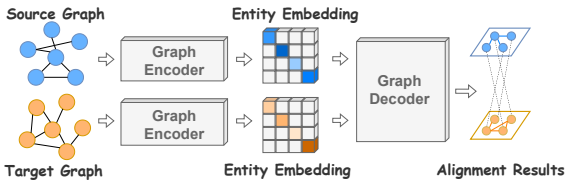

Figure 1: The architecture of existing EA methods that involve the encoding and decoding phase.

Despite progress in encoder design, research on decoders remains limited (Zhu et al., 2024). Decoders are either time-consuming or heavily dependent on encoder outputs. DATTI (Mao et al., 2022a) uses third-order tensor isomorphism to leverage structural information for decoding, but its effectiveness is constrained by the need for explicit relation embeddings, which many GNN-based encoders lack. While LightEA (Mao et al., 2022b) offers an efficient decoding framework, it remains computationally expensive in practice. Other methods like CSLS (Lample et al., 2018) and global alignment strategies, such as the Hungarian (Kuhn, 1955) or Sinkhorn (Cuturi, 2013) algorithms, are fast and simple but fail to fully exploit the structural properties of KGs.

In this work, we propose Triple Feature Propagation (TFP), a fast EA decoding method designed to maximize the smoothness of entity embeddings. TFP reconstructs KG representations using only entity embeddings by minimizing Dirichlet energy (DE), which results in a propagation-based differential equation on the graph. Discretizing this equation yields an explicit Euler solutiona sim-

ple, fast, and adaptable iterative algorithm. We also generalize the traditional adjacency matrix to multi-view matrices (*entity-to-entity*, *entity-to-relation*, *relation-to-entity*, and *relation-to-triple*) to capture the full structural diversity of KGs. TFP leverages these multi-view matrices within its iterative process instead of the traditional adjacency matrix. We evaluate TFP across both translation- and GNN-based encoders on public datasets. The results show that TFP is fast, adaptable to various encoders, and achieves results comparable to state-of-the-art methods in under six seconds, surpassing them in many cases. Our contributions are summarized as follows:

- We introduce TFP, a fast and adaptable decoding method that reconstructs entity embeddings by maximizing smoothness, suitable for various types of encoders.
- We provide a solid mathematical foundation for TFP by discretizing the smoothness-maximization differential equation, leading to an explicit Euler solution.
- Extensive experiments show that TFP improves upon existing EA methods, using only entity embeddings with minimal computational cost, typically under 6 seconds.

2 Related Work

Entity Alignment Encoders. EA is predominantly viewed as a graph representation learning endeavor, with encoders designed to intricately model KG structures (Zeng et al., 2021a). Encoders fall into two main categories: translation-based, exemplified by TransE (Bordes et al., 2013) and its variants (Chen et al., 2017; Guo et al., 2019; Pei et al., 2019), which emphasize embedding learning strategy; and GNN-based, which leverage a variety of GNN architectures to produce entity embeddings. These include simple GCNs (Wang et al., 2018), multi-hop and relational GCNs (Sun et al., 2020; Yu et al., 2020), graph attention networks (Zhu et al., 2020; Mao et al., 2021a; Sun et al., 2022), and self-supervised GCNs (Li and Song, 2022). Innovations extend to semi-supervised (Li and Song, 2022; Mao et al., 2020a) and active learning approaches (Zeng et al., 2021b; Liu et al., 2021; Berrendorf et al., 2021), plus the integration of additional information like entity attributes or temporal data (Sun et al., 2017; Trisedya et al., 2019; Wu et al., 2019a; Xu et al., 2021, 2022; Liu et al., 2023). Our TFP decoding strategy aims to reconstruct entity representation by maximizing the smoothness. TFP, adaptable across graph encoder types, enhances these encoders, including advanced models.

Entity Alignment Decoders traditionally aim at pairing entities from different KGs using learned embeddings. The *greedy search* algorithm (Ye et al., 2019; Shi and Xiao, 2019), though widely used, can produce multiple mappings for a single entity, conflicting with EA's one-to-one mapping principle. Alternative methods like Cross-Domain Similarity Local Scaling (CSLS) (Lample et al., 2018) and the deferred acceptance algorithm (Roth, 2008) have sought to refine alignment accuracy. More recent efforts (Xu et al., 2020; Zhu et al., 2021) employ global alignment strategies, using algorithms such as the Hungarian (Kuhn, 1955) or Sinkhorn (Cuturi, 2013), yet often overlook KGs' distinctive structural properties. DATTI (Mao et al., 2022a) advances decoding by utilizing third-order tensor isomorphism to capture structural information but is limited by its requirement for explicit relation embeddings, absent in many GNN-based models. While LightEA (Mao et al., 2022b) offers an adaptable decoding framework, it remains computationally expensive in practice. TFP distinguishes itself by focusing on entity embeddings, fundamental to all encoder types, and establishes a fast and adaptable decoding solution requiring less than 6 seconds, applicable across a wide range of encoder architectures. [1]

3 Preliminary

Definition 1. ***Knowledge Graph (KG)*** *stores the real-world knowledge in the form of* $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$*, given a set of entities* $\mathcal{E}$ *as nodes, a set of relations* $\mathcal{R}$ *as edges, and a set of triples* $\mathcal{T} = \{(h, r, t), |h, t \in \mathcal{E}, r \in \mathcal{R}\}$*, where* h, t *denote the head and tail entity,* r *denotes the relation.*

Definition 2. ***Entity Alignment (EA)*** *aims to discover a one-to-one mapping* $\Phi = \{(e_s, e_t), |, e_s \in \mathcal{E}_s, e_t \in \mathcal{E}_t, e_s \equiv e_t\}$ *between entities from a source KG* $\mathcal{G}_s = (\mathcal{E}_s, \mathcal{R}_s, \mathcal{T}_s)$ *to a target KG* $\mathcal{G}_t = (\mathcal{E}_t, \mathcal{R}_t, \mathcal{T}_t)$*, where with* $\equiv$ *signifying an equivalence relation between entities* e_s *and* e_t*.*

The Dirichlet energy is widely used as a regularization loss for functions defined on the embeddings of the graph (Rossi et al., 2022; Maskey

[1] The code and processed dataset are available at https://anonymous.4open.science/r/TFP-4EEF

et al., 2024).To simplify the derivation, we consider the simple undirected graph at first.

Definition 3. ***(Laplacian matrix.)*** *The adjacency matrix is defined as* $\mathbf{A}$*, the symmetrically normalized adjacency is represented by* $\widetilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ *, and the symmetrically normalized Laplacian matrix as* $\mathbf{\Delta} = \mathbf{I} - \widetilde{\mathbf{A}}$.

Definition 4. ***(Dirichlet Energy (DE).)*** *Given the graph node embedding* $\mathbf{X} \in \mathbb{R}^{N\times d}$*, the Dirichlet energy of* $\mathbf{X}$ *is defined as:*

$$\begin{aligned}\mathscr{L}(\mathbf{X}) &= tr(\mathbf{X}^\top \mathbf{\Delta}\mathbf{X}) \\ &= \frac{1}{2}\sum_{i,j=1}^{N} a_{i,j}||\frac{\mathbf{X}_i}{\sqrt{\mathbf{D}_{i,i}+1}} - \frac{\mathbf{X}_j}{\sqrt{\mathbf{D}_{j,j}+1}}||_2^2\end{aligned} \tag{1}$$

4 Theoretical Motivation

Our Triple Feature Propagation (TFP) seeks to maximize smoothness by minimizing the Dirichlet energy (DE). This process primarily utilizes the symmetric Laplacian matrix $\mathbf{\Delta}$, constructed from the undirected adjacency matrix to capture the entity-to-entity relationships. We initiate our study with gradient flows within this matrix. Before decoding, the encoder is trained to produce initial entity embeddings $\mathbf{X}^{(0)}$.

$$\mathbf{X}^{(0)} = Encoder(\mathcal{G}) \in \mathbb{R}^{|\mathcal{E}|\times d} \tag{2}$$

4.1 Existence of the Solution

Given seed alignment entity features x_s, our objective is to reconstruct the features x_o of other entities for maximizing smoothness by minimizing the Dirichlet energy $\mathscr{L}(\mathbf{X})$. We designate the set of seed alignment entities as $\mathcal{E}_s \subseteq \mathcal{E}$, with the remaining entities denoted by $\mathcal{E}_o = \mathcal{E}\backslash\mathcal{E}_s$. The entities are ordered such that:

$$\mathbf{X} = \begin{pmatrix}\mathbf{x}_s \\ \mathbf{x}_o\end{pmatrix}, \quad \mathbf{\Delta} = \begin{pmatrix}\mathbf{\Delta}_{ss} & \mathbf{\Delta}_{so} \\ \mathbf{\Delta}_{os} & \mathbf{\Delta}_{oo}\end{pmatrix} \tag{3}$$

We consider the associated gradient flow as a differential equation with boundary condition $\mathbf{x_s}(t) = \mathbf{x_s}$, which is expressed as:

$$\frac{d\mathbf{X}(t)}{dt} = -\nabla_{\mathbf{x}}\mathscr{L}(\mathbf{X}(t)) = -\mathbf{\Delta}\mathbf{X}(t) \tag{4}$$

As the features of seed alignments are regarded as stationary (Wang et al., 2024), we have the initial and boundary conditions:

$$\mathbf{x}(0) = \begin{pmatrix}\mathbf{x}_s \\ \mathbf{x}_o(0)\end{pmatrix}, \quad \mathbf{x_s}(t) = \mathbf{x_s} \tag{5}$$

By integrating the above conditions, the solution to this propagation equation provides the required decoding process.

$$\begin{aligned}\frac{d}{dt}\begin{pmatrix}\mathbf{x}_s(t) \\ \mathbf{x}_o(t)\end{pmatrix} &= -\begin{pmatrix}\mathbf{0} & \mathbf{0} \\ \mathbf{\Delta}_{os} & \mathbf{\Delta}_{oo}\end{pmatrix}\begin{pmatrix}\mathbf{x}_s \\ \mathbf{x}_o(t)\end{pmatrix} \\ &= \begin{pmatrix}\mathbf{0} \\ -\mathbf{\Delta}_{os}\mathbf{x}_s - \mathbf{\Delta}_{oo}\mathbf{x}_o(t)\end{pmatrix}\end{aligned} \tag{6}$$

Given the positive semi-definite nature of the graph Laplacian matrix, Dirichlet energy $\mathscr{L}$ is convex, and its global minimizer is the solution to the gradient equation $\nabla\mathscr{L}(\mathbf{X}(t)) = 0$. The solution in equation (6) can be expressed as:

$$\frac{d\mathbf{x}_o(t)}{dt} = -\mathbf{\Delta}_{os}\mathbf{x}_s - \mathbf{\Delta}_{oo}\mathbf{x}_o(t) = \mathbf{0} \tag{7}$$

Therefore, we present the following proposition:

Proposition 1. ***(Existence of the solution.)*** *The matrix* Δ_{oo} *is non-singular, allowing the reconstruction of other entity features* $\mathbf{x}_o$ *using seed alignment entity features* $\mathbf{x}_s$ *as* $\mathbf{x}_o(t) = -\mathbf{\Delta}_{oo}^{-1}\mathbf{\Delta}_{os}\mathbf{x}_s$.

Proof. Please refer to Appendix A. □

Remark. This special gradient flow in TFP is generalized to directed graphs, distinguishing it from FP (Rossi et al., 2022) on undirected graphs.

4.2 Explicit Euler Solution

Proposition 1 reveals the existence of the solution to maximize the smoothness. However, solving this system of linear equations is computationally intensive, with a complexity of $O(|\mathcal{E}_o|^3)$ for matrix inversion, rendering it impractical for large graphs. To tackle the computational challenges, we discretize the equation (6) and adopt an iterative approach for its resolution. By approximating the temporal derivative as a forward difference and discretizing time t with fixed steps ($t = hk$ for step size $h > 0$ and $k = 1, 2, \dots$), we employ the implicit Euler scheme:

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + h\frac{d\mathbf{X}(t)}{dt}\mathbf{X}^{(k)} \tag{8}$$

Incorporating the initial and boundary conditions (5), the explicit Euler scheme is defined as:

$$\mathbf{X}^{(k+1)} = \begin{pmatrix}\mathbf{I} & \mathbf{0} \\ -h\mathbf{\Delta}_{os} & \mathbf{I} - h\mathbf{\Delta}_{oo}\end{pmatrix}\mathbf{X}^{(k)} \tag{9}$$

When focusing on the special case of $h = 1$, we can use the following observation to rewrite the iteration formula to equation (11).

$$\widetilde{\mathbf{A}} = \mathbf{I} - \mathbf{\Delta} = \begin{pmatrix} \mathbf{I} - \mathbf{\Delta}_{ss} & -\mathbf{\Delta}_{so} \\ -\mathbf{\Delta}_{os} & \mathbf{I} - \mathbf{\Delta}_{oo} \end{pmatrix} \tag{10}$$

$$\mathbf{X}^{(k+1)} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \widetilde{\mathbf{A}}_{os} & \widetilde{\mathbf{A}}_{oo} \end{pmatrix} \mathbf{X}^{(k)} \tag{11}$$

This Euler scheme serves as a gradient descent for the Dirichlet energy. The approximation of this solution in this case is formalized as follows:

Proposition 2. ***(Approximation of the solution.)*** *With the reconstruction solution as delineated in equation (11), and considering a sufficiently large iteration count N, the entity features will approximate the results of feature propagation as follows:*

$$\mathbf{X}^{(N)} \approx \begin{pmatrix} \mathbf{x}_s \\ \mathbf{\Delta}_{oo}^{-1} \widetilde{\mathbf{A}}_{os} \mathbf{x}_s \end{pmatrix} \tag{12}$$

Proof. Please refer to Appendix B. □

The update process in Equation (11) equates to initially multiplying the entity features $\mathbf{X}$ by the matrix $\widetilde{\mathbf{A}}$, followed by resetting the seed alignment features to their original values.

$$\mathbf{X}^{(k+1)} \leftarrow \widetilde{\mathbf{A}}\mathbf{X}^{(k)}; \quad \mathbf{x}_s^{(k+1)} \leftarrow \mathbf{x}_s^{(k)} \tag{13}$$

Equation (13) defines an iterative strategy to reconstruct entity features from aligned entities, forming the basic Triple Feature Propagation. Since $\widetilde{\mathbf{A}}$ only captures entity-to-entity relationships, it is termed "basic." In the following section, we generalize the adjacency matrix to multi-view matrices to capture comprehensive structural information, leading to the complete TFP.

5 Method

In this section, we present the complete TFP. First, we generalize the traditional adjacency matrix to multi-view matrices to capture the comprehensive KG structure. These multi-view matrices are then applied to Equation 13 to derive the complete TFP. Fig.2 shows the framework and Algorithm **??** illustrates the details.

5.1 Generalized Multi-view Matrices

Equation (13) establishes the basic Triple Feature Propagation. However, KGs extend beyond the realm of simple directed graphs represented by an adjacency matrix $\mathbf{A}$. KGs encompass a richer structure, with triples $\mathcal{T} = \{(h, r, t), |h, t \in \mathcal{E}, r \in \mathcal{R}\}$, highlighting entity-to-entity $(h - t)$, entity-to-relation $(h - r)$, relation-to-entity $(r - t)$, and relation-to-triple $(r - \mathcal{T})$. These perspectives encapsulate key meta-structures in KGs:

Algorithm 1: TFP

Input: The entity embedding $\mathbf{X}^{(0)}$, the triples $\mathcal{T}$, iteration number K, dimension d_r, d_e.

Output: The reconstructed entity feature $\mathbf{X}_{out}$.

Initialize $\widetilde{\mathbf{A}}^{proximal}$, $\widetilde{\mathbf{A}}^{distal}$, $\widetilde{\mathbf{A}}^{integral}$, and $\mathbf{A}^{tri-rel}$ through the triples $\mathcal{T}$.;
for $k = 1 \rightarrow K$ **do**
  $\mathbf{X}_r^{(k)} \leftarrow \widetilde{\mathbf{A}}^{distal}\mathbf{X}_e^{(k)}$;
  $\mathbf{X}_e^{(k+1)} \leftarrow \widetilde{\mathbf{A}}^{integral}\mathbf{X}_e^{(k)} + \widetilde{\mathbf{A}}^{proximal}\mathbf{X}_r^{(k)}$;
  $\mathbf{x}_s^{(k+1)} \leftarrow \mathbf{x}_s^{(k)}$;
end
$\mathbf{X}_r \leftarrow random_projection(\mathbf{X}_r^{(K)}, d_r)$;
$\mathbf{X}_t \leftarrow \mathbf{A}^{tri-rel}\mathbf{X}_r$;
Generate $\mathbf{A}_i^{triple}$ through $\mathbf{X}_t$.;
$\mathbf{X}_e \leftarrow random_projection(\mathbf{X}_e^{(K)}, d_e)$;
$\mathbf{X}_i \leftarrow \mathbf{A}_i^{triple}\mathbf{X}_e$;
$\mathbf{X}_{out} \leftarrow [\mathbf{X}_1||\mathbf{X}_2||\ldots||\mathbf{X}_{d_r}]$;
return $\mathbf{X}_{out}$

(i) **Entity-to-Relation:** Beyond the conventional adjacency matrix, the (h, r) pairs in triples provide a unique structural perspective, indicating directional links from head entity h to relation r. This directionality is fundamental and unidirectional. For instance, in the triple $(London, CapitalOf, England)$, the entity $[England]$ is connected to $[London]$ via the relation $[CapitalOf]$, but not in reverse. Similar to the adjacency matrix $\mathbf{A}$, we define an *entity to relation* matrix $\mathbf{A}^{proximal} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{R}|}$ based on the (h, r) pairs as:

$$\exists e \in \mathcal{E}: \quad \begin{aligned} \mathbf{A}_{i,j}^{proximal} &= 1, \ (h_i, r_j, e) \in \mathcal{T} \\ \mathbf{A}_{i,j}^{proximal} &= 0, \ (h_i, r_j, e) \notin \mathcal{T} \end{aligned} \tag{14}$$

(ii) **Relation-to-Entity.** This category, akin to *entity to relation*, encompasses directed and irreversible information. It underscores the more distant positional relationships between relations and tail entities. The resulting structural matrix,

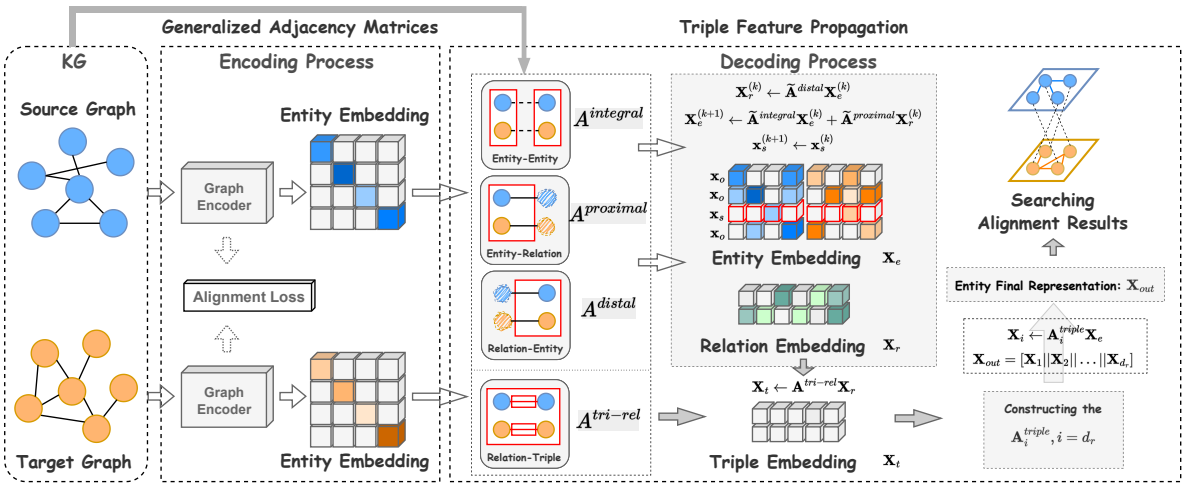

Figure 2: The illustration of Triple Feature Propagation.

$\mathbf{A}^{distal} \in \mathbb{R}^{|\mathcal{R}|\times|\mathcal{E}|}$, is defined as:

$$\exists e \in \mathcal{E}: \quad \begin{aligned} \mathbf{A}_{i,j}^{distal} &= 1,\ (e, r_i, t_j) \in \mathcal{T} \\ \mathbf{A}_{i,j}^{distal} &= 0,\ (e, r_i, t_j) \notin \mathcal{T} \end{aligned} \tag{15}$$

(iii) **Entity-to-Entity.** Representing the fundamental structural category, this is similar to the adjacency matrix $\mathbf{A}$. While $\mathbf{A}$ provides a solid foundation and is rich in spectral properties, it is insufficient to fully represent KG structures. For example, the triples $(London, CapitalOf, England)$ are structurally distinct with $(London, LocateIn, England)$ in KGs, yet a traditional adjacency matrix treats them identically as they both represent $London$ connects $English$, representing $\mathbf{A}_{i,j} = 1$. To better represent relationships between entities, we extend the adjacency matrix to $\mathbf{A}^{integral} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$:

$$\exists r \in \mathcal{R}: \quad \begin{aligned} \mathbf{A}_{i,i}^{integral} &= |\mathcal{T}_{e_i}| \\ \mathbf{A}_{i,j}^{integral} &= |\mathcal{T}_{(h_i,t_j)}|, (h_i, r, t_j) \in \mathcal{T} \\ \mathbf{A}_{i,j}^{integral} &= 0, (h_i, r, t_j) \notin \mathcal{T} \end{aligned} \tag{16}$$

$|\mathcal{T}_{e_i}|$ denotes the number of triples involving entity e_i, and $|\mathcal{T}_{(h_i,t_j)}|$ indicates the count of triples with the pair (h_i, t_j).

(iv) **Relation-to-Triple.** As discussed before, the knowledge in KG is stored in the triple $\mathcal{T} = \{(h, r, t), |h, t \in \mathcal{E}, r \in \mathcal{R}\}$, which is the core representation of the KG structure. The unique role of relation r within triple $\mathcal{T}$ motivates the extension of the adjacency matrix to the triple level through relation r. We introduce the adjacency matrix $\mathbf{A}^{tri-rel} \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{R}|}$, defined as:

$$\begin{aligned} \mathbf{A}_{i,j}^{tri-rel} &= 1,\ r_j \in \mathcal{T}_i \\ \mathbf{A}_{i,j}^{tri-rel} &= 0,\ r_j \notin \mathcal{T}_i \end{aligned} \tag{17}$$

(v) **Entity-to-Triple** matrix is also considered. This matrix is considered to represent *entity-entity-triple* relationships for head (h) and tail (t) entities connections in $\mathcal{T}$. In fact, these connections are already encapsulated within the *entity-to-entity* matrix $\mathbf{A}^{integral}$. This matrix, which details connections between head h and tail t entities, effectively doubles as an *entity-to-triple* representation.

We assume that the normalized form of $\mathbf{A}^{proximal}$, $\mathbf{A}^{distal}$, and $\mathbf{A}^{integral}$, which are similar to graph Laplacian, representing different facets of KG structure, can be regarded as $\widetilde{\mathbf{A}}$ in equation (13). The experimental results confirm our view. Consequently, the gradient flow should yield comparable solutions across these matrices. We denote their normalized form $\widetilde{\mathbf{A}}^{k,k=\{proximal,distal,integral\}}$

$$\widetilde{\mathbf{A}}_i^k = \frac{\mathbf{A}_i^k}{\sum_{j=0}^{numberofcolumn} \mathbf{A}_{i,j}^k} \tag{18}$$

5.2 Triple Feature Propagation

Utilizing the gradient flow of Dirichlet energy on these categories of KG structure, we derive a natural and straightforward generalized propagation strategy. The propagation process is articulated through the following equations:

$$\mathbf{X}_r^{(k)} = \widetilde{\mathbf{A}}^{distal}\mathbf{X}_e^{(k)} \tag{19}$$

$$\mathbf{X}_e^{(k+1)} = \widetilde{\mathbf{A}}^{integral}\mathbf{X}_e^{(k)} + \widetilde{\mathbf{A}}^{proximal}\mathbf{X}_r^{(k)} \tag{20}$$

$$\mathbf{x}_s^{(k+1)} = \mathbf{x}_s^{(k)} \tag{21}$$

Here, $\mathbf{X}_e^{(i)}, 1 \leq i \leq k+1$, denotes the entity features, initially derived from the graph encoder as $\mathbf{X}_e^{(0)}$. $\mathbf{X}_r$ represents the implicit relation features generated through the propagation. To comprehensively capture the evolving entity and relation information across iterations, we aggregate entity features from each step through concatenation, resulting in the entity feature as follows:

$$\mathbf{X}_e = [\mathbf{X}^{(0)}||\mathbf{X}^{(1)}||\ldots||\mathbf{X}^{(k)}] \tag{22}$$

We notice that our process explicitly captures the implicit relation feature $\mathbf{X}_r$, serving as an essential intermediary. By leveraging the *triple-to-relation* matrix, we further refine $\mathbf{X}_r$. We use a hyper-sphere independent random projection to reduce the dimension of $\mathbf{X}_r$ to d_r, following the method outlined in (Mao et al., 2022b). This step enables the creation of the triple feature $\mathbf{X}_t$.

$$\mathbf{X}_r = ranp(\mathbf{X}_r) \in \mathbb{R}^{|\mathcal{R}| \times d_r} \tag{23}$$

$$\mathbf{X}_t = \mathbf{A}^{tri-rel}\mathbf{X}_r \tag{24}$$

Our method enables the representation of the KG structure as a three-dimensional tensor $\mathbf{A}^{triple} \in \mathbb{R}^{|\mathcal{E}| \times |\mathcal{E}| \times d_r}$, rather than the traditional adjacency matrix. The tensor is defined as:

$$\begin{aligned} \mathbf{A}_{h,t}^{triple} &= \mathbf{X}_t(h,r,t), (h,t) \in \mathcal{T} \\ \mathbf{A}_{h,t}^{triple} &= \mathbf{0}, (h,t) \notin \mathcal{T} \end{aligned} \tag{25}$$

Here, $\mathbf{X}_t(h,r,t)$ represents the feature of triple (h,r,t) in $\mathbf{X}_t$. The KG structure is encapsulated in d_r slices as $\mathbf{A}_1^{triple}, \ldots, \mathbf{A}_{d_r}^{triple} \in \mathbb{R}^{|\mathcal{E}| \times |\mathcal{E}|}$. Parallel to the relation feature, the entity feature $\mathbf{X}_e$ is also scaled through hyper-sphere independent random projection to d_e dimension. The propagation process is based on the final entity features $\mathbf{X}_e$, allowing for the continuation of gradient flow from the triple perspective. Since each $\mathbf{A}_i^{triple}$ represents a segmented view of the overall $\mathbf{A}^{triple}$, their concatenation is essential to compile the comprehensive final feature.

$$\begin{aligned} \mathbf{X}_e &= ranp(\mathbf{X}_e) \in \mathbb{R}^{|\mathcal{R}| \times d_e}, \\ \mathbf{X}_i &= \mathbf{A}_i^{triple}\mathbf{X}_e, i = 1, \ldots, d_r \end{aligned} \tag{26}$$

$$\mathbf{X}_{out} = [\mathbf{X}_1||\mathbf{X}_2||\ldots||\mathbf{X}_{d_r}] \tag{27}$$

The $\mathbf{X}_{out}$ is the reconstructed entity embedding, which is the TFP output and is used to search alignment results.

Remark: To search alignment results, we adopt an approach from (Mao et al., 2022a,b) that frames the search as an assignment problem. Detailed procedures are described in Appendix C.

6 Experiments

All experiments are conducted on a PC with an NVIDIA RTX A6000 GPU and an Intel Xeon Gold 6248R CPU.

6.1 Experimental Settings

Datasets: We utilize two widely recognized datasets to test our decoding approach: **(1)** DBP15K (Sun et al., 2017) comprises three cross-lingual subsets from multilingual DBpedia. Each subset contains 15,000 entity pairs. **(2)** SRPRS (Guo et al., 2019). Similar to DBP15K in terms of the number of entity pairs but with fewer triples. Following (Wang et al., 2018; Chen et al., 2017), we use a 30/70 split of pre-aligned entity pairs for training and testing encoders. The dataset statistics are reported in Appendix D.

Baseline: (1) Encoder: Our evaluation includes prominent encoders, divided into GNN-based: MRAEA (Mao et al., 2020a), RREA (Mao et al., 2020b), Dual-AMN (Mao et al., 2021a) (with Dual-AMN as the structural SOTA), and translation-based: AlignE (Sun et al., 2018), RSN (Guo et al., 2019), and TransEdge (Sun et al., 2019) (with TransEdge as the premier in its class). **(2) Decoder:** For decoding baselines, we benchmark against the Hungarian algorithm (Kuhn, 1955), which is the prominent solution in previous work (Xu et al., 2020), and the SOTA of EA decoder, DATTI (Mao et al., 2022a). The LightEA (Mao et al., 2022b) and plain Sinkhorn (Cuturi, 2013) are also applied as decoders.

Remark: The hyperparameter and implementation details are provided in Appendix E.

6.2 Main Results

The main experimental results and time costs are summarized in Tables 1 and 2. TFP consistently demonstrates fast execution, adaptability to various encoders, and performance comparable to state-of-the-art methods with minimal time consumption. It strikes the elegant balance between accuracy and efficiency.

GNN-based Encoders: As shown in Tables 1 and 2, TFP consistently improves the accuracy of RREA and DualAMN across datasets, with a time

Table 1: Main experimental results on DBP15K and SRPRS datasets. All results and initial embeddings were derived using official code with default hyperparameters. The Hungarian algorithm (Hun) produces only one aligned pair per entity, thus only Hits@1 is reported. DATTI cannot decode RREA, as both entity and relation embeddings are required, whereas RREA provides only entity embeddings.

Datasets		DBP_{FR-EN}			DBP_{JA-EN}			DBP_{ZH-EN}			$SRPRS_{FR-EN}$			$SRPRS_{DE-EN}$		
	Model	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
GNN-based	RREA	73.40	94.80	81.36	70.59	94.13	79.11	70.73	93.21	78.97	42.96	73.96	53.41	56.74	81.24	65.18
	+Hun	80.42	-	-	78.44	-	-	78.69	-	-	45.68	-	-	58.96	-	-
	+ Sinkhorn	80.77	96.45	86.76	78.72	95.70	84.90	78.98	94.98	84.84	46.81	75.06	56.21	60.05	81.97	67.70
	+LightEA	**84.53**	**97.71**	**89.45**	**80.74**	**96.59**	**86.20**	**80.90**	95.20	**85.92**	**49.18**	**76.53**	**58.32**	**60.89**	**82.03**	**68.32**
	+TFP	80.79	96.98	86.79	78.83	96.00	85.23	79.43	**95.45**	85.35	47.23	75.60	56.67	59.98	82.33	67.80
	DualAMN	83.43	96.19	88.14	80.31	94.69	85.57	80.39	93.68	85.29	48.28	75.51	57.34	61.20	81.91	68.30
	+Hun	83.87	-	-	80.39	-	-	80.12	-	-	48.32	-	-	61.15	-	-
	+ Sinkhorn	84.02	95.86	88.58	80.32	93.69	85.22	80.06	92.88	84.77	48.19	73.40	56.41	61.24	80.36	67.81
	+DATTI	**87.30**	**97.90**	**91.30**	**83.60**	**96.90**	**88.40**	**83.50**	**95.30**	**88.00**	**49.50**	76.00	**58.30**	**62.30**	82.20	**69.10**
	+LightEA	84.45	95.35	89.10	82.47	93.91	86.81	81.86	92.74	86.67	48.61	76.50	57.93	61.38	**83.32**	68.86
	+TFP	85.31	97.26	89.92	81.26	95.85	86.82	81.64	95.24	86.84	48.90	**76.28**	58.02	61.84	82.61	69.01
Translation-based	RSN	63.17	86.37	71.33	59.13	81.50	67.03	60.67	82.86	68.53	35.10	63.78	44.73	51.07	74.43	59.02
	+Hun	69.25	-	-	63.33	-	-	66.08	-	-	37.42	-	-	53.79	-	-
	+ Sinkhorn	69.23	88.47	76.03	63.73	83.59	70.73	65.98	84.76	72.67	37.50	64.91	46.58	53.70	75.07	61.18
	+DATTI	72.00	91.80	79.00	68.60	89.50	75.90	72.10	90.30	78.50	40.70	69.40	50.20	55.90	78.20	63.70
	+LightEA	74.50	94.80	81.99	71.00	**93.20**	78.89	71.82	**91.96**	79.08	43.61	**74.30**	53.89	58.00	**82.06**	66.37
	+TFP	**77.79**	**92.45**	**83.73**	**73.53**	91.97	**80.04**	**75.44**	91.71	**81.33**	**44.32**	73.22	**53.97**	**59.27**	81.14	**66.84**
	TransEdge	76.90	93.97	83.03	74.66	92.93	81.10	76.19	92.16	81.81	40.81	67.66	49.73	55.65	75.30	64.28
	+Hun	79.55	-	-	77.06	-	-	78.72	-	-	42.65	-	-	57.41	-	-
	+ Sinkhorn	79.32	95.37	85.17	76.94	94.18	83.20	78.50	93.42	83.87	43.78	68.31	51.66	57.30	75.67	65.15
	+DATTI	81.80	96.50	87.30	80.40	95.70	86.10	81.40	94.70	86.30	44.10	70.70	52.10	59.30	78.20	67.30
	+LightEA	80.68	96.56	87.30	78.20	95.17	85.16	78.98	94.61	85.06	**49.37**	75.10	56.30	**64.80**	**82.10**	**71.24**
	+TFP	**82.36**	**96.90**	**87.91**	**80.52**	**95.96**	**86.19**	**81.53**	**95.21**	**86.50**	49.03	**75.22**	**56.49**	64.12	81.70	70.95

Table 2: Main results of execution time (s). TFP(C) and TFP(G) represent CPU and GPU execution, respectively. "Encoding" refers to the time spent on encoding, with all models run on the GPU except for TFP(C).

Dataset	Encoder	Time	Hun	DATTI	LightEA	**TFP(C)**	**TFP(G)**
DBP15K	RSN	3659	13.4	8.6	12.8	14.2	**4.8**
	TransEdge	1625	8.0	6.1	12.5	12.9	**4.7**
	RREA	323	12.3	6.0	13.1	16.3	**5.7**
	DualAMN	177	12.2	6.6	13.4	17.7	**5.1**
SRPRS	RSN	1279	12.7	5.4	12.3	9.2	**3.7**
	TransEdge	1625	12.5	4.7	12.1	12.9	**4.7**
	RREA	276	13.1	4.3	11.2	11.0	**3.8**
	DualAMN	163	13.3	**4.1**	11.3	11.4	4.6

cost under 6 seconds. While LightEA decodes RREA effectively and DATTI excels at decoding Dual-AMN (the state-of-the-art GNN-based structural EA decoder), DATTI's reliance on both entity and relation embeddings limits its suitability for RREA and LightEA is too time-consuming. In contrast, TFP delivers comparable results to LightEA and DATTI while being the fastest for both RREA and DualAMN.

Translation-based Encoders: TFP significantly enhances the performance of translation-based EA encoders by capturing multi-view structural information and maximizing smoothness. With less than 5 seconds of execution time, TFP achieves superior decoding performance with minimal computational cost, highlighting its ability to efficiently capture KG topology.

Time Complexity: TFP employs iterative sparse-to-dense matrix multiplications. Following (Mao et al., 2022b), we optimize these multiplications in Eqs. (19) and (20) to maintain computational complexity at $O(k(|\mathcal{T}|d_r + |\mathcal{E}|d_e))$. For Eqs. (24) and (26), Tensorflow's sparse matrix multiplications reduce complexity to $O(|\mathcal{T}|d_r)$ and $O(|\mathcal{E}|d_e)$.

Time Cost: Table 6 provides a detailed comparison of time costs for encoding and decoding phases on DBP15K and SRPRS datasets. Notably, TFP's cost is minimal in the overall runtime, with most time spent on encoder training. On a GPU, TFP runs in as little as 5.9 seconds, and even on a CPU, it peaks at just 17.7 seconds, rivaling the fastest decoders like the GPU-based Hungarian algorithm. TFP achieves the lowest time cost compared to other decoders across most cases.

Besides these results, we further evaluate TFP on other encoders. Due to the space limitation, these experiments are detailed in Appendix F.

6.3 Hyper-parameter Analysis

We conduct extensive experiments to examine the sensitivity of TFP's hyper-parameters during entity embedding decoding. Due to space limitation,

Table 3: Performances of unsupervised textual decoding.

Datasets	DBP_{FR-EN}			DBP_{JA-EN}			DBP_{ZH-EN}			$SRPRS_{FR-EN}$			$SRPRS_{DE-EN}$		
Models	H@1	MRR	Time/s	H@1	MRR	Time/s	H@1	MRR	Time/s	H@1	MRR	Time/s	H@1	MRR	Time/s
GM-Align (Xu et al., 2019)	89.4	-	26328	73.9	-	26328	67.9	-	26328	57.4	60.2	13032	68.1	71.0	13032
RDGCN (Wu et al., 2019a)	87.3	90.1	6711	76.3	76.3	6711	69.7	75.0	6711	67.2	71.0	886	77.9	82.0	886
HGCN (Wu et al., 2019b)	76.6	81.0	11275	89.2	91.0	11275	72.0	76.0	11275	67.0	71.0	2504	76.3	80.1	2504
SEU (Mao et al., 2021b)	98.8	99.2	17.0	95.6	96.9	16.2	90.0	92.4	16.2	98.2	98.6	9.6	98.3	98.7	9.8
LightEA (Mao et al., 2022b)	**99.5**	99.6	15.7	**98.1**	**98.7**	14.8	**95.2**	**96.4**	15.2	98.6	98.9	11.2	98.8	99.1	11.4
TFP(SEU)	99.05	99.4	4.9	96.21	97.19	4.7	90.51	92.63	5.1	98.55	98.95	3.7	98.57	98.95	3.8
TFP(LightEA)	99.21	99.65	5.1	97.94	**98.7**	4.9	94.34	95.62	5.4	98.6	99.03	3.9	98.6	99	3.8
TFP	99.33	**99.68**	**4.8**	96.68	97.54	**4.7**	91.54	93.59	**4.9**	**98.99**	**99.28**	**3.6**	**98.87**	**99.15**	**3.6**

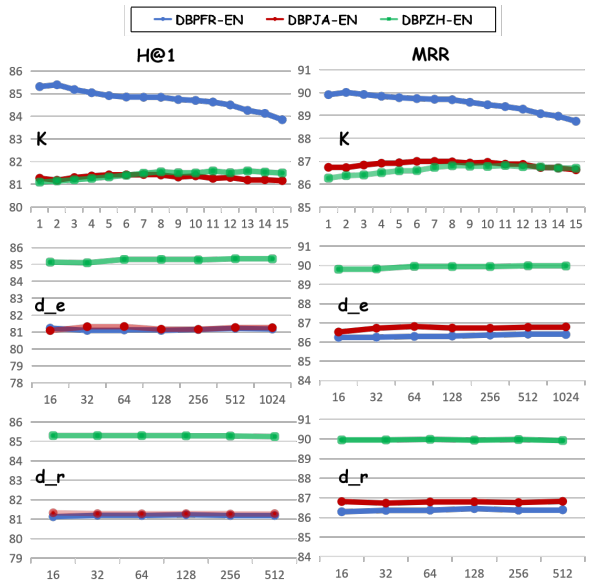

Figure 3: Hyper-parameter experiments of TFP decoding the DualAMN on DBP15K.

Figure 3 only presents the results for TFP decoding DualAMN on DBP15K.

Propagation Round. Figure 3 illustrates the performance across different propagation rounds K. Similar to increasing network depth, more rounds can cause over-smoothing. TFP achieves optimal performance at $K = 2$ and declines thereafter. This suggests that while multi-view matrices effectively capture KG structure, they have distinct spectral properties compared to the adjacency matrix. The basic TFP with only the adjacency matrix can still ensure convergence, as demonstrated in Proposition 2.

Entity Dimension. To accelerate computation, we apply hyper-sphere independent random projection to reduce entity dimensions to d_e. As shown in Figure 3, TFP remains robust to changes in these dimensions, with performance barely affected by significant dimensional reductions.

Relation Dimension. Similar to entity dimensions, random projection is used to reduce relation dimensions. However, d_r should not be too large since, as indicated by Eq. (24), the tensor-matrix product leads to a squared increase in dimension after each propagation round.

Additional hyper-parameter results are detailed in Appendix G due to space limitations.

6.4 Unsupervised Textual Decoding

Most EA research focuses on structural methods, but some (Xu et al., 2019; Wu et al., 2019b) use textual information, such as entity names, for improvement via cross-lingual embeddings or translation systems. These methods treat pre-trained word embeddings as initial embeddings for entities, which can be viewed as the decoding process that captures the topology of KGs. Similarly, TFP can serve this role, decoding initial entity features like word embeddings without seed alignments.

We used pre-trained GLoVe embeddings (Pennington et al., 2014) as initial features for TFP, treating it as an unsupervised textual decoder. Table 3 shows TFP outperforming baselines on SRPRS with minimal time cost and ranking near the top on DBP15K. The competitive results of TFP suggest propagation may be more effective than neural networks, questioning the need for complex models and seed alignments. However, TFP(SEU) and TFP(LightEA), which use SEU and LightEA as encoders, slightly underperform compared to TFP, likely due to over-smoothing. Despite this, TFP achieves top performance on SRPRS, while the extra required time is less than 4 seconds.

7 Conclusion

This paper introduces a fast and adaptable entity alignment decoding algorithm that maximizes the smoothness of entity embeddings, supported by a solid theoretical foundation. Extensive experiments on public datasets show that our method significantly enhances performance on most EA models, with running time of under six seconds.

Limitations

While TFP has shown speed, adaptability, and robustness across various public datasets, several limitations remain: (1) Although TFP achieves competitive results with state-of-the-art methods with significantly reduced runtime, some gaps in accuracy still exist on certain datasets. Future work will focus on improving the balance between performance and efficiency. (2) TFP theoretically offers high parallel efficiency with potential for linear speedup on multiple GPUs. However, due to limited hardware resources, all experiments were conducted on a single A6000 GPU. We plan to address this by running additional experiments on multiple GPUs in the future. (3) TFP is currently available in both Tensorflow and PyTorch versions. Given that it is a non-neural algorithm without trainable parameters, deep learning frameworks can be inefficient. Future work includes refactoring TFP using CUDA and C++ to enhance efficiency further.

Ethics Statement

This work does not involve discrimination, social bias, or private data. All datasets used are open-source, derived from publicly available KGs such as Wikidata, YAGO, and DBpedia, ensuring compliance with the ACL Ethics Policy. Additionally, while the introduction of literal features may raise concerns about misuse of user-generated content, TFP mitigates this risk by relying solely on structural information.

References

Max Berrendorf, Evgeniy Faerman, and Volker Tresp. 2021. Active learning for entity alignment. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part I 43*, pages 48–62. Springer.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Soumen Chakrabarti. 2022. Deep knowledge graph representation learning for completion, alignment, and question answering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3451–3454.

Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2017. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1511–1517.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26.

Lingbing Guo, Zequn Sun, and Wei Hu. 2019. Learning to exploit long-term relational dependencies in knowledge graphs. In *International conference on machine learning*, pages 2505–2514. PMLR.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

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

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *International Conference on Learning Representations*.

Jia Li and Dandan Song. 2022. Uncertainty-aware pseudo label refinery for entity alignment. In *Proceedings of the ACM Web Conference 2022*, pages 829–837.

Bing Liu, Harrisen Scells, Guido Zuccon, Wen Hua, and Genghong Zhao. 2021. Activeea: Active learning for neural entity alignment. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3364–3374.

Xiaoze Liu, Junyang Wu, Tianyi Li, Lu Chen, and Yunjun Gao. 2023. Unsupervised entity alignment for temporal knowledge graphs. In *Proceedings of the ACM Web Conference 2023*, pages 2528–2538.

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 *Proceedings of the Web Conference 2021*, pages 821–832.

Xin Mao, Wenting Wang, Huimin Xu, Man Lan, and Yuanbin Wu. 2020a. Mraea: an efficient and robust entity alignment approach for cross-lingual knowledge graph. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 420–428.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020b. Relational reflection entity alignment. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1095–1104.

Xinnian Mao, Meirong Ma, Hao Yuan, Jianchao Zhu, Zongyu Wang, Rui Xie, Wei Wu, and Man Lan. 2022a. An effective and efficient entity alignment decoding algorithm via third-order tensor isomorphism. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5888–5898.

Xinnian Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2021b. From alignment to assignment: Frustratingly simple unsupervised entity alignment. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2853.

Xinnian Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2022b. Lightea: A scalable, robust, and interpretable entity alignment framework via three-view label propagation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 825–838.

Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. 2023. A fractional graph laplacian approach to oversmoothing. *arXiv preprint arXiv:2305.13084*.

Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. 2024. A fractional graph laplacian approach to oversmoothing. *Advances in Neural Information Processing Systems*, 36.

Shichao Pei, Lu Yu, Robert Hoehndorf, and Xiangliang Zhang. 2019. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *The world wide web conference*, pages 3130–3136.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Emanuele Rossi, Henry Kenlay, Maria I Gorinova, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. 2022. On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features. In *Learning on Graphs Conference*, pages 11–1. PMLR.

Alvin E Roth. 2008. Deferred acceptance algorithms: History, theory, practice, and open questions. *international Journal of game Theory*, 36:537–569.

Xiaofei Shi and Yanghua Xiao. 2019. Modeling multi-mapping relations for precise cross-lingual entity alignment. 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)*, pages 813–822.

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 16*, pages 628–644. Springer.

Zequn Sun, Wei Hu, Chengming Wang, Yuxin Wang, and Yuzhong Qu. 2022. Revisiting embedding-based entity alignment: a robust and adaptive method. *IEEE Transactions on Knowledge and Data Engineering*.

Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *IJCAI*, volume 18.

Zequn Sun, Jiacheng Huang, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019. Transedge: Translating relation-contextualized embeddings for knowledge graphs. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part I 18*, pages 612–629. Springer.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 222–229.

Bayu Distiawan Trisedya, Jianzhong Qi, and Rui Zhang. 2019. Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 297–304.

Jihu Wang, Yuliang Shi, Han Yu, Xinjun Wang, Zhongmin Yan, and Fanyu Kong. 2023. Mixed-curvature manifolds interaction learning for knowledge graph-aware recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 372–382.

Yuanyi Wang, Haifeng Sun, Jiabo Wang, Jingyu Wang, Wei Tang, Qi Qi, Shaoling Sun, and Jianxin Liao. 2024. Towards semantic consistency: Dirichlet energy driven robust multi-modal entity alignment. *arXiv preprint arXiv:2401.17859*.

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*, pages 349–357.

Y Wu, X Liu, Y Feng, Z Wang, R Yan, and D Zhao. 2019a. Relation-aware entity alignment for heterogeneous knowledge graphs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence.

Y Wu, X Liu, Y Feng, Z Wang, and D Zhao. 2019b. Jointly learning entity and relation representations for entity alignment. 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), pages 240–249. Association for Computational Linguistics.

Chengjin Xu, Fenglong Su, and Jens Lehmann. 2021. Time-aware graph neural network for entity alignment between temporal knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8999–9010.

Chengjin Xu, Fenglong Su, Bo Xiong, and Jens Lehmann. 2022. Time-aware entity alignment using temporal relational attention. In *Proceedings of the ACM Web Conference 2022*, pages 788–797.

Kun Xu, Linfeng Song, Yansong Feng, Yan Song, and Dong Yu. 2020. Coordinated reasoning for cross-lingual knowledge graph alignment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 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 Annual Meeting of the Association for Computational Linguistics*, pages 3156–3161.

Rui Ye, Xin Li, Yujie Fang, Hongyu Zang, and Mingzhong Wang. 2019. A vectorized relational graph convolutional network for multi-relational network alignment. In *IJCAI*, pages 4135–4141.

Donghan Yu and Yiming Yang. 2023. Retrieval-enhanced generative model for large-scale knowledge graph completion. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2334–2338.

Donghan Yu, Yiming Yang, Ruohong Zhang, and Yuexin Wu. 2020. Generalized multi-relational graph convolution network. *arXiv*, page 07331.

Kaisheng Zeng, Chengjiang Li, Lei Hou, Juanzi Li, and Ling Feng. 2021a. A comprehensive survey of entity alignment for knowledge graphs. *AI Open*, 2:1–13.

Weixin Zeng, Xiang Zhao, Jiuyang Tang, and Changjun Fan. 2021b. Reinforced active entity alignment. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2477–2486.

Beibei Zhu, Ruolin Wang, Junyi Wang, Fei Shao, and Kerun Wang. 2024. A survey: knowledge graph entity alignment research based on graph embedding. *Artificial Intelligence Review*, 57(9):1–58.

Qi Zhu, Hao Wei, Bunyamin Sisman, Da Zheng, Christos Faloutsos, Xin Luna Dong, and Jiawei Han. 2020. Collective multi-type entity alignment between knowledge graphs. In *Proceedings of The Web Conference 2020*, pages 2241–2252.

Renbo Zhu, Meng Ma, and Ping Wang. 2021. Raga: relation-aware graph attention networks for global entity alignment. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 501–513. Springer.

A Proof of Proposition 1

Proof. Initially, consider an undirected connected graph scenario where $\mathbf{\Delta}_{oo}$ is a sub-matrix of the Laplacian matrix $\mathbf{\Delta}$. Given that sub-Laplacian matrices of undirected connected graphs are invertible (Rossi et al., 2022), thus Δ_{oo} is non-singular. The spectral properties of eigenvalues in undirected graphs suggest similar non-singularity for directed graphs (Maskey et al., 2023).

For general cases, assume an ordered representation of the adjacency matrix for a disconnected graph as:

$$\mathbf{A} = \mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_r) \tag{28}$$

Here, $\mathbf{A}_i, i = 1, \ldots, r$, represents each connected component. The gradient flow in equation (7) is applicable to each connected component independently for disconnected graphs. □

B Proof of Proposition 2

Proof. Commencing with the initial entity features $\mathbf{X}^{(0)}$ generated by EA encoders and applying equation (11), we iterate:

$$\begin{aligned}\begin{pmatrix}\mathbf{x}_s^{(k)}\\ \mathbf{x}_o^{(k)}\end{pmatrix} &= \begin{pmatrix}\mathbf{I} & \mathbf{0}\\ \widetilde{\mathbf{A}}_{os} & \widetilde{\mathbf{A}}_{oo}\end{pmatrix}\begin{pmatrix}\mathbf{x}_s^{(k-1)}\\ \mathbf{x}_o^{(k-1)}\end{pmatrix}\\ &= \begin{pmatrix}\mathbf{x}_s^{(k-1)}\\ \widetilde{\mathbf{A}}_{os}\mathbf{x}_s^{(k-1)} + \widetilde{\mathbf{A}}_{oo}\mathbf{x}_o^{(k-1)}\end{pmatrix}\end{aligned} \tag{29}$$

Given the stationary nature of the seed alignment entity features $\mathbf{x}_s$, we have the equation $\mathbf{x}_s^{(k)} = \mathbf{x}_s^{(k-1)} = \mathbf{x}_s$. The focus then shifts to the convergence of $\mathbf{x}_o$::

$$\mathbf{x}_o^{(k)} = \widetilde{\mathbf{A}}_{os}\mathbf{x}_s + \widetilde{\mathbf{A}}_{oo}\mathbf{x}_o^{(k-1)} \tag{30}$$

Expanding and analyzing the limit for the stationary state, we find:

$$\begin{aligned}&\lim_{k\to\infty}\mathbf{x}_o^{(k)}\\ &= \widetilde{\mathbf{A}}_{os}\mathbf{x}_s + \lim_{k\to\infty}\sum_{i=2}^{k}\widetilde{\mathbf{A}}_{oo}^{i-1}\widetilde{\mathbf{A}}_{os}\mathbf{x}_s + \lim_{k\to\infty}\widetilde{\mathbf{A}}_{oo}^{k}\mathbf{x}_o^{(0)}\\ &= \lim_{k\to\infty}\widetilde{\mathbf{A}}_{oo}^{k}\mathbf{x}_o^{(0)} + \lim_{k\to\infty}\sum_{i=1}^{k}\widetilde{\mathbf{A}}_{oo}^{i-1}\widetilde{\mathbf{A}}_{os}\mathbf{x}_s\end{aligned} \tag{31}$$

Spectral graph theory provides critical insights into the properties of the Laplacian matrix $\mathbf{\Delta}$. It establishes that the eigenvalues of $\mathbf{\Delta}$ are confined within the range [0,2). This spectral characteristic has direct implications for the matrix $\widetilde{\mathbf{A}} = \mathbf{I} - \mathbf{\Delta}$, whose eigenvalues are consequently within the range (-1,1]. A pivotal aspect of this discussion, as elucidated in Proposition 1, is the non-singularity of $\mathbf{\Delta}_{oo}$. The absence of 0 as an eigenvalue of $\mathbf{\Delta}_{oo}$ implies that $\widetilde{\mathbf{A}}$'s eigenvalues strictly occupy the interval (-1,1), thereby excluding the endpoints. This spectral behavior significantly influences the convergence properties of the iterative process. Specifically, the limit $\lim_{k\to\infty} \widetilde{\mathbf{A}}_{oo}^{n} \mathbf{x}_o^{(0)}$ approaches 0. Furthermore, the summation $\lim_{k\to\infty} \sum_{i=1}^{k} \widetilde{\mathbf{A}}_{oo}^{i-1}$ converges to $(\mathbf{I} - \widetilde{\mathbf{A}}_{\mathbf{oo}})^{-1} = \mathbf{\Delta}_{oo}^{-1}$. By integrating these insights, the long-term behavior of the iterative solution can be articulated as:

$$\lim_{k\to\infty} \mathbf{x}_o^{(k)} = \mathbf{\Delta}_{oo}^{-1} \widetilde{\mathbf{A}}_{os} \mathbf{x}_s \tag{32}$$

Therefore, when the number of iterations N is sufficiently large, the entity features in $\mathbf{x}_o^{(N)}$ approximate $\mathbf{\Delta}_{oo}^{-1} \widetilde{\mathbf{A}}_{os} \mathbf{x}_s$. □

C Searching Alignment Results

To identify alignment results, rather than use the popular distance metric of Cross-domain Similarity Local Scaling (CSLS) (Lample et al., 2018) to search the alignments in most works, we follow (Mao et al., 2022a) and (Mao et al., 2022b) to formalize the entity alignment problem as an assignment problem to enforce the one-to-one alignment constraint. Before giving the mathematical definition, it assumes $|\mathcal{E}_s| = |\mathcal{E}_t| = n^t$ to simplify the process. In addition, it uses $SIM \in \mathbb{R}^{n^t \times n^t}$ to represent the cosine similarity matrix and compute it between testing entities in two $\mathcal{KG}$s with the entity embeddings. Thus, we attempt to solve the following optimization problem:

$$\arg \max_{P \in \mathbb{P}_{n^t}} \langle P, SIM \rangle \tag{33}$$

where $\mathbb{P}_{n^t}$ is a set of permutation matrices with shape of $\mathbb{R}^{n^t \times n^t}$. Actually, we can directly obtain the optimal solution $P^\star$ by Sinkhorn operation(Cuturi, 2013).

$$P^\star = \lim_{\tau \to 0^+} \text{Sinkhorn}(\frac{SIM}{\tau}) \tag{34}$$

Table 4: Statistics of the DBP15K and SRPRS datasets, highlighting SRPRS as sparse KGs typical of real-world scenarios.

Datasets		Entity	Relation	Triple
DBP_{ZH-EN}	Chinese	19388	1701	70414
	English	19572	1323	95142
DBP_{JA-EN}	Japaense	19814	1299	77214
	English	19780	1153	93484
DBP_{FR-EN}	French	19661	903	105998
	English	19993	1208	115722
SRPRS_{FR-EN}	French	15000	177	33532
	English	15000	221	36508
SRPRS_{DE-EN}	German	15000	120	37377
	English	15000	222	38363

where the operation of Sinkhorn is as follows:

$$\begin{aligned} Sinkhorn(\mathbf{X}) &= \lim_{k\to+\infty} S^k(\mathbf{X}) \\ S^k(\mathbf{X}) &= \mathcal{N}_c(\mathcal{N}_r(S^{k-1}(\mathbf{X}))) \end{aligned} \tag{35}$$

where $S^0(\mathbf{X}) = exp(\mathbf{X})$, $\mathcal{N}_r(\mathbf{X}) = X \oslash (\mathbf{X}\mathbf{1}_{n^t}\mathbf{1}_{n^t}^\top)$ and $\mathcal{N}_c(\mathbf{X}) = X \oslash (\mathbf{1}_{n^t}\mathbf{1}_{n^t}^\top\mathbf{X})$ are the row and column-wise normalization operators of a matrix, $\oslash$ denotes the element-wise division, $\mathbf{1}_{n^t}$ is a column vector of ones. Though we can only obtain an approximate solution with a small k in practice, we empirically found that the approximate solution is enough to obtain a good alignment performance. Considering that the assumption of $n^t = |\mathcal{E}_s| = |\mathcal{E}_t|$ is easily violated, a naive reduction is to pad the similarity matrix with zeros such that its shape becomes $\mathbb{R}^{n^t \times n^t}$ where $n^t = max(n_1^t, n_2^t)$.

D Dataset

DBP15K (Sun et al., 2017) comprises three cross-lingual subsets from multilingual DBpedia. Each subset contains 15,000 entity pairs. SRPRS (Guo et al., 2019). Similar to DBP15K in terms of the number of entity pairs but with fewer triples. Table 4 lists the statistics of the DBP15K and SRPRS datasets.

E Experiment Details

E.1 Evaluation Metrics

Following most previous works (Ji et al., 2022; Zeng et al., 2021a; Mao et al., 2022b, 2020a; Wang et al., 2024), our evaluation employs cosine similarity for EA and H@k and MRR metrics for a thorough evaluation.

Table 5: Supplementary for decoding on MRAEA and AlignE. DATTI cannot decode MRAEA and RREA, as both entity and relation embeddings are required, whereas these encoders provide only entity embeddings.

Datasets		DBP_{FR-EN}			DBP_{JA-EN}			DBP_{ZH-EN}			$SRPRS_{FR-EN}$			$SRPRS_{DE-EN}$		
	Model	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
GNN	MRAEA	71.63	94.28	80.02	68.70	93.19	77.66	68.70	92.94	77.43	43.44	75.10	53.85	56.18	51.47	64.87
	+Hun	**80.33**	-	-	**76.54**	-	-	**78.04**	-	-	45.29	-	-	58.70	-	-
	+ Sinkhorn	78.00	**96.10**	**84.72**	75.91	**95.03**	**82.98**	76.29	94.51	**83.04**	**46.11**	75.39	55.80	**59.29**	82.17	**67.03**
	+LightEA	74.88	95.85	82.50	71.52	94.32	79.98	71.81	93.51	79.70	44.28	75.14	54.64	57.94	**82.61**	66.40
	+TFP	75.45	95.87	83.04	74.26	94.98	82.08	74.40	**94.54**	81.82	46.06	**75.55**	**55.89**	58.83	81.89	66.78
Translation	AlignE	53.36	86.55	64.93	50.12	83.91	61.58	50.96	82.30	61.70	34.33	65.44	44.68	44.07	69.43	52.70
	+Hun	64.16	-	-	58.41	-	-	60.37	-	-	37.24	-	-	50.15	-	-
	+ Sinkhorn	63.29	90.01	72.73	58.93	87.81	68.84	60.37	86.09	96.33	38.22	68.21	48.16	51.28	74.38	59.37
	+DATTI	58.55	85.96	68.01	55.62	82.80	64.80	57.50	82.70	66.15	39.15	69.12	49.19	53.55	75.59	61.29
	+LightEA	**69.82**	**93.53**	**78.21**	**66.40**	**92.02**	**75.53**	**66.30**	**90.12**	74.75	42.00	**72.90**	**52.30**	**55.74**	**80.57**	**64.43**
	+TFP	68.45	91.96	76.81	64.59	90.77	73.76	66.98	89.78	**74.89**	**42.11**	71.80	51.98	55.44	78.61	63.48

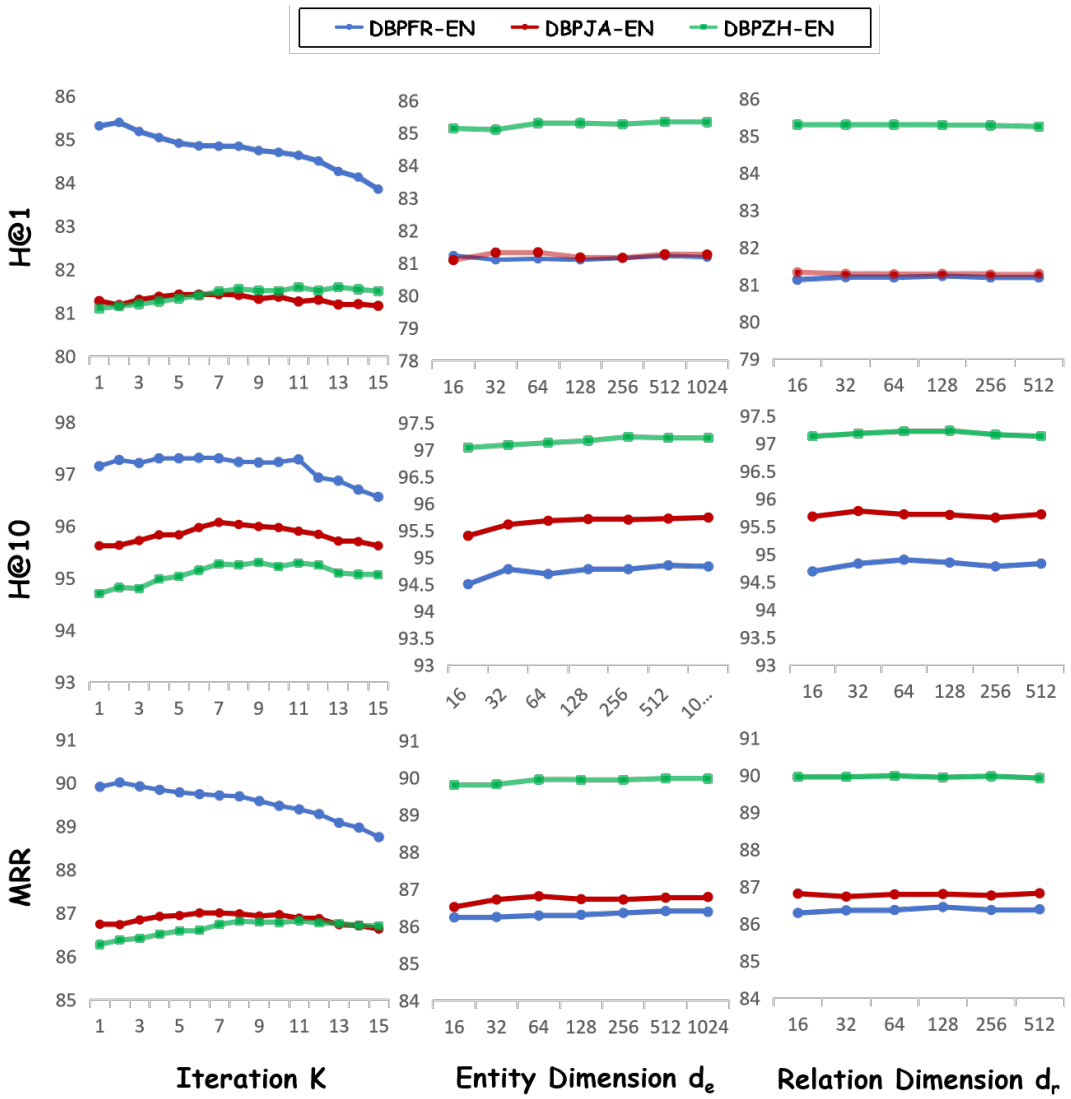

Figure 4: Hyper-parameter experiments of TFP decoding the DualAMN on DBP15K.

H@k measures the proportion of correctly aligned entities ranked within the top k positions:

$$H@k = \frac{1}{|S_t|} \sum_{i=1}^{|S_t|} \mathbb{I}[rank_i \leq k] \tag{36}$$

where $rank_i$ is the ranking of the first accurate alignment for query entity i, $\mathbb{I}$ indicates correctness ($\mathbb{I} = 1$ if $rank_i \leq k$, else 0), and S_t is the set of test alignments. MRR evaluates the average of reciprocal ranks of the first correct answer for

Table 6: Complete main results of execution time (s).

Dataset	DBP15K					
Model	Translation-based			GNN-based		
	AlignE	RSN	TransEdge	MRAEA	RREA	DualAMN
Encoding	2087	3659	1625	1743	323	177
Hun	12.2	13.4	8.0	12.4	12.3	12.2
DATTI	6.4	8.6	6.1	6.2	6.0	6.6
LightEA	12.5	12.8	12.5	12.8	13.1	13.4
TFP(CPU)	13.9	14.2	12.9	16.6	16.3	17.7
TFP(GPU)	**4.8**	**4.8**	**4.7**	**5.9**	**5.7**	**5.1**
Dataset	SRPRS					
Encoding	1190	1279	1625	558	276	163
Hun	13.5	12.7	12.5	13.8	13.1	13.3
DATTI	4.4	5.4	4.7	**4.1**	4.3	**4.1**
LightEA	10.4	12.3	12.1	10.7	11.2	11.3
TFP(CPU)	8.1	9.2	12.9	10.6	11.0	11.4
TFP(GPU)	**4.2**	**3.7**	**4.7**	4.6	**3.8**	4.6

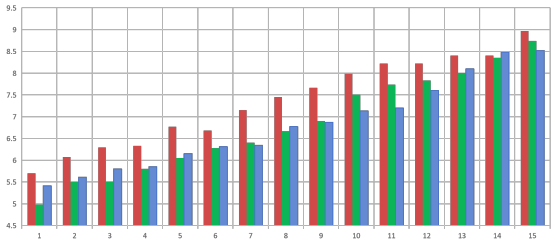

Figure 5: Execution time (seconds) on DBP15K across propagation iterations number k.

queries, reflecting the model's precision across a set of queries:

$$MRR = \frac{1}{|S_t|} \sum_{i=1}^{|S_t|} \frac{1}{rank_i} \tag{37}$$

E.2 Implementaion Hyperparameters

The output dimensions d and other hyper-parameters of all encoders adhere to their original settings in their papers: Dual-AMN ($d = 768$), RREA($d = 600$), MRAEA ($d = 600$), AlignE ($d = 75$), RSN ($d = 256$), and TransEdge ($d = 75$). The iteration k is set to their best results, which details described in Section 6.3. Other hyper-parameters remain the same for all datasets and methods: relation scale dimension $d_r = 512$, entity scale dimension $d_e = 16$.

F Decoding on other Encoders

Table 5 and Table 6 show the TFP decoding on the AlignE and MRAEA. TFP achieves performance comparable to SOTA with minimal time consumption. Among the evaluated encoders, Dual-AMN demonstrates superior performance across all datasets, underscoring the efficacy of GNN encoders. Interestingly, TransEdge shows notable success on the DBP15K dataset but underperforms on SRPRS. This can be attributed to TransEdge's reliance on existing edge semantic information for entity dependency capture, a feature less prevalent in the sparser KGs like SRPRS. Conversely, GNN-based models exhibit robustly, highlighting their aptitude for handling sparse topologies.

Table 6 provides the complete time cost. A comparative observation reveals that TFP's application is expedited on translation-based encoders relative to GNN-based ones. This acceleration is attributable to the varying entity embedding dimensions discussed in Section E.2. The output dimension for GNN-based encoders, particularly for MRAEA and Dual-AMN, is set at $d = 600$ and $d = 768$, respectively, which is larger than that of the translation-based encoders. This discrepancy in dimensionality directly influences the time efficiency of TFP, as demonstrated by the faster performance on translation-based models.

G Hyper-parameters

Table 4 presents the full performance results for varying propagation rounds K and dimensions d_e and d_r. Table 5 provides the execution time (in seconds) on DBP15K for different values of k (number of propagation iterations). All experiments were conducted using DualAMN for encoding.