NALA: an Effective and Interpretable Entity Alignment Method

Anonymous ACL submission

Abstract

Entity alignment (EA) aims to find equivalent entities between two Knowledge Graphs. Existing embedding-based EA methods usually encode entities as embeddings, triples as embeddings' constraint and learn to align the embeddings. However, the details of the underlying logical inference steps among the alignment process are usually omitted, resulting in inadequate inference process. In this paper, we introduce NALA, an entity alignment method that captures three types of logical inference paths with Non-Axiomatic Logic (NAL). Type I&II align the entity pairs and type III aligns relations. NALA iteratively aligns entities and relations by integrating the conclusions of the inference paths. Our method is logically interpretable and extensible by introducing NAL, and thus suitable for various EA settings. Experimental results show that NALA outperforms state-of-the-art methods in terms of Hits@1, achieving 0.98+ on all three datasets of DBP15K with both supervised and unsupervised settings. We offer a pioneering in-depth analysis of the fundamental principles of entity alignment, approaching the subject from a unified and logical perspective. Our code is available at https://anonymous.4open.science/r/NALA-976B.

1 Introduction

Knowledge graphs (KGs), which store massive facts about the real world, expresses massive information in a form closer to human cognition. KGs can be used by various application domains, such as question answering, recommender systems and language representation learning (knowledge graph enhanced language model) (Ji et al., 2021; Logan IV et al., 2019). The information contained in each individual KG project, such as DBpedia (Auer et al., 2007) and YAGO (Suchanek et al., 2007) is limited. So the task of entity alignment

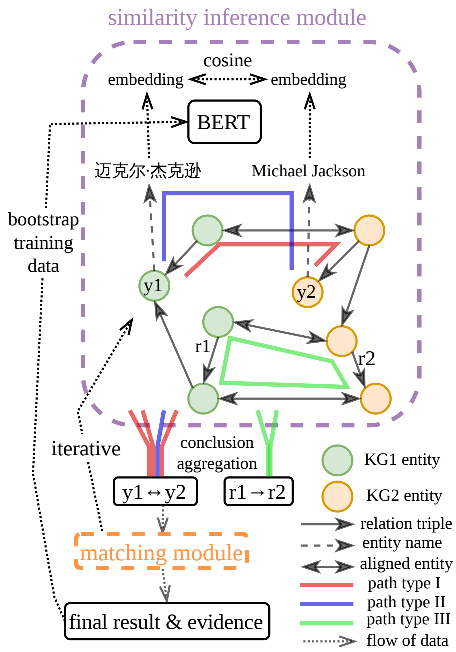

Figure 1: An overview illustration of NALA.

(EA) is proposed to increase KG completeness. The EA task consists of integrating two or more KGs into a same KG by aligning nodes that refer to the same entity.

There are many embedding-based EA methods (Fanourakis et al., 2023) that leverage deep learning techniques to represent entities with low-dimensional embeddings, and align entities with a similarity function on the embedding space. KGs' triples and seed alignments are usually seen as embeddings' constraint during the training process of such embedding model. The structural and side information of KGs are usually utilized via embedding propagation, aggregation or interaction. Generally speaking, there are some crucial shortcomings of embedding-based EA methods: *First*, they lack complex reasoning capability. Some of them are enhanced by paths (Cai et al., 2022), however, due to the nature of vector representation, it is not easy to perform or approximate symbolic reasoning on such paths. *Second*, they lack interpretability in the models, so they have to rely solely on numerical evaluation metrics to evaluate their performance. Thus the cons and pros of their model design may not be properly evaluated. *Third*, the absence of a unified framework explaining the mechanism of embedding learning and processing renders their semantic or structural learning capability quite mysterious.

Apart from embedding-based methods, path-based methods directly estimates entity similarities from the contextual data (path) that are available in the two input KGs. A "path" usually refers to an interconnected sequence of edges that links two entities of different KGs. The edges can be either relations or entity similarities. We refer to the estimation of entity similarities by processing and aggregating the paths as "similarity inference". There is a potential advantage that path-based methods can capture fine-grained matches of neighbors while the traditional embedding-based methods can't. There are also emerging methods that combine the idea of embedding learning and path reasoning. More recently, path-based (such as PARIS+ (Leone et al., 2022)) and combined methods (such as BERT-INT (Tang et al., 2020) and FGWEA (Tang et al., 2023)) are starting to surpass the performance of traditional embedding-based methods. However, they failed to handle the similarity inference appropriately to some extent, possibly due to the lack of proper formalization of the inference paths and steps.

To address the aforementioned issues of existing methods, we carefully examine the similarity inference of EA from the logical perspective. Thus we propose a path-based EA method NALA, where NAL stands for Non-Axiomatic Logic (Wang, 2013) and "A" for align. NAL is a term logic with a specific semantic theory and its design suits KG tasks (see Section 2.3).

NALA first loads the KGs and then aligns entities iteratively. In each iteration, it first performs similarity inference, then uses a matching module to obtain EA results. In the similarity inference module, it exhaustively searches the KGs for instances of three type of paths. The path instances are formalized in NAL *sentences* (as premises) and then path inference is conducted for each instance. We use BERT embedding to obtain similarity between entity names and attribute values, which constitutes some premises of the paths. For each path instance, a conclusion *sentence* is reached and conclusions of different path instances are aggregated using a specific inference rule. The similarity inference module ends up with a list of *similarity sentences* for each entity of KG1. NALA also infers the matching of relations in each iteration with path inference. In the matching module, we propose the rBMat algorithm to obtain 1-to-1 EA results.

Experiments on cross-lingual EA dataset DBP15K demonstrate that NALA outperforms SOTA EA methods in 5 different setting groups (including both supervised and unsupervised scenarios), showcasing the effectiveness of our proposed logical similarity inference module and matching module.Ablation study shows that our design choices jointly boost the overall performance of NALA.

Our contributions can be summarized as:

- We propose an interpretable EA framework NALA, which tackle the EA problem with similarity inference phase and matching phase. Various types of logical paths are formalized within the similarity inference phase.
- NALA aligns entities and relations simultaneously with a unified yet extensible logical framework.
- Our framework bridges the gap between embedding-based and path-based EA.
- Our proposed method achieves SOTA on a widely used EA dataset DBP15K's various settings.
- We present the first in-depth analysis of EA's basic principles from a unified logical perspective, and help explain the mechanism of other EA methods.

2 Preliminaries

2.1 Knowledge Graph and Entity Alignment

KGs. Knowledge graphs (KGs) are knowledge bases that store knowledge in the form of triples (or "facts"). We refer to (head, relation, tail) and (head, attribute, literal) as relation and attribute triples, respectively. Examples of both triple types are (New_Zealand, capital, Wellington) and (New_-Zealand, establishedDate, "1947-11-25"), respectively. To summarize, a KG is characterized with a number of relation triples from $\mathcal{E} \times \mathcal{R} \times \mathcal{E}$ and a number of attribute triples from $\mathcal{E} \times \mathcal{A} \times \mathcal{L}$, where

$\mathcal{E}, \mathcal{R}, \mathcal{A}$, and $\mathcal{L}$ indicate the set of entities, relations, attributes and literals, respectively.

EA. The entity alignment (EA) problem is typically defined between two KGs, $\mathcal{KG}_1$ and $\mathcal{KG}_2$, where the task consists of finding equivalences (so-called alignment) between the set of entities $\mathcal{E}_1$ and $\mathcal{E}_2$ of the two KGs. Sometimes there exists a set of given equivalences that can be used as supervision. This set $\mathcal{S}$ is known as seed alignment set. We assume that there exists a ground truth set $\mathcal{G} = \{(x_1, x_2) \in \mathcal{E}_1 \times \mathcal{E}_2 | \, x_1 \equiv x_2\}$ that includes all known equivalences between pairs of entities. We use the ground truth set to evaluate the performance of our method.

2.2 Represent KGs with NAL

A brief introduction to NAL is presented in Appendix B.

In this paper, every entity, literal or relation is regarded as an *atomic term* in NAL. Triple (x_1, r_1, y_1) is reinterpreted as *inheritance statement* (*, x_1, y_1) $\rightarrow r_1$. Its intuitive meaning is "The relation between x_1 and y_1 is a specialization of relational term r_1". The triples (or "facts") of the KGs can be seen as absolutely true (for *frequency*) and with sufficient evidence (for *confidence*) to some extent, so the *truth-value* attached to the *statement* is $\langle 1, 1 \rangle$. Entity equivalency $x_1 \equiv x_2$ can be seen as an extreme case of entity similarity $x_1 \leftrightarrow x_2$, so we align entities by similarity inference. As for relations, the *inheritance statement* $r_1 \rightarrow r_2$ intuitively represents a correspondence of two relations of different KGs such that one relational fact of r_1 in $\mathcal{KG}_1$ implies the existence of a corresponding relational fact of r_2 in $\mathcal{KG}_2$.

Inference path. We define an instance of inference path as a premise set of NAL *sentences* (triples, similarities, etc.) and a series of corresponding inference steps which will eventually lead to a conclusion *sentence*. The premise *sentences* are either in the KGs or inferred from the KGs. A type of inference path is a shared form of paths and it can be instantiated with concrete entities and relations. It is usually utilized for a certain purpose, such as aligning entities or aligning relations.

2.3 Why NAL

Actually there might be many different logical systems that are qualified to represent the similarity inference process of EA. However, we believe that the non-axiomatic nature of NAL fits in the domain of knowledge graph better than those axiomatic logical systems, because real world KGs need to deal with the problem of open-domain and alterable/incomplete/conflicting facts. Fundamentally, the tasks of knowledge graph (such as EA), fits well with the assumption of insufficient knowledge and resources (Wang, 2013), which is the basic assumption of NAL.

Technically speaking, NAL can represent entities, relations and relational triples, which are essential for EA. It can also perform formal reasoning and evidence aggregation, which is useful to align entities. The frequency/confidence measurement of *truth-value* is suitable to represent fuzziness and unknownness in the similarity inference process. The high expressiveness of NAL makes our approach extensible, which may benefit subsequent studies.

2.4 Related Work of EA

Generally speaking, there are three families of EA methods: embedding-based, path-based and combined methods, as elaborated in this section.

In recent years, embedding-based methods have become mainstream for addressing the EA task (Tang et al., 2023; Fanourakis et al., 2023). Their main idea is to embed the nodes (entities) and edges (relations or attributes) of a KG into a low-dimensional vector space that preserves their similarities in the original KG.

In addition to embedding-based methods, there exist path-based methods that directly estimates entity similarities from the contextual data (path) that are available in the two input KGs. There is a potential advantage that path-based methods can capture fine-grained matches of neighbors while the traditional embedding-based methods can't. Embedding-based methods may suffer from the negative influence from the dissimilar neighbors, according to (Tang et al., 2020). The distinction between embedding-based and path-based methods is sometimes obscure.

There are also emerging methods that combine the idea of embedding learning and path reasoning. More recently, path-based and combined methods are starting to surpass the performance of traditional embedding-based methods. Our proposed method NALA inherits and develops the ideas of two path-based methods PARIS and PARIS+. The two methods as well as some other EA methods that will be compared with our results are introduced in Appendix A.

3 The Proposed Method

The overall structure of NALA (as illustrated in Figure 1 and Algorithm 3) adopts an iterative aligning strategy, and for each iteration it first performs similarity inference, then it uses the matching module (rBMat algorithm with modification in Section 3.2) to obtain EA results. Note that the inference within each iteration benefits from the alignment results (both entities and relations) of the previous iteration. We register evidential information while performing path inference, that is memorizing which premises constitute the specific path instance and such information will be used to generate evidence log file.

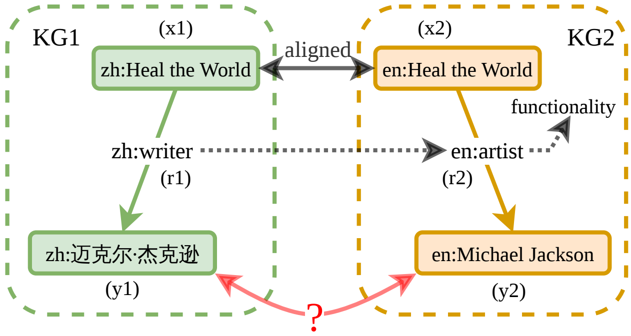

Figure 2: An instance of *type I path*, fetching from DBP15K zh-en and omitting irrelevant triples. Grey dashed arrow represents inheritance between the relations and "functionality".

3.1 Similarity Inference Module

We formalize the similarity inference module as using NAL's *revision* inference rule to aggregate three types of inference paths. The first two types calculates similarity for entities and the third type for relations.

3.1.1 Type I Path: Align Entities by Triples

Inspired by the probabilistic alignment method of PARIS, we formalize the key point of the similarity inference process as *type I path*. *Type I paths* are bridge-like inference paths between to-be-aligned entity pairs. Valid *type I paths* are retrieved from the KGs in a depth-first manner. An example of it is shown in Figure 2 and the NAL formalization is in Appendix F. Entity y_1 and entity y_2 belong to different KGs (where subscripts represent different KGs) and (y_1, y_2) forms a to-be-aligned entity pair. Triple (x_1, r_1, y_1) is a relational or attribute triple in $\mathcal{KG}_1$, where x_1 is either an entity or a literal respectively. Note that NALA automatically duplicates every original KG triple (a, b, c) with a reversed triple (c, b^{-1}, a) upon KG loading, so the attribute triple (x_1, r_1, y_1) with a literal x_1 is a reversed attribute triple. Similarly, triple (x_2, r_2, y_2) is a relational or attribute triple in $\mathcal{KG}_2$.

Note that in the case of entity pair, the similarity statement in the premise comes from either seed alignments or alignments of the previous iteration. We omit the entity *similarity statement* which has a f or c lesser than $theta$, a hyper-parameter. And in the case of literal pair (attribute value), see Section 3.1.2.

The relation inheritance is inferred in Section 3.1.3. PARIS evaluates the degree of functionality of relation r_2 with precomputed functionalities of each relation. We interpret it as an *inheritance statement* $r_2 \to [fun]$ with the degree reflected in the *truth-value*. The statement intuitively means "r_2 has the functional property (to some extent)".

The idea of *type I path* can be explained with the example. We would like to figure out whether "zh:迈克尔·杰克逊" and "en:Michael Jackson" refers to the same entity. We find out that a related pair of entity: "zh:Heal the World" and "en:Heal the World" are known aligned entity pair (or, inferred to be aligned). "zh:Heal the World"'s writer is "zh:迈克尔·杰克逊" and "en:Heal the World"'s artist is "en:Michael Jackson". We also know that being the writer of something probably implies being the artist of it (the relation inheritance). We look through the KG and find out that a certain work usually has only one artist. We conclude that these premises together form a certain amount of positive evidence that supports "zh:迈克尔·杰克逊" and "en:Michael Jackson" being the same entity. *Type I path* can be seen as the fundamental entity alignment evidence (signal).

The conclusions with the same statement but obtained from different *type I paths* are merged by *probabilistic revision* rule because of the probabilistic nature of functionality. For example, the functionality of relation "zh:writer" is 0.78 which means that the majority of works approximately have one to two writers. While reasoning with *type I paths*, we could not know how many writers does "zh:Heal the World" have, the conclusion has a probabilistic nature because we don't know whether "zh:迈克尔·杰克逊" and "en:Michael Jackson" is the same writer of "Heal the World". The *probabilistic revision* rule is similar with the continued multiplication of PARIS's probability formula for $Pr\,(x_1 \equiv x_2)$ (given in Appendix A), except for the introduction of *confidence*.

3.1.2 Type II Path: Align Entities by Name

Type II path is the direct path linking the to-be-aligned entities with their name/description similarity. It only have a conclusion statement:

$$y_1 \leftrightarrow y_2 \;\; \langle sim(name(y_1), name(y_2)),\; C_{name} \rangle$$

where sim is the cosine similarity of entity name/description embedding and C_{name} is a hyper-parameter. NALA adopts BERT as the embedding model. The BERT unit is finetuned on the name/description of seed alignment entity pairs before embedding generation, similar with BERT-INT. The conclusion of a type II path is seen as a piece of evidence and fused with other evidences by *revision* rule. See more discussion of utilizing literal value in appendix C.1 and C.2.

We implement an adaptive method to automatically set C_{name} to avoid excessive parameter tuning. First, with a specific setting and dataset, we run NALA for 5 iterations (with a default $C_{name} = 0.5$ which represents a unit amount of evidence) and calculate the alignment output's average *confidence*. We set $C_{name} = halve\_evidence(average\_confidence)$, where $halve\_evidence$ is a function that outputs a *confidence* value that corresponds to half of evidence amount of the input *confidence*. The idea is to balance the influence of structural information and name information, preventing the name information's evidence from being too strong or too weak. Then, the we restart NALA from the first iteration and C_{name} remain unchanged. If translated name is available, the evidence amount is equally divided between translated and original name's C_{name}. If the BERT unit is un-finetuned, we penalize C_{name}'s evidence amount by a factor $C_{penalty}$.

Attribute value embeddings' cosine similarity is used to convert to the *truth-value* of premise (5) where x_1 and x_2 are distinct attribute values:

$$x_1 \leftrightarrow x_2 \;\; \langle f = sim(x_1, x_2),\; c = sim(x_1, x_2) \rangle$$

The idea is that the deep learning model's result which has higher similarity is usually more verifiable. For identical attribute values, the *truth-value* is simply $\langle 1,\ 1 \rangle$. There are thousands of distinct attribute values in a KG, so for an attribute value we only consider the K_{value} most similar (but not identical) values in the other KG to prevent an explosive number of value similarities. K_{value} is a hyper-parameter and in implementation we set K_{value} to 1.

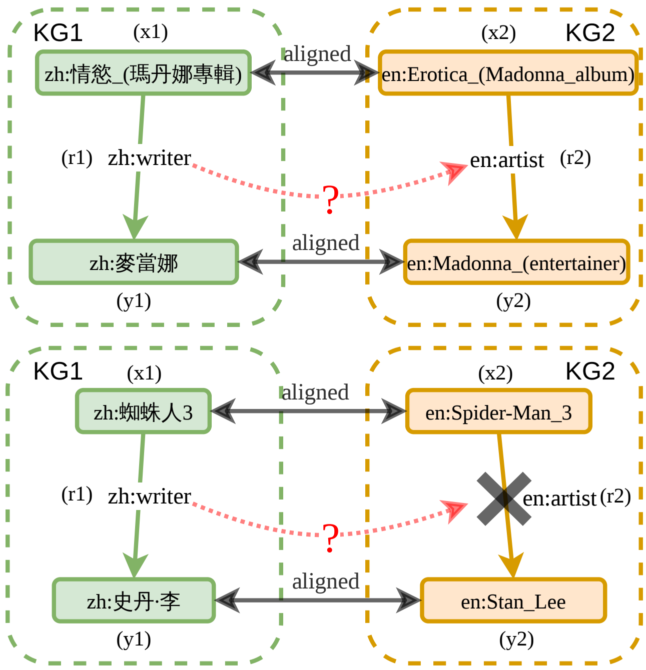

Figure 3: An illustration of *type III path*. The upper half represents positive version of the path and the lower half represents negative version. The dark cross represents the absence of the triple.

3.1.3 Type III Path: Aligning Relations

NALA align relations by path inference, which is a different approach from PARIS's probabilistic relation aligning method. We formalize the inference process as *type III path* in Appendix F.

Two instances of *type III path* are illustrated in Figure 3. We would like to figure out the inheritance between relations "zh:writer" and "en:artist", so multiple path instances are collected, including both positive version and negative version of the type of path. The conclusions of the two versions are supposed to be merged by the *revision* rule. The relation inheritance sentence of *type I path* use computation result of *type III path* in the previous iteration or a default value $\langle 1,\ iota \rangle$ (in the first two iterations), where $iota$ is a hyper-parameter.

3.2 Matching Module

In the matching module, first we consider the 1-to-1 range assumption (see Appendix C.3).

Then the similarity sentences are rearranged. *Type I path*'s similarities (*type I path*) are naturally sparse, because it only considers the entity pairs which is effectively linked by the logical path. Entity name/description's similarities (*type II path*) are dense, however, it is noisy and most of the similarities are useless. NALA's similarity inference module exhaustively search for and aggregates the two types of similarity sentences for a specific to-be-aligned entity versus any entity in the other KG. Then, because of the sparsity of informative similarity signal, the similarity sentences is rearranged into ordered linked list, one list for a specific to-be-aligned entity. The sentences are ordered (descending) by its $expectation$ value. We only store the top K_{sim} similarity sentences in the linked list, where K_{sim} is a hyper-parameter.

Then we perform a recursive bidirectional matching algorithm (rBMat) which has similar idea with BMat (Tang et al., 2020) but different implementation. See Algorithm 1 and Algorithm 2 for details. The main idea is to recursively delete the similarity sentences that don't conform the 1-to-1 assumption. Considering sorting cost, our rBMat has $O(kn^2)$ time complexity and $O(kn)$ space complexity, where k represents K_{sim} and $k \ll n$.

We found that there are still some mismatches after performing rBMat algorithm and most of them share a same pattern. For example, $e_1 \leftrightarrow e_2$ and $e_3 \leftrightarrow e_4$ are two ground truth pairs, however, rBMat's result is $e_1 \leftrightarrow e_3$ and $e_2 \leftrightarrow e_4$. We implements a simple swapping technique to handle this. For every pair of similarity sentences, we swap their alignment if the swapped similarity sentences have a higher total $expectation$ value than their original form.

3.3 Unsupervised Learning

The seed alignment set is not always available for different EA tasks or real-world EA applications. So an unsupervised scenario is sometimes adopted to evaluate the industrial applicability of EA methods. We adapt our method to the unsupervised scenario, that is, without using seed alignments. The BERT embedding model need to finetune on seed alignments, so we adopt a bootstrapping strategy. First, a NALA instance performs alignment on the dataset with 0% seed and no literal embedding information. Then, filter the initial alignment results' $expectation$ with a threshold θ_{filter} and use the filtered result as the training set of BERT. Next, another 0% seed NALA instance performs alignment with the help of BERT's literal embedding information to obtain the final result.

4 Interpretability of NALA

Following (Rudin, 2019; Marcinkevičs and Vogt, 2020), interpretable ML (machine learning) focuses on designing models that are inherently interpretable, while explainable ML tries to provide post hoc explanations for existing black box models. NALA is highly interpretable and self-explanatory. It is arguably more interpretable than PARIS for the following two reasons. First, with the introduction of evidence amount (confidence) and logical inference rules, NALA processes data with more information and generates a more informative explanation. Second, NALA manages value similarity, name similarity and structural similarity in a unified logical framework, while PARIS doesn't leverage such side information.

NALA is self-explanatory in the sense that it generates a log file of evidences for the alignments so we can inspect the file after an iteration. This feature enhances the troubleshooting capacity of us to some extent during the development process of NALA. For example, inspecting the faulty alignments in the evidence file inspired many decision choices in this paper. The generated evidences are displayed in our GitHub repository.

Using the neural BERT model does not weaken the interpretability of *type I path* because utilizing literal value similarity does not affect the interpretable inference steps. Moreover, as we only keep the attribute value similarities with a score above the threshold, these similarities are easily understood and self-explanatory, except the wrong ones. Our method tolerates faulty attribute value similarity because *type I path* needs a conjunction of all premises, while faulty similarities usually can't form a complete premise set.

We discuss NALA's relation with other methods and help explain the mechanism of those methods in Appendix C.4.

5 Experiments and Results

5.1 Datasets and Settings

We evaluate our model on two EA datasets: the widely used cross-lingual dataset DBP15K (see (Sun et al., 2017) for details) and a monolingual multi-source dataset OpenEA benchmark (D-W-15K-V2, D-Y-15K-V2 and D-Y-100K-V2) (Sun et al., 2020). DBP15K consists of three subsets of cross-lingual KG pairs extracted from DBpedia. Each sub-dataset of OpenEA benchmark consists of two English KGs. The statistics of the datasets are listed in Table 3.

The settings of our main results on DBP15K (Table 1) consists of five sub-settings: $Attr.$, $Name$,-

Group	Model	Settings					ZH_EN	JA_EN	FR_EN
		Attr.	Name	Trans.	Desc.	Seed	Hits@1	Hits@1	Hits@1
1	JAPE	✓				30%	0.412	0.363	0.324
	GCNAlign	✓				30%	0.413	0.399	0.373
	PARIS+	✓				30%	0.904	0.874	0.928
	NALA	✓				30%	**0.985**	**0.972**	**0.990**
2	PARIS	✓				0%	0.777	0.785	0.793
	FGWEA*	✓				0%	0.929	0.922	0.967
	NALA	✓				0%	**0.982**	**0.968**	**0.987**
3	RDGCN		✓	✓		30%	0.708	0.767	0.886
	CUEA		✓	✓		30%	0.921	0.946	0.956
	UPL-EA		✓	✓		30%	0.949	0.970	0.995
	SE-UEA		✓	✓		0%	0.935	0.951	0.957
	LightEA		✓	✓		0%	0.952	0.981	0.995
	FGWEA*		✓	✓		0%	**0.959**	0.982	0.994
	NALA		✓	✓		0%	0.952	**0.985**	**0.996**
4	BERT-INT	✓	✓		✓	30%	0.968	0.964	0.995
	NALA	✓	✓		✓	30%	**0.998**	**0.996**	**0.999**
5	TEA	✓	✓			30%	0.941	0.941	0.979
	FGWEA*	✓	✓			0%	0.976	0.978	0.997
	NALA	✓	✓			0%	**0.993**	**0.988**	**0.998**

Table 1: Evaluation Results of all compared EA methods on DBP15K in five different setting groups. Methods marked with * use the additional information of relation names.

$Trans.$, $Desc.$ and $Seed$, explained as follows. $Attr.$ is for utilizing the attribute triples. $Name$ is for utilizing the entity name information. $Trans.$ is for utilizing translators for entity name. We use the Google translator, which is consistent with many other studies.. $Desc.$ is for utilizing the information of entity description. $Seed$ is for the percentage of seed alignments, 30% for the conventional supervised scenario and 0% for the unsupervised scenario.

We categorize baselines into five setting groups and run NALA using the settings for each group. The five groups cover a vast majority of different method's settings. Group 1 is the supervised scenario with attribute triples. Group 2 is the unsupervised scenario with attribute triples. Group 3 is the supervised or unsupervised scenario with entity name information and translator. Group 4 is the supervised scenario with entity name and description information, which is the same scenario as BERT-INT. Group 5 is the unsupervised scenario with attribute triples and entity name information.

Most hyper-parameters of our model remain the same across different datasets and setting groups, except for group 3 which will be discussed later. The hyper-parameters are selected manually. We set $iota$ = 0.5, $theta$ = 0.1, $C_{penalty} = 4$ and $end\_iteration$ = 19 (20 iterations in total). K_{sim} is set to 80. θ_{filter} is set to 0.9. The BERT unit is finetuned for 15 epochs. The dimension of the BERT CLS embedding is 768 and the dimension of BERT unit's embedding output is 300.

5.2 Main Results

We compare NALA with the following methods, most of which are new and well-performing: JAPE (Sun et al., 2017), GCNAlign (Wang et al., 2018), PARIS+ (Leone et al., 2022), PARIS (Suchanek et al., 2011), FGWEA (Tang et al., 2023), RDGCN (Wu et al., 2019) ,CUEA (Zhao et al., 2022), UPL-EA (Ding et al., 2023), SE-UEA (Jiang et al., 2023b), LightEA (Mao et al., 2022), BERT-INT (Tang et al., 2020), TEA (Zhao et al., 2023). Their results are fetched from their original papers.

The experimental settings and results of NALA and all compared baselines on DBP15K are in Table 1. As observed, NALA achieves the best performance in term of Hits@1 in all five groups except group 3. NALA outperforms BERT-INT significantly with identical setting and the same embedding method, verifying the effectiveness of our similarity inference combined with the matching algorithm. NALA outperforms FGWEA in group 2 and 5, indicating that it successfully utilizes the information of attribute triples. In group 1, two classic EA model JAPE and GCNAlign are outperformed by the newer approaches (PARIS+ and NALA) by a significant margin, indicating the effective innovation of the new EA approaches in the recent years. The performance of NALA in unsupervised group 2 approaches its performance in supervised group 1 with a minor gap, indicating that our proposed bootstrapping strategy effectively adapts to the unsupervised setting (with the help of attribute information).

As for setting group 3, the attribute information is unavailable and we have to rely on the name and translation information to bootstrap the alignment process. We adjust the hyper-parameter K_{sim} to 400 and other hyper-parameters are unchanged. We use two BERT units instead of one to separately embed the original entity names and the translated entity names. The BERT units are finetuned separately. We adapt the bootstrapping strategy in Section 3.3 into three steps. In each step, we perform alignment with a NALA instance and filter the alignment results as the training set of the BERT units of the subsequent step. The results of each step of the three datasets are shown in Figure 5. As expected, the alignment performance increases with the steps, because the BERT unit obtains better finetuning data every step and thus produces better embeddings for alignment. The results of each iteration of the second bootstrap step are shown in Figure 6. NALA outperforms other methods in setting group 3 on JA_EN and FR_EN, including three supervised ones. However, on ZH_EN unsupervised FGWEA yields better performance. This is possibly due to FGWEA's utilization of additional information of relation names. The error accumulation effect of NALA's strategy in group 3 is left for further study.

5.3 Ablation study

To validate the effectiveness of each component in NALA, we compare it with several ablations. We demonstrate the results in Table 4, where w/o represents without and E_{value} represents attribute value embedding information. $all\_revision$ represents replacing *probabilistic revision* rule with *revision* rule and $all\_prob\_revision$ is the opposite. $1v1\_range$ is the 1-to-1 matching range information that is utilized in Section 3.2 and $swapping$ is a proposed technique in Section 3.2.

NALA performs the best compared with its variants. The *revision* rule can deal with negative evidences of similarity sentences, while *probabilistic revision* rule cannot. The ablation results together with the main results show that NALA seems to have good monotonicity in Hits@1 performance in the sense that when adding extra information or procedure (component) into the model, the Hits@1 increases monotonically. Arguably, this is because introducing two-dimensional *truth-values* in every inference step separates confidence from truth degree (frequency) in every statement, thus the information of relative reliability level is stored for further usage.

NALA also achieves competitive performance compared with LightEA and FGWEA on OpenEA benchmark datasets, as shown in Table 2 and Table 4.

Model	D-W-15K-V2	D-Y-15K-V2	D-Y-100K-V2
NALA	0.908	**0.981**	**0.980**
LightEA	**0.951**	0.976	0.977

Table 2: Performance (Hits@1) of NALA on OpenEA benchmark datasets.

6 Conclusion and Future Work

In this paper, we propose an entity alignment method named NALA, tackling the EA problem by modeling similarity inference and performing a matching algorithm. Similarity inference obtains similarity through paths that connect the entities. NALA leverages three type of paths, exploiting both structural and side information of KGs. Using the similarities, NALA matches the entities by the proposed rBMat algorithm. NALA is also successfully adapted to the unsupervised scenario and a scenario without attribute triples. Compared with up-to-date EA methods, NALA attains competitive result on OpenEA benchmark datasets and various settings of DBP15K, indicating that it successfully handles the most effective part of similarity inference.

We also take a step in re-evaluating the design choices of different EA models, by providing some interesting insights (explanations) of different methods and competitive results compared with them. Hopefully, our approach may broaden the view and deepen the understanding of the EA research community. How to combine embedding models with path inference and facilitate its full potential is a research question to be further studied.

NAL can express and process many different reasoning patterns and logical structures, so NALA can be extended to tackle other challenges in the EA process in future research, such as integrating ontological information.

Limitations

Roughly speaking, NALA has slightly more hyper-parameters than some other EA methods, which may be a drawback.

NALA costs more time compared with the fastest EA methods (2418 seconds compared with 34.5 seconds by LightEA-L on D-Y-100K-V2), possibly due to the inability to utilize GPU in its logical design, thus being more difficult to be parallelized.

The performance of NALA on a hard setting of the datasets, that is, without both attribute triples and entity name information is moderate. Our approach is not yet optimized for utilizing pure structure information of KGs.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *international semantic web conference*, pages 722–735. Springer.

Ali Beikmohammadi and Sindri Magnússon. 2023. Comparing nars and reinforcement learning: An analysis of ona and q-learning algorithms. In *International Conference on Artificial General Intelligence*, pages 21–31. Springer.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, volume 2, pages 2787–2795.

Weishan Cai, Wenjun Ma, Jieyu Zhan, and Yuncheng Jiang. 2022. Entity alignment with reliable path reasoning and relation-aware heterogeneous graph transformer. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22)*, pages 1930–1937.

Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 3998–4004.

Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2016. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1511–1517.

Nhat-Minh Dao, Thai V Hoang, and Zonghua Zhang. 2023. A benchmarking study of matching algorithms for knowledge graph entity alignment. *arXiv preprint arXiv:2308.03961*.

Qijie Ding, Jie Yin, Daokun Zhang, and Junbin Gao. 2023. Combating confirmation bias: A unified pseudo-labeling framework for entity alignment. *arXiv preprint arXiv:2307.02075*.

Nikolaos Fanourakis, Vasilis Efthymiou, Dimitris Kotzinos, and Vassilis Christophides. 2023. Knowledge graph embedding methods for entity alignment: experimental review. *Data Mining and Knowledge Discovery*, 37(5):2070–2137.

Genet Asefa Gesese, Russa Biswas, Mehwish Alam, and Harald Sack. 2021. A survey on knowledge graph embeddings with literals: Which model links better literal-ly? *Semantic Web*, 12(4):617–647.

Lingbing Guo, Yuqiang Han, Qiang Zhang, and Huajun Chen. 2022. Deep reinforcement learning for entity alignment. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2754–2765.

Aidan Hogan, Axel Polleres, Jürgen Umbrich, and Antoine Zimmermann. Some entities are more equal than others: statistical methods to consolidate linked data. In *4th International Workshop on New Forms of Reasoning for the Semantic Web: Scalable and Dynamic (NeFoRS2010)*.

Mengting Hu, Zhen Zhang, Shiwan Zhao, Minlie Huang, and Bingzhe Wu. 2023. Uncertainty in natural language processing: Sources, quantification, and applications. *arXiv preprint arXiv:2306.04459*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514.

Chuanyu Jiang, Yiming Qian, Lijun Chen, Yang Gu, and Xia Xie. 2023a. Unsupervised deep cross-language entity alignment. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 3–19. Springer.

Shangpu Jiang, Daniel Lowd, and Dejing Dou. 2012. Learning to refine an automatically extracted knowledge base using markov logic. In *2012 IEEE 12th International Conference on Data Mining*, pages 912–917. IEEE.

Tingting Jiang, Chenyang Bu, Yi Zhu, and Xindong Wu. 2023b. Integrating symbol similarities with knowledge graph embedding for entity alignment: an unsupervised framework. *Intelligent Computing*, 2:0021.

Hugo Latapie, Mina Gabriel, and Ramana Kompella. 2022. Hybrid ai for iot actionable insights real-time data-driven networks. In *International Workshop on Self-Supervised Learning*, pages 127–131. PMLR.

Manuel Leone, Stefano Huber, Akhil Arora, Alberto García-Durán, and Robert West. 2022. A critical re-evaluation of neural methods for entity alignment. In *Proceedings of the VLDB Endowment*, volume 15, pages 1712–1725.

Lin Lin, Lizheng Zu, Feng Guo, Song Fu, Yancheng Lv, Hao Guo, and Jie Liu. 2023. Using combinatorial optimization to solve entity alignment: An efficient unsupervised model. *Neurocomputing*, 558:126802.

Bing Liu, Tiancheng Lan, Wen Hua, and Guido Zuccon. 2023. Dependency-aware self-training for entity alignment. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 796–804.

Xiao Liu, Haoyun Hong, Xinghao Wang, Zeyi Chen, Evgeny Kharlamov, Yuxiao Dong, and Jie Tang. 2022. Selfkg: Self-supervised entity alignment in knowledge graphs. In *Proceedings of the ACM Web Conference 2022*, pages 860–870.

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)*, pages 6355–6364.

Robert L Logan IV, Nelson F Liu, Matthew E Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge-graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5962–5971.

Shengxuan Luo and Sheng Yu. 2022. An accurate unsupervised method for joint entity alignment and dangling entity detection. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2330–2339.

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, 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.

Xin Mao, Wenting Wang, Yuanbin Wu, and Man Lan. 2022. 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.

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.

Ričards Marcinkevičs and Julia E Vogt. 2020. Interpretability and explainability: A machine learning zoo mini-tour. *arXiv preprint arXiv:2012.01805*.

Francis Jeffry Pelletier and Allen Hazen. 2021. Natural deduction systems in logic.

Gabriel Peyré, Marco Cuturi, and Justin Solomon. 2016. Gromov-wasserstein averaging of kernel and distance matrices. In *International conference on machine learning*, pages 2664–2672. PMLR.

C. Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215.

Fabian M Suchanek, Serge Abiteboul, and Pierre Senellart. 2011. Paris: Probabilistic alignment of relations, instances, and schema. In *Proceedings of the VLDB Endowment*, volume 5, pages 157–168.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

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*, pages 628–644. Springer.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020. A benchmarking study of embedding-based entity alignment for knowledge graphs. In *Proceedings of the VLDB Endowment*, volume 13, pages 2326–2340.

Jianheng Tang, Kangfei Zhao, and Jia Li. 2023. A fused gromov-wasserstein framework for unsupervised knowledge graph entity alignment. *arXiv preprint arXiv:2305.06574*.

Xiaobin Tang, Jing Zhang, Bo Chen, Yang Yang, Hong Chen, and Cuiping Li. 2020. Bert-int: a bert-based interaction model for knowledge graph alignment. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3174–3180.

Xiaobin Tian, Zequn Sun, and Wei Hu. 2023. Generating explanations to understand and repair embedding-based entity alignment. *arXiv preprint arXiv:2312.04877*.

Pei Wang. 2005. Experience-grounded semantics: a theory for intelligent systems. *Cognitive Systems Research*, 6(4):282–302.

Pei Wang. 2013. *Non-axiomatic logic: A model of intelligent reasoning*. World Scientific.

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.

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*, pages 5278–5284.

Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. 2019. Gromov-wasserstein learning for graph matching and node embedding. In *International conference on machine learning*, pages 6932–6941. PMLR.

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.

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

Weixin Zeng, Xiang Zhao, Jiuyang Tang, and Xuemin Lin. 2020. Collective entity alignment via adaptive features. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1870–1873. IEEE.

Xiang Zhao, Weixin Zeng, Jiuyang Tang, Xinyi Li, Minnan Luo, and Qinghua Zheng. 2022. Toward entity alignment in the open world: an unsupervised approach with confidence modeling. *Data Science and Engineering*, 7(1):16–29.

Yu Zhao, Yike Wu, Xiangrui Cai, Ying Zhang, Haiwei Zhang, and Xiaojie Yuan. 2023. From alignment to entailment: A unified textual entailment framework for entity alignment. *arXiv preprint arXiv:2305.11501*.

Xiaojin Zhu and Ghahramani Zoubin. 2002. Learning from labeled and unlabeled data with label propagation. *ProQuest number: information to all users*.

A Related Work

A.1 Embedding-based EA

Embedding-based EA methods usually consist of three parts: the embedding module, the alignment module and the matching module. For the embedding module, translational methods and graph neural network (GNN) methods are the most popular. Translational methods, such as MTransE (Chen et al., 2016), usually optimize a margin-based loss function to learn the structural information (relation triples) of a KG. On the other hand, GNN methods recursively aggregate the representations of neighboring nodes with graph convolutional networks (GCNs) or graph attention networks (GATs). The representative ones are RDGCN (Wu et al., 2019) and RREA (Mao et al., 2020b), respectively. The alignment module maps the entity embeddings in different KGs into a unified space. There are generally three techniques (Fanourakis et al., 2023) for this module: 1. Sharing the embedding space by using the margin-based loss to enforce the seed alignment entities' embeddings from different KGs to be close. 2. Swapping the triples of seed alignment entities. 3. Mapping the entity vectors from one embedding space to the other using a transformation matrix. The matching module generates the final alignment result. Common practices use the cosine similarity, the Manhattan distance, or the Euclidean distance between entity embeddings to measure their similarities and then perform a specific matching algorithm based on the similarity scores.

A.2 Path-based EA

PARIS (Suchanek et al., 2011) is a classic unsupervised non-neural EA method with competitive performance on benchmark datasets (Leone et al., 2022). It is purely path-based. Following previous works (Hogan et al.), PARIS introduces the probabilistic usage of "functionality" into the field of EA to enhance the validity of similarity inference paths. Functionality generally corresponds to the uniqueness of related things, for example a man can only have one father but multiple friends, so $fun(father)$ is close to 1 and $fun(friend)$ is relatively lower, where $fun()$ represents functionality of a relation or attribute. See (Suchanek et al., 2011) for more details about functionality. With functionality, PARIS constructs a probabilistic model that estimates the probabilities of entity equivalences:

$$Pr\left(x_1 \equiv x_2\right) = 1 - \prod_{r_1(x_1,y_1),r_2(x_2,y_2)} (1 - Pr\left(r_1 \subset r_2\right) \times fun(r_2^{-1}) \times Pr\left(y_1 \equiv y_2\right))$$

As depicted in the above formula, PARIS estimates the equivalence probabilities by integrating paths that connects corresponding entities. It also find subrelations between the two ontologies of KG. Subrelations, such as $r_1 \subset r_2$, intuitively means a correspondence of two relations of different KGs such that one relational fact of r_1 in $\mathcal{KG}_1$ implies the existence of a corresponding relational fact of r_2 in $\mathcal{KG}_2$. Here is the formula for $Pr\left(r_1 \subset r_2\right)$:

$$\frac{\Sigma_{r_1(x_1,y_1)}\left(1-\prod_{r_2(x_2,y_2)}(1-Pr(x_1\equiv x_2)\times Pr(y_1\equiv y_2))\right)}{\Sigma_{r_1(x_1,y_1)}\left(1-\prod_{x_2,y_2}(1-Pr(x_1\equiv x_2)\times Pr(y_1\equiv y_2))\right)}$$

With the help of subrelations' measurement, PARIS generalizes the equation of $Pr\,(x_1 \equiv x_2)$ to the case where the two ontologies do not share common relations. Therefore, PARIS recursively aligns the entities and the equivalence probability of $x_1 \equiv x_2$ depends recursively on other equivalence probabilities. In each iteration, the probabilities are re-calculated based on the equivalences and subrelations of the previous iteration. Initial equivalences are computed between attribute literals based on a certain string distance measurement.

PARIS+ (Leone et al., 2022) is a variant of PARIS that makes a simple refinement and works in the absence of attribute triples. It processes the seed alignment information to generate synthetic attribute triples. That is, for every pair of seed alignments (x_1, x_2), it creates the attribute triples (x_1, EA:label, string(x_1)) and (x_2, EA:label, string(x_1)), where EA:label is a synthetic relation. Thus, the reverse of the relation EA:label is designed to be highly functional in order to let the model match the seed alignments easily. NALA adopts the same refinement as PARIS+.

A.3 Combined EA

BERT-INT (Tang et al., 2020), an embedding-path EA method, uses the well-known transformer model BERT to embed the entities and literals. It calculates the cosine similarity of the entity name/description embedding. Then it proposes an interaction model that compares each pair of neighbors or attributes (which forms a path from the source entity to the target entity) to obtain the neighbor/attribute similarity score. The name/description similarity vector, neighbor similarity vector and attribute similarity vector are concatenated and applied to a MLP layer to get the final similarity score.

FGWEA (Tang et al., 2023) is a three-step progressive optimization algorithm for EA and it can be classified as an embedding-path EA method. First, the entity names and concatenated attribute triples are used for semantic embedding matching to obtain initial anchors. Then in order to approximate GWD (Gromov-Wasserstein Distance (Peyré et al., 2016)), FGWEA computes cross-KG structural and relational similarities, which are then used for iterative multi-view optimal transport alignment. Finally, the Bregman Proximal Gradient algorithm (Xu et al., 2019) is employed to refine the GWD's coupling matrix.

A.4 Other EA methods

There are also a few works that focus on the interpretability or explanability of EA, such as LightEA (Mao et al., 2022) and ExEA (Tian et al., 2023). **LightEA** is an interpretable non-neural EA method. It is inspired by a classical graph algorithm, label propagation (Zhu and Zoubin, 2002). First, it generates a random orthogonal label for each seed alignment entity pair. Then, the labels of entities and relations are propagated according to the three views of adjacency tensor. Finally, LightEA utilizes sparse sinkhorn iteration to address the assignment problem of alignment results.

The **ExEA** framework, proposed by (Tian et al., 2023), aims to explain the results of embedding-based EA. It generates semantic matching subgraphs as explanation by matching semantically consistent triples around the two aligned entities. ExEA devises an alignment dependency graph structure to gain deeper insights into the explanation.

The recent literature of EA is abundant, focusing on many different aspects or procedures of entity alignment apart from the aforementioned ones, such as utilizing attribute triples (Liu et al., 2020; Sun et al., 2017), utilizing literals (Gesese et al., 2021; Chen et al., 2018) , sample mining (Liu et al., 2022; Mao et al., 2021a), reinforcement learning (Guo et al., 2022), matching algorithm (Lin et al., 2023; Dao et al., 2023; Mao et al., 2021b; Xu et al., 2020; Zeng et al., 2020), iterative strategy (Liu et al., 2023; Mao et al., 2020a) and unsupervised learning (Jiang et al., 2023a,b; Liu et al., 2022; Luo and Yu, 2022; Zhao et al., 2022). There are also some surveys for EA (Fanourakis et al., 2023; Zeng et al., 2021; Sun et al., 2020; Mao et al., 2022). Besides graph structural, attribute and literal information, there are other information forms researched by the EA community, such as temporal, spatial and graphical information, however, these topics are beyond the scope of this paper.

B Introduction of NAL

NAL (Non-Axiomatic Logic) (Wang, 2013) is a logic designed for the creation of general-purpose AI systems, by formulating the fundamental regularities of human thinking in a general level. It can be used as the logical foundation of a (non-axiomatic) inference system and it has been explored to be utilized in various AI tasks (Beikmohammadi and Magnússon, 2023; Latapie et al., 2022). Traditional inference systems are usually based on model-theoretic semantics, while under the assumption of insufficient knowledge and resources, NAL is a term logic basing on *experience-grounded semantics* (Wang, 2005). The meaning of a *term* in NAL, to the inference system, is determined by its role in the *experience* (which will be explained later), that is, how it has been related to other *terms* in the past. The *truth-value* of a *statement* in NAL is determined by how it has been supported or refuted by other *statements* in the past.

In this paper we only utilize a fraction of NAL's syntax and inference capability (for EA). We will now introduce the relevant parts of its syntax. A *term* in NAL can either be atomic or compound. An *atomic term* is a word (string) or a *variable term*. *Independent variable*, such as "$\$x$", represents any unspecified *term* under a given restriction, and intuitively correspond to the universally quantified variable in first-order predicate logic. *Dependent variable*, such as "$\#y$", represents a certain unspecified *term* under a given restriction, and intuitively correspond to the existentially quantified variable. A *compound term* consists of *term connector* and *components* (which are themselves *terms*).

A basic *statement* has the form of "*subject copula predicate*", where *subject* and *predicate* are *terms*. There are multiple types of *copula* and each type has a corresponding *statement* type, including: 1.*Inheritance* ("$A \rightarrow B$", where A and B are *terms*) which intuitively means "B is a general case of A"; 2.*Similarity* ("$A \leftrightarrow B$") which intuitively means "A is similar with B"; 3.*Implication*, which is a *higher-order copula* ("$P \Rightarrow Q$", where P and Q are *statements*), intuitively means "P implies Q" (different from the "material implication", it requires P to be related to Q in content because NAL is a term logic that uses syllogistic inference rules and only derives conclusions that are related in content). A *sentence* is a *statement* together with its *truth-value*. An *intensional set* with only one component, for example, "$[red]$" intuitively means "red things". *Term connector* "$*$" (*product*) combines multiple *component terms* into an ordered *compound term* such as $(*, A, B)$, which intuitively means "an anonymous relation between A and B". Compound terms are usually written in the prefix format, that is the *term connector* is written in the first place. *Statement connector* "$\wedge$" can be seen as the conjunction operator of propositional logic.

NAL is "non-axiomatic" in the sense that the *truth-value* of a conclusion in the inference system does not indicate how much the conclusion agrees with the "state of affair" in the world, or with a constant set of assumptions (the axioms), but how much it is supported by the evidence provided by the past *experience* of the system. *Experience* means the inference system's history of interaction with the environment or equivalently the input *sentences*. The acquisition of *experience* may involve sensorimotor mechanism and sensation-perception process, which is beyond our scope. The information source of a *sentence* is characterized as its *evidence*. The inference rules of NAL coherently pass on the evidential information from the premises to the conclusion, so the premises can be seen as the *evidence* of the conclusion. The input *sentences* can be seen as a synthesis of virtual positive and negative evidences. Assume the available amount of positive evidence and negative evidence of a statement are written as w^+ and w^-, respectively, then the total amount of evidence is $w = w^+ + w^-$. The *frequency* of the statement is $f = w^+/w$, and the *confidence* of the statement is $c = w/(w + k)$, where k is a positive constant representing "evidential horizon". We take k = 1 in our implementation. *Frequency* intuitively means "the degree of truth" and *confidence* intuitively represents "the total amount of evidences". The more evidences that the statement have considered, the higher *confidence* value. The *truth-value* attached to the statement is the ordered pair $\langle f, c\rangle$ and it is often written right after the statement. *Expectation* of the *truth-value* is a combined measurement of f and c, defined as $expectation = f \times c$.

NAL uses *syllogistic* (rather than truth-functional) inference rules, that is, the two premises have to share at least one common *term*. Among them the *revision* rule merges evidences for the same statement collected from different sources together, so it can settle inconsistency among the system's *sentences*. It is very useful in our approach. The relevant rules with corresponding truth functions are all listed in Table 5. Note that the inference rules are not domain-specific. There are three extended boolean operators (Wang, 2013) in the calculation of truth functions:

$$\begin{cases} and(x_1, ..., x_n) = \prod_{i=1}^{n} x_i \\ or(x_1, ..., x_n) = 1 - \prod_{i=1}^{n}(1 - x_i) \\ not(x) = 1 - x \end{cases}, \text{ where}$$

$x_i \in [0, 1]$.

C Discussion

C.1 Problem of understanding literal value

Literal values in real-world KGs act as entity names, entity descriptions, relation/attribute names or attribute values, carrying enormous information. Literal values include texts (strings), numerical values and dates. Deep neural network language models provide an interim solution to the problem of understanding literal values. For example, BERT-INT (Tang et al., 2020) utilize BERT to embed names/descriptions and values into vector space, thus use similarities between the feature vectors for alignment. Literals' deficiency of its outer semantic structure (triples) contrasts with its abundant internal semantics. However, symbolic reasoning languages (systems) like NAL currently can't effectively handle the subtle semantics in texts for the following reasons: semantic parsing/understanding requires processing capacity and efficiency of complex logical forms and it also requires automatic learning capacity; the lacking of KGs with complex logical forms; the lacking of KGs with detailed and comprehensive common sense knowledge. In a certain perspective, the literal values in real-world KGs are not really "literal" but rather under-characterized entities, concepts, triples, common sense knowledge and/or statements with complex logical forms. The real-world KG project may not have enough information or adequate paradigm to deal with them. For example, the literal value of attribute triple (John Lennon, deathPlace, "Manhattan, New York City, United States"@en) referred to entities "Manhattan", "New York" and "United States", and its form indicates a specific relation between these places.

C.2 Understanding of type II path

Type II path seems straightforward, however we can have a deeper understanding of it. Language models used for the embedding process of EA are distinct information sources other than the KG itself. The deep language model which has the ability of aligning or translating entity names can be seen as a generalized alignment model that aligns morphemes, words, entities and concepts. The pretraining corpus of it consists of sentences, although the sentences do not possesses explicit structures, they can be understood/parsed by the model by transforming them into complex logical forms. However, such transformation (if exist) and the logical forms are implicitly expressed in the model parameters and intermediate layer vector representations. To summarize, our similarity inference's *type II path* can be seen as the aggregation of multiple virtual complex logical paths. The aggregation result is represented into the vector space by the language model.

C.3 1-to-1 Assumption

There are 1-to-1 assumptions in some EA datasets (such as $DBP15K$) and it is a useful information for alignment. Formally, we define the 1-to-1 assumption as follows: first, there is a range of alignable entities $A_1 \subset \mathcal{E}_1$ and $A_2 \subset \mathcal{E}_2$ (for DBP15K, $A_1 \subsetneqq \mathcal{E}_1$). Second, the equivalence between A_1 and A_2 is a bijection. Note that the assumption does not have aligning regularity for entities outside the range except that they can't be aligned with entities inside the range. Many ranking-based EA methods leverages the 1-to-1 range assumption, however, PARIS do not. Therefore, in implementation in order to leverage the range assumption we take the set A_1 and A_2 as input and filters out any alignment sentence that aligns A_1 to $\mathcal{E}_2 \setminus A_2$ or $\mathcal{E}_1 \setminus A_1$ to A_2.

C.4 Relation with Other Methods

In this section, we will discuss the relation between our proposed method and methods with other forms. We will propose some preliminary explanations of certain translational embedding methods and embedding-path EA methods from a theoretical perspective.

The way NAL models KG information and the inference process has a similar part with "uncertainty estimation" (Hu et al., 2023) in the natural language processing domain. The *truth-value* of alignments shares some similarity with the distributive view of facts or beliefs which views facts as probability distribution of random variables. Also, the concept of confidence is shared with some information extraction systems such as Markov logic network (Jiang et al., 2012), which assigns confidence to extracted facts or logical formulas in some intermediate steps.

C.5 Relation with Translational Embedding Methods

The well-known KG embedding model TransE (Bordes et al., 2013) is initially proposed for link prediction tasks. It may be partially explained from a logical perspective of NAL (or equivalently other logic with similar expressive power). Consider a specific type of Horn clauses $((*, A, B) \rightarrow R_1 \wedge (*, B, C) \rightarrow R_2) \Rightarrow (*, A, C) \rightarrow R_3 \langle f_1, c_1 \rangle$, the following three triples

$$(Martin\_Luther\_King\_Jr,\ birthPlace,\ Georgia\_(U.S.\_state))$$

$$(Georgia\_(U.S.\_state),\ country,\ United\_States)$$

$$(Martin\_Luther\_King\_Jr,\ citizenship,\ United\_States)$$

together forms a piece of positive evidence of an instantiated Horn clause, in which R_1, R_2 and R_3 is replaced by $birthPlace$, $country$ and $citizenship$ respectively. We conjecture that the gradient descent optimization process of TransE implicitly performs approximate logical inference and evidence aggregation. In the above example for each of the three triples, $||\mathbf{h}+\mathbf{r}-\mathbf{t}||$ (where bold format represent a vector) is minimized once per epoch (ignoring margin-based criterion), leading to **birthPlace** + **country** $\approx$ **citizenship**. Thus, the instantiated Horn clause together with its *truth-value* may be represented by the vector representations' correlation, and the *truth-value* may be reflected in distance $||$**birthPlace** + **country** − **citizenship**$||$. Note that these three relations may appear in more than one Horn clauses, so the gradients from the evidences of a Horn clause may confuse with (or conflict with) those from another Horn clause, for example **manufacturer** + **country** $\approx$ **madeInCountry**. The training process may force vector **birthPlace** to be nearly perpendicular with **manufacturer**, otherwise, there may be hallucination in link prediction or EA results. A similar explanation of hallucination may apply to LLMs. A similar analysis applies to the vector representations of two relations which frequently appear on the same head entity (or tail entity). It's arguable that the test set link prediction process of TransE mainly relies on Horn clauses, because from a logical perspective there is no other information. In this paper Horn clauses will not be extracted and managed, leaving for further research.

MTransE (Chen et al., 2016) is a translational embedding-based EA method. It encodes the two KGs' relational triples separately with the TransE loss criterion $S_K = \Sigma_{(h,r,t)}||\mathbf{h} + \mathbf{r} - \mathbf{t}||$. It proposed a "distance-based axis calibration" alignment model in order to coincide the vectors of counterpart entities/relations. The corresponding loss is $S_{a_2} = \Sigma||\mathbf{e}_1 - \mathbf{e}_2|| + ||\mathbf{r}_1 - \mathbf{r}_2||$ (S_{a_2} only has the first item if there is no available seed relation alignment). The seed and derived alignments are assumed to have $\mathbf{e}_1 \approx \mathbf{e}_2$ and we see it as the embedding representation of the similarity statement $e_1 \leftrightarrow e_2$, with its *truth-value* somehow represented by the distance $||\mathbf{e}_1 - \mathbf{e}_2||$. Theoretically, the distance can't simultaneously represent *frequency* and *confidence* by itself, but more possibly a combined effect. We argue that MTransE performs approximate inference that is similar with the type III path, because if the learned embedding constraints of the four premises are considered simultaneously, we can get $\mathbf{r}_1 \approx \mathbf{r}_2$ which we interpret as $r_1 \leftrightarrow r_2$. Similarly, MTransE performs approximate inference of the type I path (with functionality omitted and $r_1 \rightarrow r_2$ replaced by $r_1 \leftrightarrow r_2$) to obtain derived alignment results.

C.6 Relation with Embedding-path EA Methods

Here we propose some preliminary explanations of the similarity inference aspect of some embedding-path EA methods from a theoretical perspective.

The first method to be discussed is BERT-INT. It generates entity embedding using the name/description information with BERT unit and the embedding is $C(e) = MLP(CLS(e))$. It uses pairwise margin loss to approximately enforce $C(e) \approx C(e')$. Different from MTransE which performs path inference implicitly with the gradient optimization of loss criterions, BERT-INT explicitly performs path inference with its proposed interaction model. Every element of the neighbor-view interaction matrix represents a inference process of a type I path. Its path omits functionality and relation alignment (for BERT-INT fails to utilize its proposed relation mask matrix). Because of the ignorance of relation type, its premise (1) and (4) has the form of $(*, x_1, y_1) \rightarrow \#r$ and $(*, x_2, y_2) \rightarrow \#r$ which represents "There exists an unspecified relation between x_1/y_1, and (another) unspecified relation between x_2/y_2". Moreover, its premise (5) fails to utilize derived alignments, because BERT-INT is not iterative. With such premises, BERT-INT's type I path inference's effectiveness is supposed to be lower than that of NALA's. Similarly, every element of the attribute-view interaction matrix represents a type I path which has attribute triples as premises (1) and (4). BERT-INT's evidence aggregation method is different from NALA which uses *probabilistic revision* and *revision* rules.

The second method to be discussed is FGWEA. Its multi-view Optimal Transport (OT) alignment step combines four cost matrices for the OT problem, that is, $C_{sum} = C_{stru} + C_{rel} + C_{name} + C_{attr}$. Obtaining the cost matrices corresponds to the similarity inference process and different matrices correspond to different groups of inference paths. Among them, C_{rel} corresponds to a degenerated type I path inference where relation alignment is obtained by relation names and without the consideration of functionality. C_{stru} corresponds to a further degenerated type I path inference (similar with BERT-INT's neighbor-view interaction). C_{name} corresponds to type II path inference. C_{attr} fails to model the (fine-grained) attributive type I path because it uses the concatenation of all attribute triples of an entity.

In this paper, BERT-INT and FGWEA are classified as embedding-path EA methods because their embedding module couples with the path inference to some extent. In contrast, NALA, which we classify as path-based, performs inference wherever it can and uses embeddings minimally.

D Experiments

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}_{\mathcal{R}}\rvert$	$\lvert\mathcal{T}_{\mathcal{A}}\rvert$
$DBP15K_{ZH\_EN}$	19,388	1,701	70,414	379,684
	19,572	1,323	95,142	567,755
$DBP15K_{JA\_EN}$	19,814	1,299	77,214	354,619
	19,780	1,153	93,484	497,230
$DBP15K_{FR\_EN}$	19,661	903	105,998	528,665
	19,993	1,208	115,722	576,543
D-W-15K-V2	15,000	167	73,983	66,813
	15,000	121	83,365	175,686
D-Y-15K-V2	15,000	72	68,063	65,100
	15,000	21	60,970	131,151
D-Y-100K-V2	100,000	230	576,547	547,026
	100,000	31	865,265	855,161

Table 3: Dataset statistics. $\lvert\mathcal{E}\rvert$, $\lvert\mathcal{R}\rvert$, $\lvert\mathcal{T}_{\mathcal{R}}\rvert$ and $\lvert\mathcal{T}_{\mathcal{A}}\rvert$ represent the number of entities, relation types, relation triples and attribute triples in each KG, respectively.

D.1 Evaluation Metric& Environment

We use Hits@1 (which is the same metric as recall for EA) as the sole evaluation metric of our main results of DBP15K for the following reasons. Mean Reciprocal Rank (MRR) is unavailable for NALA because it does not provide a alignment ranking for the test entities. There exist a non-negligible number of equivalent entity pairs that are not in the ground-truth of DBP15K, so the precision and F1-score can't be measured properly. We use the precision (P), recall (R), and F1 score for OpenEA benchmark datasets.

Our NALA model is implemented in java and the BERT unit is implemented in python with PyTorch. All experiments are performed on a Linux server with an Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz, 251G RAM and a NVIDIA GeForce RTX 3090 GPU.

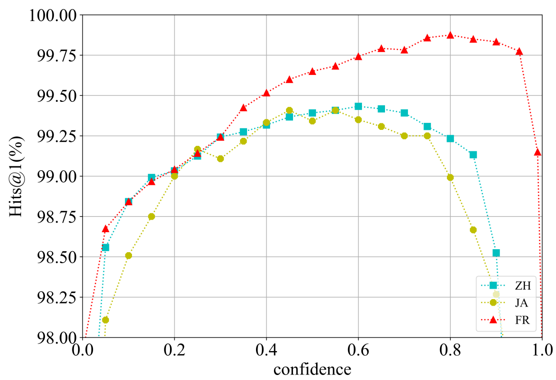

Figure 4: Influence of C_{name}.

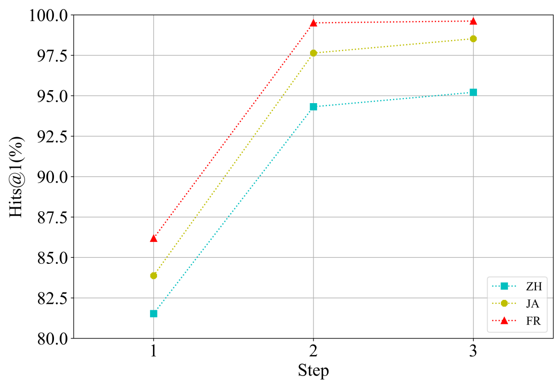

Figure 5: Results of bootstrap steps of setting group 3.

D.2 Influence of Confidence Hyper-parameter

The experiment results of Figure 4 shows how entity name/description embedding similarity confidence C_{name} (without the adaptive setting of C_{name}) affects Hits@1. These experiments are performed on setting group 4 without using attribute value embedding information. We adjust C_{name} with other conditions unchanged. The

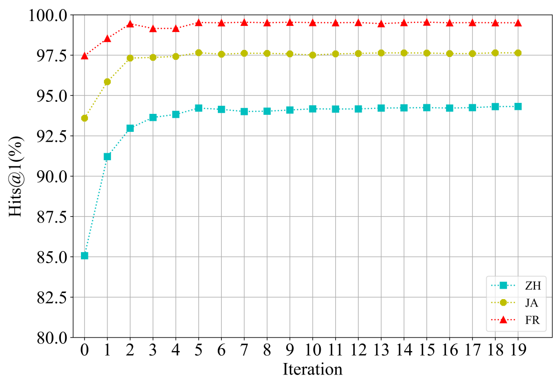

Figure 6: Results of the iterations of the second bootstrap step in setting group 3.

Hits@1 curve is approximately concave and for ZH_EN, JA_EN and FR_EN respectively, it reaches maximum performance at 0.6, 0.55 and 0.8. It shows that the informative embedding similarity enhances the performance to different extents. French is often regarded as more closely related to English than Chinese or Japanese, so the BERT unit learns representation easier and thus produces more confident embedding similarity. Pretraining corpus of the BERT unit may include relevant triples (in the form of natural language sentences) which may have same informational origin with DBpedia. So the embedding similarity's evidences may have an overlap part with *type I path*'s evidences. The *revision* rule is only appropriately used when the two premises don't share same evidence (or equivalently their evidential bases do not overlap). So the appropriate confidence value need to be lower than the confidence of the BERT output (if it provides such information) in order to exclude the overlap. The best-performance confidence of each dataset is conjectured to reflect the combined influence of embedding quality of the BERT unit and the evidence overlapping effect. The C_{name} confidence value can be alternatively set equal to the cosine similarity of the embeddings, resulting in a slightly decreased performance. This is a good choice if you want to avoid hyper-parameter tuning.

E Algorithms

Algorithm 1: recursive bidirectional matching

input : An array of linked list of *similarity sentences* $KG1\_to\_KG2$, with each linked list storing top-k *similarity sentences* of an entity with descending order.

output : Optimized 1-to-1 *similarity sentences* (alignment results)

populates $KG2\_to\_KG1$ *with all of the sentences in* $KG1\_to\_KG2$;

/* $KG2\_to\_KG1$ is another array of linked list, arranging the similarity sentences in the other direction */

for e_1 ***in*** $\mathcal{E}_1$ **do**

  recursively_delete($e_1, null$);

F Inference Paths

The inference paths are formally represented in a form similar to natural deduction (Pelletier and Hazen, 2021). Each step of inference is characterized by two premises (on the top of the inference line) and a conclusion (on the bottom of the inference line). The inference rule is indicated on the right edge of the inference line.

$(type\ I\ path):$

$$\frac{(*, x_1, y_1) \to r_1 \quad (1),\ r_1 \to r_2 \quad (2)}{(*, x_1, y_1) \to r_2 \quad (3)} Deduction$$

$$\frac{(*, x_2, y_2) \to r_2 \quad (4),\ x_1 \leftrightarrow x_2 \quad (5)}{(*, x_1, y_2) \to r_2 \quad (6)} Analogy*$$

$$\frac{((*, \#a, \$b) \to \#r\ \wedge\ (*, \#a, \$c) \to \#r\ \wedge\ \#r \to [fun])\ \Rightarrow \$b \leftrightarrow \$c \quad (7),\ (*, x_1, y_1) \to r_2 \quad (3)}{((*, x_1, \$c) \to r_2\ \wedge\ r_2 \to [fun])\ \Rightarrow y_1 \leftrightarrow \$c \quad (8)} Conditional\ deduction$$

$$\frac{((*, x_1, \$c) \to r_2\ \wedge\ r_2 \to [fun])\ \Rightarrow y_1 \leftrightarrow \$c \quad (8),\ (*, x_1, y_2) \to r_2 \quad (6)}{r_2 \to [fun]\ \Rightarrow y_1 \leftrightarrow y_2 \quad (9)} Conditional\ deduction$$

$$\frac{r_2 \to [fun]\ \Rightarrow y_1 \leftrightarrow y_2 \quad (9),\ r_2 \to [fun] \quad (10)}{y_1 \leftrightarrow y_2 \quad (11)} Conditional\ deduction$$

In the path listed above, we omit two auxiliary inference steps right before arriving at conclusion (6) which performs *structural transformation* in order to dismount x_2 from the *product* of (4) without modifying its *truth-value*. The last conditional deduction of (11) degenerates into a case without conjunction in its premises (similar with Modus Ponens) and its *truth function* remains the same. Note that in the path listed above only one direction of the relational inheritance is considered ($r_1 \to r_2$) and there exists a symmetrical variation of the path that utilizes the other direction ($r_2 \to r_1$). The conclusions of the two paths are aggregated by *probabilistic revision* rule.

Statement (11) is the conclusion of the above inference steps and the whole steps act as a summarizing or validation process of the *type I path*. Implication statement (8) is regarded as a definition or a piece of *essence* of the concept "functionality". Relations' functionality seems to reflect a widespread orderliness of reality or human cognition and PARIS leverages such orderliness.

$(type\ III\ path):$

$$\frac{(*, x_1, y_1) \to r_1 \quad (12),\ x_1 \leftrightarrow x_2 \quad (13)}{(*, x_2, y_1) \to r_1 \quad (14)} Analogy$$

$$\frac{(*, x_2, y_1) \to r_1 \quad (14),\ y_1 \leftrightarrow y_2 \quad (15)}{(*, x_2, y_2) \to r_1 \quad (16)} Analogy$$

$$\frac{(*, x_2, y_2) \to r_2 \quad (17),\ (*, x_2, y_2) \to r_1 \quad (16)}{r_1 \to r_2 \quad (18)} Induction$$

The only difference between the two versions of type III path is the *truth-value* of premise (17) ($(*, x_2, y_2) \to r_2$). The positive version's *truth-value* is $\langle 1,\ 1\rangle$ and the negative version's is $\langle 0,\ C_{absent}\rangle$, where C_{absent} is a hyper-parameter for absent/missing fact. We argue that when there is a fact present in the KG, it is usually confident. However, when there is an absent fact in the KG, its denial is not as confident because the KG may be incomplete. In implementation we set $C_{absent} = 0.5$ (which represents a unit amount of evidence).

Note that in *type III path* the *induction* inference rule is a *weak inference rule*, so the upper bound of its conclusion's confidence is lower than the *strong inference rules* (such as *deduction* and *analogy*). The positive version only generates positive evidence for the conclusion and the negative version only generates negative evidence, because of the characteristic of *induction* rule.

Algorithm 2: recursively delete

input : Entity e_1, entity e_{prev}.

/* e_1 is the entity to be processed and we assume that e_1 belongs to the left graph, similarly otherwise. Entity e_{prev} represents the previous entity, that is the processed entity of the recursion parent. */

output : entity e_{return} which represents the final alignment for e_1

for $sentence$ ***in*** $KG1\_to\_KG2(e_1)$ **do**
  $e_2 \leftarrow predicate\_term$ of $sentence$;
  /* $predicate\_term$ means the other entity of the similarity sentence */
  if $e_2 == e_{prev}$ **then**
    $e_{return} \leftarrow e_{prev}$;
    break;
  else
    $e_3 \leftarrow$ recursively_delete(e_2, e_1);
    if $e_3 == e_1$ **then**
      $e_{return} \leftarrow e_2$;
      break;
for $sentence$ ***in*** $KG1\_to\_KG2(e_1)$ *except the first node* **do**
  /* now that the first sentence for e_1 is bidirectionally matched, we delete other sentences */
  removes $sentence$ *from the linked list*;
  removes $sentence$*'s counterpart in* $KG2\_to\_KG1$ *which expresses the same similarity in the other direction*;
return e_{return};

Algorithm 3: NALA(supervised)

input : Two knowledge graphs $\mathcal{KG}_1$ and $\mathcal{KG}_2$.

output : Alignment result and other information.

run finetuning for BERT unit;
compute entity/value embeddings with the BERT unit;
generate synthetic attribute triples for seed alignments (for supervision);
load the knowledge graphs;
for $iteration \leftarrow 0$ **to** $end\_iteration$ **do**
  for y_1 ***in*** $\mathcal{E}_1$ **do**
    /* aligning for different entities of $\mathcal{E}_1$ is divided into multiple parallel threads */
    for x_1, x_2, y_2 *that forms a sound type I path with* y_1 *(depth-first)* **do**
      perform inference of type I path;
      perform inference of type III path;
    for y_2 ***in*** $\mathcal{E}_2$ **do**
      retrieve embedding similarity for $y_1 \leftrightarrow y_2$;
      perform inference of type II path;
    filter the similarity sentences with 1-to-1 range assumption;
    insert the sentences into a top-k ordered linked list;
  perform recursive bidirectional matching;
  swapping;
  save alignment results and evidence log file;

Model	ZH_EN	JA_EN	D-W-15K-V2			D-Y-15K-V2			D-Y-100K-V2		
	Hits@1	Hits@1	P	R	F_1	P	R	F_1	P	R	F_1
NALA	**0.993**	**0.988**	0.917	0.908	0.912	**0.983**	**0.981**	**0.982**	**0.985**	**0.980**	**0.983**
- w/o E_{value}	0.980	0.980	-	-	-	-	-	-	-	-	-
- $all\_revision$	0.964	0.912	0.857	0.814	0.835	0.899	0.871	0.885	0.402	0.312	0.351
- $all\_prob\_revision$	0.985	0.987	-	-	-	-	-	-	-	-	-
- w/o $1v1\_range$	0.989	0.978	-	-	-	-	-	-	-	-	-
- w/o $swapping$	0.991	0.982	0.912	0.901	0.907	0.975	0.972	0.973	0.981	0.976	0.978
FGWEA	0.976	0.978	**0.952**	0.903	**0.927**	-	-	-	-	-	-
LightEA	0.812	0.821	-	**0.951**	-	-	0.976	-	-	0.977	-

Table 4: Ablation study of NALA.

Inference rule	Premises	Conclusion
Deduction	$A \to B\ \langle f_1, c_1\rangle$ $B \to C\ \langle f_2, c_2\rangle$	$A \to C\ \langle f = and(f_1, f_2), c = and(f_1, f_2, c_1, c_2)\rangle$
Analogy	$A \to B\ \langle f_1, c_1\rangle$ $A \leftrightarrow C\ \langle f_2, c_2\rangle$	$C \to B\ \langle f = and(f_1, f_2), c = and(f_2, c_1, c_2)\rangle$
Conditional Deduction	$(P \wedge Q) \Rightarrow R\,\langle f_1, c_1\rangle$ $Q\ \langle f_2, c_2\rangle$	$P \Rightarrow R\ \langle f = and(f_1, f_2), c = and(f_1, f_2, c_1, c_2)\rangle$
Induction	$A \to B\ \langle f_1, c_1\rangle$ $A \to C\ \langle f_2, c_2\rangle$	$C \to B\,\langle w^+ = and(f_2, c_2, f_1, c_1), w^- = and(f_2, c_2, not(f_1), c_1)\rangle$
Revision	$P\ \langle f_1, c_1\rangle$ $P\ \langle f_2, c_2\rangle$	$P\ \langle w^+ = w_1^+ + w_2^+, w = w_1 + w_2\rangle$
Probabilistic Revision	$P\ \langle f_1, c_1\rangle$ $P\ \langle f_2, c_2\rangle$	$P\ \langle f = or(f_1, f_2), w = w_1 + w_2\rangle$

Table 5: The table of relevant truth functions.