Facing Changes: Continual Entity Alignment for Growing Knowledge Graphs

Yuxin Wang[1], Yuanning Cui[1], Wenqiang Liu[3], Zequn Sun[1], Yiqiao Jiang[3], Kexin Han[3], and Wei Hu[1,2(✉)]

[1] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
whu@nju.edu.cn,yuxinwangcs@outlook.com,yncui.nju@gmail.com
[2] National Institute of Healthcare Data Science, Nanjing University, Nanjing, China
[3] Interactive Entertainment Group, Tencent Inc, Shenzhen, China
{masonqliu,gennyjiang,casseyhan}@tencent.com

Abstract. Entity alignment is a basic and vital technique in knowledge graph (KG) integration. Over the years, research on entity alignment has resided on the assumption that KGs are static, which neglects the nature of growth of real-world KGs. As KGs grow, previous alignment results face the need to be revisited while new entity alignment waits to be discovered. In this paper, we propose and dive into a realistic yet unexplored setting, referred to as continual entity alignment. To avoid retraining an entire model on the whole KGs whenever new entities and triples come, we present a continual alignment method for this task. It reconstructs an entity's representation based on entity adjacency, enabling it to generate embeddings for new entities quickly and inductively using their existing neighbors. It selects and replays partial pre-aligned entity pairs to train only parts of KGs while extracting trustworthy alignment for knowledge augmentation. As growing KGs inevitably contain non-matchable entities, different from previous works, the proposed method employs bidirectional nearest neighbor matching to find new entity alignment and update old alignment. Furthermore, we also construct new datasets by simulating the growth of multilingual DBpedia. Extensive experiments demonstrate that our continual alignment method is more effective than baselines based on retraining or inductive learning.

Keywords: Knowledge graphs · Continual entity alignment · Representation learning

1 Introduction

Entity alignment, also known as entity matching or entity resolution [22], has been a long-standing research topic in the Semantic Web and Database communities. The task aims at matching the identical entities with different URIs in different

U. Sattler et al. (Eds.): ISWC 2022, LNCS 13489, pp. 196–213, 2022.
https://doi.org/10.1007/978-3-031-19433-7_12

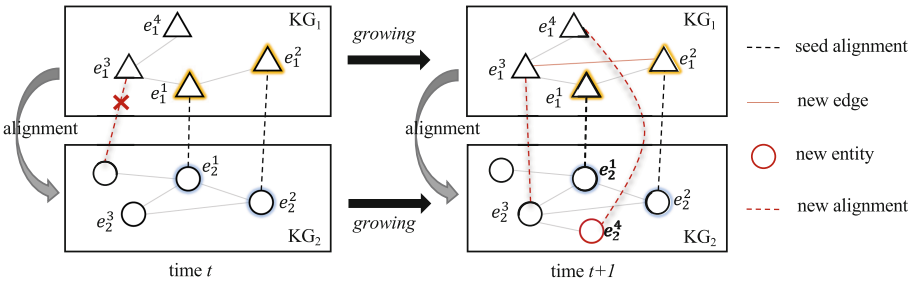

Fig. 1. Illustration of continual entity alignment. Given two pre-aligned entity pairs (e_1^1, e_2^1) and (e_1^2, e_2^2) between KG_1 and KG_2, we expect to find the identical counterparts for e_1^3 and e_1^4. At time t, due to the incompleteness of both KGs, e_1^3 can be falsely matched to a wrong entity, and the expecting counterpart of e_1^4 does not even appear yet. At time $t+1$, as new triples emerge over time, e_1^3 and e_1^4 gain more chance to be correctly matched with richer supportive information.

knowledge graphs (KGs). For example, two entities http://dbpedia.org/resource/Hangzhou and http://zh.dbpedia.org/resource/杭州 from DBpedia [13] in different languages both refer to the same Chinese city, Hangzhou, which is the venue of ISWC 2022 conference. Early studies [11,22] mainly explore the literal similarities with probabilistic or semantic inference to match entities. However, these methods are hampered by the symbolic heterogeneity of different KGs, particularly the cross-lingual KGs. To resolve this issue, recent embedding-based methods strive to construct a unified vector space to represent different KGs, with entity embeddings used to infer entity similarity [24]. Furthermore, the embeddings from the unified space built by aligning various KGs are shown to be useful for downstream tasks, such as cross-lingual knowledge transfer and multi-lingual KG completion [7,20]. Thus, as a backbone of knowledge fusion and transfer, embedding-based entity alignment has received increasing attention [28,41,42].

However, existing embedding-based entity alignment methods assume an idealized scenario of static KGs, neglecting many real-world difficulties like alignment incompleteness, KG growth, and alignment growth. In this paper, we argue that entity alignment is not a one-time task. We propose and study a new setting, i.e., *continual entity alignment*, between growing and incomplete KGs. Our motivation comes from the growth and incompleteness nature of real-world KGs. For example, the release bot of DBpedia [13] extracts about 21 billion new triples per month [10], and Wikidata [30] releases data dumps in a weekly cycle.[1] The new entities and triples bring about new alignment to be found and provide new clues for correcting the previous alignment. Figure 1 presents an illustration.

This real scenario poses new challenges to embedding-based entity alignment. The first challenge is *how to learn embeddings for the new entities in an effective and efficient manner*. When KGs grow, the pre-trained entity alignment model sees new entities for the first time, as new triples bring structural changes to KGs.

[1] https://dumps.wikimedia.org/wikidatawiki/entities/.

To handle new entities, retraining the model from scratch is costly. Also, inductive entity embedding is less adaptable to changes of structure. Thus, it requires non-trivial updates of the pre-trained model to incorporate new entities and new triples. The second challenge is *how to capture the potential alignment of both old and new entities*. In real cases, KGs always contain unknown non-matchable entities [23], which necessitates a more reliable alignment retrieval strategy than simply ranking candidates from test sets. Furthermore, as new entities typically are few-linked [2], capturing the potential alignment for new entities becomes more difficult. The third challenge is *how to integrate the old predicted alignment with the new predictions*. In our setting, we output alignment results each time the KGs grow. The old and new alignment inevitably have conflicts. We need an effective integration strategy to combine them and update the final alignment.

As the first attempt to address these challenges, we propose a continual entity alignment method **ContEA**. Our key idea is to finetune the pre-trained model to incorporate new entities and triples, meanwhile capturing the potential entity alignment. Specifically, we use Dual-AMN [16], a prominent alignment model, as our basal encoder. To enable it to effectively handle new entities, we design an entity reconstruction objective, which allows the encoder to generate entity embeddings using solely neighboring subgraphs. To retrieve alignment from the embedding space, we propose a bidirectional nearest neighbor search strategy. Two entities are predicted to be aligned if and only if they are the nearest neighbors to each other. When new entities and triples emerge, ContEA finetunes the pre-trained model according to the changed structures. To capture potential entity alignment, we replay partial pre-known alignment to avoid knowledge oblivion and select high-confidence predictions for knowledge augmentation.

To support the research on this new and practical task, we build three new datasets based on the widely-used benchmark DBP15K [24], which contains three cross-lingual datasets, i.e., ZH-EN, JA-EN and FR-EN. For each dataset, we construct six snapshots (i.e., $t = 0, 1, 2, 3, 4, 5$) by adding new entities and new triples into the preceding snapshot, to simulate KGs' growth. We conduct extensive experiments on our datasets. Our method outperforms strong baselines that use retraining or inductive embedding techniques while at a lower time cost. Our datasets and source code are publicly available to foster future research.

2 Problem Statement

We define a KG as a 3-tuple $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the sets of entities and relations, respectively. $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of relational triples. Given two KGs $\mathcal{G}_1 = \{\mathcal{E}_1, \mathcal{R}_1, \mathcal{T}_1\}$ and $\mathcal{G}_2 = \{\mathcal{E}_2, \mathcal{R}_2, \mathcal{T}_2\}$, *entity alignment* aims to identify entities in $\mathcal{G}_1$ and $\mathcal{G}_2$ that refer to the same real-world object, i.e., seeking a set of alignment $\mathcal{A} = \{(e_1, e_2) \in \mathcal{E}_s \times \mathcal{E}_t \mid e_1 \equiv e_2\}$, where "$\equiv$" indicates equivalence. A small set of seed entity alignment $\mathcal{A}_s \subset \mathcal{A}$ is usually provided as anchors (i.e., training data) beforehand to help align the remaining entities.

From time to time, new triples emerge and are added into KGs, which brings KGs' size growth. We propose the definition of *growing KGs* as follows:

Definition 1 (Growing knowledge graphs). *A growing KG $\mathcal{G}$ is a sequence of snapshots $\mathcal{G} = (\mathcal{G}^0, \mathcal{G}^1, \ldots, \mathcal{G}^T)$, where the superscript numbers denote different timestamps. For any two successive timestamps $\mathcal{G}^t = \{\mathcal{E}^t, \mathcal{R}^t, \mathcal{T}^t\}$ and $\mathcal{G}^{t+1} = \{\mathcal{E}^{t+1}, \mathcal{R}^{t+1}, \mathcal{T}^{t+1}\}$, there exist $\mathcal{E}^t \subseteq \mathcal{E}^{t+1}$, $\mathcal{R}^t = \mathcal{R}^{t+1}$ and $\mathcal{T}^t \subseteq \mathcal{T}^{t+1}$.*

In this definition, each newly added triple in $\Delta\mathcal{T}^{t+1}$ between t and $t+1$ contains zero, one, or two new entities. Considering that the set of relations in KGs is much less diverse than that of entities, we dismiss the emergence of new relations in this paper and assume that the relations in KGs are pre-defined.

To practice entity alignment on growing KGs. We propose the task of *continual entity alignment* and give its definition below:

Definition 2 (Continual entity alignment). *Given two growing KGs $\mathcal{G}_1$ and $\mathcal{G}_2$, and the seed entity alignment $\mathcal{A}_s$ at time $t = 0$, continual entity alignment at time t aims to find potential entity alignment $\mathcal{A}_p^t$ between $\mathcal{G}_1^t$ and $\mathcal{G}_2^t$ based on the currently learned KG embeddings and alignment model.*

In this definition, the size of $\mathcal{A}_\mathrm{s}$ is constant, while $\mathcal{A}_\mathrm{p}^t$ grows over time as new entities may bring new entity alignment to be found. Considering that the seed entity alignment is usually deficient and difficult to obtain [21], we do not assume that new snapshots bring new seed alignment to augment training data. That is to say, $\mathcal{A}_\mathrm{s}$ of snapshot at time $t > 0$ is the same as that at time $t = 0$.

3 Methodology

In this section, we introduce the proposed continual entity alignment method ContEA. Figure 2 depicts its framework. It consists of two modules: the subgraph-based entity alignment module, and the embedding and alignment update module. The following is a brief overview of them:

- In the subgraph-based entity alignment module, the input is the two KGs at time $t = 0$ and the seed entity alignment across them. A graph neural network (GNN) is employed over the two KGs to represent entities based on their subgraph structures. The alignment learning objective is to minimize the embedding distance of similar entities while separating dissimilar ones. Additionally, an entity reconstruction design is used to encourage entities similar to their contexts. When the learning process is completed, the trustworthy alignment is predicted based on bidirectional nearest neighbor search.
- At time $t > 0$, the embedding and alignment update module first incorporates new entities into previously learned KG embeddings. It reconstructs new entities' embeddings based on their neighborhood subgraphs. Then, partial seed entity alignment and trustworthy alignment predicted in the previous snapshot are used for finetuning the GNN model. Last, after new alignment is predicted, we use it to update the previously-found old alignment.

We introduce the details of the two modules in the following two subsections.

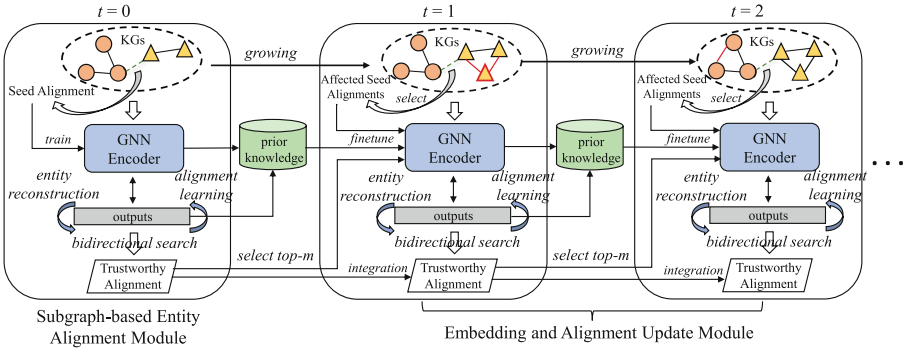

Fig. 2. Framework of the proposed continual entity alignment method ContEA.

3.1 Subgraph-Based Entity Alignment

This module is built upon a GNN that represents an entity by aggregating its neighborhood subgraph. The key assumption behind GNN is that the entities with similar neighborhoods appear to be close, which makes GNNs extensible to represent new entities. Please note that we do not focus on how to develop a powerful GNN for entity alignment, but on how to incorporate new entities and triples in an effective and efficient manner for continual entity alignment.

Subgraph Encoder. We adopt the GNN-based encoder of Dual-AMN [16] as our subgraph encoder for its effectiveness and simplicity. The encoder of Dual-AMN consists of an inner-graph layer (namely $\texttt{Aggregator}_1$) capturing the structural information within a single KG, and a cross-graph layer ($\texttt{Aggregator}_2$) capturing cross-graph matching information based on the outputs of $\texttt{Aggregator}_1$. Technically, $\texttt{Aggregator}_1$ is a 2-layered relation-aware GNN, and $\texttt{Aggregator}_2$ is a proxy attention network connecting entities with a list of proxy nodes. Overall, given an entity e, its representation after being encoded by Dual-AMN is

$$\texttt{Encoder}(e) = \texttt{Aggregator}_2\big(\texttt{Aggregator}_1(e, \mathcal{N}_e), \mathcal{E}_{\text{proxy}}\big), \tag{1}$$

where $\texttt{Aggregator}_1()$ aggregates the entity itself and its relational neighbors $\mathcal{N}_e$ to generate its embedding, and $\texttt{Aggregator}_2()$ combines the output embeddings with proxy nodes $\mathcal{E}_{\text{proxy}}$ to generate the final representations of entities. To save space, we do not present the detailed techniques of Dual-AMN here. Interested readers can refer to its original paper [16] for more details.

Entity Reconstruction. As KGs grow, the pre-trained GNN encoder encounters new entities and triples. The critical challenge is how to incorporate unseen entities into the encoder. Randomly initializing the embeddings of new entities could be detrimental to the previously optimized embedding space and cause representation inconsistency. A typical assumption in embedding-based entity

alignment is that two entities are similar if their neighborhood subgraphs are similar (i.e., the two subgraphs have similar or pre-aligned entities). Motivated by this, we propose a self-supervised learning objective that enables the encoder to reconstruct an entity using its neighborhood subgraphs:

$$\mathcal{L}_{\text{reconstruct}} = \sum_{e \in \mathcal{E}} \left\| \mathbf{e} - \frac{1}{|\mathcal{N}_e|} \sum_{e' \in \mathcal{N}_e} \mathbf{e}' \right\|_2^2. \tag{2}$$

Here, $\mathcal{N}_e$ denotes the set of one-hop neighbors of e. This objective minimizes the distance between an entity and its neighbor subgraph embedding (the mean vector of all neighbor embeddings).

Alignment Learning. Given the outputs of the encoder, alignment learning aims to gather similar entity pairs and distance dissimilar entity pairs. The dissimilar entity pairs is modeled by negative sampling. Following Dual-AMN [16], we also adopt the `LogSumExp` function to compute the loss:

$$\mathcal{L}_{\text{align}} = \log \left[1 + \sum_{(e_1,e_2) \in \mathcal{A}_s} \sum_{(e_1,e_2') \in \mathcal{A}_{e_1}^{\text{neg}}} \exp\left(\gamma(\lambda + \texttt{sim}(e_1,e_2) - \texttt{sim}(e_1,e_2'))\right) \right], \tag{3}$$

where $\mathcal{A}_{e_1}^{\text{neg}}$ denotes the negative alignment generated for entity e_1. γ is a scale factor, and λ is the margin for separating the similarities of seed alignment pairs and negative pairs. `Cosine` is used to compute embedding similarity, i.e., $\texttt{sim}(e_1, e_2) = \cos(\texttt{Encoder}(e_1), \texttt{Encoder}(e_2))$. We employ the in-batch negative generating method. Specifically, for entity e_1, other entities (e.g., e_2') in a training batch act as its negative counterparts to generate the negative pairs $\mathcal{A}_{e_1}^{\text{neg}}$. The final learning objective of subgraph-based entity alignment module $\mathcal{L}_1$ is a combination of $\mathcal{L}_{\text{align}}$ and $\mathcal{L}_{\text{reconstruct}}$ with a weight α on $\mathcal{L}_{\text{reconstruct}}$:

$$\mathcal{L}_1 = \mathcal{L}_{\text{align}} + \alpha \cdot \mathcal{L}_{\text{reconstruct}}. \tag{4}$$

Trustworthy Alignment Search. After the alignment learning is complete, we retrieve trustworthy entity alignment as predictions based on the optimized embedding space. Previous embedding-based entity alignment methods assume that each entity in one KG must have a counterpart in the other KG. A typical inference process is the nearest neighbor search, i.e., it seeks

$$\hat{e}_2 = \arg\min_{e_2 \in \mathcal{E}_2} \pi(\texttt{Encoder}(e_1), \texttt{Encoder}(e_2)), \tag{5}$$

where $\pi()$ is a measure for alignment search, and $\hat{e}_2$ is the predicted counterpart for e_1. However, such an "idealized" assumption may not stand in a realistic setting as there are many no-match entities in the two KGs [23]. To resolve this issue and improve alignment search, we propose a parameter-free strategy called bidirectional nearest alignment search. It searches for the nearest neighbor in one KG for the entities in the other. An alignment pair (e_1, e_2) is a trustworthy alignment if and only if $e_2 = \hat{e}_2$ and $e_1 = \hat{e}_1$. Other alignment pairs are discarded.

3.2 Embedding and Alignment Update

At time $t > 0$, the relational structure of KGs get changed as new triples come. It needs to generate embeddings for new entities while capturing the structure changes. To resolve this challenge, we propose to *finetune* the GNN encoder and new entity embeddings with partial seed alignment and selected trustworthy alignment. After finetuning, the new trustworthy entity alignment is retrieved based on the updated model and embeddings. The new predicted alignment is used to complete and update the old alignment discovered at time $t-1$ using a heuristic strategy.

Encoder Finetuning. We initialize the encoder with the parameters learned in the previous module/time. Thanks to our entity reconstruction objective, the encoder is able to initialize the embedding of a new entity e as follows:

$$\texttt{Encoder}(e) = \texttt{Aggregator}_2\big(\texttt{Aggregator}_1(\texttt{MP}(\mathcal{N}'_e)), \mathcal{E}_{\text{proxy}}\big), \tag{6}$$

where $\mathcal{N}'_e$ denotes the seen neighbors of the new entity e. $\texttt{MP}()$ is mean-pooling process to generate embedding for e using $\mathcal{N}'_e$.

Based on the output embeddings of new and existing entities, we finetune the GNN encoder. Specifically, we freeze the inner-graph layer $\texttt{Aggregator}_1$ while make the cross-graph $\texttt{Aggregator}_2$ learnable. For a single KG, the coming of new data does not change the neighbor aggregation pattern, as a KG's schema stays consistent (no new relations or entity domains). But the two KGs grow independently and asymmetrically in the proposed scenario. It is necessary to fine-tune the matching network to make adjustments and new discoveries.

For training data, considering that the potential entity alignment is more likely to occur near anchors [37], we replay only the affected seed entity alignment that contains anchors involved in new triples. This helps the alignment of new entities, which is originally difficult due to their low degrees. Also, to help align entities from wider and more dynamic areas, we select top-m predicted trustworthy alignment with the highest similarity scores and treat them as "new anchors".

We finetune the GNN encoder and new entity embeddings on the obtained affected seed alignment (ASA for short) and m selected trustworthy alignment (TA for short). We use a weight β on the learning loss over m trustworthy alignment to balance its importance. The final loss function $\mathcal{L}_2$ of finetuning is

$$\mathcal{L}_2 = \mathcal{L}_{\text{align}}(ASA) + \alpha \cdot \mathcal{L}_{\text{reconstruct}} + \beta \cdot \mathcal{L}_{\text{align}}(TA). \tag{7}$$

Trustworthy Alignment Update. After finetuning, a new set of trustworthy alignment can be retrieved using the updated entity embeddings and model. It is necessary to combine it with the previously discovered trustworthy alignment because they are gathered from different snapshots and may complement each other to produce superior outcomes. Here, we carry out a heuristic strategy to integrate them. We keep new trustworthy alignment which is between two new entities. But for new ones that cause alignment conflicts [25] with the previous

Algorithm 1: Process of ContEA

Input : Two growing KGs G_1^t and G_2^t at time t, prior learned knowledge $\mathbf{K}$ (none for $t = 0$), seed alignment $\mathcal{A}_s$, previous trustworthy alignment TA (none for $t = 0$), hyperparameters α, β;
Output: Updated trustworthy alignment TA;
1 **if** $t = 0$ **then**
2 Training encoder on G_1^0 and G_2^0 using $\mathcal{L}_1$ loss in Eq. (4);
3 Generating TA though trustworthy alignment search;
4 **else**
5 Initializing embeddings and encoder parameters using $\mathbf{K}$, G_1^t and G_2^t;
6 Selecting affected $\mathcal{A}_s$ as ASA and top-m TA with highest similarity;
7 Finetuning encoder using $\mathcal{L}_2$ loss in Eq. (7);
8 Updating TA with new trustworthy alignment;

trustworthy alignment (i.e., an entity is aligned with different entities), we decide to keep the alignment that has higher similarity scores. With KGs growing, the size of trustworthy entity alignment is accumulative.

3.3 Put It All Together

Algorithm 1 describes the training and finetuning details of ContEA for continual entity alignment. Lines 1–3 describe the process of the subgraph-based entity alignment module at time $t = 0$. Lines 4–8 describe the process of embedding and alignment updating modules at time $t > 0$.

4 Experiments

4.1 New Datasets for Continual Entity Alignment

Due to the lack of off-the-shelf benchmarks for proposed setting, we construct new datasets based on DBP15K [24]. For each DBP15K's cross-lingual entity alignment dataset, we use its two KGs as the first snapshots (i.e., $t = 0$). DBP15K only considers entity alignment between the head entities of triples and overlooks other entity alignment pairs. Hence, we first complete the reference entity alignment using the inter-language links in DBpedia[2], resulting in more than 15K reference alignment pairs in the first snapshot. Then, the reference entity alignment is divided into training, validation and test sets (i.e., $\mathcal{A}_s$, $\mathcal{A}_v$ and $\mathcal{A}_p^0$) with a ratio of 2 : 1 : 7. We further build five snapshots to simulate KGs' growth:

– At time $t > 0$, we first collect the relation triples from DBpedia that contain entities in $\mathcal{G}_1^{t-1}$ and $\mathcal{G}_2^{t-1}$. Then, among these triples we remove seen ones at time $t-1$, and sample new triples from the remaining with the size of 20% of the triples in previous snapshots. Adding the new triples into $\mathcal{G}_1^{t-1}$ and $\mathcal{G}_2^{t-1}$ and we create snapshots $\mathcal{G}_1^t$ and $\mathcal{G}_2^t$.

[2] We use the infobox-based relation triples (version 2016-10) following DBP15K.

Table 1. Statistics of the three datasets. Each consists of two growing KGs in six snapshots from consecutive timestamps. In a snapshot, $|\mathcal{T}|$ is the current triple size, and $|\mathcal{A}_s|, |\mathcal{A}_v|, |\mathcal{A}_p|$ are the sizes of training, validation and test alignment, respectively.

	$\text{DBP}_{\text{ZH-EN}}$					$\text{DBP}_{\text{JA-EN}}$					$\text{DBP}_{\text{FR-EN}}$				
	$\lvert\mathcal{T}\rvert_{\text{ZH}}$	$\lvert\mathcal{T}\rvert_{\text{EN}}$	$\lvert\mathcal{A}_s\rvert$	$\lvert\mathcal{A}_v\rvert$	$\lvert\mathcal{A}_p\rvert$	$\lvert\mathcal{T}\rvert_{\text{JA}}$	$\lvert\mathcal{T}\rvert_{\text{EN}}$	$\lvert\mathcal{A}_s\rvert$	$\lvert\mathcal{A}_v\rvert$	$\lvert\mathcal{A}_p\rvert$	$\lvert\mathcal{T}\rvert_{\text{FR}}$	$\lvert\mathcal{T}\rvert_{\text{EN}}$	$\lvert\mathcal{A}_s\rvert$	$\lvert\mathcal{A}_v\rvert$	$\lvert\mathcal{A}_p\rvert$
$t=0$	70,414	95,142	3,623	1,811	12,682	77,214	93,484	3,750	1,875	13,127	105,998	115,722	3,727	1,863	13,048
$t=1$	103,982	154,833	3,623	1,811	14,213	112,268	150,636	3,750	1,875	15,079	148,274	184,132	3,727	1,863	15,875
$t=2$	137,280	213,405	3,623	1,811	16,296	147,097	207,056	3,750	1,875	18,092	191,697	251,591	3,727	1,863	20,481
$t=3$	173,740	278,076	3,623	1,811	18,716	185,398	270,469	3,750	1,875	21,690	239,861	326,689	3,727	1,863	25,753
$t=4$	213,814	351,659	3,623	1,811	21,473	227,852	341,432	3,750	1,875	25,656	293,376	411,528	3,727	1,863	31,564
$t=5$	258,311	434,683	3,623	1,811	24,678	274,884	421,971	3,750	1,875	29,782	352,886	507,793	3,727	1,863	37,592

– Then, we complete $\mathcal{G}_1^t$ and $\mathcal{G}_2^t$ by adding additional relation triples from DBpedia of which the head and tail entities are both in the snapshots, leading to more than 20% growth of triple size.
– Finally, we retrieve the new entity alignment pairs brought by the newly added entities, and add them into the test set $\mathcal{A}_p^t$ of snapshot t. The training set $\mathcal{A}_s$ or validation set $\mathcal{A}_v$ still follows that in the first snapshot at time $t = 0$. We do not assume that the new snapshot introduces new training data because obtaining seed alignment for emerging entities is usually more difficult than finding seed alignment for old entities in the real world.

The detailed statistics of our dataset are present in Table 1.

4.2 Baselines

We compare ContEA with two groups of entity alignment methods.

– **Retraining baselines.** Since most existing embedding-based EA methods are designed for static KGs, they need retraining each time new triples come. Here, we choose the representative translation-based method MTransE [6], and several state-of-the-art GNN-based methods, including GCN-Align [33], AlignE [25], AliNet [27], KEGCN [40] and Dual-AMN [16] as our baselines.
– **Inductive baselines.** The only entity alignment method focusing on KGs' growth is DINGAL [39]. We choose one of the proposed variants, DINGAL-O, as a baseline, which can handle our scenario. Additionally, since there are some inductive KG embedding (KGE) methods which can generate embeddings for new entities, we explore their combination with static methods to tackle our task. Here, we select two representative inductive KGE methods MEAN [8] and LAN [31] as the entity representation layer and incorporate them with Dual-AMN. We denote the two baselines by MEAN^+ and LAN^+.

4.3 Experiment Settings

Evaluation Metrics. At each time t, the bidirectional nearest neighbor search and alignment integration are used to obtain the final trustworthy alignment.

The details are described in Sect. 3.1 and Sect. 3.2. Then, we compare the final trustworthy alignment with gold test pairs $\mathcal{A}_{\mathrm{p}}^{t}$. We report the precision, recall, and F1 scores as the evaluation metrics.

Implementation. We implement ContEA, Dual-AMN, MEAN^{+} and LAN^{+} using PyTorch. For other retraining baselines, we use the implementations in an open-source library.[3] We set the embedding dimensions to 100. The embedding similarity metric is CSLS [12]. We use grid search on hyperparameters and early stop to find the best performance. Specifically for ContEA, we set $\alpha = 0.1$, $\beta = 0.1$ and $m = 500$. More detailed hyperparameter settings can be found on our GitHub repository. For a fair comparison, all baselines only rely on KGs' structural information and do not use pre-trained models for initialization.

4.4 Results

General Results. We conduct experiments on the constructed datasets and present the results in Tables 2, 3 and 4. Compared with baselines, ContEA reaches the best performance in discovering potential entity alignment. Its F1 scores outperform the best baseline Dual-AMN by 27.1%, 19.4%, and 15.2% averagely on six snapshots of $\mathrm{DBP}_{\mathrm{ZH\text{-}EN}}$, $\mathrm{DBP}_{\mathrm{JA\text{-}EN}}$, and $\mathrm{DBP}_{\mathrm{FR\text{-}EN}}$, respectively. The superior performance of ContEA over retraining methods is because ContEA can iteratively leverage the prior knowledge (e.g., previously predicted alignment and model parameters) from the past snapshots. Also, ContEA collectively obtains predicted entity alignment by integrating new and old trustworthy alignment rather than totally neglecting old predictions in retraining. As for inductive baselines, MEAN^{+} and LAN^{+} perform worse than ContEA and Dual-AMN, which indicates that straightway adding the inductive KGE layer without adjusting the alignment network does not give satisfactory performance. DINGAL-O also shows unsatisfactory results, because it is purely inductive and does not update the alignment network. Besides, we can notice that the performance of all methods declines over time. This is due to the expansion of the searching space for alignment candidates, and the drop in the ratio of seed alignment against to-be-aligned alignment. Both of these increase the probability of entities being mismatched.

Ablation Study. To investigate the impact of each design of ContEA, also to give a fairer comparison between ContEA and baselines, we discard certain parts of ContEA and present three variants as follows:

- ContEA w/o TA. In the finetuning process, we discard the selected trustworthy entity alignment and only train on the affected seed alignment.
- ContEA w/o TA & ASA. We discard both the selected trustworthy alignment and the affected seed alignment. Thus, our method requires no finetuning and reduces to an inductive method. The entity reconstruction method generates embeddings for new entities using their neighbors.

[3] https://github.com/nju-websoft/OpenEA.

Table 2. Results of entity alignment on $\text{DBP}_{\text{ZH-EN}}$. NA stands for not applicable.

		$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
		P / R / F1	P / R / F1	P / R / F1	P / R / F1	P / R / F1	P / R / F1
Retraining	MTransE	.552/.178/.269	.242/.111/.152	.159/.078/.105	.094/.054/.068	.080/.041/.055	.049/.030/.037
	GCN-Align	.550/.249/.343	.212/.152/.177	.133/.115/.123	.096/.091/.094	.076/.075/.076	.062/.062/.062
	AlignE	.721/.364/.484	.382/.272/.317	.282/.222/.248	.206/.173/.188	.191/.152/.169	.127/.112/.119
	AliNet	.641/.358/.459	.285/.311/.297	.195/.279/.230	.146/.244/.183	.129/.232/.166	.105/.199/.128
	KEGCN	.664/.200/.308	.315/.129/.183	.198/.093/.127	.160/.075/.102	.136/.064/.087	.120/.052/.072
	Dual-AMN	.834/.596/.695	.482/.443/.462	.357/.356/.356	.285/.286/.286	.249/.254/.251	.227/.227/.227
Induct.	MEAN^+	.828/.576/.679	.483/.422/.450	.357/.341/.349	.267/.264/.265	.225/.226/.225	.198/.197/.198
	LAN^+	.827/.576/.679	.488/.426/.455	.360/.345/.352	.274/.271/.272	.231/.229/.230	.205/.199/.202
	DINGAL-O	.497/.195/.280	.370/.158/.222	.315/.135/.189	.251/.111/.154	.229/.093/.132	.209/.080/.116
ContEA		**.843/.604/.703**	**.555/.539/.546**	**.444/.473/.458**	**.373/.421/.396**	**.324/.375/.348**	**.291/.336/.312**
w/o TA		NA / NA / NA	.543/.531/.537	.419/.469/.443	.357/.414/.384	.316/.371/.341	.286/.332/.307
w/o TA & ASA		NA / NA / NA	.543/.527/.535	.422/.463/.442	.352/.410/.379	.309/.365/.335	.278/.324/.300
Retraining		NA / NA / NA	.493/.455/.473	.364/.357/.361	.300/.301/.301	.265/.266/.265	.245/.240/.243

Table 3. Results of entity alignment on $\text{DBP}_{\text{JA-EN}}$. NA stands for not applicable.

		$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
		P / R / F1	P / R / F1	P / R / F1	P / R / F1	P / R / F1	P / R / F1
Retraining	MTransE	.599/.200/.299	.293/.121/.172	.213/.082/.118	.151/.061/.087	.128/.046/.067	.117/.035/.054
	GCN-Align	.594/.279/.379	.263/.183/.216	.177/.142/.158	.140/.117/.127	.116/.099/.107	.099/.084/.091
	AlignE	.738/.359/.483	.433/.282/.342	.320/.218/.260	.270/.178/.214	.228/.148/.180	.193/.122/.149
	AliNet	.661/.364/.469	.305/.312/.308	.216/.270/.240	.167/.231/.194	.149/.215/.176	.126/.189/.151
	KEGCN	.663/.198/.305	.389/.153/.219	.280/.110/.157	.245/.087/.128	.200/.070/.104	.194/.063/.096
	Dual-AMN	**.861**/.606/.711	.517/.437/.474	.398/.347/.370	.348/.292/.318	.313/.251/.278	.300/.231/.261
Induct.	MEAN^+	.847/.571/.682	.528/.420/.468	.407/.330/.365	.330/.261/.292	.287/.221/.250	.265/.193/.223
	LAN^+	.845/.575/.684	.528/.424/.470	.410/.333/.368	.335/.265/.296	.296/.226/.257	.274/.200/.231
	DINGAL-O	.540/.227/.320	.391/.174/.241	.328/.137/.194	.271/.113/.159	.249/.092/.134	.231/.078/.116
ContEA		.858/**.610/.713**	**.586/.519/.551**	**.483/.440/.461**	**.417/.381/.398**	**.375/.336/.354**	**.344/.299/.320**
w/o TA		NA/NA/NA	.572/.518/.544	.466/.439/.452	.398/.377/.387	.357/.332/.344	.333/.294/.312
w/o TA & ASA		NA/NA/NA	.580/.514/.545	.466/.436/.450	.399/.374/.386	.359/.328/.343	.331/.291/.310
Retraining		NA/NA/NA	.530/.449/.486	.415/.356/.383	.369/.298/.330	.349/.272/.306	.327/.244/.280

Table 4. Results of entity alignment on $\text{DBP}_{\text{FR-EN}}$. NA stands for not applicable.

		$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
		P / R / F1	P / R / F1	P /R / F1	P /R / F1	P /R/ F1	P /R/ F1
Retraining	MTransE	.570/.188/.283	.246/.100/.142	.145/.062/.087	.108/.040/.059	.104/.032/.049	.073/.024/.036
	GCN-Align	.561/.262/.357	.233/.161/.190	.148/.111/.127	.113/.086/.098	.089/.066/.076	.077/.056/.065
	AlignE	.757/.394/.518	.399/.274/.325	.305/.202/.243	.245/.154/.189	.210/.121/.154	.195/.104/.136
	AliNet	.653/.361/.465	.275/.289/.282	.187/.226/.205	.144/.180/.160	.124/.155/.138	.115/.138/.126
	KEGCN	.716/.214/.330	.344/.125/.184	.260/.090/.134	.237/.076/.115	.201/.058/.089	.169/.045/.071
	Dual-AMN	.862/.629/.727	.503/.443/.471	.394/.331/.359	.351/.273/.307	.322/.237/.273	.313/.214/.254
Induct.	MEAN^+	.840/.585/.690	.514/.415/.459	.387/.305/.341	.314/.235/.269	.273/.191/.225	.254/.169/.203
	LAN^+	.845/.594/.697	.506/.410/.453	.379/.300/.335	.304/.227/.260	.269/.188/.222	.247/.162/.195
	DINGAL-O	.540/.224/.317	.381/.165/.231	.329/.124/.180	.258/.092/.136	.247/.073/.112	.227/.061/.096
ContEA		**.866/.634/.732**	**.569/.520/.543**	**.453/.421/.436**	**.387/.351/.369**	**.351/.301/.324**	**.325/.265/.292**
w/o TA		NA/NA/NA	.559/.516/.537	.443/.417/.430	.379/.348/.363	.342/.299/.319	.315/.263/.287
w/o TA & ASA		NA/NA/NA	.548/.511/.528	.431/.413/.421	.367/.342/.354	.334/.293/.312	.311/.256/.281
Retraining		NA/NA/NA	.516/.437/.473	.409/.339/.370	.372/.284/.322	.348/.247/.289	.331/.224/.267

– ContEA retraining. Same as the retraining baselines, ContEA treats each snapshot as at $t = 0$. Old predicted entity alignment is totally replaced by newly predicted entity alignment rather than being integrated.

Table 5. Recall of the alignment containing new entities on $\text{DBP}_{\text{ZH-EN}}$.

		$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
Retraining	MTransE	.075	.055	.032	.023	.013
	GCN-Align	.049	.031	.028	.014	.012
	AlignE	.137	.099	.067	.057	.040
	AliNet	.148	.149	.118	.124	.085
	KEGCN	.059	.046	.026	.026	.021
	Dual-AMN	.204	.164	.128	.113	.094
Induct.	MEAN^{+}	.170	.142	.106	.098	.078
	LAN^{+}	.167	.140	.109	.095	.076
	DINGAL-O	.003	.007	.007	.008	.006
ContEA		**.205**	**.167**	**.140**	**.116**	**.095**

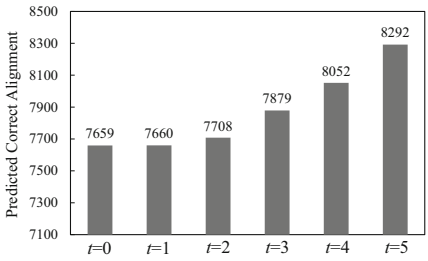

Fig. 3. Size growth of predicted correct alignment on $\text{DBP}_{\text{ZH-EN}}$.

We show the results of three variants in Tables 2, 3 and 4. The variants inherit the trained ContEA at $t = 0$ and perform respectively afterwards. We can notice a performance drop when discarding selected trustworthy alignment. Bigger declines are seen if further dropping the affected seed alignment. This demonstrates the effectiveness of both selected trustworthy alignment and affected seed alignment replay. For ContEA retraining, though it performs much worse than ContEA, it still outperforms all retraining baselines, including Dual-AMN, which indicates the effectiveness of entity reconstruction.

Discovering New Alignment. Next, we present the performance of ContEA on discovering alignment for new entities. At time $t = \{1, 2, 3, 4, 5\}$, we collect the final predicted alignment that involves new entities, and calculate the recall value by comparing it with the gold test alignment containing new entities. We show the results on $\text{DBP}_{\text{ZH-EN}}$ in Table 5. ContEA reaches the highest recall against all baselines, which indicates the advantage of our method in discovering alignment for new entities. We can also notice that the recalls on gold alignment about new entities are significantly lower than those on all gold alignment. This is because new entities tend to be sparsely-linked, which hinders the alignment models from matching them correctly. Also, Fig. 3 illustrates the growth of the total correctly predicted alignment of ContEA. At time t, the size of total correctly predicted alignment is calculated as $|\mathcal{A}^t_{\text{p}}| \times$ Recall (R in Table 2). The results show that ContEA can find an increasing size of correct entity alignment as KGs grow, which fulfills the proposal of continual entity alignment.

Efficiency. We compare the training efficiency of ContEA with retraining baselines. Note that, since inductive baselines have no training process as new triples come, we do not include them here. We run all experiments on a server outfitted with 512 GB memory, two Xeon Gold 6326 CPUs, and four RTX A6000 GPUs. Figure 4 depicts the average time cost on three datasets at different snapshots. We set the ceiling of vertical axis to 2,000 s for better presentation. We can see that ContEA has significantly less training time, which shows a part of its superiority in tackling the continual entity alignment task.

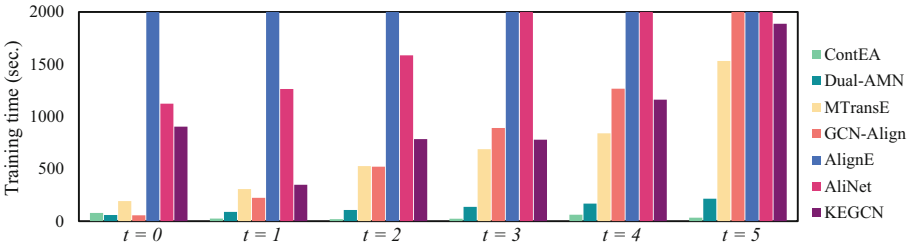

Fig. 4. Time cost comparison of ContEA and retraining baselines. We report the average time cost on the three datasets.

Table 6. F1 results comparison when incorporating name attribute of entities.

	DBP_{ZH-EN}						DBP_{JA-EN}						DBP_{FR-EN}					
	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
Google Translate	.550	.504	.473	.451	.434	.420	.677	.635	.617	.595	.587	.586	.749	.685	.658	.645	.636	.627
ContEA (fasttext)	.709	.556	.471	.411	.367	.334	.754	.615	.521	.456	.409	.374	.780	.613	.510	.446	.404	.368
ContEA ∪ G.T	.826	.645	.571	.506	.466	.441	.886	.742	.677	.637	.612	.604	.892	.740	.682	.650	.633	.623

4.5 Further Analysis

Incorporating Entity Names. Here we explore the advantage of leveraging entities' names. Practically, we use `fasttext` library to generate name embeddings for entities. Since the original word embedding dimension of `fasttext` is 300, to make the embedding space scalable, we reduce the dimension to 100 using the official dimension reducer.[4] Also, we involve Google Translate[5] (G.T.) as a competing method. For a cross-lingual dataset, we first translate entities from two KGs into the same language (both in English or non-English), and then calculate name similarity using Levenshtein distance, a popular measurement in linguistics [18] and ontology matching [4]. The bidirectional nearest neighbor search is also used later to obtain predicted trustworthy entity alignment, which are compared with the gold test set to calculate P, R, and F1 scores. We list the F1 results in Table 6. By utilizing name attributes, ContEA (`fasttext`) outperforms ContEA with a large margin. Google Translate gives satisfactory and robust performance over time, as powerful as expected. It performs more stably and is less sensitive to KGs' size, with outperforming ContEA (`fasttext`) on most snapshots of the three datasets except the first snapshot.

We further explore the combination of ContEA and Google Translate. To do so, we combine their predicted alignment when searching the nearest neighbor from one KG to the other, then take a bidirectional intersection to get the final combined predicted alignment. The results of this combination are shown in the last row. We can see that their combination outperforms both Google Translate and ContEA in almost all snapshots of three datasets. We believe that when Google Translate fails to align an entity, ContEA can be a practical alternative.

[4] https://fasttext.cc/docs/en/crawl-vectors.html.
[5] https://translate.google.com/.

Table 7. Case study on previously predicted alignment getting corrected.

	$t = 0$		$t = 5$	
	Predicted alignment	Sim.	Predicted alignment	Sim.
$\text{DBP}_{\text{ZH-EN}}$	(我是歌手, *Hunan Television*)	.204	(湖南卫视, *Hunan Television*)	.530
	(呼和浩特市, *Baotou*)	.243	(包头市, *Baotou*)	.541
$\text{DBP}_{\text{JA-EN}}$	(スウェーデン語, *Finnish language*)	.210	(フィンランド語, *Finnish language*)	.316
	(フランス人, *Spaniards*)	.265	(フランス人, *French people*)	.367
$\text{DBP}_{\text{FR-EN}}$	(*Révolution française, Réunion*)	.262	(*La Réunion, Réunion*)	.403
	(*Stade de Wembley, White Hart Lane*)	.302	(*Stade de Wembley, Wembley Stadium*)	.380

Case Study on Correcting Previous Alignment. Last, we present several cases in Table 7 about the previously predicted alignment getting corrected in later finetuning processes. We save the predicted alignment and their similarity scores at time $t = 0$ and $t = 5$, and juxtapose two alignment pairs from each time that involve the same entity. We manually check the list of juxtaposition and notice that the predicted alignment at $t = 0$ is usually incorrect with smaller similarity scores, while their counterparts at $t = 5$ are correct with higher similarity scores. This indicates the ability of ContEA on self-correction. Meanwhile, the two entities in falsely predicted alignment at $t = 0$ are not totally irrelevant. For example, in the second case from the $\text{DBP}_{\text{FR-EN}}$ dataset, both *Stade de Wembley* and *White Hart Lane* are Stadiums in London. In the first case from the $\text{DBP}_{\text{ZH-EN}}$ dataset, 我是歌手 is a popular TV show made by *Hunan Television*. And in the first case from the $\text{DBP}_{\text{JA-EN}}$ dataset, スウェーデン語 means Swedish language (Sweden and Finland are two neighboring Nordic countries). This gives an interesting insight on how ContEA predicts entity alignment with slight inaccuracy.

5 Related Work

Static Entity Alignment. Most existing embedding-based entity alignment methods focus on static KGs. They can usually be classified into two categories regarding the techniques of their KG encoders: translation-based [6,14,19,25,26,43] and GNN-based [16,17,27,33–36]. The former family adopts translation-based KG embedding (KGE) techniques [3,32] to embed entities, and map cross-graph entities into a unified space based on pre-aligned entity pairs. The encoder of GNN-based entity alignment methods learns a shared neighborhood aggregator to embed entities in different KGs. They have gained overwhelming popularity in recent years due to their strong ability to capture the structural information using a subgraph around an entity, rather than a single triple. For more details, there are several surveys [28,41] that comprehensively summarize the recent advances.

Dynamic Entity Alignment. As far as we know, DINGAL [39] is the only entity alignment method that addresses the dynamics of KGs. In its dynamic

scenario, new triples are added into KGs as well as new pre-known alignment provided along with these new entities. A variant of DINGAL, named DINGAL-O, is also proposed in their work to handle a similar setting as ours where the pre-known alignment does not grow. DINGAL-O is an inductive method that leverages prior-learned model parameters to predict new alignment. Particularly, they use name attributes to generate word embeddings for entity initialization.

Inductive Knowledge Graph Embedding. The study on dynamic KG embedding has drawn lots of attention over the years. Powered by GNN, many inductive embedding methods for KG completion are proposed to generate embeddings for new entities. Early inductive methods either focus on semi-inductive settings where new entities are connected to existing KG and making inferences between new entities and existing entities [8,9,31,38], or fully-inductive settings where new entities form independent graphs and making inferences among new entities [5,29]. Later inductive method [15] intend to tackle both settings. Meanwhile, some inductive KG embedding methods focus on special tasks like few-shot learning [38] and hyper-relational KG completion [1]. Specifically, as the first inductive KG embedding method, MEAN [8] learns to represent entities using their neighbors by simply mean-pooling the information of neighboring entity-relation pairs. LAN [31] advances MEAN by incorporating a rule-based attention and a GNN-based attention on entity-relation pairs in the pooling process.

6 Conclusion and Future Work

In this paper, considering the growth nature of real-world KGs, we focus on an entity alignment scenario where both graphs are growing, and address a new task named continual entity alignment. We propose a novel method ContEA as a solution to the task. Also, we construct three datasets to imitate the scenario and conduct extensive experiments. The experimental results show the superiority of ContEA in terms of effectiveness and efficiency against a list of retraining and inductive baselines. For future work, there are many promising improvements and extensions to the current proposal. Regarding the setting, future studies can consider more complex scenarios such as the addition of new relations, the addition of new pre-known alignment, and even the deletion of entities and triples. As to the method, more reliable and comprehensive trustworthy alignment update strategies are necessary to handle intricate alignment conflict cases.

Supplemental Material Statement: The source code, detailed hyperparameters, and constructed datasets are available at our GitHub repository.[6]

Acknowledgments. This work was supported by National Natural Science Foundation of China (No. 61872172), Beijing Academy of Artificial Intelligence (BAAI),

[6] https://github.com/nju-websoft/ContEA.

and Collaborative Innovation Center of Novel Software Technology & Industrialization. Zequn Sun was also grateful for the support of Program A for Outstanding PhD Candidates of Nanjing University.

References

1. Ali, M., Berrendorf, M., Galkin, M., Thost, V., Ma, T., Tresp, V., Lehmann, J.: Improving inductive link prediction using hyper-relational facts. In: ISWC, pp. 74–92 (2021)
2. Baek, J., Lee, D.B., Hwang, S.J.: Learning to extrapolate knowledge: transductive few-shot out-of-graph link prediction. In: NeurIPS (2020)
3. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS, pp. 2787–2795 (2013)
4. Cheatham, M., Hitzler, P.: String similarity metrics for ontology alignment. In: ISWC, pp. 294–309 (2013)
5. Chen, J., He, H., Wu, F., Wang, J.: Topology-aware correlations between relations for inductive link prediction in knowledge graphs. In: AAAI, pp. 6271–6278 (2021)
6. Chen, M., Tian, Y., Yang, M., Zaniolo, C.: Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In: IJCAI, pp. 1511–1517 (2017)
7. Chen, X., Chen, M., Fan, C., Uppunda, A., Sun, Y., Zaniolo, C.: Multilingual knowledge graph completion via ensemble knowledge transfer. In: Findings of EMNLP, pp. 3227–3238 (2020)
8. Hamaguchi, T., Oiwa, H., Shimbo, M., Matsumoto, Y.: Knowledge transfer for out-of-knowledge-base entities: a graph neural network approach. In: IJCAI, pp. 1802–1808 (2017)
9. He, Y., Wang, Z., Zhang, P., Tu, Z., Ren, Z.: Vn network: embedding newly emerging entities with virtual neighbors. In: CIKM, pp. 505–514 (2020)
10. Hofer, M., Hellmann, S., Dojchinovski, M., Frey, J.: The new DBpedia release cycle: increasing agility and efficiency in knowledge extraction workflows. In: Blomqvist, E., Groth, P., de Boer, V., Pellegrini, T., Alam, M., Käfer, T., Kieseberg, P., Kirrane, S., Meroño-Peñuela, A., Pandit, H.J. (eds.) SEMANTICS 2020. LNCS, vol. 12378, pp. 1–18. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59833-4_1
11. Jiménez-Ruiz, E., Grau, B.C.: LogMap: logic-based and scalable ontology matching. In: ISWC, pp. 273–288 (2011)
12. Lample, G., Conneau, A., Ranzato, M., Denoyer, L., Jégou, H.: Word translation without parallel data. In: ICLR (2018)
13. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web **6**(2), 167–195 (2015)
14. Lin, X., Yang, H., Wu, J., Zhou, C., Wang, B.: Guiding entity alignment via adversarial knowledge embedding. In: ICDM (2019)
15. Liu, S., Grau, B.C., Horrocks, I., Kostylev, E.V.: INDIGO: GNN-based inductive knowledge graph completion using pair-wise encoding. In: NeurIPS (2021)
16. Mao, X., Wang, W., Wu, Y., Lan, M.: Boosting the speed of entity alignment 10 ×: Dual attention matching network with normalized hard sample mining. In: WWW, pp. 821–832 (2021)

17. Mao, X., Wang, W., Xu, H., Lan, M., Wu, Y.: MRAEA: an efficient and robust entity alignment approach for cross-lingual knowledge graph. In: WSDM, pp. 420–428 (2020)
18. Medhat, D., Hassan, A., Salama, C.: A hybrid cross-language name matching technique using novel modified levenshtein distance. In: ICCES, pp. 204–209 (2015)
19. Pei, S., Yu, L., Zhang, X.: Improving cross-lingual entity alignment via optimal transport. In: IJCAI, pp. 3231–3237. IJCAI (2019)
20. Singh, H., Chakrabarti, S., Jain, P., Choudhury, S.R., Mausam: Multilingual knowledge graph completion with joint relation and entity alignment. In: AKBC (2021)
21. Song, Y., Karras, P., Xiao, Q., Bressan, S.: Sensitive label privacy protection on social network data. In: SSDBM, pp. 562–571 (2012)
22. Suchanek, F.M., Abiteboul, S., Senellart, P.: PARIS: probabilistic alignment of relations, instances, and schema. Proc. VLDB Endow. **5**(3), 157–168 (2011)
23. Sun, Z., Chen, M., Hu, W.: Knowing the no-match: Entity alignment with dangling cases. In: ACL, pp. 3582–3593 (2021)
24. Sun, Z., Hu, W., Li, C.: Cross-lingual entity alignment via joint attribute-preserving embedding. In: ISWC, pp. 628–644 (2017)
25. Sun, Z., Hu, W., Zhang, Q., Qu, Y.: Bootstrapping entity alignment with knowledge graph embedding. In: IJCAI, pp. 4396–4402 (2018)
26. Sun, Z., Huang, J., Hu, W., Chen, M., Guo, L., Qu, Y.: TransEdge: translating relation-contextualized embeddings for knowledge graphs. In: ISWC, pp. 612–629 (2019)
27. Sun, Z., et al.: Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In: AAAI, pp. 222–229 (2020)
28. Sun, Z., Zhang, Q., Hu, W., Wang, C., Chen, M., Akrami, F., Li, C.: A benchmarking study of embedding-based entity alignment for knowledge graphs. Proc. VLDB Endow. **13**(11), 2326–2340 (2020)
29. Teru, K.K., Denis, E.G., Hamilton, W.L.: Inductive relation prediction by subgraph reasoning. In: ISWC (2020)
30. Vrandecic, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
31. Wang, P., Han, J., Li, C., Pan, R.: Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In: AAAI, pp. 7152–7159 (2019)
32. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI, pp. 1112–1119 (2014)
33. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: EMNLP, pp. 349–357 (2018)
34. Wang, Z., Lv, Q., Lan, X., Zhang, Y.: Cross-lingual knowledge graph alignment via graph convolutional networks. In: ACL, pp. 349–357 (2018)
35. Wu, Y., Liu, X., Feng, Y., Wang, Z., Yan, R., Zhao, D.: Relation-aware entity alignment for heterogeneous knowledge graphs. In: IJCAI, pp. 5278–5284 (2019)
36. Wu, Y., Liu, X., Feng, Y., Wang, Z., Zhao, D.: Neighborhood matching network for entity alignment. In: ACL, pp. 6477–6487 (2020)
37. Xia, Y., Gao, J., Cui, B.: iMap: Incremental node mapping between large graphs using GNN. In: CIKM, pp. 2191–2200 (2021)
38. Xie, R., Liu, Z., Jia, J., Luan, H., Sun, M.: Representation learning of knowledge graphs with entity descriptions. In: AAAI, pp. 2659–2665 (2016)
39. , Yan, Y., Liu, L., Ban, Y., Jing, B., Tong, H.: Dynamic knowledge graph alignment. In: AAAI, pp. 4564–4572 (2021)
40. Yu, D., Yang, Y., Zhang, R., Wu, Y.: Knowledge embedding based graph convolutional network. In: WWW, pp. 1619–1628 (2021)

41. Zeng, K., Li, C., Hou, L., Li, J., Feng, L.: A comprehensive survey of entity alignment for knowledge graphs. AI Open **2**, 1–13 (2021)
42. Zhao, X., Zeng, W., Tang, J., Wang, W., Suchanek, F.: An experimental study of state-of-the-art entity alignment approaches. IEEE Trans. Knowl. Data Eng. (2020)
43. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative entity alignment via joint knowledge embeddings. In: IJCAI, pp. 4258–4264 (2017)