applied sciences

MDPI

Article

A Novel Embedding Model for Knowledge Graph Entity Alignment Based on Graph Neural Networks

Hongchan Li, Zhaoyang Han, Haodong Zhu * and Yuchao Qian

School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450002, China; 2011017@zzuli.edu.cn (H.L.); aa1522802412@163.com (Z.H.); supaor66@163.com (Y.Q.)
* Correspondence: 2011016@zzuli.edu.cn; Tel.: +86-13592697657

Abstract: The objective of the entity alignment (EA) task is to identify entities with identical semantics across distinct knowledge graphs (KGs) situated in the real world, which has garnered extensive recognition in both academic and industrial circles. Within this paper, a pioneering entity alignment framework named PCE-HGTRA is proposed. This framework integrates the relation and property information from varying KGs, along with the heterogeneity information present within the KGs. Firstly, by learning embeddings, this framework captures the similarity that exists between entities present across diverse KGs. Additionally, property triplets in KGs are used to generate property character-level embeddings, facilitating the transfer of entity embeddings from two distinct KGs onto an identical space. Secondly, the framework strengthens the property character-level embeddings using the transitivity rule to increase the count of entity properties. Then, in order to effectively capture the heterogeneous features in the entity neighborhoods, a heterogeneous graph transformer with relation awareness is designed to model the heterogeneous relations in KGs in the framework. Finally, comparative experimental results on four widely recognized real-world datasets demonstrate that PCE-HGTRA performs exceptionally well. In fact, its Hits@1 performance exceeds the best baseline by 7.94%, outperforming seven other state-of-the-art methods.

Keywords: knowledge graphs; entity alignment; character embeddings; heterogeneous features

Citation: Li, H.; Han, Z.; Zhu, H.; Qian, Y. A Novel Embedding Model for Knowledge Graph Entity Alignment Based on Graph Neural Networks. *Appl. Sci.* **2023**, *13*, 5876. https://doi.org/10.3390/app13105876

Academic Editor: Yu-Dong Zhang

Received: 7 April 2023
Revised: 29 April 2023
Accepted: 4 May 2023
Published: 10 May 2023

1. Introduction

As technology for storing complex structured and unstructured data, knowledge base (KB) has been widely applied in various fields relating to artificial intelligence. Among them, knowledge graphs (KGs), as the most common representation of knowledge bases, have made significant progress and have been extensively applied across diverse application scenarios such as recommendation systems, information retrieval, machine translation, and so on, drawing high levels of attention from both industry and academia [1]. However, different institutions and organizations construct knowledge graphs using different technologies and languages for their own purposes, resulting in heterogeneous structures and complementary contents. The same entity can exist in various forms across distinct knowledge graphs. Therefore, how to efficiently organize this redundant information to form a more comprehensive knowledge graph for downstream tasks is an important challenge currently faced by this field. The objective of entity alignment is to establish connections between identical entities present in two distinct knowledge graphs. This facilitates the transfer of valuable information from one KG to its corresponding entity in another KG, thus enriching the content of both and contributing significantly to the performance of downstream applications.

Traditional entity alignment methods [2] typically focus on structured data sources, such as relational databases, and use heuristic or data mining methods to calculate the similarity between different entities, aiming to improve the effectiveness of entity matching. However, with the growth of data, the efficiency of traditional methods for entity matching

needs to be improved. Moreover, knowledge graphs are semi-structured data structures, and the accuracy of the traditional entity alignment techniques is limited, while heuristic algorithms are also difficult to generalize.

With the emergence of Word2Vec [3], the task of entity alignment has gradually been bifurcated into approaches that are based on translation and those that are based on graph neural networks [4] (GNNs). The fundamental concept behind both of these approaches is consistent. It involves acquiring an efficient vector representation of the KG within a low-dimensional space and subsequently executing entity alignment tasks based on the learned vector representation. This technique is collectively known as representation learning. A large number of experiments have evinced that translation-based methods are more suitable for link prediction, whereas GNNs excel at incorporating the neighborhood features of nodes. These capabilities can be harnessed to devise pertinent techniques for feature acquisition in entity alignment tasks, thereby enabling the attainment of superior accuracy and generalization capacity [5].

Although GNNs methods have achieved remarkable results, there are still three limitations. Firstly, most methods [6–8] regard KGs as homogeneous graphs and do not consider the heterogeneity of edges between different entities, whereas heterogeneous information can augment the accuracy and resilience of the model. Secondly, although many methods consider some semantic information beyond the relational structure, such as entity property information [9], entity description information [10], and entity name information [11], the more semantic information integrated into the methods, the more data are required, which is difficult to satisfy in many practical scenarios where the seed entities are often insufficient. Thirdly, some other works [12] only use the relational structures of different KGs to extract inter-graph information using graph matching networks (GMN) [13] to explore more analogous features between aligned entities. However, the introduction of the matching module throughout the training process results in an increase in the space and time complexity of the model, thereby impacting its efficiency.

The proposed heterogeneous graph transformer with relation awareness (HGTRA) in this paper addresses the first limitation by effectively extracting similar features from aligned entities within their respective heterogeneous structures. To address the aforementioned latter two limitations, a new embedding model is introduced. The model initially generates property embeddings from the knowledge graph's property information and then relocates the entity embeddings of two knowledge graphs to the same vector space by leveraging the property embeddings. Thus, the similarity of properties between two knowledge graphs is crucial for generating a unified embedding space, which is also a major challenge in knowledge graph alignment tasks. Utilizing property embeddings, the PCE-HGTRA model can reciprocally transform the entity embeddings of both knowledge graphs to the identical vector space, allowing the entity embeddings to capture the property similarity from both knowledge graphs. This paper's similarity model between entities includes predicate alignment, which renames the predicates of two knowledge graphs to a unified scheme, ensuring that the relation embeddings of both knowledge graphs are also embedded in the identical vector space.

This article further employs the transitivity rule, whereby if there exists (A, r_1, B) and (B, r_1, C) it is possible to deduce the existence of (A, r_1, C), to enrich the property triples and relation triples used in calculating property embeddings and enhancing relation triples. Therefore, the fundamental concept of this article is to fully exploit the wealth of knowledge graphs by simultaneously evaluating the similarity of entity relationships and property information. By fusing the relationship information and property information, the two complement each other and alleviate the heterogeneity between different types of information when aligning entities.

We have integrated the two approaches mentioned above into an entity alignment framework called PCE-HGTRA, which fully takes into account the heterogeneous information of properties and relations in knowledge graphs. Extensive experimentation on three well-known benchmark datasets has demonstrated that PCE-HGTRA surpasses seven

state-of-the-art models in both accuracy and effectiveness, while also exhibiting remarkable robustness and scalability.

2. Related Work

Entity Alignment based on Translation. This methodology presented in this study is chiefly rooted in TransE [14] and certain of its adaptations, with the core idea of representing the relationship between two entities as the transformation between their embedded representations in order to ensure that entities with analogous structures in different KGs are in close proximity in the embedding space, achieving the goal of preserving entity structural information. MtransE [15] is the first work to introduce embeddings in a multilingual setting. This work models entities and relationships using TransE, embedding each entity and relationship in different embedding spaces in each knowledge graph, and based on pre-aligned entities, the transformation between the two vector spaces is evaluated. The model includes a knowledge module for encoding and an alignment module for learning. This work proposes three learning strategies, including linear transformations, translation vectors, and distance-based axis alignment, with linear transformations having the best performance. JAPE [16] utilizes embeddings of relations and properties to optimize the embedding effect of the knowledge graph. Specifically, the method jointly embeds two distinct knowledge graphs into a shared vector space and improves the effect by embedding property information. In addition, customized data preprocessing techniques are used to facilitate the sharing of the same or similar embeddings among aligned entities in the seed alignment, allowing the model to achieve cross-lingual entity alignment. IPTransE [17] adopts semi-supervised learning and a margin-based loss function and uses bootstrapping techniques to add newly aligned entities to the seed entities, thereby expanding the number of available resources while ensuring quality. This model improves upon the underlying TransE method with PTransE, which captures indirectly connected entities by observing the paths between entities and constructing transformations between entities based on the path information composed of relation predicates connecting multiple entities. This model relies on seed entities and divides the transformation between the embedding spaces of both knowledge graphs into three strategies: translation, linear transformation, and parameter sharing, with parameter sharing being the most effective.

Entity alignment based on GNNs. The main approach for entity alignment using graph attention networks (GATs) and graph convolutional networks (GCNs) involves aggregating the neighborhood features of each entity to obtain neighborhood similarities between the corresponding aligned entities. GCN_align [6] was the first GNN-based EA study, which achieved alignment through the margin-based loss function. This study treated property triplets as relation triplets, learned entity embeddings from structural information, and used two GCNs to embed entities from two knowledge graphs into a unified space with a shared weight matrix. RDGCN [7], also a margin-based loss function, integrated relation information through an attentional interaction mechanism and extended GCNs with relation information and a high-speed gating mechanism to capture neighborhood structural information, similar to HGCN [11]. SEA [18] achieved alignment by utilizing cyclic consistency constraints and aligning unaligned entities.

Entity alignment based on heterogeneous GNNs. Recently, numerous academic studies have attempted to apply graph neural networks (GNNs) for modeling heterogeneous graphs. Among them, RGCNs [19] and RGATs [20] describe heterogeneous graphs by utilizing weight matrices for each relationship. HAN [21] has innovatively proposed a hierarchical attention mechanism, learning weights of nodes and meta-paths from both the node level and semantic level. Meanwhile, HetGNN [22] employs various recurrent neural networks (RNNs) to integrate multimodal features to deal with various types of nodes. However, due to the existence of a large number of relations in knowledge graphs (KG), the application of these methods to KG models results in high training complexity. Recently, HGT [23] and RHGT [24] have attempted to describe heterogeneity using heterogeneous graph transformers. Nevertheless, these methods are not specifically designed to capture

neighbor similarities; thus, they are not directly applicable to entity alignment tasks. Consequently, we propose an enhanced heterogeneous graph transformer that takes into account the heterogeneity of knowledge graphs and provides high-quality entity embeddings for entity alignment tasks.

Self-Supervised learning models for knowledge graphs. To capture the semantic discrepancies between entities and relationships in knowledge graphs, contrastive learning has emerged as a viable technique. Cutting-edge research has recently merged knowledge graph representation with contrastive learning, giving rise to the development of a universal knowledge graph contrastive learning framework, KGCL [25]. The framework aims to reduce noise in the underlying data supporting recommendation systems and provides stronger knowledge representation capabilities. The CKGC [26] method differentiates between descriptive attributes and traditional relationships in the knowledge graph, connecting the remaining parts as a structure to broaden the descriptive information scope of the knowledge graph.

3. Proposed Framework

3.1. Problem Definition

As a type of graph structure, in a knowledge graph entities are represented as nodes and the relations between entities are represented as edges. However, the symbolic features of triples make processing difficult. Obtaining more effective entity embedding representations has become a major challenge in entity alignment tasks. For generality, this paper uses uppercase letters to represent sets and lowercase letters to represent vectors. Let $G = (E, R, P, V, T)$ represent a knowledge graph, where E represents the set of all entities; R represents the set of relationship predicates; P represents the set of property predicates; V represents the set of property values; and T represents the set of triples that relate entities and their properties. $T = T_r \cup T_p$, where T_r represents relationship triples (h, r, t) in the knowledge graph—where h stands for head entity, t stands for tail entity, and r stands for relationship (predicate) between them—T_p represents property triples (e, p, v)—where e stands for entity, p stands for property name, and v stands for property value. Entity alignment endeavors to unearth the corresponding entities across divergent knowledge graphs.

Given two knowledge graphs, $G_1 = (E_1, R_1, P_1, V_1, T_1)$ and $G_2 = (E_2, R_2, P_2, V_2, T_2)$, we aim to discover (e_1, e_2), where $e_1 \in E_1$, $e_2 \in E_2$, and $e_1 \equiv e_2$, indicating that e_1 and e_2 denote the identical real-world entity, with " $\equiv$ " indicating an equivalence relation. We employ an embedding-based model to assign a continuous representation to every element of two types of triples (h, r, t) and (e, p, v), represented in bold font $(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})$, $(\boldsymbol{e}, \boldsymbol{p}, \boldsymbol{v})$.

3.2. Overview Framework of PCE-HGTRA

In this chapter, we will present our novel and robust entity alignment (EA) framework, the PCE-HGTRA, which incorporates property character embedding and heterogeneous graph transformers with relation awareness. PCE-HGTRA consists of three main modules, as shown in Figure 1 PCE-HGTRA framework overview: (1) property character-level embedding (PCE), which proposes a novel embedding approach to locate partially similar predicates and uses a unified naming scheme for renaming; (2) heterogeneous graph transformer with relation awareness (HGTRA), where we have designed HGTRA with the aim of capturing distinctive pattern features of relations and properties while utilizing fewer parameters. This involves incorporating the heterogeneous neighborhood features of aligned entities in both relations and properties; and (3) alignment learning, which computes the loss function of entity embeddings by taking into account both their properties and relations and, subsequently, evaluating the likelihood of entity alignment (EA).

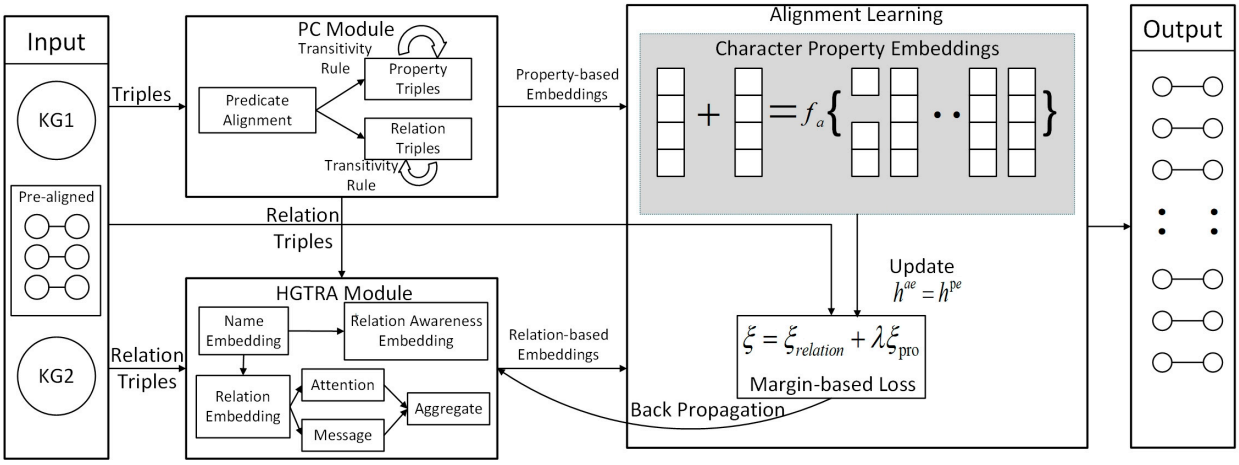

Figure 1. Overview of the framework of PCE-HGTRA.

3.3. Property Character Embedding

After applying the TransE algorithm, we proceeded with the property character-level embedding. Here, the predicate r is interpreted as the transformation from the head entity (h) to the property value (p) within the context of this paper. However, in the two knowledge graphs, the same property value p may manifest in different forms, for example, as 20.445 and 20.445444 in financial data, or as "Li Bai" and "Qinglian Jushi" in personal names. Therefore, to encode the property values, a composite function is used in this paper, and the relationship for each element in the Tp is defined as $h + r \approx f_p(p)$. Here, $f_p(p)$ is a composite function, and p is the property value, $p = \{a_1, a_2, \ldots, a_t\}$. The composite function is employed in this paper to encode the property values into a single vector while mapping similar property values to similar vector representations. Three composite functions are defined in this paper.

Summation composite function (SUM). The initial composite function pertains to the summation function (SUM), which is defined as the total sum of all character embeddings of the property values. The definition of the summation composite function is as follows:

$$f_p(p) = a_1 + a_2 + \ldots + a_t \tag{1}$$

The characters $a_1, a_2, \cdots, p_t$ represent the character embeddings of the property value. However, this composite function is not without limitations. Its inadequacy lies in the fact that when two strings share the same character set but in different orders, they will have identical vector representations. For instance, the values "20.18" and "18.02" would result in the same vector representation, rendering the function less effective.

Composite function based on LSTM (LSTM). In order to surmount the restrictions posed by the SUM composite function, this paper puts forth a novel composite function founded on Long Short-Term Memory (LSTM). This function employs an LSTM network to encode the character sequence of the property value into a solitary vector. The ultimate hidden state of the LSTM network is utilized as the vector representation of the property value. The composite function based on LSTM is delineated as follows:

$$f_p(p) = f_{lstm}(a_1, a_2, \ldots, a_t) \tag{2}$$

N-gram-based composite function (N-gram). This paper further proposes an N-gram-based composite function as a viable solution to mitigate the limitations of the SUM composite function. Specifically, this function uses the sum of all n-tuples (N-grams) in the

property value as the vector representation. The definition of the N-gram-based composite function is shown as follows:

$$f_p(p) = \sum_{n=1}^{N} \left(\frac{\sum_{i=1}^{t} \sum_{j=i}^{n} a_j}{t-i-1} \right) \tag{3}$$

where N represents the upper limit of N-gram combinations utilized (in this study, N = 15); and t signifies the length of the property value.

To acquire the property character embedding, the following objective function is minimized in this study, the detailed definition of J_A is as follows:

$$J_A = \sum_{t_p \in T_p} \sum_{t'_p \in T'_p} max\left(0, \left[\gamma + \alpha\left(f(t_p) - f\left(t'_p\right)\right)\right]\right) \tag{4}$$

The detailed definitions of T_p and T'_p are as follows:

$$T_p = \{\langle h, r, p\rangle \in G\}; f(t_p) = ||h + r - f_p(p)|| \tag{5}$$

$$T'_p = \left\{\langle h', r, p\rangle | h' \in E\right\} \cup \left\{\langle h, r, p'\rangle | p' \in A\right\} \tag{6}$$

where T_p denotes the collection of authentic property triplets within the training dataset; and T'_p represents the collection of defective property triplets (where A signifies the collection of properties in G). The erroneous triplets serve as negative samples, where a random entity replaces the head entity or a random property value replaces the property. $f(t_p)$ symbolizes the confidence score of the vector representation of the property value, which is rooted in the embedding of the head entity "h", the embedding of the relationship "r", and the vector representation of the property value derived via the composite function $f_p(p)$.

3.4. Heterogeneous Graph Transformer with Relation Awareness (HGTRA)

The process by which the graph transformer assimilates all the neighboring features of node "h" can be elegantly formulated as follows:

$$e_h^{(l)} \leftarrow \underset{\forall t \in N(h)}{Aggregate}(Attention(h,t) \cdot Message(h,t)) \tag{7}$$

In this equation, *Attention* is used to calculate the importance of each neighboring node; *Message* extracts features from each neighboring node; and *Aggregate* aggregates neighbor information using attention weights. However, as illustrated in Equation (7), the graph transformer fails to consider edge features. To address this, we designed a novel heterogeneous graph transformer with relation awareness (HGTRA) in this paper, inspired by previous work [26]. The proposed HGTRA enables our model to differentiate between the heterogeneous features of relations and properties, thus facilitating a better capture of neighborhood similarities among aligned entities. Let $E^{(l)}$ represents the output of the (l)-th layer of HGTRA, which also serves as the input to the $(l+1)$-th layer. At first, the value of $E^{(0)}$ is equal to $E^{(n)}$.When HGTRA takes a relation triplet as input, the output is relation-based embedding. When it takes a property triplet as input, the output is property-based embedding. HGTRA mainly consists of the following four layers, as shown in Figure 2:

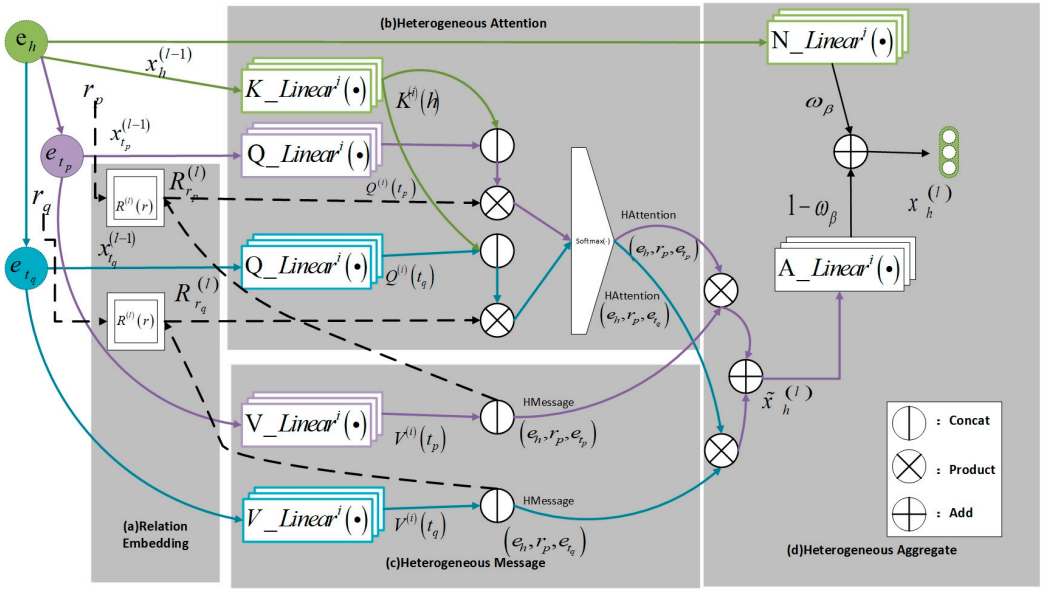

Figure 2. Overview of the framework of HGTRA.

(a) Relation Embedding. Considering the possible similarity between the aligned relation and the head entity and tail entity, this paper generates relation features by combining the relevant entity features. In particular, the relation embedding r is approximately calculated by taking an average of the embeddings of its related head entity H_r and tail entity T_r, as demonstrated in the subsequent formula:

$$R^l(r) = \sigma\left[\frac{\sum_{e_i \epsilon H_r} b_h e_i^{(l-1)}}{|H_r|} || \frac{\sum_{e_j \epsilon T_r} b_t e_j^{(l-1)}}{|T_r|}\right] \tag{8}$$

In this equation, $|\cdot|$ denotes the size of a set; b_h and b_t are attention vectors; $[\cdot]$ denotes the operation of concatenation; and σ denotes the activation function Rectified Linear Unit (ReLU);

(b) Heterogeneous Attention. In this work, entity h is mapped to a key vector $K^i(h)$, and its neighboring entity t is mapped to a query vector $Q^i(t)$. In contrast to other methods, this work uses the dot product of their concatenation and $R^l(r)$ as the value of attention, rather than directly using the dot product of the key and query vectors. $R^l(r)$ is derived from the feature aggregation of the related head and tail entities (refer to Equation (8)), hence it does not stray too far from the embeddings of its linked entities. In addition, $R^l(r)$ signifies the heterogeneous feature of the edge, thereby causing distinct effects on the contribution of neighboring pairs linked to different edges towards the entity h. More specifically, this work calculates the multi-head attention of each neighbor relation (h, r, t), evaluated in the following manner in this study:

$$HAttention(h,r,t) = \Big|\Big|_{i\in[1,h_n]\forall(r,t)\in RN(h)} Softmax(HATT_{head^i}(h,r,t)) \tag{9}$$

Among them, the detailed expression of $HATT_{head^i}(h,r,t)$ is as follows:

$$HATT_{head^i}(h,r,t) = \frac{a^T\left(\left[K^i(h)||Q^i(t)\right]R^l(r)\right)}{\sqrt{d/h_n}} \tag{10}$$

where $K^i(h) = K\_Linear^i\left(e_h^{(l-1)}\right)$; $Q^i(t) = Q\_Linear^i\left(e_t^{(l-1)}\right)$; the symbol $RN(h)$ denotes the set of entities neighboring h; the parameter a is the attention parameter of dimensionality $d/h_n \times 1$, where h_n denotes the number of attention heads. It should be noted that the

Softmax operation ensures that the sum of attention weights assigned to all neighboring entities is equal to one;

(c) Heterogeneous Message. Likewise, this paper aims to integrate relationships into the message-passing mechanism in order to differentiate the disparities between diverse categories of edges. For any given $(h, r, t) \in t$, the calculation of its multi-head message is carried out as follows:

$$HMessage(h, r, t) = \Big\|_{i \in [1, h_n]} HMSG_{head^i}(h, r, t) \tag{11}$$

The detailed expression of $HMSG_{head^i}(h, r, t)$ is shown below:

$$HMSG_{head^i}(h, r, t) = \left[{V\_Linear}^i \left(e_t^{(l-1)} \right) \middle\| R^{(l)}(r) \right] \tag{12}$$

In order to obtain the (i)-th message head, $HMSG_{head^i}(h, r, t)$, this paper first applies the linear projection V_Linear^i to project the characteristics of the tail entity t. Subsequently, it concatenates the features of t and the relation r, and connects all h_n message heads to obtain the final heterogeneous message;

(d) Heterogeneous Aggregation. The final step is heterogeneous aggregation, depicted in Figure 2d, where the heterogeneous multi-head attention and messages are merged into entities. By using attention coefficients to weigh the messages of neighboring entities, we can aggregate information from neighbors with different features and update the vector representation of entity h. The specific formula is shown below:

$$\widetilde{e_h}^{(l)} = \forall (r, t) \in RN(h) \oplus HAttention(h, r, t) \cdot HMessage(h, r, t) \tag{13}$$

In this context, the symbol $\oplus$ represents the operation of superimposition. In order to combine the characteristics of names and the features derived from a multi-layer neural network, we employ residual connections [27] to create the ultimate modified embedding, as demonstrated in the subsequent equation:

$$e_h^{(l)} = \varpi_\beta A_{Linear(\widetilde{e_h}^{(l)})} + (1 - \varpi_\beta) N_{Linear(e_h^{(l-1)})} \tag{14}$$

where ϖ_β is a trainable weight; and $A\_Linear(\cdot)$ and $N\_Linear(\cdot)$ are linear projections. Finally, based on the entire relation structure T_{rel} and the property structure T_{attr}, this paper can generate relation-based embedding E_{rel} and property-based embedding E_{attr}, respectively, and employ them for end-to-end entity alignment tasks.

3.5. Learning Alignment

Upon obtaining the ultimate entity representations, this paper uses the Manhattan distance to gauge the similarity among potential pairs of entities. The more negligible the distance, the greater the likelihood of entity alignment. To calculate the similarity between candidate entity pairs, this paper uses E_{rel} and E_{attr}, and the specific equation is stated as follows:

$$d_f \left(e_i^1, e_j^1 \right) = ||e_{f,i}^1 - e_{f,i}^2||_{L1} \tag{15}$$

where $f = \{rel, attr\}$; $L1$ denotes the Manhattan distance.

Previous methods generally concatenated the embeddings of entities from multiple sources and employed them directly in the loss function to capture the entity features comprehensively. Nevertheless, we opine that relation-based embedding and property-based embedding may contribute to EA differently, since these two entities' structures may have notable dissimilarities. Hence, we did not embrace the concatenation embedding method outright. Instead, we allotted distinct weights to these two embeddings to differentiate their contributions during training. Bearing this in mind, we integrated a margin-based ranking loss function in the model training process, intended to reduce the embedding

distance of positive pairs and enlarge that of negative pairs. The particular equation is stated as follows:

$$\xi = \sum_{(p,q)\in\mathbb{L},(p',q')\in\mathbb{L}'_{rel}} \left[d_{rel}(p,q) - d_{rel}(p',q') + \gamma_1\right]$$

$$+\Theta\left(\sum_{(p,q)\in\mathbb{L},(p',q')\in\mathbb{L}'_{attr}} \left[d_{attr}(p,q) - d_{attr}(p',q') + \gamma_2\right]\right) \quad (16)$$

Here $[\cdot]^+ = max\{\cdot, 0\}$; $\mathbb{L}'_{rel}$ and $\mathbb{L}'_{attr}$ represent negative pairs based on relation and property embeddings, correspondingly; γ_1 and γ_2 (both > 0) are the margin hyperparameters that separate positive and negative pairs.

3.6. Enriching Triplets with Transitivity Rules

Although the relational embeddings implicitly learn the information of relation transitivity, incorporating this information explicitly augments the number of properties and related entities for each entity, thereby facilitating the identification of similarities between entities. For instance, let us consider the triplet $\langle zhengzhou, locatedIn, HeNan\rangle$ and $\langle HeNan, country, China\rangle$, from this, we can infer the existence of a relationship, namely, "$locatedInCountry$", between the entities "$zhengzhou$" and "$China$". In actuality, this information can be leveraged to enhance the relevant entities "$zhengzhou$". This paper addresses the handling of single-hop transitive relations as follows: Given the relationship triplets $\langle h_1, r_1, t_1\rangle$ and $\langle t_1, r_2, t_2\rangle$, we interpret r_1 and r_2 as the relationships from the head entity h_1 to the tail entity t_2. Therefore, the relationships between these transitive triplets are defined as $h_1 + (r1.r2) \approx t_2$, and by replacing the relationship vector r with $r1.r2$, we can obtain the relationship between h_1 and t_2.

4. Experiments

4.1. Experiment Settings

Environment Information. The experiments were executed on the CentOS 7.5 operating system, utilizing the Intel(R) Xeon(R) CPU E5-2697 v2 @ 2.70 GHz with 256 GB of memory and NVIDIA Quadro K4000 GPU. The programming languages and frameworks employed were Python, TensorFlow, and Torch.

Dataset. This paper uses four 15 K bilingual datasets from OpenEA [28], covering multiple languages including English, French, and German. The data are sourced from well-known knowledge graphs such as YAGO 3 [29], Dbpedia [30], and Wikidata [31]. To assess the performance of the model on datasets of varying densities, OpenEA constructed sparse and dense versions of each dataset. The sparse dataset (V1 version) was directly generated using the IDS algorithm and has features that are more closely aligned with real-world datasets. The dense dataset (V2 version) is based on the V1 version but with low-degree entities (d < 5) removed from the source knowledge graph and then reconstructed using the IDS algorithm, making it more similar to real-world datasets. It should be noted that since the data collection methods for DBpedia, Wikidata, and YAGO 3 are similar, the labels between their entities are highly similar, which may have a certain impact on the evaluation of actual performance. Therefore, in the data preprocessing stage, the label information for the entities was removed. Table 1 presents the statistical details for each dataset, comprising the entity count, relationship count, as well as relationship and property triple counts.

Table 1. Statistics of datasets.

Datasets	KGs	Entities	Relations	Rel.Triples	Pro.Triples
EN-DE-15K-V1	English	15,000	215	47,676	83,755
	German	15,000	131	50,419	156,148
EN-DE-15K-V2	English	15,000	169	84,867	81,988
	German	15,000	96	92,632	186,333
EN-FR-15K-V1	English	15,000	267	47,334	73,121
	French	15,000	210	40,864	67,167
EN-FR-15K-V2	English	15,000	193	96,318	52,355
	French	15,000	166	80,112	56,113

Implementation details. To ensure fairness in the evaluation, this study employed a five-fold cross-validation method. Specifically, the dataset was divided into five non-overlapping parts, with each part accounting for 20% of the total dataset. In each experiment, one part was chosen as the training data, 10% of the remaining data was allocated for validation, and the remaining 70% of the data was used for testing. The hyperparameters were set to a maximum epoch number of 2000; batch size of 5000; embedding dimension of 100; margin of 1.5; and learning rate of 0.001.

Evaluation metrics. In this study, Hits@k (where k = 1, 5, 10) and mean rank (MR) were used as evaluation metrics to measure the performance of aligning entities. These metrics assess the proportion of correctly aligned entities within the top k predicted entities and overall performance: lower MR scores and Higher Hits@k scores indicate superior performance. Specifically, we focus on the Hits@1 performance metric, as it corresponds to the conventional accuracy metric in the field of traditional entity alignment.

Baseline. To evaluate the performance of the proposed PCE-HGTRA model, it was compared against seven contemporary EA models. For an elaborate exposition of the model specifications, please refer to Section 2.

4.2. Main Results and Analysis

Tables 2 and 3 illustrate the cross-lingual performance of various models, all exhibiting bidirectional best alignment outcomes. The tables provide a breakdown of Hits@k in percentages, with the best results of the baseline denoted in bold, while the PCE-HGTRA model's superior performance is signified by underlined numbers.

Table 2. Results from the experiments conducted on EN-DE-15K-V1 and EN-DE-15K-V2 datasets.

Datasets	EN-DE-15K-V1				EN-DE-15K-V2			
Model	**Hits@1**	**Hits@5**	**Hits@10**	**MR**	**Hits@1**	**Hits@5**	**Hits@10**	**MR**
MTransE	11.85	24.15	30.42	586.78	1.9	5.39	7.96	1045.27
JAPE	12.82	26.86	34.13	342.67	2.57	6.92	10.04	847.1
IPTransE	50.23	68.32	76.2	183.01	60.74	76.43	82.02	23.93
RDGCN	43.38	55.31	59.47	828.38	58.57	69.83	73.35	560.06
SEA	53.45	72.31	80.05	125.25	60.55	78.14	**84.23**	**16.07**
IMUSE	**61.01**	**75.38**	**80.33**	**76.75**	**63.03**	**78.32**	82.82	19.71
GCN-Align	47.01	67.86	75.05	150.45	34.97	55.02	62.89	130.81
PCE-HGTRA	62.63	76.77	80.84	68.74	64.97	80.76	85.52	15.93
Improv.best	1.62	1.39	0.51	8.01	1.94	2.44	1.29	0.14

Table 3. Results from the experiments conducted on EN-FR-15K-V1 and EN-FR-15K-V2 datasets.

Datasets	**EN-FR-15K-V1**				**EN-FR-15K-V2**			
Model	**Hits@1**	**Hits@5**	**Hits@10**	**MR**	**Hits@1**	**Hits@5**	**Hits@10**	**MR**
MTransE	18.67	37.56	46.75	284.34	8.88	19.48	25.38	476.9
JAPE	20.94	41.65	51.28	196.16	20.17	38.97	47.73	141.81
IPTransE	23.09	46.56	57.63	454.66	34.08	62.73	74.47	39.65
RDGCN	20.12	33.28	38.27	1513.92	22.72	37.7	43.56	1209.87
SEA	28.64	53.63	64.91	319.98	**35.27**	**64.35**	76.4	**32.81**
IMUSE	**54.20**	**69.00**	**75.11**	**121.81**	34.97	60.08	70.99	34.11
GCN-Align	35.44	60.91	70.01	228.37	35.63	63.81	**79.93**	80.27
PCE-HGTRA	62.14	74.8	79.05	51.95	34.64	61.97	72.85	21.99
Improv.best	7.94	5.48	3.94	69.86	-	-	-	10.82

Results of OpenEA: According to the data in Tables 2 and 3, PCE-HGTRA performed the best on three out of four datasets and still had the best MR index on the fourth dataset. Additionally, it exceeded the best baseline by 1.62% to 7.94% on the Hits@1 indicator. IMUSE's performance on the OpenEA dataset was almost equivalent to that of PCE-HGTRA, effectively utilizing the properties and relationship information existing in the KG. However, PCE-HGTRA still achieved outstanding performance. The OpenEA model reduced the number of relationships and triplets, challenging the modeling ability of sparse KGs. The PCE-HGTRA achieved significant improvements over both dense and sparse KG baselines. Notably, the improvement achieved by our approach on the Hits@1 metric exceeded that on Hits@5, implying that PCE-HGTRA can more precisely identify true entities among the top five alignment candidates that are difficult to distinguish. The experiments showed that PCE-HGTRA can partially address the problem of neighborhood sparsity for certain entities. It is noteworthy that, despite not being able to precisely identify the top 10 alignment candidates on the EN_FR_15K_V2 dataset, our model still achieved an optimal MR index, highlighting the superior recall performance of our method compared to the baseline.

As expected, GNN-based methods generally achieve good results, while entity alignment models based on TransE perform poorly in the EA task due to embeddings of different KGs residing in different vector spaces and a lack of proper conversion methods. MTransE represents entities across distinct vector spaces, albeit with a potential loss of information during the transformation learning process. IPTransE improves upon TransE by specifying corresponding conversion rules between entities through their paths. Methods that overly rely on the number of seed alignments also perform suboptimally on the dataset. In our model, character-level embeddings of properties accurately preserve the similarity of property strings when mapping them to their vector representations, and the transitivity rule allows for the use of more property information during the alignment process, improving the model's performance. Our method outperforms existing embedding-based and GNN-based EA models on the EA task.

4.3. Ablation Study of PCE-HGTRA

Tables 4 and 5 record the results of the ACE-HGTRA model in four different datasets for the ablation study. PCE and HGTRA refer to the property character embedding module and the heterogeneous graph transformer with relation awareness, respectively. The HGTRA module has made significant contributions to the model's performance improvement, indicating that processing the corresponding heterogeneous graphs inside the knowledge graph is of great help in achieving entity alignment tasks. This also reflects that the knowledge graph stores a large amount of heterogeneous information, and our approach can relatively well address such issues by using the model's HGTRA module. We also found that the property character embedding module played an important role in the model's final performance. This module can preliminarily reduce the heterogeneity of the knowledge graph through predicate alignment, providing a foundation for subsequent

work. Furthermore, it can further enrich the available triplets by using transitivity rules. The processed relationship triplets can be transferred to the HGTRA module by the PC module, thereby improving the model's performance. In summary, our model has tried its best to reduce the heterogeneity of the knowledge graph. The experimental results show that the design of these two modules is relatively effective for entity alignment tasks.

Table 4. Ablation study of PCE-HGTRA on the EN-DE-15K-V1 and EN-DE-15K-V2 datasets.

Datasets	**EN-DE-15K-V1**				**EN-DE-15K-V2**			
Model	**Hits@1**	**Hits@5**	**Hits@10**	**MR**	**Hits@1**	**Hits@5**	**Hits@10**	**MR**
PCE-HGTRA	62.63	76.77	80.84	68.74	64.97	80.76	85.52	15.93
PCE-HGTRA w/o PCE	51.85	69.71	76.71	175	62.68	78.87	84.72	20.15
PCE-HGTRA w/o HGTRA	43.38	55.31	59.47	828.38	58.57	69.83	73.35	560.06

Table 5. Ablation study of PCE-HGTRA on the EN-FR-15K-V1 and EN-FR-15K-V2 datasets.

Datasets	**EN-FR-15K-V1**				**EN-FR-15K-V2**			
Model	**Hits@1**	**Hits@5**	**Hits@10**	**MR**	**Hits@1**	**Hits@5**	**Hits@10**	**MR**
PCE-HGTRA	62.14	74.8	79.05	51.95	34.64	61.97	72.85	21.99
PCE-HGTRA w/o PCE	31.03	52.36	61.57	384.8	33.75	64.62	76.33	27.53
PCE-HGTRA w/o HGTRA	20.12	33.28	38.27	1513.95	22.72	37.7	43.56	1209.89

4.4. Efficiency Study of Entity Alignment Method

In this section, we conducted a comparative evaluation of the efficiency of various models, and early stopping was applied to all models. Please refer to Tables 6 and 7 for details.

Table 6. The train+validation+test time (execution time in seconds) of the method in each dataset.

Model/ Datasets	**EN-DE-15K-V1**	**EN-DE-15K-V2**	**EN-FR-15K-V1**	**EN-FR-15K-V2**
MTransE	520.89 + 618.73 + 33.35	407.15 + 562.43 + 40.33	469.68 + 598.83 + 29.64	572.99 + 552.66 + 32.64
JAPE	589.79 + 682.37 + 36.61	538.12 + 617.01 + 41.19	566.28 + 657.26 + 31.01	510.89 + 624.47 + 36.02
IPTransE	20,783.12 + 20,411.01 + 33.91	17,016.05 + 25,541.36 + 40.87	12,777.05 + 13,019.38 + 32.51	20,759.59 + 21,517.86 + 41.32
RDGCN	1911.26 + 6017.83 + 136.29	4239.44 + 2433.15 + 140.60	3901.33 + 3782.80 + 132.75	4206.55 + 7754.57 + 136.24
SEA	5035.92 + 6096.29 + 24.22	5035.92 + 6215.20 + 31.18	3443.45 + 2947.20 + 22.44	5780.66 + 3723.51 + 25.69
IMUSE	8429.14 + 10,718.46 + 194.77	9830.71 + 11,757.12 + 207.65	8375.05 + 10,654.46 + 161.94	9956.91 + 12,242.45 + 132.78
GCN-Align	8276.27 + 9297.59 + 73.61	6503.66 + 10,508.86 + 100.04	5395.27 + 6911.49 + 100.01	6356.06 + 10,320.45 + 104.57
PCE-HGTRA	7290.52 + 10,545.98 + 26.56	7828.82 + 9892.24 + 31.76	5167.42 + 6095.57 + 25.95	4524.95 + 7217.73 + 27.88

Table 7. Number of epochs used for training and validation.

Model/Datasets	**EN-DE-15K-V1**	**EN-DE-15K-V2**	**EN-FR-15K-V1**	**EN-FR-15K-V2**
MTransE	40 + 40	30 + 30	40 + 40	30 + 30
JAPE	40 + 40	30 + 30	40 + 40	30 + 30
IPTransE	710 + 560	410 + 610	310 + 210	410 + 350
RDGCN	70 + 120	70 + 60	70 + 70	70 + 160
SEA	270 + 380	270 + 290	240 + 180	330 + 180
IMUSE	1000 + 1000	1000 + 1000	1000 + 1000	1000 + 1000
GCN-Align	2000 + 2000	2000 + 2000	2000 + 2000	2000 + 2000
PCE-HGTRA	360 + 360	270 + 280	220 + 230	170 + 260

It is worth mentioning that in Table 6, in order to objectively evaluate the running efficiency of each model, we measured the average execution time of training, validation, and testing during five runs. Among all the methods, MTransE has the lightest implementation with only two layers of entity and relation embeddings and a simpler scoring

function, thus it has the shortest running time. Except for MTransE and JAPE, RDGCN achieved the best performance on the four datasets mentioned above due to its utilization of a wider entity neighborhood in the knowledge graph during training. This method requires more training time, i.e., longer epochs, and for specific information please refer to Table 7. However, overall, RDGCN's efficiency is considerable, and its convergence speed is faster. In addition, RDGCN is faster than our unsupervised method PCE-HGTRA, because PCE-HGTRA needs to iterate multiple times in the character embedding module, using all possible descriptive text values to reinforce the quality of embedding, while the latter can better adapt to situations where seeds are often unavailable in real-world environments. Overall, our method can achieve model convergence with fewer epochs compared to other methods. In terms of total running time, our model does not have an advantage because frequent interactions between the two modules are required, and an iterative mechanism is added to increase the number of available entities to achieve better performance to cope with various complex situations in real-world environments.

5. Conclusions

The translation-based approach typically embeds two different knowledge graphs into separate vector spaces and then uses technical means to transform the content of the two vector spaces into the same space to achieve entity alignment. However, this method generally achieves poor results when faced with entity alignment tasks. This is primarily due to their inadequate consideration of the heterogeneous information present in knowledge graphs and their inability to effectively extract useful heterogeneous information for the alignment of entities. Consequently, we present a novel entity alignment framework, PCE-HGTRA, which capitalizes on relationship and property triplets, as well as other forms of heterogeneous information to fully leverage the inherent information present in the knowledge graph, thereby enhancing alignment accuracy. We improved the heterogeneous model HGTRA for property and relationship structures, enabling the model to better capture heterogeneous neighborhood similarity during entity alignment. Experimental results demonstrate the superior performance of PCE-HGTRA compared to most existing models. Moving forward, we plan to explore more effective methods for mining heterogeneous information in the knowledge graph to improve entity alignment effectiveness.

Author Contributions: Validation, H.L.; Formal analysis, Y.Q.; Resources, H.Z.; Writing—original draft, Z.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Project of Science and Technology Tackling Key Problems in Henan Province of China under Grant 222102210234 and 232102210035.

Institutional Review Board Statement: Research does not require ethical approval, which excludes this statement.

Informed Consent Statement: Research not involving humans excludes this statement.

Data Availability Statement: All the data used in the article can be found in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, F.; Yang, L.; Li, J.; Cheng, J. A survey of entity alignment research. *J. Comput. Sci.* **2022**, *45*, 1195–1225.
2. Zhuang, Y.; Li, G.; Feng, J. A survey on knowledge base entity alignment techniques. *J. Comput. Res. Dev.* **2016**, *53*, 165–192.
3. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781.
4. Meng, P. A survey on entity alignment based on graph neural networks. *Mod. Comput.* **2020**, *2020*, 37–40.
5. Xu, Y.; Zhang, H.; Cheng, K.; Liao, X.; Zhang, Z.; Li, L. A survey of knowledge graph embedding. *J. Comput. Eng. Appl.* **2022**, *58*, 30–50.
6. Wang, Z.; Lv, Q.; Lan, X.; Zhang, Y. Cross-lingual Knowledge Graph Alignment via Graph Convolutional Networks. In Proceedings of the EMNLP, Brussels, Belgium, 31 October–4 November 2018.

7. Wu, Y.; Liu, X.; Feng, Y.; Wang, Z.; Yan, R.; Zhao, D. Relation-Aware Entity Alignment for Heterogeneous Knowledge Graphs. In Proceedings of the IJCAI, Macao, China, 10–16 August 2019.
8. Sun, Z.; Wang, C.; Hu, W.; Chen, M.; Dai, J.; Zhang, W.; Qu, Y. Knowledge Graph Alignment Network with Gated Multi-Hop Neighborhood Aggregation. In Proceedings of the AAAI, New York, NY, USA, 7–12 February 2020.
9. Teong, K.S.; Soon, L.K.; Su, T.T. Schema-Agnostic Entity Matching using pre-trained Language Models. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Virtual Event, Ireland, 19–23 October 2020; pp. 2241–2244.
10. Kang, S.; Ji, L.; Liu, S.; Ding, Y. A Cross-lingual Entity Alignment Model Based on Entity Descriptions and Knowledge Vector Similarity. *J. Electron.* **2019**, *47*, 1841–1847.
11. Wu, Y.; Liu, X.; Feng, Y.; Wang, Z.; Zhao, D. Jointly Learning Entity and Relation Representations for Entity Alignment. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 240–249.
12. Wu, Y.; Liu, X.; Feng, Y.; Wang, Z.; Zhao, D. Neighborhood Matching Network for Entity Alignment. In Proceedings of the ACL, Online, 5–10 July 2020.
13. Li, Y.; Gu, C.; Dullien, T.; Vinyals, O.; Kohli, P. Graph Matching Networks for Learning the Similarity of Graph Structured Objects. In Proceedings of the ICML, Long Beach, CA, USA, 9–15 June 2019.
14. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-relational Data. In Proceedings of the NeurIPS, Lake Tahoe, Nevada, USA, 5–10 December 2013.
15. Chen, M.; Tian, Y.; Yang, M.; Zaniolo, C. Multilingual Knowledge Graph Embeddings for Cross-lingual Knowledge Alignment. In Proceedings of the IJCAI, Melbourne, Australia, 19–25 August 2017.
16. Sun, Z.; Hu, W.; Li, C. Cross-lingual entity alignment via joint property-preserving embedding. In Proceedings of the International Semantic Web Conference (ISWC), Vienna, Austria, 21–25 October 2017; pp. 628–644.
17. Zhu, H.; Xie, R.; Liu, Z.; Sun, M. Iterative Entity Alignment via Joint Knowledge Embeddings. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017.
18. Pei, S.; Yu, L.; Hoehndorf, R.; Zhang, X. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In Proceedings of the WWW, San Francisco, CA, USA, 13–17 May 2019; pp. 3130–3136.
19. Schlichtkrull, M.; Kipf, T.N.; Bloem, P.; Van Den Berg, R.; Titov, I.; Welling, M. Modeling relational data with graph convolutional networks. In Proceedings of the ESWC, Heraklion, Crete, Greece, 3–7 June 2018.
20. Busbridge, D.; Sherburn, D.; Cavallo, P.; Hammerla, N.Y. Relational Graph Attention Networks. In Proceedings of the ICLR, New Orleans, LA, USA, 6–9 May 2019.
21. Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; Yu, P.S. Heterogeneous Graph Attention Network. In Proceedings of the WWW, San Francisco, CA, USA, 13–17 May 2019.
22. Zhang, C.; Song, D.; Huang, C.; Swami, A.; Chawla, N.V. Heterogeneous Graph Neural Network. In Proceedings of the SIGKDD, Anchorage, AK, USA, 4–8 August 2019.
23. Hu, Z.; Dong, Y.; Wang, K.; Sun, Y. Heterogeneous Graph Transformer. In Proceedings of the WWW, Taiwan, China, 20–24 April 2020.
24. Mei, X.; Cai, X.; Yang, L.; Wang, N. Relation-aware Heterogeneous Graph Transformer based drug repurposing. *Expert Syst. Appl.* **2022**, *190*, 116165. [CrossRef]
25. Yang, Y.; Huang, C.; Xia, L.; Li, C. Knowledge Graph Contrastive Learning for Recommendation. In Proceedings of the SIGIR, Madrid, Spain, 11–15 July 2022.
26. Cao, X.; Shi, Y.; Wang, J.; Yu, H.; Wang, X.; Yan, Z. Cross-modal Knowledge Graph Contrastive Learning for Machine Learning Method Recommendation. In Proceedings of the ACM MM, Lisboa, Portugal, 10–14 October 2022.
27. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the CVPR, Las Vegas, NV, USA, 27–30 June 2016.
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. In Proceedings of the PVLDB, Online, 31 August–4 September 2020.
29. Rebele, T.; Suchanek, F.M.; Hoffart, J.; Biega, J.; Kuzey, E.; Weikum, G. YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames. In Proceedings of the ISWC, Kobe, Japan, 17–21 October 2016; Volume LNCS 9982, pp. 177–185.
30. Vrandecic, D.; Krotzsch, M. Wikidata: A free collaborative knowledgebase. *Commun. ACM* **2014**, *57*, 78–85. [CrossRef]
31. Lehmann, J.; Isele, R.; Jakob, M.; Jentzsch, A.; Kontokostas, D.; Mendes, P.N.; Hellmann, S.; Morsey, M.; van Kleef, P.; Auer, S.; et al. DBpedia–A large-scale, multilingual knowledge base extracted from wikipedia. *Semant. Web J.* **2015**, *6*, 167–195. [CrossRef]