electronics

Article

Embedding Hierarchical Tree Structure of Concepts in Knowledge Graph Embedding

Jibin Yu [1], Chunhong Zhang [2], Zheng Hu [1,*] and Yang Ji [2]

[1] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China; yujibin@bupt.edu.cn
[2] Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China; zhangch@bupt.edu.cn (C.Z.); jiyang@bupt.edu.cn (Y.J.)
* Correspondence: huzheng@bupt.edu.cn

Abstract: Knowledge Graph Embedding aims to encode both entities and relations into a continuous low-dimensional vector space, which is crucial for knowledge-driven application scenarios. As abstract entities in knowledge graphs, concepts inherently possess unique hierarchical structures and encompass rich semantic information. Although existing methods for jointly embedding concepts and instances achieve promising performance, they still face two issues: (1) They fail to explicitly reconstruct the hierarchical tree structure of concepts in the embedding space; (2) They ignore disjoint concept pairs and overlapping concept pairs derived from concepts. In this paper, we propose a novel concept representation approach, called Hyper Spherical Cone Concept Embedding (HCCE), to explicitly model the hierarchical tree structure of concepts in the embedding space. Specifically, HCCE represents each concept as a hyperspherical cone and each instance as a vector, maintaining the anisotropy of concept embeddings. We propose two variant methods to explore the impact of embedding concepts and instances in the same or different spaces. Moreover, we design score functions for disjoint concept pairs and overlapping concept pairs, using relative position relations to incorporate them seamlessly into our geometric models. Experimental results on three benchmark datasets show that HCCE outperforms most existing state-of-the-art methods on concept-related triples and achieves competitive results on instance-related triples. The visualization of embedding results intuitively shows the hierarchical tree structure of concepts in the embedding space.

Keywords: knowledge graph embedding; concept; hierarchical structure; representation learning

Citation: Yu, J.; Zhang, C.; Hu, Z.; Ji, Y. Embedding Hierarchical Tree Structure of Concepts in Knowledge Graph Embedding. *Electronics* **2024**, *13*, 4486. https://doi.org/10.3390/electronics13224486

Academic Editor: Arkaitz Zubiaga

Received: 22 October 2024
Revised: 11 November 2024
Accepted: 12 November 2024
Published: 15 November 2024

1. Introduction

Concepts, as an important part of ontology, and instances collectively compose entities in knowledge bases such as Wordnet [1], YAGO [2] and Freebase [3]. Knowledge Graph Embedding (KGE) encodes entities and relations into continuous low-dimensional vector spaces while capturing their semantics and preserving the inherent structure of the Knowledge Graph (KG). Compared to traditional knowledge graph embeddings, KGE methods with concept embeddings offer significant improvements in semantic understanding, generalization, and integration with language models, making them highly effective for various downstream applications such as relation extraction [4], question answering [5], dialogue agent [6], and Concept-Enhanced Pre-Training Model [7].

Some studies have achieved promising performance by differentiating and jointly embedding concepts and instances. Depending on the representation form of concepts, these methods can be classified into vectorial methods (e.g., JOIE [8]) and geometric methods (e.g., TransC [9]). Specifically, vectorial methods map both instances and concepts into vectors. This uniform representation makes it challenging for models to distinguish between their different characteristics. In comparison, geometric methods represent each concept as a hyper geometric region and each instance as a vector to explicitly distinguish them.

Furthermore, geometric methods exploit relative positions between embedding representations to model isA relations (instanceOf and subClassOf), thus providing an interpretation that aligns with human intuition for maintaining the transitivity of isA relations.

Nevertheless, existing geometric methods do not explicitly model the unique hierarchical tree structure of concepts in embedding space. This structure naturally forms through the subClassOf relation between concepts and their sub-concepts (or super-concepts), as illustrated on the left of Figure 1. The concept tree structure effectively demonstrates the hierarchical relationships between concepts. Higher-level concepts are typically more abstract and general, while lower-level concepts are more specific and detailed. Moreover, a concept can be reified by all of its actual or potential instances [9]. The instanceOf relation between instances and different levels of concepts in a concept tree reflects the hierarchy of classification and the scope of concepts (i.e., the higher-level concepts contain more instances than the lower-level concepts). Therefore, establishing a hierarchical tree structure in vector space can link the symbolic representation and vector representation in knowledge graphs. This provides an intuitive explanation for concept reasoning and the transitivity of isA relations and aids in positioning concept and instance representations within the semantic space.

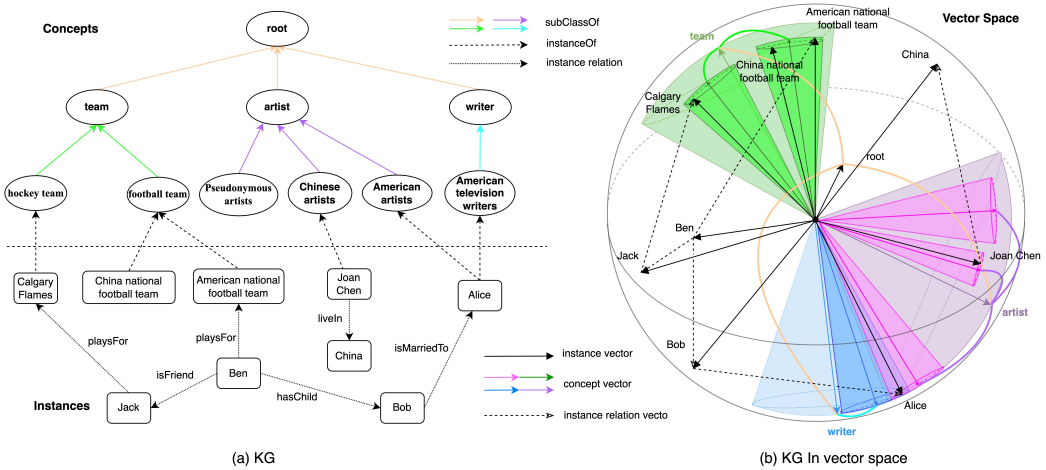


Figure 1. An example of knowledge graph consisting of concepts, instances, and relations (**a**); the representation of instances, concepts, and relations in the vector space (**b**).

To preserve the hierarchical tree structure between concepts and perform logical inference in a vector space, which was inspired by JoSH [10] and geometric methods, we represent each concept embedding as a hyperspherical cone region and each instance as a vector point in a unit of high-dimensional spherical space. Specifically, a hyperspherical cone region is determined by a concept central vector as the axis and a half-vertex angle. As shown in the Figure 1b, higher-level concepts, represented as larger hyperspherical cone regions, iteratively encompass their sub-concepts, which are represented as smaller hyperspherical cone regions, thus illustrating the hierarchical structure of concepts in the embedding space. Starting from the root vector and connecting the central vectors of these concept regions layer by layer reconstructs the tree structure of concepts in the knowledge graph in Figure 1a.

In contrast to existing geometric methods, HCCE represents the anisotropy of concept embeddings by allowing the lengths of the spherical cap sections of the hyperspherical cone to vary across different dimensions, which can model the anisotropy of concept embeddings. The sphere [9] ignores the anisotropy of concept embeddings. Compared to the cuboid and ellipsoid [11] and box [12], HCCE requires fewer parameters and simpler calculations. Compared to using hyperbolic embedding for modeling hierarchical structures [13], our approach to modeling high-dimensional spherical cones in Euclidean space makes the model easier to understand and easier to apply to downstream tasks

while reducing computational complexity. Unlike existing conical modeling methods [14], HCCE represents each concept as a high-dimensional spherical cone region composed of all dimensions. This not only strengthens the connections between dimensions but also provides better interpretability and simpler computational methods.

Models that do not differentiate between concepts and instances can only embed entities in the same embedding space. However, distinguishing between instances and concepts provides the option to embed concepts and instances into two different vector spaces. Based on JOIE [8] and DGS [15], we not only explore embedding concepts and instances in the same space (HCCE-SS) but also model them in different spaces to capture their distinct structures (HCCE-DS). In our models, the primary distinction between these two methods lies in the modeling of the instanceOf relation. Specifically, for the same space, the instanceOf relation is modeled by having instances fall within the concept region. For different spaces, we first use non-linear transformations to map instances from the instance embedding space to the concept embedding space, then determine if the transformed instance vector falls within the concept region in order to model the instanceOf relation.

Additionally, based on the tree structure of concepts, two special types of concept pairs naturally occur in knowledge graphs. To the best of our knowledge, we are the first to propose and define these concept pairs as follows: (1) Overlapping concept pairs: These two concepts contain the same instance, but the subClassOf relation does not exist between them. For example, two concepts "American artists" and "American television writers" have the same instance, "Alice", in Figure 1. (2) Disjoint concept pairs: These two concepts belong to the same super-concept, but they do not comprise a subClassOf relation and do not have the same instance. In Figure 1, two concepts "Chinese artists" and "American artists" belong to the same super-concept "artist". These two pairs of concepts can be seen as intrinsic enhancement samples for concepts. They can accelerate the formation of concept regions, improve concept representation, and enhance the performance of the subClassOf relation. However, existing geometric methods do not account for the above two concept pairs.

In this paper, we propose a novel concept representation, HCCE, where each concept is encoded as a hyperspherical cone, and each instance is encoded as a vector point. We explicitly model the hierarchical tree structure of concepts in the embedding space while preserving the transitivity of isA relations as geometric methods. Additionally, we utilize the relative positions to design score functions for overlapping concept pairs and disjoint concept pairs, thus enhancing the model's ability to capture relationships between concepts and improving the performance of concept representations. We explore the impact of spatial configuration on HCCE by modeling concepts and instances in both the same space (SS) and different spaces (DS). Finally, we perform exhaustive empirical experiments on three benchmark datasets, which demonstrate the superiority of HCCE. We use t-SNE visualization to demonstrate the embedding effect of our models on the concept tree structure, concepts, and instances in the embedding space. Additionally, by illustrating the relationship between the size of the concept embedding regions and the number of concepts and instances, we validate the model's rationality and effectiveness, providing a basis for geometric modeling methodologies.

Our contributions can be summarized as follows:

1. To the best of our knowledge, our method, HCCE, is the first to explicitly model the hierarchical tree structure of concepts by representing each concept as a hyperspherical cone region in the embedding space.
2. We propose two methods, HCCE-SS and HCCE-DS, to explore the impact of embedding instances and concepts in the same or different embedding spaces.
3. Based on the tree structure of concepts, we identify two types of concept pairs—overlapping concept pairs and disjoint concept pairs—and design two score functions for them by utilizing relative positions to enhance the representation of concept embeddings.

4. Through exhaustive experiments on three benchmark datasets, HCCE achieves the best performance on the subClassOf relation and competitive performance on instance-related relations.

2. Related Works

2.1. Instance Knowledge Graph Embedding

Early knowledge embedding methods, called factual knowledge graph embedding, treat both concepts and instances as instances. They can be divided into three types: Tensor Decomposition Models, Geometric Models, and Neural Network Models.

Tensor Decomposition Models model the KG as a three-way tensor, which is decomposed into a combination of low-dimensional vectors or matrices. RESCAL [16] is an early model in this line, introducing three-way rank-r factorization across each relational slice of the knowledge graph tensor. DistMult [17] simplifies RESCAL by restricting relation matrices to diagonals, effectively modeling symmetric relations but lacking support for asymmetric ones. ComplEx [18] also employs diagonal matrices but extends into a complex vector space, allowing for both symmetric and asymmetric relations. HolE [19] reduces time and space complexity by using circular correlation rather than bilinear products, thus accommodating various types of relations. SimplE [20] builds on DistMult by giving each entity and relation two embeddings, which enables it to capture asymmetry. TuckER [21] employs Tucker decomposition [22] to factorize a tensor, achieving better performance than the aforementioned models.

Geometric Models interpret relations as geometric transformations in the embedding space. TransE [23] represents the relation in each triple as a translation from the head entity to the tail entity and is the foundational geometric model. Building on TransE, various methods have been proposed to improve its performance in handling complex and multiple relations, including TransM [24], TransH [25], TransR/CtransR [26], TransD [27], TransMS [28], Trans4E [29], etc. RotatE [30] represents each relation as a rotation from the source entity to the target entity in the complex vector space to model and infer various relation patterns, including symmetry/antisymmetry, inversion, and composition. Furthermore, HAKE [31] uses the polar coordinate system to model the semantic hierarchies in the knowledge graph. Additionally, an ensemble model [32] combines translation and rotation models to capture structural features and employs a bi-encoder architecture based on BERT to capture semantic features. HBE [13] applies an extended Poincaré Ball for KGE to capture hierarchical structures. HMI [33] utilizes hyperbolic embeddings to handle structured multi-label prediction tasks. However, hyperbolic models involve more complex calculations than Euclidean operations, which limits their application in knowledge graph embeddings. Compared to hyperbolic model RotH [34], RotL [35] proposes a lightweight variant based on Euclidean calculations to reduces half of the training time.

Neural Network Models use neural networks for encoding semantic matching, yielding remarkable predictive performance. (1) Neural Networks: SMEs [36] first utilize linear/bilinear blocks as neural networks to capture semantic and structural features in knowledge graphs. In addition, multi-layer perceptrons (MLPs) [37], neural tensor networks (NTNs) [38], and neural association models (NAMs) [39] take entities or relations, or both of them, into deep neural networks and compute a matching score. (2) Convolutional Neural Networks: ConvE [40] uses 2D convolution over embeddings and multiple layers of nonlinear features to model the interactions between entities and relations by reshaping the head entity and relation into a 2D matrix. ConvKB [41] adopts CNNs for encoding the concatenation of entities and relations without reshaping. ConvR [42] applies convolutional filters adaptively constructed from relation representations across entity representations to generate convolutional features. (3) Recurrent Neural Networks (RNNs). RSN [43] designs a recurrent skip mechanism to enhance semantic representation learning by distinguishing relations and entities. (4) Graph Neural Networks: R-GCN [44] proposes relation-specific transformation to model the directed nature of knowledge graphs. CompGCN [45] proposes entity-relation composition operations over each edge in the neighborhood of a cen-

tral node and generalizes previous GCN-based models. (5) Pre-trained language models. KG-BERT [46] borrows the idea from language model pre-training and takes Bidirectional Encoder Representations from the Transformer (BERT) model as an encoder for entities and relations. LMKE [47] adopts language models as knowledge embeddings and utilizes textual descriptions to solve the problem of long-tail entities. KG-LLM [48] explores Large Language Models for KGC. Although neural network models often enhance knowledge graph performance and facilitate the integration of textual information, they also introduce challenges such as low interpretability and high computational complexity.

2.2. Concept Knowledge Graph Embedding

Concept knowledge graph embedding treats both instances and concepts (or ontology) as concepts. On2Vec [49] proposed a novel translation-based graph embedding method for the ontology population. It integrates two components, a Component-specific Model and a Hierarchy Model, to effectively characterize comprehensive relation facts in ontology graphs. The research [50] studied how to use the knowledge graph embedding to ontology embedding. It firstly shows that the most popular method in the previous KGE is not applicable to the ontology population, and it proposes a novel method of modeling the relation as an arbitrary convex region, which can properly express the class of so-called quasi-chained existential rules. Embeds [51] proposes a method of using geometric interpretation to incorporate ontology information in the knowledge graph embedding. CosE [52] defined two score functions based on angle-based semantic space and translation-based semantic space to simultaneously preserve the transitivity of subClassOf and the symmetry of and disjointWith. Concept2Vec [53] utilizes a random walk strategy to learn concept embedding through the similarity between concepts. Furthermore, OWL2Vec* [54] encodes the semantics of an OWL ontology by taking into account its graph structure, lexical information, and logical constructors. However, due to the nature of their learning strategies like random walk, the methods like OWL2Vec* fail to distinguish concepts and instances clearly.

Additionally, some researchers introduce concepts as auxiliary information in knowledge graph embeddings but overlook that concepts are themselves a type of entity. For example, Entity Hierarchy Embedding [55] learns a distance metric for each category node and measures entity vector similarity under an aggregated metric. SSE [56] combines the semantic categories of entities to smoothly embed entities belonging to the same category into the semantic space. KEC [57] incorporates concept information into instance embedding by characterizing the semantic correlation between concepts and instances to improve the representation of knowledge graphs. TransO [58] utilizes ontology information to design relation and type constraints and hierarchical structure information constraints to enhance instance-view knowledge representation. The research. [59] brings in feature embeddings of concepts corresponding to entities to learn the semantic information implicit in the ontology and builds type-constrained regular loss to alleviate the problem of missing ontology information in order to improve the inductive inference performance on newly emerging entities. RMPI [60] injects the KG's ontological schema for richer relation semantics to improve inductive relation inference.

2.3. Distinguish Concepts and Instances in Knowledge Graph Embedding

Some studies have tried to distinguish the difference between concepts and instances in entities in knowledge graph embedding. TransC [9] is the first to propose and formalize the problem. This approach enables the handling of the transitivity of the isA relation. IBKE [12] addresses insufficient concept representation by proposing a box-based model, where concepts are embedded as boxes and instances are represented by vectors in the same semantic space. TransFG [61] is based on the TransC model, replacing TransC's handling of instance relation triples with TransD [27]. Additionally, it embeds concepts and instances into separate spaces to resolve the ambiguity of instances. However, these methods do not consider the anisotropy of concept embeddings. To address this issue, TransEllipsoid and TransCuboid [11] represent each concept as either an ellipsoid or a cuboid, allowing

for a flexible adjustment of the boundaries of concept regions to enhance representational capacity. EIKE [62], based on TransEllipsoid, uses a pre-trained model to process the name of each concept or the concatenation of the name and description as the initial vector for the concept, thereby enhancing concept embedding representation. Although it incorporates semantic information into the concept embedding representation, it increases the number of parameters of the model.

JOIE [8] represents both concepts and instances as vectors. It explores the impact of modeling concepts and instances in the same or different spaces from two perspectives: the ontology view and the instance view. It also examines how different methods for modeling relationships between concepts and instances affect knowledge graph representation. However, it cannot provide interpretability for the transitivity of the isA relations. Furthermore, DGS [15] models instance-view entities in spherical space to capture cyclical relations and ontology-view concepts in hyperbolic space to capture hierarchical relations. Experimental results demonstrate the effectiveness of using non-Euclidean spaces with different curvatures and properties to capture the various structures within the knowledge graph. However, non-Euclidean spaces increase the complexity of applying models to downstream applications. JECI [63] also represents both concepts and instances as vectors, but it organizes hierarchical concepts as a tree structure. It attempts to determine more fine-grained concepts for instances in the instanceOf relation, but the subClassOf relation is not modeled. In addition, OntoEA [64] utilizes the embedding of ontologies (concepts) and instances for entity alignment tasks.

3. Model

In this section, we first introduce our model HCCE, which encodes each concept as a hyperspherical cone region and each instance as a vector in the embedding space. Next, based on relative positions, we use an interpretable approach to design score functions for different triples and concept pairs including instanceOf, subClassOf, instance relation triples, overlapping concept pairs, and disjoint concept pairs in Figure 2. Then, we formally define the loss function to be optimized by our model. Finally, we analyze the space and time complexity of relative models to demonstrate HCCE's advantages.

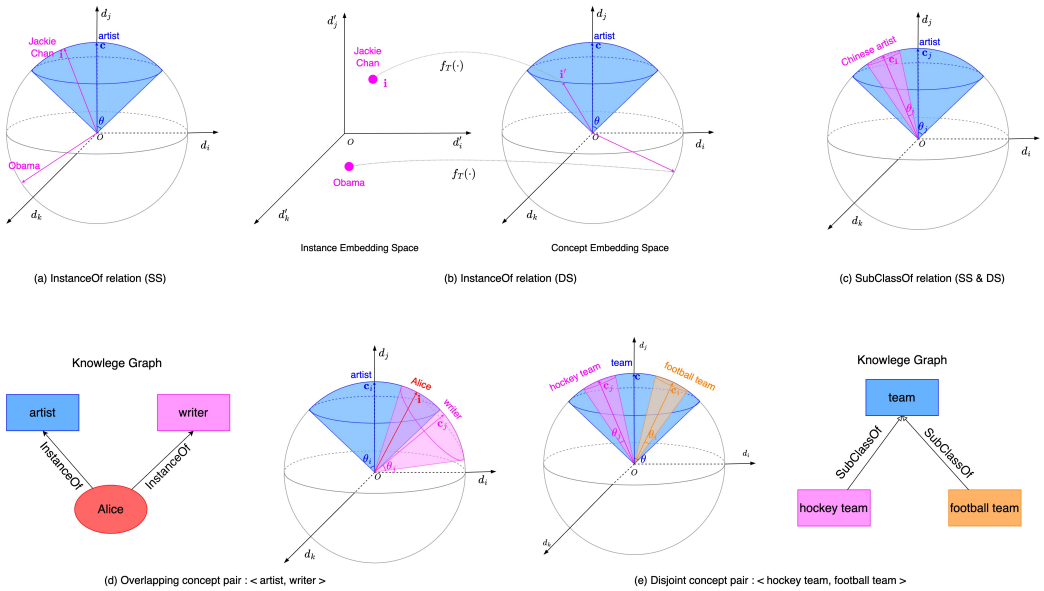


Figure 2. Hyperspherical cone embedding: concepts and their embedding regions are marked with the same color.

3.1. Preliminaries

Given a knowledge graph $\mathcal{KG}$, we represent knowledge graph elements as embeddings in geometric form. Specifically, each instance $i \in \mathcal{I}$ is encoded as a point vector $\mathbf{i} \in \mathbb{R}^d$ in the

embedding space. Each concept $c \in \mathcal{C}$ is encoded as a hyperspherical cone region $G(\mathbf{c}, \theta)$, where concept central vector $\mathbf{c} \in \mathbb{R}^d$ serves as the axis and $\theta \in \mathbb{R}^1$ represents the half-vertex angle. Moreover, to accelerate convergence and reduce computational complexity, we constrain the embedding vectors of both concepts and instances on the unit sphere, which means their norm is equal to 1. Thus, the concept region $G(\mathbf{c}, \theta)$ can be defined by a set as follows:

$$G(\mathbf{c}, \theta) = \left\{ \mathbf{x} \mid \arccos\left(\frac{\mathbf{x} \cdot \mathbf{c}}{\|\mathbf{x}\|_2 \cdot \|\mathbf{c}\|_2} \right) \leqslant \theta, \mathbf{c} \in \mathbb{R}^k, \mathbf{x} \in \mathbb{R}^k, \|\mathbf{x}\|_2 = \|\mathbf{c}\|_2 = 1 \right\}, \quad (1)$$

where $\| \cdot \|_2$ represents the L2-norm. Equation (1) can be interpreted as defining the concept region $G(\mathbf{c}, \theta)$ in the embedding space as the set of vectors $\mathbf{x}$ whose angle with the concept central vector $\mathbf{c}$ is less than or equal to θ.

3.2. Triples and Pairs

3.2.1. InstanceOf Relation Triple

We propose two methods to model instanceOf relation triple to explore the impact of spatial configuration on HCCE: Same Space (SS) and Different Space (DS).

Same Space (SS) We embed instances and concepts in the same space, as shown in Figure 2a. For the triple (*Jackie Chan*, instanceOf, *artist*), the instance *Jackie Chan* is inside the concept *artist* region. In contrast, the instance *Obama* is outside the concept *artist* region because there is no instanceOf relation between them. For a given instanceOf relation triple (i, r_e, c), the instance embedding vector $\mathbf{i}$ should be inside the concept embedding region $G(\mathbf{c}, \theta)$, which is denoted as $\mathbf{i} \in G(\mathbf{c}, \theta)$. The score function is designed to measure whether the instanceOf relation holds:

$$f_{ins}(i, c) = \arccos(\mathbf{i} \cdot \mathbf{c}) - \theta, \quad (2)$$

where $f_{ins} \leqslant 0$, the instanceOf relation r_e holds; otherwise, it does not hold. $\arccos(\cdot)$ is the inverse function of the cosine function.

Different Space (DS) We embed instances and concepts in different vector spaces, as shown in Figure 2b. The instances *Jackie Chan* and *Obama* are embedded in an instance embedding space and the concept *artist* is embedded in concept embedding space. For the triple (*Jackie Chan*, instanceOf , *artist*), the instance *Jackie Chan* in the instance embedding space is mapped inside the region of the concept *artist* in the concept embedding space. Correspondingly, the instance *Obama* is mapped outside the region of the concept *artist* because there is no instanceOf relation between them.

Therefore, for a given instanceOf relation triple (i, r_e, c), we first design a transformation $f_T(\cdot)$ to map the instance vector $\mathbf{i}$ in instance embedding space to an embedding vector point $\mathbf{i}'$ in the concept vector space:

$$\mathbf{i}' = f_T(\mathbf{i}) = \sigma(\mathbf{W} \cdot \mathbf{i} + \mathbf{b}), \quad (3)$$

where $\mathbf{W} \in \mathbb{R}^{d_i \times d_c}$ is a weight matrix and $\mathbf{b}$ is a bias vector. $\sigma(\cdot)$ is a non-linear activation function (we adopt tanh). Thus, the score function is designed to measure whether the instanceOf relation holds:

$$f_{ins}(i, c) = \arccos(f_T(\mathbf{i}) \cdot \mathbf{c}) - \theta, \quad (4)$$

where $f_{ins} \leqslant 0$, the instanceOf relation r_e holds; otherwise, it does not hold.

3.2.2. SubClassOf Relation Triple

For a given subClassOf triple (c_i, r_c, c_j), it should be satisfied that the concept embedding region $G_i(\mathbf{c}_i, \theta_j)$ is contained by the concept embedding region $G_j(\mathbf{c}_j, \theta_j)$, which is denoted as $G_i(\mathbf{c}_i, \theta_j) \subseteq G_j(\mathbf{c}_j, \theta_j)$. For example, as shown in Figure 2c, the *Chinese artist* concept region is contained by the *artist* concept region. Furthermore, according to the

algebraic condition of determining the relative positions of two regions, we design a score function to determine whether the subClassOf relation holds.

$$f_{sub}(c_i, c_j) = \arccos(\mathbf{c}_i \cdot \mathbf{c}_j) - (\theta_j - \theta_i), \tag{5}$$

where $f_{sub} \leqslant 0$, the subClassOf relation holds, otherwise not.

3.2.3. Overlapping Concept Pair

An instance may belong to multiple concepts in a knowledge graph, and two of these concepts can be called overlapping concept pair. For example, as shown in Figure 2d, an instance *Alice* belongs to the concept *artist* and the concept *writer*. Accordingly, in the embedding space, the concept regions corresponding to the concepts *artist* and *writer* have overlapping parts. Extended to the general situation, if two concepts $c_i, c_j \in \mathcal{C}$, $\exists i \in \mathcal{I}$ satisfies $(i, r_e, c_i) \in \mathcal{T}_e$ and $(i, r_e, c_j) \in \mathcal{T}_e$, then $< c_i, c_j >$ can be called overlapping concept pair $\mathcal{P}_o$. Based on the relative position of the two concept regions in the embedding space, we design the following score function to measure the degree of overlap between the concept regions.

$$f_{\mathcal{P}_o}(c_i, c_j) = \arccos(\mathbf{c}_i \cdot \mathbf{c}_j) - (\theta_j + \theta_i). \tag{6}$$

The term $\arccos(\mathbf{c}_i \cdot \mathbf{c}_j)$ calculates the angle between the vectors representing the concepts c_i and c_j. The sum $\theta_j + \theta_i$ represents the combined angular spread of the two concepts. If the angle between the vectors is less than or equal to the combined angular spread, the score function will be non-positive, indicating that the two concept regions are overlapping.

It is worth noting that a concept pair satisfying the subClassOf relation also belongs to overlapping concept pairs, but the Equation (5) imposes stronger limitation than Equation (6). Therefore, if the two aforementioned concepts do not exist in the subClassOf relation, we can add a lower limited condition:

$$f'_{\mathcal{P}_o}(c_i, c_j) = |\theta_j - \theta_i| - \arccos(\mathbf{c}_i \cdot \mathbf{c}_j). \tag{7}$$

If the angle $\arccos(\mathbf{c}_i \cdot \mathbf{c}_j)$ between the vectors is less than or equal to the difference $|\theta_j - \theta_i|$, the score function will be non-positive, ensuring that two concept regions do not collapse into a 'contain' relationship.

3.2.4. Disjoint Concept Pair

Multiple sub-concepts may belong to the same concept in the knowledge graph, two of these sub-concepts can be called disjoint concept pair if there is no subClassOf relation between them. For example, as shown in Figure 2e, the concept *hockey team* and the concept *hockey team* belong to the concept *team*. Accordingly, in the embedding space, the embedding regions corresponding to the two concepts in the concept pair <*hockey team*, *hockey team*> have no overlapping parts. Extending to the general situation, if three concepts $c_i, c_j, c \in \mathcal{C}$ are such that $(c_i, r_c, c) \in \mathcal{T}_c$, $(c_j, r_c, c) \in \mathcal{T}_c$, $(c_j, r_c, c_i) \notin \mathcal{T}_c$, and $(c_i, r_c, c_j) \notin \mathcal{T}_c$, then $< c_i, c_j >$ can be called a disjoint concept pair $\mathcal{P}_d$. Based on the relative position of the two concept regions in the embedding space, the following score function is designed:

$$f_{\mathcal{P}_d}(c_i, c_j) = (\theta_j + \theta_i) - \arccos(\mathbf{c}_i \cdot \mathbf{c}_j). \tag{8}$$

If the combined angular spread $\theta_j + \theta_i$ is greater than the angle $\arccos(\mathbf{c}_i \cdot \mathbf{c}_j)$ between the vectors, the score function will be positive, indicating that two concept regions are disjoint.

3.2.5. Instance Relation Triple

For the instance triple (i_m, r_l, i_n), we will learn low-dimensional vectors $\mathbf{i}_m, \mathbf{r}_l, \mathbf{i}_n \in \mathbb{R}^d_i$ for instances and relations. In order to be consistent with concept embedding, we utilize TransE [23] models to design the score function:

$$f_{r_l}(i_m, i_n) = \|\mathbf{i}_m + \mathbf{r}_l - \mathbf{i}_n\|_{L_1/L_2}. \tag{9}$$

3.3. Loss Function

For instance relation triples, we use ϕ and ϕ' to denote a positive triple and a negative triple. $\mathcal{T}_e$ and $\mathcal{T}_e'$ are used to describe the positive triple set and negative triple set. As we adopt a margin-based ranking loss:

$$\mathcal{L}_{r_l} = \sum_{\phi \in \mathcal{T}_{r_l}} \sum_{\phi' \in \mathcal{T}_{r_l}'} \left[\gamma_{r_l} + f_{r_l}(\phi) - f_{r_l}(\phi')\right]_+, \tag{10}$$

where γ_{r_l} is the margin separating positive triples and negative triples.

Unlike Equation (9), the score functions (2), (4), and (5) for the instanceOf relation triples and subClassOf relation triples allow the score of positive triples to be less than zero. To more accurately constrain the model's judgment of positive and negative triples, we design a separate margin-based loss to constrain positive and negative examples separately. Specifically, the loss function of instanceOf relation triples is

$$\mathcal{L}_{r_e} = \sum_{\phi \in \mathcal{T}_{r_e}} \left[\gamma_{r_e} + f_{r_e}(\phi)\right]_+ + \sum_{\phi' \in \mathcal{T}_{r_e}'} \left[\gamma_{r_e} - f_{r_e}(\phi')\right]_+, \tag{11}$$

where $[x]_+ \triangleq \max(0, x)$. Similarly, for subClassOf relation triples, we will have a loss function as follows:

$$\mathcal{L}_{r_c} = \sum_{\phi \in \mathcal{T}_{r_c}} \left[\gamma_{r_c} + f_{r_c}(\phi)\right]_+ + \sum_{\phi' \in \mathcal{T}_{r_c}'} \left[\gamma_{r_c} - f_{r_c}(\phi')\right]_+, \tag{12}$$

where γ_{r_e} and γ_{r_c} are the margins separating positive and negative triples of instanceOf relation and subClassOf relation.

For overlapping concept pairs, based on Equations (6) and (7), we design the loss function as follows:

$$\mathcal{L}_{\mathcal{P}_o} = \sum_{\phi \in \mathcal{P}_o} \left[f_{\mathcal{P}_o}(\phi)\right]_+ + \sum_{\phi \in \mathcal{P}_o} \left[f'_{\mathcal{P}_o}(\phi)\right]_+ \tag{13}$$

For disjoint concept paris, based on Equation (8), we design the loss function as follows:

$$\mathcal{L}_{\mathcal{P}_d} = \sum_{\phi \in \mathcal{T}_{\mathcal{P}_d}} \left[f_{\mathcal{P}_d}(\phi)\right]_+ \tag{14}$$

Finally, we define the overall loss function as linear combinations of the aforementioned five functions:

$$\mathcal{L} = \mathcal{L}_{r_e} + \mathcal{L}_{r_c} + \mathcal{L}_{r_l} + \mathcal{L}_{\mathcal{P}_o} + \mathcal{L}_{\mathcal{P}_d}. \tag{15}$$

3.4. Complexity of Models

In Table 1, we list the space and time complexity of all the models used in the Section of Experiments. As the most classic KGE model, TransE has the fewest parameters. In comparison, HCCE-SS only adds approximately N_c parameters, which is the same as TransC. Compared to Transellipsoid, EIKE, and IBKE, HCCE-SS reduces the number of parameters by $N_c d_c$. When N_i is similar to N_c (e.g., in the YAGO39K dataset), HCCE-SS reduces about 33% fewer parameters, and this reduction becomes more significant as N_c increases. Compared to HCCE-SS, HCCE-DS only introduces one additional mapping matrix and a set of bias parameters. Moreover, these parameters depend solely on the embedding dimensions of concepts (d_c) and instances (d_i), which are generally negligible compared to the total number of concepts (N_c) and instances (N_i). The high time complexity of HCCE-DS is due to its approach of mapping instances to the concept space when handling instanceOf relations, which increases computational demands. Overall, HCCE-SS, TransE, and TransC have the lowest time and space complexity among models of the same level.

Table 1. Complexity of several embedding models. N_i, N_c, N_r denote the numbers of instances, concepts, and relations. d_i, d_c, d_r denote the dimension of embeddings of instances, concepts, and relations. For time complexity, d denotes the dimension of embeddings.

Model	O_{space}	O_{Time}
TransE [23]	$O(N_i d_i + N_c d_c + N_r d_r)$	$O(d)$
TransH [25]	$O(N_i d_i + N_c d_c + 2N_r d_r)$	$O(d)$
TransR [26]	$O(N_i d_i + N_c d_c + N_r d_r + N_r d_r d_i)$	$O(d d_r)$
TransD [27]	$O(2N_i d_i + 2N_c d_c + 2N_r d_r)$	$O(d^2)$
HolE [19]	$O(N_i d_i + N_c d_c + N_r d_r)$	$O(d \log d + d)$
DistMult [17]	$O(N_i d_i + N_c d_c + N_r d_r)$	$O(d)$
ComplEx [18]	$O(N_i d_i + N_c d_c + N_r d_r)$	$O(d)$
TransC [9]	$O(N_i d_i + N_c(d_c + 1) + (N_r - 2)d_r)$	$O(d)$
TransFG [61]	$O(3N_i d_i + N_c(2d_c + 1) + 2(N_r - 2)d_r)$	$O(d^2)$
JECI [63]	$O(N_i d_i + N_c d_c + N_r d_r)$	$O(d \log d)$
IBKE [12]	$O(N_i d_i + 2N_c d_c + (N_r - 2)d_r)$	$O(d^2)$
TransCuboid [11]	$O(N_i d_i + 2N_c d_c + (N_r - 2)d_r)$	$O(d)$
TransEllipsoid [11]	$O(N_i d_i + 2N_c d_c + (N_r - 2)d_r)$	$O(d)$
EIKE-UNP-EYE [62]	$O(N_i d_i + 2N_c d_c + (N_r - 2)d_r + d_c)$	$O(d^2)$
EIKE-UNP-MAT [62]	$O(N_i d_i + 2N_c d_c + (N_r - 2)d_r + d_c d_i)$	$O(d^2)$
HCCE-SS (ours)	$O(N_i d_i + N_c(d_c + 1) + (N_r - 2)d_r)$	$O(d)$
HCCE-DS (ours)	$O(N_i d_i + N_c(d_c + 1) + (N_r - 2)d_r + d_c(d_r + 1))$	$O(d^2)$

4. Experimental Preliminaries

4.1. Datasets

We used the datasets YAGO39K, M-YAGO39K, and DB99K-242 in TransEllipsoid [11]. YAGO39K was created by TransC [9] and is based on YAGO, which contains lots of concepts from WordNet and instances from Wikipedia. M-YAGO39K is an extension of the YAGO39K dataset, which is designed to evaluate the model's reasoning ability about the transitivity of isA relations. DB99K-242 is created by removing other relations between concepts except for the subClassOf relation on DB111K-174 [8] extracted from DBpedia. These three datasets are described in detail in Table 2.

Table 2. Statistics of datasets.

Datasets	**YAGO39K**	**M-YAGO39K**	**DB99K-242**
#Instance	39,374	39,374	99,744
#Concept	46,110	46,110	242
#Relation	39	39	298
#Training Relational Triple	354,997	354,997	592,654
#Training InstanceOf Triple	442,836	442,836	89,744
#Training subClassOf Triple	30,181	30,181	111
#Valid (Relational Triple)	9341	9341	32,925
#Test (Relational Triple)	9364	9364	32,925
#Valid (InstanceOf Triple)	5000	8650	4987
#Test (InstanceOf Triple)	5000	8650	4987
#Valid (subClassOf Triple)	1000	1187	13
#Test (subClassOf Triple)	1000	1187	13

4.2. Baselines

Following TransEllipsoid [11], we selected 13 state-of-the-art knowledge representation learning models as baselines for evaluating our proposed approach. Based on whether to distinguish between concepts and instances, baselines are divided into two categories: (1) Methods distinguishing concepts and instances, including TransC [9], TransFG [61],

JECI [63], IBKE [12], TransEllipsoid [11], and TransCuboid [11]. (2) Methods that do not distinguish between concepts and instances, including TransE [23], TransH [25], TransR [26], TransD [27], DistMult [17], HolE [19], ComplEx [18].

Our baselines do not include concept embedding methods such as JOIE and TransO. JOIE [8] requires numerous meta-relations between concepts, which are not present in our datasets. Methods like TransO [58], which utilize concepts as supplementary information without directly modeling them, are not suitable as our baselines. EIKE [62] introduces additional textual information to pre-encode concept vectors, while other models do not include this information, leading to an unfair comparison. Therefore, we only use EIKE-UNP, which initializes concept vectors without encoding ontology information, for comparison.

4.3. Experiment Settings

We implement HCCE on NVIDIA TITAN RTX. As for the training hyper-parameters, we select them from the following ranges: learning rate $\lambda \in \{0.1, 0.01, 0.001\}$, the dimension of the embedding vector $d_i, d_c \in \{50, 100, 150\}$, the three margins $\gamma_e, \gamma_c, \gamma_l \in \{0, 0.1, 0.3, 0.5, 1\}$, norm L_1, L_2. We use grid search to determine the hyper-parameters for different tasks and datasets, which are shown in Table 3.

Table 3. Hyperparameters for different tasks and datasets.

Task	Dataset	Model	Hyperparameters						
			γ_l	γ_e	γ_c	λ	d_i	d_c	norm
Triple Classification	YAGO39K	HCCE-SS	1	0.3	0.1	0.001	100	100	L_2
		HCCE-DS	1	0.1	0.1	0.001	100	100	L_2
	M-YAGO39K	HCCE-SS	1	0.3	0.1	0.001	100	100	L_1
		HCCE-DS	1	0.1	0.1	0.001	100	100	L_2
	DB99K-242	HCCE-SS	1	0.1	0	0.001	100	100	L_2
		HCCE-DS	1	0.1	0	0.001	100	100	L_2
Link Prediction	YAGO39K	HCCE-SS	1	0.1	0.1	0.001	100	100	L_1
		HCCE-DS	1	0.4	0.3	0.001	100	100	L_1
	DB99K-242	HCCE-SS	1	0.1	0	0.001	100	100	L_2
		HCCE-DS	1	0.1	0	0.001	100	100	L_2

5. Experimental Results and Discussion

In this section, we first evaluate the performance of our methods on two standard tasks for Knowledge Graph Embedding (KGE): link prediction [23] and triple classification [38]. Next, we assess the impact of overlapping and disjoint concept pairs through ablation experiments. Then, we present visualizations of concept embeddings and their corresponding instance embeddings to illustrate the hierarchical tree structure of concepts in the embedding space. Additionally, we analyze the half-vertex angle θ to validate the rationality of our models. Finally, we analyze the theoretical implications, practical implications, and limitations of our methods.

5.1. Triple Classification

Triple classification is a binary classification task to judge whether a given triple is correct or not. We conduct classification experiments on three datasets: YAGO39K, M-YAGO39K, and DB99K-242. We trained three types of triples—instance triples, instanceOf triples, and subClassOf triples—simultaneously and show their performance separately.

Evaluation protocol

Following most previous works [9,38], we employ the micro-averaged evaluation metric on four metrics—Accuracy, Precision, Recall, and F1-Score—to evaluate a triple classification task. This approach captures the overall effectiveness of the embedding model across all relations without being biased by class size on three datasets YAGO39K, M-YAGO39K, and DB99K-242. The decision rule for classification is simple: We set a relation-

specific δ_r for each relation triple prediction, including instance relations, instanceOf relation, and subClassOf relation. Then, for a triple, if its score f_r is below the threshold δ_r, the triple is predicted as positive, otherwise negative. The relation-specific threshold δ_r is determined by maximizing classification accuracy on the validation set. The optimal configurations are determined by accuracy in the validation set, which are shown in Table 3.

Result

Table 4 shows the experimental results of subClassOf triple classification on the YAGO39K and M-YAGO39K datasets. Compared with recent competitive models, HCCE-SS and HCCE-DS demonstrate clear improvements in almost all metrics for both datasets. This indicates that using hyperspherical cones to model concepts enhances the representation of concept embeddings. At a significant cost to either precision or recall performance, EIKE-UNP-EYE and TransEllipsoid achieve the highest recall and precision results, respectively. This likely stems from the design of each model: TransEllipsoid restricts concept regions more tightly, while EIKE-UNP-EYE allows broader concept regions. Specifically, compared to our method (HCCE-DS), EIKE-UNP-EYE shows a slight improvement in recall (1% in YAGO39K dataset and 0.85% in the M-YAGO39K dataset), but at a substantial decrease in precision (12.23% lower in the YAGO39K dataset and 12.59% lower in the M-YAGO39K dataset). Similarly, TransEllipsoid achieves higher precision than HCCE-DS (1.1% higher in the YAGO39K dataset and 2.23% higher in M-YAGO39K dataset) but exhibits a notable decline in recall (7.9% lower in YAGO39K dataset and 9.48% lower in M-YAGO39K dataset). As a result, HCCE-DS achieves higher overall F1-Scores, surpassing TransEllipsoid by 2.83% on the YAGO39K dataset and 2.9% on the M-YAGO39K dataset, and outperforming EIKE-UNP-EYE by 7.64% on YAGO39K dataset and 7.55% on the M-YAGO39K dataset. These results suggest that HCCE effectively balances classification metrics to enhance the performance of the subClassOf triplet classification task. The superior performance of HCCE-DS compared to HCCE-SS indicates that modeling concepts separately in the concept embedding space helps capture the hierarchical tree structure of concepts, thereby improving the effectiveness of concept embeddings. The enhanced performance of the HCCE model on M-YAGO39K compared to YAGO39K indicates that the HCCE model can effectively model the transitivity of the subClassOf relation.

Table 4. Experimental results on subClassOf triple classification (%).

Datasets	**YAGO39K**				**M-YAGO39K**			
Metric	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
TransE [23]	77.6	72.2	89.8	80.0	76.9 (−0.7)	72.3 (+0.1)	87.2 (−2.6)	79.0 (−1)
TransH [25]	80.2	76.4	87.5	81.5	79.1 (−1.1)	72.8 (−3.6)	92.9 (+5.4)	81.6 (+0.1)
TransR [26]	80.4	74.7	91.9	82.4	80.0 (−0.4)	73.9 (−0.8)	92.9 (+1)	82.3 (−0.1)
TransD [27]	75.9	70.6	88.8	78.7	76.1 (+0.2)	70.7 (+0.1)	89.0 (+0.2)	78.8 (+0.1)
HolE [19]	70.5	73.9	63.3	68.2	66.6 (−3.9)	72.3 (−1.6)	53.7 (−9.6)	61.7 (−6.5)
DistMult [17]	61.9	68.7	43.7	53.4	60.7 (−1.2)	71.7 (+3)	35.5 (−8.2)	47.7 (−5.7)
ComplEx [18]	61.6	71.5	38.6	50.1	59.8 (−1.8)	65.6 (−5.9)	41.4 (+2.8)	50.7 (+0.6)
TransC (unif) [9]	82.9	77.1	93.7	84.6	83.0 (+0.1)	77.5 (+0.4)	93.1 (−0.6)	84.7 (+0.1)
TransC (bern) [9]	83.7	78.1	93.9	85.2	84.4 (+0.7)	80.7 (+2.6)	90.4 (−3.5)	85.3 (+0.1)
TransFG (unif) [61]	82.8	75.7	96.5	84.8	83.1 (+0.3)	76.5 (+0.8)	95.5 (−1)	84.9 (+0.1)
TransFG (bern) [61]	84.5	78.6	95.2	86.1	84.7 (+0.2)	78.7 (+0.1)	94.1 (−1.1)	85.7 (−0.4)
TransEllipsoid [11]	85.1	**82.1**	89.7	85.7	85.5 (−0.4)	**84.2** (+2.1)	87.4 (−2.3)	85.8 (+0.1)
TransCuboid [11]	74.7	68.3	92.3	78.5	75.5 (+0.8)	69.1 (+0.8)	92.4 (+0.1)	79.1 (+0.6)
EIKE-UNP-EYE [62] *	76.8	68.77	**98.2**	80.89	77.3 (+0.5)	69.38 (+0.61)	**97.73** (−0.47)	81.15 (+0.26)
EIKE-UNP-MAT [62] *	77.8	70.23	96.5	81.3	77.55 (−0.25)	69.99 (−0.24)	96.46 (−0.04)	81.12 (−0.18)
HCCE-SS	85.9	79.72	96.3	87.23	86.52 (+0.62)	80.59 (+0.87)	96.21 (−0.09)	87.71 (+0.48)
HCCE-DS	**87.35**	81.00	97.60	**88.53**	**87.66** (+0.31)	81.97 (+0.97)	96.88 (−0.72)	**88.70** (+0.17)

The values in parentheses (in the M-YAGO39K column) represent the difference in scores from YAGO39K to M-YAGO39K. Results of * are taken from [62], and other results are taken from [11]. Bold font represents the best results.

Table 5 displays the experimental results of instanceOf triple classification on the YAGO39K and M-YAGO39K datasets. On YAGO39K, TransEllipsoid and EIKE-UNP-EYE outperform HCCE-SS and HCCE-DS, potentially because the former models have more parameters. Specifically, according to Tables 1 and 2, TransEllipsoid and EIKE-UNP-EYE have nearly 50% more parameters than HCCE-SS. Compared to TransC, which has a similar number of parameters, HCCE-SS achieves better results across all four metrics. On M-YAGO39K, HCCE-SS outperforms all baselines on almost all metrics. This indicates that using hyperspherical cones for concept modeling enhances the model's performance in handling instanceOf relation. Similar to the results for subClassOf relation, EIKE-UNP-EYE performs slightly better than HCCE-SS in terms of precision but falls short in the other three metrics. This further demonstrates that HCCE can effectively balance classification metrics. Therefore, these results suggest that our HCCE model can effectively improve the performance of the instanceOf relation classification task while significantly reducing the parameter count. Moreover, on both datasets, HCCE-SS outperforms HCCE-DS. We speculate that this may be due to the difficulty of mapping functions accurately and assigning all instances to their corresponding concept regions. This suggests that modeling instances and concepts in the same space can reduce errors from spatial mapping, thereby benefiting the modeling of the instanceOf relation.

Table 5. Experimental results on instanceOf triple classification (%).

Datasets	**YAGO39K**				**M-YAGO39K**			
Metric	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
TransE [23]	82.6	83.6	81.0	82.3	71.0 (−11.6)	81.4 (−2.2)	54.4 (−26.6)	65.2 (−17.1)
TransH [25]	82.9	83.7	81.7	82.7	70.1 (−12.8)	80.4 (−3.3)	53.2 (−28.5)	64.0 (−18.7)
TransR [26]	80.6	79.4	82.5	80.9	70.9 (−9.7)	73.0 (−6.4)	66.3 (−16.2)	69.5 (−11.4)
TransD [27]	83.2	84.4	81.5	82.9	72.5 (−10.7)	73.1 (−11.3)	71.4 (−10.1)	72.2 (−10.7)
HolE [19]	82.3	86.3	76.7	81.2	74.2 (−8.1)	81.4 (−4.9)	62.7 (−14)	70.9 (−10.3)
DistMult [17]	83.9	86.8	80.1	83.3	70.5 (−13.4)	86.1 (−0.7)	49.0 (−31.1)	62.4 (−20.9)
ComplEx [18]	83.3	84.8	81.1	82.9	70.2 (−13.1)	84.4 (−0.4)	49.5 (−31.6)	62.4 (−20.5)
TransC (unif) [9]	80.2	81.6	80.0	79.7	85.5 (+5.3)	88.3 (+6.7)	81.8 (+1.8)	85.0 (+5.3)
TransC (bern) [9]	79.7	83.2	74.4	78.6	85.3 (+5.6)	86.1 (+2.9)	84.2 (+9.8)	85.2 (+6.6)
TransFG (unif) [61]	80.2	82.4	78.6	80.4	85.5 (+5.3)	88.3 (+5.9)	82.0 (+3.4)	85.0 (+4.6)
TransFG (bern) [61]	81.7	83.8	75.5	79.4	85.9 (+4.2)	88.4 (+4.6)	82.2 (+6.7)	85.2 (+5.8)
JECI (cbow) [63]	82.7	84.1	81.0	82.8	86.0 (+3.3)	88.2 (+4.1)	81.2 (+0.2)	84.6 (+1.8)
JECI (sg) [63]	83.9	86.6	83.0	84.8	86.1 (+2.2)	88.7 (+2.1)	84.1 (+1.1)	86.3 (+1.5)
TransEllipsoid [11]	**87.23**	**87.89**	86.36	87.12	87.84 (+0.61)	87.88 (−0.01)	87.78 (+1.42)	87.83 (+0.71)
TransCuboid [11]	79.3	80.43	77.44	78.91	84.77 (+5.47)	87.59 (+7.16)	81.03 (+3.59)	84.18 (+5.27)
EIKE-UNP-EYE [62] *	87.13	86.63	**87.54**	**87.18**	87.77 (+0.64)	**89.91** (+3.28)	85.21 (−2.33)	87.45 (+0.27)
EIKE-UNP-MAT [62] *	85.23	86.43	84.06	85.23	85.24 (+0.01)	86.33 (−0.1)	83.75 (−0.31)	85.02 (−0.21)
HCCE-SS	84.06	86.28	81	83.56	**88.16** (+4.10)	88.32 (+2.04)	**87.95** (+6.95)	**88.14** (+4.58)
HCCE-DS	83.69	84.71	82.22	83.45	88.00 (+4.31)	89.83 (+5.12)	85.70 (+3.48)	87.72 (+4.27)

The values in parentheses (in the M-YAGO39K column) represent the difference in scores from YAGO39K to M-YAGO39K. Results of * are taken from [62], and other results are taken from [11]. Bold font represents the best results.

Table 6 shows the experimental results of subClassOf and instanceOf triple classification on the DB99K-242 dataset. On the DB99K-242 dataset, HCCE achieves only competitive results across all metrics, without the outstanding performance seen on the M-YAGO39K and YAGO39K datasets. This may be due to the DB99K-242 dataset having fewer concepts, fewer subClassOf and instanceOf triples, and a lower ratio of concepts to instances compared to the other two datasets. As a result, the advantages of HCCE in concept modeling are not fully realized, and HCCE may overfit the structure of the instances due to a larger quantity of instances. This is also reflected in the instance triple classification experiments.

Table 6. Triple Classification results on the DB99K-242 dataset.

Task	instanceOf (%)				subClassOf (%)			
Metric	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
TransE [23]	78.82	79.76	77.26	78.49	61.54	**100**	23.08	38
TransC [9]	83.59	92.38	73.21	81.69	57.69	75	23.08	35
TransEllipsoid [11]	93.1	94.81	91.64	93.2	61.54	**100**	23.08	38
TransCuboid [11]	69.91	65.23	85.28	73.92	**76.92**	73.33	**84.62**	**79**
EIKE-UNP-EYE [62] *	**94.21**	**96.17**	**92.1**	**94.09**	73.08	68.75	84.62	75.86
EIKE-UNP-MAT [62] *	78.34	72.4	91.62	80.88	53.85	53.33	61.54	57.14
HCCE-SS	92.33	93.01	91.54	92.27	73.08	71.43	76.92	74.07
HCCE-DS	89.89	92.47	86.87	89.58	69.23	77.78	53.85	63.64

Results of * are taken from [62], and other results are taken from [11]. Bold font represents the best results.

Table 7 shows the instance triple classification results on the YAGO39K and DB99K-242 datasets. On the YAGO39K dataset, HCCE achieves competitive results on all metrics compared to state-of-the-art models. Compared with TransE, HCCE-SS yields better results on all metrics, illustrating that using relative position modeling for instanceOf and subClassOf relations can indirectly benefit instance embedding. Compared to TransEllipsoid and EIKE-UNP-EYE, HCCE-SS achieves better results. The combined performance on instanceOf and subClassOf relations suggests that using hyperspherical cones for concept embedding is more accurate than ellipsoids for spatial distribution, improving instance embedding through the instanceOf relation. TransFG and JECI achieve better performance than HCCE-SS and HCCE-DS. It is because TransFG enhances instance performance beyond TransE, while JECI does not consider subClassOf relations. On the DB99K-242 dataset, HCCE-SS and HCCE-DS outperform other models on three metrics. This echoes the previous results of HCCE on subClassOf and instanceOf triple classification.

Table 7. Experimental results on instance triple classification (%).

Datasets	YAGO39K				DB99K-242			
Metric	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
TransE [23]	92.1	92.8	91.2	92.0	90.5	91.5	89.29	90.38
TransH [25]	90.8	91.2	90.3	90.8	-	-	-	-
TransR [26]	91.7	91.6	91.9	91.7	-	-	-	-
TransD [27]	89.3	88.1	91.0	89.5	-	-	-	-
HolE [19]	92.3	92.6	91.9	92.3	-	-	-	-
DistMult [17]	93.5	93.9	93.0	93.5	-	-	-	-
ComplEx [18]	92.8	92.6	93.1	92.9	-	-	-	-
TransC [9]	93.8	94.8	92.7	93.7	90.17	90.9	89.27	90.08
TransFG [61]	**94.4**	94.7	**93.3**	94.0	-	-	-	-
JECI (cbow) [63]	93.4	94.8	92.6	93.7	-	-	-	-
JECI (sg) [63]	93.9	**95.2**	93.1	**94.1**	-	-	-	-
TransEllipsoid [11]	92.64	92.68	92.6	92.64	93.63	**94.67**	92.46	93.55
TransCuboid [11]	92.18	93.8	90.26	92.02	88.4	90.26	86.1	88.13
EIKE-UNP-EYE [62] *	92.69	92.54	92.87	92.7	92.38	93.46	91.14	92.29
EIKE-UNP-MAT [62] *	93.14	93.5	92.74	93.12	63.6	60.96	75.64	67.51
HCCE-SS	93.17	93.72	92.55	93.13	**94.19**	94.62	**93.71**	**94.16**
HCCE-DS	92.05	91.24	93.03	92.13	94.08	94.59	93.51	94.04

Results of * are taken from [62], and other results are taken from [11]. Bold font represents the best results.

Additionally, we observe that HCCE-DS and HCCE-SS perform similarly across the three datasets with slight differences; for example, HCCE-DS performs better on subClassOf, while HCCE-SS excels on instanceOf, indicating that different space configurations affect the performance of HCCE.

5.2. Instance Link Prediction

Link prediction [65] aims to predict the missing head entity $(?, r, t)$ or tail entity $(h, r, ?)$. For each testing instance triple, we use the method proposed in [66] to replace the head or tail instance from the instance set to construct the corrupted triples, and then such triples are ranked in descending order according to the scores by the score function of Equation (9). We conduct link prediction experiments for instance triples on YAGO39K and DB99K-242 datasets because the instanceOf relation and subClassOf relation are not suitable for this task [11].

Evaluation protocol Following most previous works [9,65], we adopt two evaluation metrics as follows: (1) The mean reciprocal rank of all correct instances (MRR); (2) The proportion of correct instances that rank no larger than N (Hits@N). The higher the value of a model in MRR and Hits@N, the better its performance is. Note that a corrupted triple may have already existed in the knowledge graph, which should be regarded as a correct prediction. The settings "Raw" and "Filter" distinguish whether or not to consider the impact of a corrupted triple already existing in the knowledge graph.

Result Table 8 shows the experimental results of link prediction on the YAGO39K and DB111K-242 datasets. On the YAGO39K dataset, TransEllipsoid and EIKE perform better than HCCE, likely because former models use more parameters to enhance performance. However, with the same number of parameters, the performance of HCCE-SS outperforms TransC's. This indicates that using hyperspherical cones to model concept embeddings is more effective than spheres for improving embedding quality and instance inference. DistMult achieves the the best performance in MRR, possibly because we determine the best configurations based solely on Hits@10, which may lead to a lower MRR. The performance of HCCE-DS is better than HCCE-SS, which may indicate that using the independent embedding space for instances is beneficial for models to instance inference. On the DB111K-242 dataset, the performance of HCCE is similar to it on the YAGO39K dataset, except that HCCE-DS achieves the best performance in Hits@10.

Table 8. Experimental results on link prediction.

Datasets	**YAGO39K**					**DB99K-242**				
Metric	**MRR**		**Hits@N(%)**			**MRR**		**Hits@N(%)**		
	Raw	**Filter**	**1**	**3**	**10**	**Raw**	**Filter**	**1**	**3**	**10**
TransE [23]	0.114	0.248	12.3	28.7	51.1	0.170	0.232	10.2	31.5	45.7
TransH [25]	0.102	0.215	10.4	24.0	45.1	-	-	-	-	-
TransR [26]	0.112	0.289	15.8	33.8	56.7	-	-	-	-	-
TransD [27]	0.113	0.176	8.9	19.0	35.4	-	-	-	-	-
HolE [19]	0.063	0.198	11.0	23.0	38.4	-	-	-	-	-
DistMult [17]	**0.156**	0.362	22.1	43.6	66.0	-	-	-	-	-
ComplEx [18]	0.058	0.362	29.2	40.7	48.1	-	-	-	-	-
TransC [9]	0.112	0.420	29.8	50.2	69.8	0.147	0.188	6.6	25.7	40.8
TransFG [61]	0.114	0.475	32.5	52.1	70.1	-	-	-	-	-
JECI (cbow) [63]	0.088	0.418	28.6	49.7	69.8	-	-	-	-	-
JECI (sg) [63]	0.122	0.441	30.0	51.1	70.1	-	-	-	-	-
IBKE [12]	-	0.522	40.4	60.5	73.1	-	-	-	-	-
TransEllipsoid [11]	0.112	0.536	41.6	**68.7**	75.1	0.170	0.248	11	34.4	48.1
TransCuboid [11]	0.095	0.475	36.3	55.6	66.9	0.149	0.200	4.2	30.4	47.8
EIKE-UNP-EYE [62] *	0.115	**0.577**	**47.1**	65.4	**76.2**	**0.175**	**0.256**	**11.9**	**34.9**	49.4
EIKE-UNP-MAT [62] *	0.113	0.545	42.4	62	74.7	0.045	0.056	0.1	9.2	15
HCCE-SS	0.108	0.439	28.7	52.9	73.9	0.169	0.233	7.38	34.46	49.83
HCCE-DS	0.096	0.442	29.8	53.2	72.3	0.170	0.234	7.49	34.50	**50.06**

Hits@N uses result of "Filter" evaluation setting. Results of * are taken from [62], and other results are taken from [11]. Bold font represents the best results.

5.3. Ablation

In this section, to prove the effectiveness of disjoint concept pairs and overlapping concept pairs, We implement three sets of experiments by separately removing overlapping concept pairs ($\mathcal{L}_{\mathcal{P}_o}$), disjoint concept pairs ($\mathcal{L}_{\mathcal{P}_d}$), and both on YAGO39K and M-YAGO39K dataset. Table 9 shows the results. (1) When we exclude overlapping concept pairs, the performance of subClassOf and instanceOf shows a slight decline. This shows that the overlapping concept pairs are important and can prompt the effectiveness of jointly embedding concepts and instances. (2) When we exclude disjoint concept pairs, the performance degrades again, but significant drop in subClassOf relation effect (more than 10% F1-Score). This may be due to the fact that the entities involved in disjoint concept pairs are all concepts, so the performance on subClassOf relation is significantly improved. Additionally, incorporating disjoint concept pairs allows the model to separate multiple concept regions that belong to the same super-concept in embedding space. This helps improve the model's discrimination of the subClassOf and instanceOf relation triples related to these concepts. (3) When we simultaneously exclude overlapping concept pairs and disjoint concept pairs, compared with other results, this scheme has the worst effect. This continues to prove the effectiveness of overlapping and disjoint concept pairs to improve the performance of jointly embedding concepts and instances.

Table 9. Ablation experimental results on F1-Score of triple classification (%).

Datasets	**YAGO39K**		**M-YAGO39K**	
Triple	**instanceOf**	**subClassOf**	**instanceOf**	**subClassOf**
HCCE (one)	83.94	87.09	88.15	87.35
HCCE (one) w/o $\mathcal{L}_{\mathcal{P}_o}$	82.53	85.59	86.68	85.80
HCCE (one) w/o $\mathcal{L}_{\mathcal{P}_d}$	82.47	76.04	86.31	75.85
HCCE (one) w/o $\mathcal{L}_{\mathcal{P}_o}$ & $\mathcal{L}_{\mathcal{P}_d}$	81.85	76.21	86.18	75.82
HCCE (two)	83.45	88.53	87.72	88.70
HCCE (two) w/o $\mathcal{L}_{\mathcal{P}_o}$	83.05	83.61	87.05	83.96
HCCE (two) w/o $\mathcal{L}_{\mathcal{P}_d}$	82.34	74.33	86.94	74.86
HCCE (two) w/o $\mathcal{L}_{\mathcal{P}_o}$ & $\mathcal{L}_{\mathcal{P}_d}$	82.68	73.89	87.00	73.32

5.4. Tree Structure Visualization

To understand how concepts and instances are distributed in the embedding space and how the hierarchical tree structure of concepts is modeled, we apply t-SNE [67] on YAGO39K to visualize the embedding space in Figure 3. We utilize t-SNE to depict the instance embeddings $\mathbf{i}$ and the concept embeddings $\mathbf{c}$ of $G(\mathbf{c}, \theta)$. For HCCE-DS, each instance is mapped to an embedding point in the concept vector space.

In Figure 3(a3,b3), we observe the following: (1) Instances surround their corresponding concepts. In the visualizations, each star representing a concept is surrounded by points of the same color, which represent instances. (2) Sub-concepts surround their super-concepts, forming a concept tree structure. To clearly illustrate this structure, we connect super-concepts and sub-concepts layer by layer from the "ROOT" concept, reconstructing the hierarchical tree structure of concepts in the knowledge graph. Furthermore, sub-concepts belonging to the same super-concept are closer together, which include the sub-concepts belonging to the concepts "party" and "hockey team". To enhance clarity regarding the distribution of these two concepts along with their respective subconcepts and instances, we have separately depicted two subtree structures, as shown in Figure 3(a1,a2,b1,b2). From these visualizations, we observe the following: (1) Instances belonging to the super-concepts fill the entire space, with black dots spread throughout sub-figure (e.g., Figure 3(a1)). (2) Instances belonging to different sub-concepts exhibit distinct partition clusters. In Figure 3(a1), the three different color dots are concentrated in the upper left, upper right, and lower left parts of the sub-figure. These visualizations

demonstrate that HCCE can distinguish and model different sub-concepts belonging to the same super-concepts. Importantly, as anticipated, both HCCE-SS and HCCE-DS yield similar t-SNE visualization results, affirming that the HCCE method can universally model concepts and instances in either the same or different spaces.

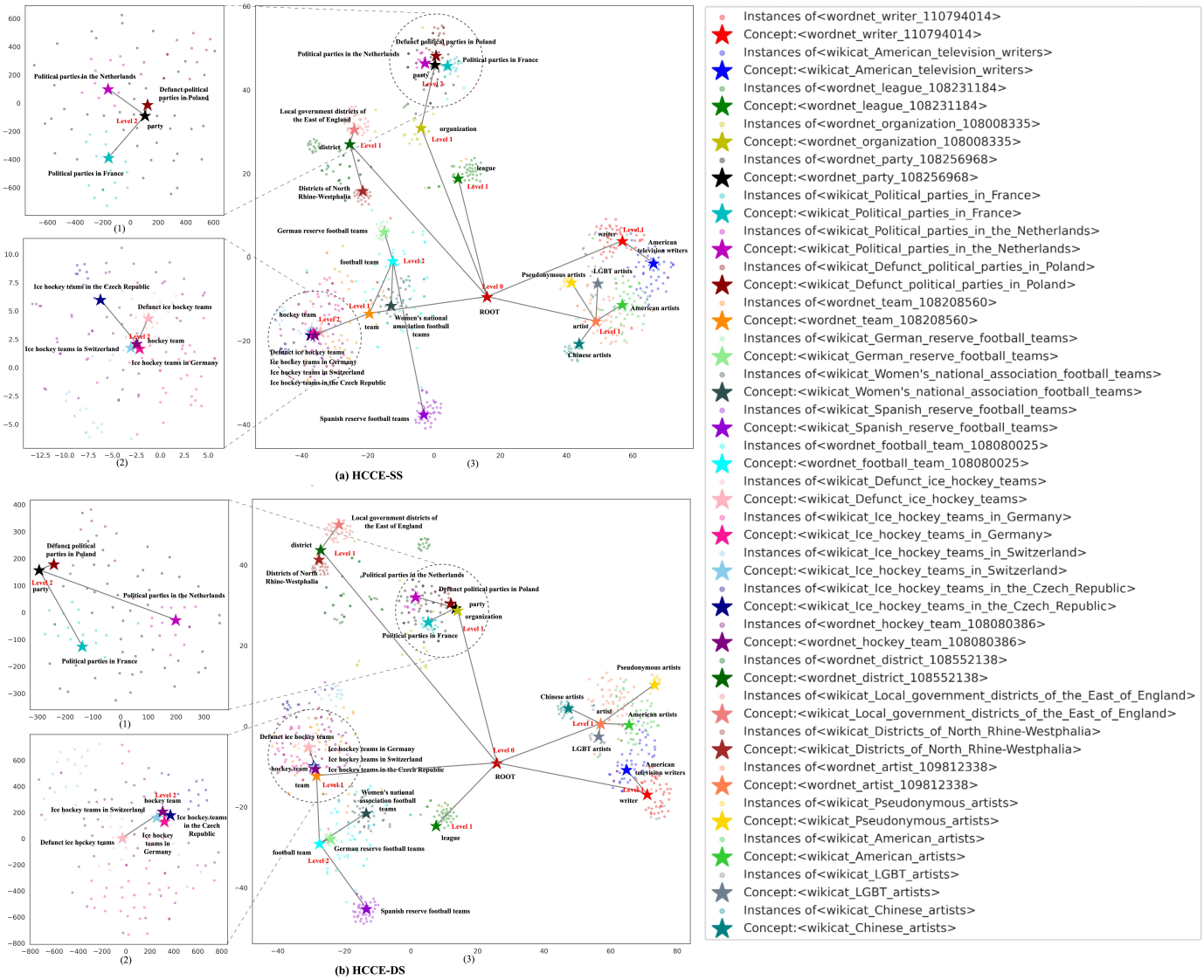
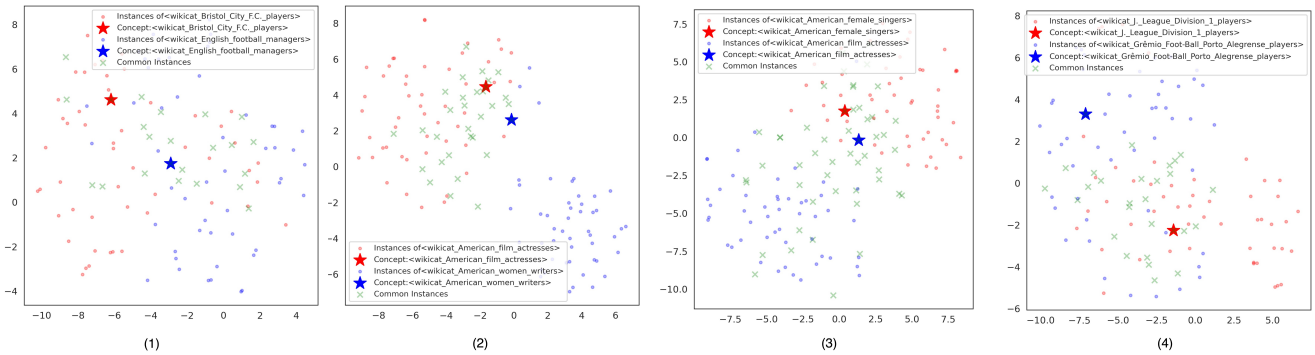


Figure 3. Concept embedding space visualization. Each concept embedding vectors is denoted as a star; each instance embedding vector is denoted as a dot in the same color with the concept to which it belongs. Level-X indicates the position of a concept at the Xth level within the hierarchical tree of concepts.

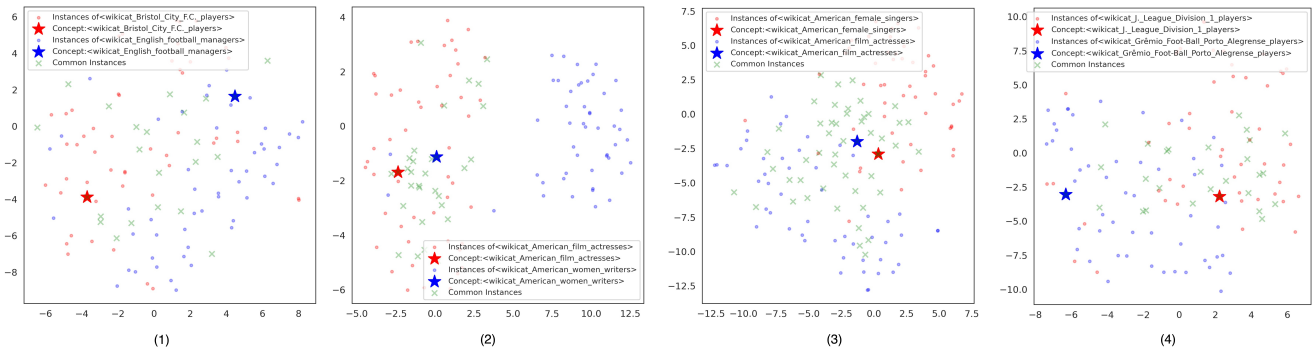
An interesting observation is that some sub-concepts that belong to different super-concepts are embedded closer, e.g., "American artists" under "artist" and "American television writers" under "writer". Clearly, for concept "American artists" and concept "American television writers", we can easily identify their commonality: these two sub-concepts are "American". Furthermore, in the YAGO39K dataset, we find that the aforementioned two concepts neither share common instances nor have the same super-concepts (except "ROOT"). These show that HCCE not only models the concept tree structure but also captures the semantic correlation among concepts via jointly training concepts, instances, and their relation embedding.

5.5. Overlapping Concept Pair Visualization

To demonstrate the impact of overlapping concept pairs on Knowledge Graph Embedding, we employed t-SNE to illustrate their distributions in the embedding space, as depicted in Figure 4. For HCCE-DS, each instance is mapped to an embedding point in the concept vector space. According to the definition of overlapping concept pairs in the introduction, we separately denote instances shared by two concepts by $\times$. By observing the distribution of these instances, we examine the overlap between the concept regions.



(**a**) HCCE-SS



(**b**) HCCE-DS

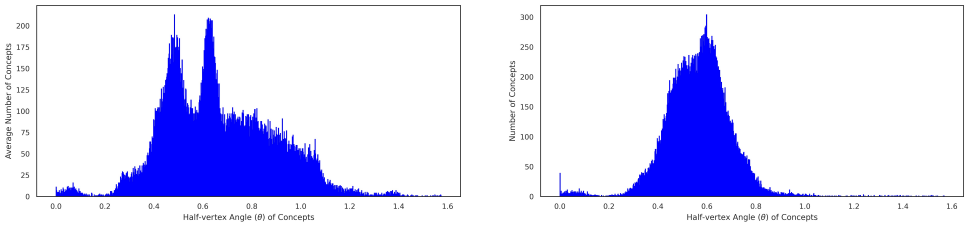
Figure 4. Overlapping concept pair visualization. Each concept embedding vectors is denoted as a star; each instance embedding vector that belongs to both two concepts is represented as a $\times$; an instance vector that belongs only to one of the two concepts is represented as a dot of the same color as the concept to which it belongs.

In each sub-figure, instance vectors form two independent clusters according to the concept to which they belong, and the instance vectors belonging to both concepts are located at the intersection of these clusters. This indicates that the regions of the two concepts are independent but partially overlap due to the shared instances, confirming the model design described in Section 3.2.3. Additionally, in some sub-figures (Figure 4(a2,b2)), we observe that the clusters of stars and instance points of the same color are widely separated, which seems unreasonable. However, we should consider the instance vectors represented by $\times$ in conjunction with the instance vectors of the same color as the concept, as they all belong to the same concept. When viewed together, the concept is located within the cluster formed by these instances, which aligns with our expectations. Furthermore, the visualization of HCCE-SS and HCCE-DS exhibit similar distributions, suggesting that HCCE can universally embed concepts and instances in either the same or different spaces.

5.6. Analysis of the Half-Vertex Angle θ

In the preceding visualization section, we have illustrated the spatial distribution of concept vectors. However, representing angles (θ) in high-dimensional space accurately in lower-dimensional spaces poses challenges. Therefore, we opted to depict the distribution

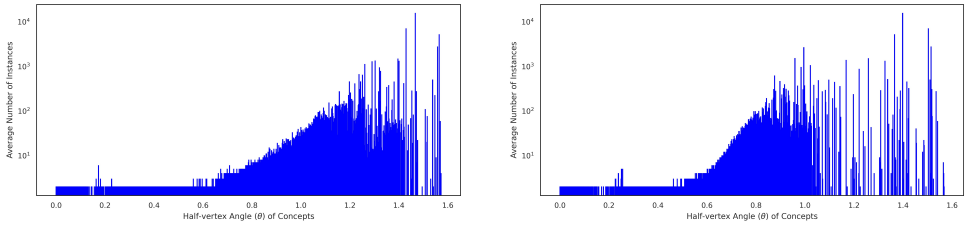
of all concept angles using frequency distribution to demonstrate the pattern of half-vertex angle distribution, as depicted in Figure 5. Here, we partitioned the entire range of angles into 1000 equal intervals based on the maximum and minimum angles. Observing the images of HCCE-SS, we notice that the half-vertex angles of concepts are concentrated between 0.3 and 1.1. HCCE-DS, on the other hand, exhibits an approximating Gaussian distribution, with concept half-vertex angles predominantly falling between 0.3 and 0.8. In both spatial scenarios, concepts with excessively small or large angles are relatively rare. We attribute concepts with small angles to either their limited number of instances or their granularity (concepts closer to leaf nodes in the concept tree). Conversely, concepts with large angles are likely to be closer to the root node in the concept tree (and inherently less frequent), and due to spatial constraints, it is reasonable for concepts with large angles to be less.



(**a**) HCCE-SS (**b**) HCCE-DS

Figure 5. Frequency distribution histogram of θ of concept region $G(\mathbf{c}, \theta)$ of HCCE-SS and HCCE-DS.

To further validate our hypothesis, we computed the average number of instances for all concepts within each angle interval, which are normalized by the number of concepts within the interval, as shown in Figure 6. In HCCE-SS and HCCE-DS, concepts encompassing more instances tend to have larger half-vertex angles, which indicates a broader conceptual domain. This observation aligns with our hypothesis and modeling assumptions, as well as human intuition.



(**a**) HCCE-SS (**b**) HCCE-DS

Figure 6. Average instance numbers of concept angle interval.

5.7. Theoretical and Practical Implications

In this section, we provide a scientifically detailed analysis of the theoretical and practical implications of our HCCE model, which is supported by mathematical modeling and experimental evidence.

From a theoretical perspective, our model addresses a gap in knowledge graph embeddings (KGE) by introducing high-dimensional spherical cones to represent concept embeddings, enabling the modeling of hierarchical and relational structures such as instanceOf, subClassOf, and instance relations. As detailed in Section 3, our mathematical framework produces anisotropic embeddings that capture both hierarchy and relationships within knowledge graphs. This approach allows for a refined representation of overlapping and disjoint concept pairs, providing a more nuanced geometry than traditional embeddings. By leveraging these high-dimensional cones, HCCE encapsulates hierarchical tree structures directly within the embedding space, making it, to our knowledge, the first KGE model to achieve this level of hierarchy-preserving detail.

Experimental results in Section 5.1 validate the efficacy of HCCE, demonstrating that our model maintains anisotropy while significantly reducing the number of parameters, thereby offering a compact yet expressive representation of concepts. Furthermore, HCCE significantly outperforms prior models on concept-related triples and achieves competitive performance on instance-related triples. Experimental results in Section 5.6 demonstrate the validity of the design of our approach and provide evidence for using geometric regions to model concept embeddings.

From a practical perspective, our method has far-reaching implications in various applications. HCCE's explicit geometric representation of hierarchical and relational structures enhances knowledge graph construction by enabling precise, hierarchy-aware concept embeddings. Furthermore, this concept-based approach can benefit concept-based question answering systems, where accurate hierarchical and relational inferences are essential. The model's capability to represent structured knowledge makes it a valuable asset for large language models, providing them with concept-aware knowledge that improves both generalization and interpretability.

In summary, our HCCE model advances the theoretical understanding of hierarchical embeddings in KGEs and offers practical tools for applications that rely on structured, concept-aware knowledge.

5.8. Limitation

Our model employs an interpretable geometric framework that focuses on leveraging structural information among instances, concepts, and relationships. Concepts are abstract descriptions of entities, which contain a wealth of semantic information. However, our model lacks the utilization of textual descriptions of concepts to enhance concept embedding representations.

Our model primarily focuses on the design of concept representation while retaining the original instance embeddings. Experimental results indicate that the HCCE method can implicitly improve performance on instance-related triples through the instanceOf relationship, although the effect is limited. Future work could explore modifying the instance embeddings, for example, by employing alternative approaches or fine-tuning the mapping functions to enhance the model's performance on instance-related triples.

6. Conclusions

In summary, this paper proposes a novel geometric model named HCCE to jointly embed concepts and instances. First, HCCE is the first to explicitly model the hierarchical tree structure of concepts by representing each concept as a hyperspherical cone region in the embedding space. It significantly reduces the model's parameter count while maintaining the anisotropy of concept embeddings. Additionally, we design two techniques, HCCE-SS and HCCE-DS, to explore the impact of embedding concepts and instances in either the same or different embedding spaces. Moreover, based on the tree structure of concepts, we identify two types of concept pairs—overlapping concept pairs and disjoint concept pairs—and design two score functions for them by utilizing relative positions. These approaches enable HCCE to capture more structural information, thereby enhancing the representation of concept embeddings. Empirical experiments conducted on the YAGO39K, MYAGO39K, and DB99K242 datasets demonstrate that HCCE significantly outperforms previous models on concept-related triples and achieves competitive performance on instance-related triples. The visualization of embedding results intuitively shows the hierarchical tree structure of concepts in the embedding space, providing evidence for interpretable reasoning regarding the transitivity of the isA relations. For future research, we intend to explore the following directions: (1) The enhancement of concept, which embeds at the semantic level by introducing text information through pre-trained models, to generate and process descriptions of concepts. (2) The investigation of additional inter-concept relations and the examination of their impact on the relations among associated instances. We hope

that these relations can mutually reinforce each other in enhancing the performance of knowledge graph embedding.

Author Contributions: Conceptualization, J.Y. and C.Z.; methodology, J.Y.; validation, J.Y.; formal analysis, J.Y.; investigation, J.Y.; resources, Z.H.; data curation, J.Y.; writing—original draft preparation, J.Y.; writing—review and editing, C.Z., Z.H. and Y.J.; visualization, J.Y.; supervision, C.Z. and Y.J.; project administration, Z.H.; funding acquisition, C.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Beijing Association of Higher Education Project MS2023151 and Training and Application of Large Language Models for Intelligent Operations Assistants (Project No. A2024091).

Data Availability Statement: The data that support the findings of this study are available upon reasonable request from the corresponding author.

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

References

1. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
2. Suchanek, F.M.; Kasneci, G.; Weikum, G. Yago: A core of semantic knowledge. In Proceedings of the 16th International World Wide Web Conference, WWW2007, Banff, AB, Canada, 8–12 May 2007 ; pp. 697–706.
3. Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; Taylor, J. Freebase: A collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, Sigmod '08, New York, NY, USA, 10–12 June 2008; pp. 1247–1250. [CrossRef]
4. Zhang, N.; Deng, S.; Sun, Z.; Wang, G.; Chen, X.; Zhang, W.; Chen, H. Long-tail Relation Extraction via Knowledge Graph Embeddings and Graph Convolution Networks. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 3016–3025. [CrossRef]
5. Saxena, A.; Tripathi, A.; Talukdar, P. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 4498–4507. [CrossRef]
6. Tan, Y.; Wang, B.; Liu, A.; Zhao, D.; Huang, K.; He, R.; Hou, Y. Guiding dialogue agents to complex semantic targets by dynamically completing knowledge graph. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2023, Toronto, ON, Canada, 9–14 July 2023; pp. 6506–6518. [CrossRef]
7. Wang, Y.; Wang, Z.; Zhang, H.; Liu, Z. Microblog Retrieval Based on Concept-Enhanced Pre-Training Model. *ACM Trans. Knowl. Discov. Data* **2023**, *17*, 1–32. [CrossRef]
8. Hao, J.; Chen, M.; Yu, W.; Sun, Y.; Wang, W. Universal Representation Learning of Knowledge Bases by Jointly Embedding Instances and Ontological Concepts. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, New York, NY, USA, 4–8 August 2019; pp. 1709–1719. [CrossRef]
9. Lv, X.; Hou, L.; Li, J.; Liu, Z. Differentiating Concepts and Instances for Knowledge Graph Embedding. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1971–1979. [CrossRef]
10. Meng, Y.; Zhang, Y.; Huang, J.; Zhang, Y.; Zhang, C.; Han, J. Hierarchical Topic Mining via Joint Spherical Tree and Text Embedding. In Proceedings of the KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 23–27 August 2020; pp. 1908–1917. [CrossRef]
11. Yu, J.; Zhang, C.; Hu, Z.; Ji, Y.; Fu, D.; Wang, X. Geometry-Based Anisotropy Representation Learning of Concepts for Knowledge Graph Embedding. *Appl. Intell.* **2023**, *53*, 19940–19961. [CrossRef]
12. Dong, Y.; Wang, L.; Xiang, J.; Liu, K. Modeling IsA Relations via Box Structure for Knowledge Graph Embedding. In Proceedings of the Advances in Knowledge Discovery and Data Mining—26th Pacific-Asia Conference, PAKDD 2022, Proceedings, Part II, Lecture Notes in Computer Science, Chengdu, China, 16–19 May 2022; Volume 13281, pp. 303–315. [CrossRef]
13. Pan, Z.; Wang, P. Hyperbolic Hierarchy-Aware Knowledge Graph Embedding for Link Prediction. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021, Punta Cana, Dominican Republic, 7–11 November 2021; pp. 2941–2948. [CrossRef]
14. Zhang, Z.; Wang, J.; Chen, J.; Ji, S.; Wu, F. ConE: Cone Embeddings for Multi-Hop Reasoning over Knowledge Graphs. In Proceedings of the Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, Virtual, 6–14 December 2021; pp. 19172–19183.
15. Iyer, R.G.; Bai, Y.; Wang, W.; Sun, Y. Dual-Geometric Space Embedding Model for Two-View Knowledge Graphs. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 14–18 August 2022; pp. 676–686. [CrossRef]

16. Nickel, M.; Tresp, V.; Kriegel, H.P. A Three-Way Model for Collective Learning on Multi-Relational Data. In Proceedings of the 28th International Conference on International Conference on Machine Learning, ICML'11, Madison, WI, USA, 28 June–2 July 2011; pp. 809–816.
17. Yang, B.; Yih, W.t.; He, X.; Gao, J.; Deng, L. Embedding entities and relations for learning and inference in knowledge bases. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015—Conference Track Proceedings, San Diego, CA, USA, 7–9 May 2015.
18. Trouillon, T.; Welbl, J.; Riedel, S.; Ciaussier, E.; Bouchard, G. Complex Embeddings for Simple Link Prediction. In Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York, NY, USA, 19–24 June 2016; Volume 5, pp. 3021–3032.
19. Nickel, M.; Rosasco, L.; Poggio, T.A. Holographic Embeddings of Knowledge Graphs. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 1955–1961.
20. Kazemi, S.M.; Poole, D. SimplE Embedding for Link Prediction in Knowledge Graphs. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, Red Hook, NY, USA, 3–8 December 2018; pp. 4289–4300.
21. Balazevic, I.; Allen, C.; Hospedales, T. TuckER: Tensor factorization for knowledge graph completion. 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. 5185–5194. [CrossRef]
22. Tucker, L.R. Some mathematical notes on three-mode factor analysis. *Psychometrika* **1966**, *31*, 279–311. [CrossRef] [PubMed]
23. Bordes, A.; Usunier, N.; Garcia-Durán, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-relational Data. In Proceedings of the 26th International Conference on Neural Information Processing Systems—Volume 2, NIPS'13, Red Hook, NY, USA, 5–10 December 2013; pp. 2787–2795.
24. Fan, M.; Zhou, Q.; Chang, E.; Zheng, T.F. Transition-based Knowledge Graph Embedding with Relational Mapping Properties. In Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing, Phuket, Thailand, 4–6 December 2014; pp. 328–337.
25. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge Graph Embedding by Translating on Hyperplanes. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; pp. 1112–1119.
26. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; pp. 2181–2187.
27. Ji, G.; He, S.; Xu, L.; Liu, K.; Zhao, J. Knowledge Graph Embedding via Dynamic Mapping Matrix. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, 26–31 July 2015; pp. 687–696. [CrossRef]
28. Yang, S.; Tian, J.; Zhang, H.; Yan, J.; He, H.; Jin, Y. TransMS: Knowledge Graph Embedding for Complex Relations by Multi-directional Semantics. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19. International Joint Conferences on Artificial Intelligence, Macao, China, 10–16 August 2019; pp. 1935–1942. [CrossRef]
29. Nayyeri, M.; Cil, G.M.; Vahdati, S.; Osborne, F.; Rahman, M.; Angioni, S.; Salatino, A.; Recupero, D.R.; Vassilyeva, N.; Motta, E.; et al. Trans4E: Link Prediction on Scholarly Knowledge Graphs. *Neurocomputing* **2021**, *461*, 530–542. [CrossRef]
30. Sun, Z.; Deng, Z.H.; Nie, J.Y.; Tang, J. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In Proceedings of the International Conference on Learning, New Orleans, LA, USA, 6–9 May 2019.
31. Zhang, Z.; Cai, J.; Zhang, Y.; Wang, J. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In Proceedings of the AAAI Conference on Artificial, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 3065–3072. [CrossRef]
32. Wang, Y.; Peng, Y.; Guo, J. Enhancing knowledge graph embedding with structure and semantic features. *Appl. Intell.* **2024**, *54*, 2900–2914. [CrossRef]
33. Xiong, B.; Cochez, M.; Nayyeri, M.; Staab, S. Hyperbolic Embedding Inference for Structured Multi-Label Prediction. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 5–8 December 2022; Volume 35, pp. 33016–33028.
34. Chami, I.; Wolf, A.; Juan, D.C.; Sala, F.; Ravi, S.; Ré, C. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020; pp. 6901–6914. [CrossRef]
35. Wang, K.; Liu, Y.; Lin, D.; Sheng, M. Hyperbolic Geometry is Not Necessary: Lightweight Euclidean-Based Models for Low-Dimensional Knowledge Graph Embeddings. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021, Punta Cana, Dominican Republic, 7–11 November 2021; pp. 464–474. [CrossRef]
36. Bordes, A.; Glorot, X.; Weston, J.; Bengio, Y. A semantic matching energy function for learning with multi-relational data. *Mach. Learn.* **2014**, *94*, 233–259. [CrossRef]
37. Dong, X.; Gabrilovich, E.; Heitz, G.; Horn, W.; Lao, N.; Murphy, K.; Strohmann, T.; Sun, S.; Zhang, W. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA, 24–27 August 2014; pp. 601–610. [CrossRef]
38. Socher, R.; Chen, D.; Manning, C.D.; Ng, A. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–8 December 2013; Volume 26.
39. Liu, Q.; Jiang, H.; Ling, Z.H.; Wei, S.; Hu, Y. Probabilistic Reasoning via Deep Learning: Neural Association Models. *arXiv* **2016**, arXiv:1603.07704.

40. Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2D Knowledge Graph Embeddings. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, LA, USA, 2–7 February 2018; pp. 1811–1818.
41. Nguyen, D.Q.; Nguyen, T.D.; Nguyen, D.Q.; Phung, D. A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 327–333. [CrossRef]
42. Jiang, X.; Wang, Q.; Wang, B. Adaptive Convolution for Multi-Relational Learning. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 978–987. [CrossRef]
43. Guo, L.; Sun, Z.; Hu, W. Learning to Exploit Long-term Relational Dependencies in Knowledge Graphs. In Proceedings of the 36th International Conference on Machine Learning, ICML 2019, Long Beach, CA, USA, 9–15 June 2019; Volume 2019, pp. 4441–4450.
44. 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 European Semantic Web Conference, Anissaras, Greece, 3–7 June 2018; pp. 593–607.
45. Vashishth, S.; Sanyal, S.; Nitin, V.; Talukdar, P.P. Composition-based Multi-Relational Graph Convolutional Networks. In Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020.
46. Yao, L.; Mao, C.; Luo, Y. KG-BERT: BERT for Knowledge Graph Completion. *arXiv* **2019**, arXiv:1909.03193.
47. Wang, X.; He, Q.; Liang, J.; Xiao, Y. Language Models as Knowledge Embeddings. In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, Vienna, Austria, 23–28 July 2022; pp. 2291–2297. [CrossRef]
48. Yao, L.; Peng, J.; Mao, C.; Luo, Y. Exploring Large Language Models for Knowledge Graph Completion. *arXiv* **2024**, arXiv:2308.13916.
49. Chen, M.; Tian, Y.; Chen, X.; Xue, Z.; Zaniolo, C. On2Vec: Embedding-based Relation Prediction for Ontology Population. In Proceedings of the 2018 SIAM International Conference on Data Mining (SDM), San Diego, CA, USA, 3–5 May 2018; pp. 315–323. [CrossRef]
50. Gutiérrez-Basulto, V.; Schockaert, S. From Knowledge Graph Embedding to Ontology Embedding? An Analysis of the Compatibility between Vector Space Representations and Rules. In Proceedings of the Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October–2 November 2018; pp. 379–388.
51. Diaz, G.I.; Fokoue, A.; Sadoghi, M. EmbedS: Scalable, Ontology-aware Graph Embeddings. In Proceedings of the 21st International Conference on Extending Database Technology, Vienna, Austria, 26–29 March 2018; pp. 433–436. [CrossRef]
52. Gao, H.; Zheng, X.; Li, W.; Qi, G.; Wang, M. Cosine-Based Embedding for Completing Schematic Knowledge. In Proceedings of the 8th CCF International Conference on Natural Language Processing and Chinese Computing, Dunhuang, China, 9–14 October 2019; pp. 249–261. [CrossRef]
53. Qiu, J.; Wang, S. Learning the Concept Embeddings of Ontology. In Proceedings of the Advanced Data Mining and Applications, Foshan, China, 12–14 November 2020; pp. 127–134. [CrossRef]
54. Chen, J.; Hu, P.; Jimenez-Ruiz, E.; Holter, O.M.; Antonyrajah, D.; Horrocks, I. OWL2Vec*: Embedding of OWL ontologies. *Mach. Learn.* **2021**, *110*, 1813–1845. [CrossRef]
55. Hu, Z.; Huang, P.; Deng, Y.; Gao, Y.; Xing, E. Entity Hierarchy Embedding. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, 26–31 July 2015; pp. 1292–1300. [CrossRef]
56. Guo, S.; Wang, Q.; Wang, B.; Wang, L.; Guo, L. SSE: Semantically Smooth Embedding for Knowledge Graphs. *IEEE Trans. Knowl. Data Eng.* **2017**, *29*, 884–897. [CrossRef]
57. Guan, N.; Song, D.; Liao, L. Knowledge Graph Embedding with Concepts. *Knowl.-Based Syst.* **2019**, *164*, 38–44. [CrossRef]
58. Li, Z.; Liu, X.; Wang, X.; Liu, P.; Shen, Y. TransO: A knowledge-driven representation learning method with ontology information constraints. In Proceedings of the World Wide Web, Austin, TX, USA, 1–5 May 2023; pp. 1–23.
59. Zhou, W.; Zhao, J.; Gui, T.; Zhang, Q.; Huang, X. Inductive relation inference of knowledge graph enhanced by ontology information. In Proceedings of the Findings of the association for computational linguistics: EMNLP, Singapore, 6–10 December 2023; pp. 6491–6502. [CrossRef]
60. Geng, Y.; Chen, J.; Pan, J.Z.; Chen, M.; Jiang, S.; Zhang, W.; Chen, H. Relational Message Passing for Fully Inductive Knowledge Graph Completion. In Proceedings of the 2023 IEEE 39th International Conference on Data Engineering (ICDE), Los Alamitos, CA, USA, 3–7 April 2023; pp. 1221–1233. [CrossRef]
61. Yu, Y.; Xu, Z.; Lv, Y.; Li, J. TransFG: A Fine-Grained Model for Knowledge Graph Embedding. In Proceedings of the Web Information Systems and Applications, Wuhan, China, 19–20 October 2019; pp. 455–466. [CrossRef]
62. Wang, K.; Qi, G.; Chen, J.; Wu, T. Embedding Ontologies via Incorporating Extensional and Intensional Knowledge. *arXiv* **2024**, arXiv:2402.01677.
63. Zhou, J.; Wang, P.; Pan, Z.; Xu, Z. JECI: A Joint Knowledge Graph Embedding Model for Concepts and Instances. In Proceedings of the Semantic Technology, Shanghai, China, 28–30 October 2020; Volume 12032, pp. 82–98. [CrossRef]

64. Xiang, Y.; Zhang, Z.; Chen, J.; Chen, X.; Lin, Z.; Zheng, Y. OntoEA: Ontology-guided Entity Alignment via Joint Knowledge Graph Embedding. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online, 1–6 August 2021; pp. 1117–1128. [CrossRef]
65. Jenatton, R.; Roux, N.; Bordes, A.; Obozinski, G.R. A latent factor model for highly multi-relational data. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; Volume 25.
66. Bordes, A.; Weston, J.; Collobert, R.; Bengio, Y. Learning Structured Embeddings of Knowledge Bases. In Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, CA, USA, 7–11 August 2011.
67. Maaten, L.V.D.; Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.