

Link Prediction with Attention Applied on Multiple Knowledge Graph Embedding Models

Cosimo Gregucci*

University of Stuttgart
Stuttgart, Germany

cosimo.gregucci@ipvs.uni-stuttgart.de

Daniel Hernández

University of Stuttgart
Stuttgart, Germany

daniel.hernandez@ipvs.uni-stuttgart.de

Mojtaba Nayyeri*

University of Stuttgart
Stuttgart, Germany

mojtaba.nayyeri@ipvs.uni-stuttgart.de

Steffen Staab

University of Stuttgart
University of Southampton
Stuttgart, Germany

ABSTRACT

Predicting missing links between entities in a knowledge graph is a fundamental task to deal with the incompleteness of data on the Web. Knowledge graph embeddings map nodes into a vector space to predict new links, scoring them according to geometric criteria. Relations in the graph may follow patterns that can be learned, e.g., some relations might be symmetric and others might be hierarchical. However, the learning capability of different embedding models varies for each pattern and, so far, no single model can learn all patterns equally well. In this paper, we combine the query representations from several models in a unified one to incorporate patterns that are independently captured by each model. Our combination uses attention to select the most suitable model to answer each query. The models are also mapped onto a non-Euclidean manifold, the Poincaré ball, to capture structural patterns, such as hierarchies, besides relational patterns, such as symmetry. We prove that our combination provides a higher expressiveness and inference power than each model on its own. As a result, the combined model can learn relational and structural patterns. We conduct extensive experimental analysis with various link prediction benchmarks showing that the combined model outperforms individual models, including state-of-the-art approaches.

CCS CONCEPTS

- Computing methodologies → Knowledge Representation and Reasoning;
- Information systems → Entity relationship models.

KEYWORDS

Knowledge graph embedding, link prediction, ensemble, geometric integration

*Both authors contributed equally to this research.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

WWW '23, April 30–May 04, 2023, Austin, TX, USA

© 2023 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-9416-1/23/04.

<https://doi.org/10.1145/3543507.3583358>

ACM Reference Format:

Cosimo Gregucci, Mojtaba Nayyeri, Daniel Hernández, and Steffen Staab. 2023. Link Prediction with Attention Applied on Multiple Knowledge Graph Embedding Models. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA*. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3543507.3583358>

1 INTRODUCTION

In the last few years large knowledge graphs (KGs) (e.g. Wikidata [36], YAGO [32], etc) have emerged to represent complex knowledge in the form of multi-relational directed labeled graphs [15]. KGs attracted great attention in industry [7, 9, 10, 13, 25, 30, 31] and academia [1, 14, 16, 32, 36], and became the core part of many artificial intelligence systems, e.g., question answering, etc.

Knowledge graphs are typically stored using the W3C standard RDF (Resource Description Framework) [4] which models graphs as sets of triples (h, r, t) where h , r , and t represent resources that are described on the Web. The link prediction community refers to them as *head entity*, *relation*, and *tail entity*, respectively. Each triple corresponds to a known fact involving entities h and t and relation r . For example, the fact that Berlin is the capital of Germany is modeled as the triple $(\text{Berlin}, \text{capitalOf}, \text{Germany})$.

A relevant problem for knowledge graphs, called link prediction, is predicting unknown facts (links) based on known facts, and knowledge graph embedding (KGE) is a prominent approach for it. To predict links, KGEs map entities h and t and relations r into elements \mathbf{h} , \mathbf{r} , and \mathbf{t} in a low-dimensional vector space, and score the plausibility of a link (h, r, t) using a *score function* on \mathbf{h} , \mathbf{r} , and \mathbf{t} . Most KGE models [5, 22, 33, 35, 46] score a link (h, r, t) by splitting it into the *query* $q = (h, r, ?)$ and the corresponding *candidate answer* t . The query is embedded to an element in the same space as the candidate answers with a transformation function $q = g_r(\mathbf{h})$ that depends on the relation r and is applied to \mathbf{h} . The score of a link is then a measure of the similarity or proximity between q and t .

KGE models can learn logical and other patterns (example in Figure 1) to predict links. For instance, the facts that co-author is a symmetric relation and part-of is hierarchical can be learned from the data. However, the capability of different KGE models to learn and express patterns for predicting missing links varies widely and, so far, no single model does it equally well for each pattern. Logical patterns exhibit the form *Premise* → *Conclusion* where *Premise* is the conjunction of several atoms and *Conclusion* is an atom.

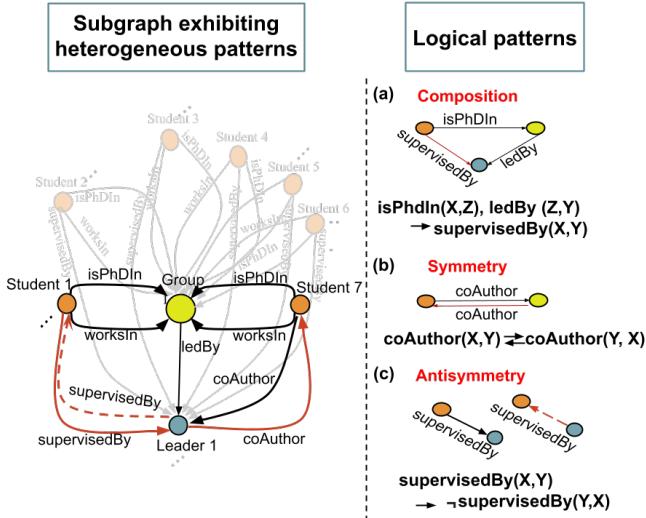


Figure 1: Subgraph exhibiting heterogeneous patterns [23].

Structural patterns refer to the arrangements of elements in a graph. A relation forms a hierarchical pattern when its corresponding graph is close to tree-like [6], e.g., *(eagle, type-of, bird)*. For example, RotatE defines transformations as rotations $g_r^{\text{RotatE}}(\mathbf{h}) = \mathbf{h} \circ \mathbf{r}$ in Complex space (\circ is an element-wise complex product). In this way, RotatE can enforce both $\mathbf{h} \circ \mathbf{r} = \mathbf{t}$, $\mathbf{t} \circ \mathbf{r} = \mathbf{h}$ if $\mathbf{r}^2 = 1$ and, thus, it is able to model symmetric relations. In Table 1, we present a summary of the query representation of some state-of-the-art baselines. We indicate whether a KGE can or cannot model a specific pattern. If it can model a pattern, we further include the number of constraints they have to satisfy to express this pattern. For instance, antisymmetry for RotatE requires two constraints $\mathbf{r} \neq -1$ and $\mathbf{r} \neq 1$ to be expressed. Further explanation of Table 1 can be found in Appendix A.4.

Beyond the KGes surveyed in Table 1, further works have defined query representations successfully dealing with different subsets of patterns, such as 5*E [22], AttE/H [6], TransH [41], or ProjE [29]. However, there is neither a single transformation function that can model all patterns nor a single approach that can take advantage of all the different transformation functions.

In this paper, we tackle this problem and propose a general framework to integrate different transformation functions from several KGE models, \mathbb{M} , in a low-dimensional geometric space such that heterogeneous relational and structural patterns are well represented. In particular, we employ spherical geometry to unify different existing representations of KGE queries, $(\mathbf{h}, \mathbf{r}, ?)$. In our framework, representations of KGE queries, $g_r^i(\mathbf{h})$ with $i \in \mathbb{M}$, define the centers of hyperspheres, and candidate answers lie inside or outside of the hyperspheres whose radii are derived during training. Plausible answers mostly lie inside the convex hull formed by the centers of the hyperspheres. Based on this representation, we learn how to pay attention to the most suitable representations of a KGE query. Thereby, attention is acquired to adhere to applicable patterns (see Figure 1 and Figure 2).

For instance, given a KGE query *(Leader1, coAuthor, ?)*, attention will focus on the representation of this query defined by RotatE, as our framework has learned about the symmetry of the relation *coAuthor*. Likewise, TransE and RotatE will be preferred for KGE query *(Student1, supervisedBy, ?)* accounting for the pattern $(X, \text{isPhDIn}, Y), (Y, \text{ledBy}, Z) \rightarrow (X, \text{supervisedBy}, Y)$, while TransE will be favored for KGE query *(Leader1, supervisedBy, ?)* due to the anti-symmetry of *supervisedBy*.

Furthermore, we also project our model onto a non-Euclidean manifold, the Poincaré ball, to also facilitate structural preservation.

In summary, our key contributions are as follows:

- We propose a spherical geometric framework for combining several existing KGE models. To our knowledge, this is the first approach to integrate KGE models taking advantage of the different underlying geometric transformations.
- We utilize an attention mechanism to focus on query representations depending on the characteristics of the underlying relation in the query. Therefore, our method can support various relational patterns. Furthermore, structural patterns are captured by projecting the model onto the Poincaré ball.
- We present various theoretical analyses to show that our model subsumes various existing models.

2 RELATED WORK

We review the related works in three parts, namely the baseline models we used for combination, the models which provide other approaches for combinations, and models that combine spaces.

2.1 KGE Model Baselines

Various models [11, 17, 38] have been proposed for KGE in the last few years. Each KGE defines a score function $f(\mathbf{h}, \mathbf{r}, \mathbf{t})$ which takes embedding vectors of a triple $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ and scores the triple. In our work we integrate and compare them to the following baselines:

- TransE [5] computes the score of a triple by computing the distance between the tail and the *translated* head. Thanks to the translation-based transformation, this KGE is particularly suited for modeling inverse and composition patterns.
- RotatE [33] uses a relation-specific rotation $\mathbf{r}_i = e^{i\theta}$ to map each element of the head to the corresponding tail. RotatE can infer symmetrical patterns if the angle formed by the head and tail is either 0 or θ . Besides, rotations are also effective in capturing antisymmetry, composition, or inversion.
- DistMult [45] represents each relation as a diagonal matrix. Its score function captures pairwise interaction between *the same dimension* of the head and tail embedding. Thus, DistMult treats symmetric relations well, but scores so highly inverse links of non-symmetric and antisymmetric relations.
- ComplEx [35] extends DistMult in the complex space to effectively capture symmetric and antisymmetric patterns.
- AttH [6] combines relation-specific rotations and reflections using hyperbolic attention and applies a hyperbolic translation. Rotation can capture antisymmetrical and symmetrical patterns, reflection can naturally represent symmetrical relations, while the hyperbolic translation can capture hierarchy. We also compared our models against AttE [6], a variant of AttH with curvature set to zero.

Table 1: Specification of query representation of baseline and state-of-the-art KGE models and respective pattern modeling and inference abilities. AttE/H include both rotation (RotateE) and reflection (RefH), hence are not mentioned in the table to avoid repetitions. \circ is element-wise complex product together with relation normalization.

Model	Query	Embeddings	Symmetry	Antisymmetry	Inversion	Composition	Hierarchy
TransE [5]	$q = h + r$	$q, h, r \in \mathbb{R}^d$	\times	$\checkmark - 0$	$\checkmark - 0$	$\checkmark - 0$	\times
RotatE [33]	$q = h \circ r$	$q, h, r \in \mathbb{C}^d$	$\checkmark - 2$	$\checkmark - 2$	$\checkmark - 2$	$\checkmark - 2$	\times
ComplEx [35]	$q = h \times r$	$q, h, r \in \mathbb{C}^d$	$\checkmark - 2$	$\checkmark - 2$	$\checkmark - 2$	\times	\times
DistMult [45]	$q = h \cdot r$	$q, h, r \in \mathbb{R}^d$	$\checkmark - 0$	\times	\times	\times	\times
RefH [6]	$q = \text{Ref}(\theta_r)h$	$q, h \in \mathbb{H}^d$	$\checkmark - 0$	\times	\times	\times	$\checkmark - 0$

2.2 KGEs Combination

Combinations between KGEs of the same kind. Authors in [44] showed that, under some conditions, the ensemble generated from the combination of multiple runs of low-dimensional embedding models of the same kind outperforms the corresponding individual high-dimensional embedding model. Unlike our approach, the ensemble model will still be able to express only a subset of existing logical patterns.

Combination between different KGE models. Prior works [20] proposed to combine different knowledge graph embeddings through score concatenation to improve the performance in link prediction. [40] proposed a relation-level ensemble, where the combination of individual models is performed separately for each relation. A recent work [39], proposed to combine the scores of different embedding models by using a weighted sum. Such methods combine scores either per model or per relation, while we provide a query attention mechanism for the combination.

A different approach has been proposed in MulDE [37], where link prediction is improved by *correcting* the prediction of a “student” embedding through the use of several pre-trained embeddings that act as “teachers”. The student embedding can be considered to constitute an ensemble model. However, this ensemble cannot steer decisions towards the strengths of individual models but can only decide randomly or based on majority guidance by teachers.

Further ensemble approaches between KGE and machine learning models can be found in the Appendix A.3

2.3 Combination Of Spaces

A different line of research aims at improving link prediction performance by combining different geometrical spaces. [12] improves link prediction by combining Hyperbolic, Spherical, and Euclidean space. Similarly, [42] embedded knowledge graphs into an Ultra-hyperbolic manifold, which generalizes Hyperbolic and Spherical manifolds. On the other hand, we combine queries rather than geometric spaces.

3 PROPOSED APPROACH

In this section, we present our geometric query integration model using Euclidean and Hyperbolic geometries, and introduce our approach in the following four items: a) entity, relation, and query representation, b) spherical query embedding, c) Riemannian attention-based query combination, d) expressivity analysis.

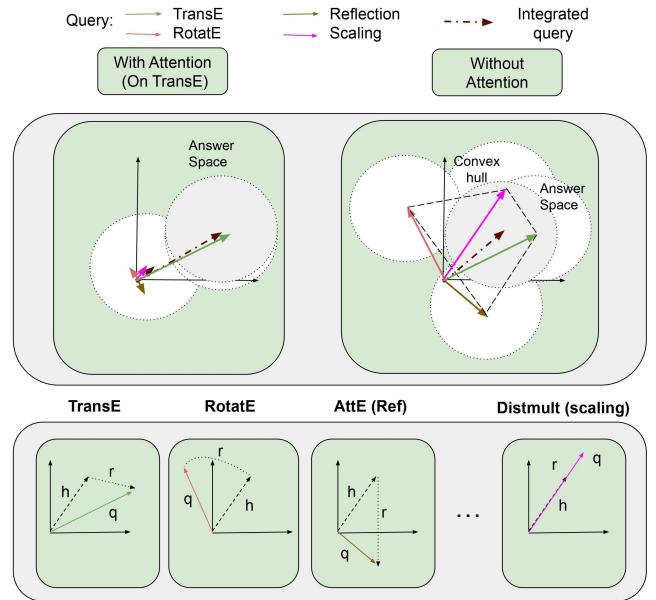


Figure 2: The overall architecture of our proposed model with spherical geometry. We combine query representations of TransE, RotatE, AttE (with Reflection), and DistMult (per dimension scaling). The left part shows query integration with attention to TransE model. The right part represents query combination without attention.

a) Entity, Relation and Query Embedding. Let \mathcal{E}, \mathcal{R} be the entity and relation sets. We represent each entity $e \in \mathcal{E}$ and relation $r \in \mathcal{R}$ as d_e and d_r dimensional vectors which are denoted by \mathbf{e} and \mathbf{r} , respectively. Thus, each triple (h, r, t) has a vector representation $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ where \mathbf{h}, \mathbf{t} are the corresponding entity embeddings.

We split each triple (h, r, t) into two parts, namely the tail query $q = (h, r, ?)$ and the candidate answer t , and represent their embeddings by \mathbf{q}, \mathbf{t} respectively.

In our model, we aim at combining the queries from several existing KGE models that are specified in Table 1. We denote the query representation set by $\mathcal{Q} = \{\mathbf{q}_i | \mathbf{q}_i = g_r^i(\mathbf{h}), i \in \mathbb{M}\}$ where \mathbb{M} is a set of several existing KGE models such as TransE, RotatE, ComplEx, DistMult, etc, and the function $g_r^i(\mathbf{h})$ is a relation-specific transformation from a head embedding to a query representation for model i . Note that we assume that query representations by

different models lie on the same space. In this paper, we stay in Euclidean space for query combination. In this regard, we can combine models lying either directly in Euclidean space (e.g., TransE and DistMult) and models that can be rewritten to lie in the Euclidean space (e.g., models lying in Complex or Hypercomplex spaces as ComplEx, RotatE, and QuatE by assuming $\mathbb{R}^4 = \mathbb{R}^2 \times \mathbb{R}^2 = \mathbb{C}^1 \times \mathbb{C}^1$, where $\mathbb{C}^d, \mathbb{R}^d$ are d -dimensional Complex and Euclidean spaces). We then project such query vectors on a hyperbolic manifold to handle hierarchical patterns.

b) Spherical Query Embedding. In this part, first, we propose a spherical query embedding to represent each query as a sphere whose center is the vector embedding of the query. This sphere defines the answer space of the query. Second, we propose an approach to combine query representations of several already existing embedding models in one spherical query representation to enhance the modeling of heterogeneous patterns. In “*radius and ranking*”, we will show that the spherical representation is connected to the ranking metric Hits@k. In particular, the top k candidate answers for a query q are embedded in a sphere whose center is a combination of the vector embeddings q_i of query q . To practically enforce this, the radius in our spherical query embedding needs to be set. Therefore, in “*radius and loss*”, we will show that a loss function can enforce the improvement of Hits@k by enforcing top k candidate answers of a query inside the sphere.

Here, we formalize the combination of n spherical KGEs. Let us assume that $q_1, q_2, \dots, q_n \in Q$ be the n vector query embeddings of a query $q = (h, r, ?)$ from n distinct KGE models, and $a = t$ be the embedding of the candidate answer. We represent each query as a hypersphere with a pair $q_i^c = (q_i, \epsilon_i)$, $q_i \in Q$, where $q_i \in \mathbb{R}^d$ is the center of the i th sphere associated to the i th model and ϵ_i is the radius. By using the function

$$p(q_i, a) = \|a - q_i\|, \quad q_i \in Q, \quad (1)$$

we define the answer space \mathcal{A} and non-answer space \mathcal{N} as decision boundaries in the embedding space for each query as follows:

$$\begin{cases} \mathcal{A}_i &= \{e \in \mathbb{R}^d \mid \|e - q_i\| \leq \epsilon_i\}, \\ \mathcal{N}_i &= \{e \in \mathbb{R}^d \mid \|e - q_i\| > \epsilon_i\}. \end{cases} \quad (2)$$

In this case, all embeddings of answers a are supposed to lie on or inside a sphere with a radius of ϵ_i and center q_i , i.e., $a \in \mathcal{A}_i$, and the ones which are not answers lie outside of the sphere [24, 47]. We combine spherical query embeddings of several existing KGE models in one spherical query embedding as follows:

Combination. Given the vector embeddings $q_1, q_2, \dots, q_n \in Q$ we can set a radius ϵ_i^k for each q_i such that the answer space \mathcal{A}_i covers the top k candidate answers.

$$\begin{cases} \mathcal{A}_1 &= \{e \in \mathbb{R}^d \mid \|e - q_1\| \leq \epsilon_1^k\}, \\ &\vdots \\ \mathcal{A}_n &= \{e \in \mathbb{R}^d \mid \|e - q_n\| \leq \epsilon_n^k\}. \end{cases} \quad (3)$$

Summing up the above inequalities, we have $\|a - q_1\| + \dots + \|a - q_n\| \leq \epsilon_1^k + \dots + \epsilon_n^k$. Because of triangle inequality of the metric $\|\cdot\|$, this can be extended to the following inequality $\|a - q_1 + \dots + a - q_n\| \leq \epsilon_1^k + \dots + \epsilon_n^k$, that concludes $\|a - \frac{q_1 + \dots + q_n}{n}\| \leq \frac{\epsilon_1^k + \dots + \epsilon_n^k}{n}$.

Therefore, the combined spherical query embedding is the spherical embedding $q_E^c = (q_E, \epsilon_E)$ where

$$\begin{cases} q_E &= \frac{q_1 + \dots + q_n}{n}, \\ \epsilon_E &= \frac{\epsilon_1^k + \dots + \epsilon_n^k}{n}. \end{cases} \quad (4)$$

This leads to the following top k candidate answer space of the combined spherical query embedding:

$$\mathcal{A}_E = \{e \in \mathbb{R}^d \mid \|e - q_E\| \leq \epsilon_E\}. \quad (5)$$

Figure 2 (top right) shows the query representations, and candidate answer spaces of TransE, RotatE, RefE, and DistMult, together with the combined query (without attention to a particular model). The combined query mainly lies inside the convex hull of all the models within the answer space. We later show that most answers lie within the convex hull covered by the combined query. Therefore, the combined model takes the advantage of all models. Before theoretically justifying this, we bridge the radius ϵ in spherical query embedding and ranking metrics, as well as the practical way of modeling radius using the loss function in the following parts.

Radius and Ranking. Most KGE models are evaluated based on ranking metrics such as Hits@k [5]. Here we explain the connection between the ranking metrics and radius in our spherical query embedding. Because the overall ranking is computed by taking the average over ranks of all test triples, we explain the connection between ranking and our model by considering an individual test triple. During testing, for each given positive test triple (h, r, t) , the tail t is replaced one by one with all entities $e \in \mathcal{E}$. We denote $\mathbb{T}_e = (h, r, e)$ the corrupted triple generated by replacing t by e . Therefore, $\mathcal{T} = \{\mathbb{T}_e \mid e \in \mathcal{E} - \{t\}\}$ is the set of all corrupted triples generated from the correct triple (h, r, t) . After computing the score of each triple in \mathcal{T} and sorting them based on their scores in descending way, we select top k high score samples and generate a new set \mathcal{T}_k containing these samples. The spherical query embedding $q_E^c = (q_E, \epsilon_E)$ associated to a query $q = (h, r, ?)$ defines a spherical answer space \mathcal{A}_E that contains the vector embeddings e for the top k entities $e \in \mathcal{T}_k$. That is, \mathcal{T}_k contains top k candidates for a query q , and \mathcal{A}_E in Equation 5 is the candidate answer embedding space. We want the vectors of answers in \mathcal{T}_k to lie inside \mathcal{A}_E , and to be as close as possible to the query center to improve ranking results. To enforce this, we define a loss function to optimize the embeddings, as is explained below.

Radius and Loss Function. In this part, we show that the existing loss functions implicitly enforce a particular (implicit) radius around the vector query embedding q_E . Let us focus on the widely used loss function shown in the following [6]:

$$\mathcal{L} = \sum_{e \in \mathcal{E}} \log(1 + \exp(y_e(-p(q_i, e) + \delta_h + \delta_e))), \quad (6)$$

where $y_e = 1$ if $e = a$, and $y_e = -1$ if $e \neq a$, and δ_h and δ_e are trainable entity biases. Minimization of this loss function leads to maximizing the function $-p(q_i, e) + \delta_h + \delta_e$. This can be approximately represented as $-p(q_i, e) + \delta_h + \delta_e \geq M$ where M is a large number. Therefore, we have $p(q_i, e) \leq \delta_h + \delta_e - M = \delta_{he} - M = \epsilon_i$ which forms boundaries for classification as well as ranking. In the next part, we theoretically show that q_E lies within the convex hull

of the set of vectors $\{q_1, \dots, q_n\}$. Thus, the combined model takes advantage of each model in ranking.

Theoretical Analysis. Equation 5 indicates that if the query is represented by (q_E, ϵ_E) , then the score given by the combined model to a plausible answer is lower than the average of the scores given by the individual models, and higher than the lowest individual model score because, without loss of generality, we have

$$\min(p(q_1, \epsilon), p(q_2, \epsilon)) \leq p(q_E, \epsilon) \leq \max(p(q_1, \epsilon), p(q_2, \epsilon)). \quad (7)$$

This equation shows that for a particular k , the combined model gets a better score than the worst model, but it gets a lower score than the best one. However, by increasing k , the combined model covers the answers provided by both models because most of the answers lie in the convex hull between the queries (later it will be proved), and the combined model with arbitrary large k covers the answers represented by each model. Therefore, the combined model improves Hits@k with a sufficiently large k . Later in this section, we present the attention-based model which enables us to improve Hits@k for small k .

The following proposition states that the best embedding for an answer to a query lies in the convex hull of the query embeddings given by two models. This implies that if two models are trained jointly with the combined model, the answers of each query lie between the centers of the two spheres associated with the two embeddings of the query. This facilitates the answer space of combined spherical query embedding to cover the answer embedding from each individual model. This can be generalized for an arbitrary number of models.

PROPOSITION 3.1. *Let q_1 and q_2 be two query embeddings for a query q . Then, the following two statements are equivalent for every vector a in the vector space:*

$$\begin{cases} a = \operatorname{argmin}_e (p(q_1, e) + p(q_2, e)), \\ a \text{ lies in the convex hull of vectors } q_1 \text{ and } q_2. \end{cases} \quad (8)$$

c) Riemannian Attention-Based Query Combination.

Weighted Combined Query Embedding. A consequence of proposition 3.1 is that the combined query embedding can improve the performance when k is sufficiently large (e.g., Hits@20). However, for a low k (e.g., Hits@1) the performance is degraded because one model gets a better ranking, and the combined model with an average query does not cover it. In addition, among several models, there might be possible that some models return wrong answers which might also influence the combined model. Therefore, allowing the combined spherical query embedding q_E to slide to q_1 or q_2 is beneficial. Hence, without loss of generality, we combine two query embeddings as the convex combination of the inequalities:

$$\begin{cases} \alpha \|a - q_1\| \leq \alpha \epsilon_1^k, \\ \beta \|a - q_2\| \leq \beta \epsilon_2^k, \quad \alpha, \beta \geq 0, \alpha + \beta = 1. \end{cases} \quad (9)$$

By computing this convex combination, we have $\alpha \|a - q_1\| + \beta \|a - q_2\| \leq \alpha \epsilon_1^k + \beta \epsilon_2^k$. This inequality implies $\|\alpha a - \alpha q_1 + \beta a - \beta q_2\| \leq \alpha \|a - q_1\| + \beta \|a - q_2\| \leq \alpha \epsilon_1^k + \beta \epsilon_2^k$, which subsequently leads to

$$\|a - (\alpha q_1 + \beta q_2)\| \leq \alpha \epsilon_1^k + \beta \epsilon_2^k. \quad (10)$$

Therefore, the combined spherical query embedding is $q_E^c = (q_E, \epsilon_E^k)$ where $q_E = (\alpha q_1 + \beta q_2)$ and $\epsilon_E^k = \alpha \epsilon_1^k + \beta \epsilon_2^k$. This combination is generalized for n models:

$$\|a - \sum_i \alpha_i q_i\| \leq \sum_i \alpha_i \epsilon_i^k, \quad (11)$$

Attention Calculation. Given a combined spherical query embedding $q_E^c = (q_E, \epsilon_E)$ with

$$q_E = \sum_i \alpha_i q_i, \quad \epsilon_E^k = \sum_i \alpha_i \epsilon_i^k, \quad (12)$$

we can compute α_i s by providing an attention mechanism [6]

$$\alpha_i = \frac{\exp(g(wq_i))}{\sum_j \exp(g(wq_j))}, \quad (13)$$

where $g(x) = wx$ is a function with a trainable parameter w . We call this version of our model Spherical Embedding with Attention (SEA).

Riemannian Query Combination. We next extend our attention-based query combination to Riemannian manifolds to model both relational patterns (via various transformations used in different models) and structural patterns as hierarchy via the manifolds (e.g., Poincaré ball). Similarly to [6], we perform attention on tangent space. We consider all models in Euclidean space and combine their query embeddings. The resulting query embedding on the tangent space is then projected to the manifold via the exponential map. This attention-based model combination is defined as follows:

$$\begin{cases} q_E^{euc} = \sum_i \frac{\exp(g(q_i))}{\sum_j \exp(g(q_j))} q_i, \\ q_E^M = \exp_0(q_E^{euc}). \end{cases} \quad (14)$$

We compute the score as $p(q, a) = d(q_E^M \oplus r, a)$, where h, r, t, q are points on a manifold M , $\exp_0(\cdot)$ is the exponential map from origin, and \oplus is Möbius addition. In terms of the Poincaré ball, the manifold, exponential map, and Möbius addition are defined as follows [2, 6]:

$$\begin{cases} M = \{p \in \mathbb{R}^d | \|p\| \leq \frac{1}{c}\}, \\ \exp_0(v) = \tanh(\sqrt{c}\|v\|) \frac{v}{\sqrt{c}\|v\|}, \\ d^c(p, q) = \frac{2}{c} \tanh^{-1}(\sqrt{c}\|p - q\|), \\ p \oplus q = \frac{(1+2c\langle p, q \rangle + c\|q\|^2)p + (1-c\|p\|^2)q}{1+2c\langle p, q \rangle + c^2\|p\|^2\|q\|^2}, \end{cases} \quad (15)$$

where c is the curvature, \exp is the exponential map from a point on tangent space to the manifold, d^c is the distance function with curvature c , and v is the point on the tangent space to be mapped to manifold via the exponential map. We call the hyperbolic version of our model Spherical Embedding with Poincaré Attention (SEPA).

d) Expressivity Analysis. In this section, we analyze our models in terms of expressive power as well as the subsumption of other models. Our model is a generalization of various existing Euclidean and non-Euclidean KGE models. We say that a model m_1 subsumes a model m_2 if for every given KG G and scoring by model m_2 , there exists a scoring by model m_1 such that the score of every triple $t \in G$ by m_1 approximates the score of t by m_2 [19, 22].

PROPOSITION 3.2. *SEPA subsumes AttH, and SEA subsumes TransE, RotateE, ComplEx, DistMult and Atte.*

COROLLARY 3.3. *SEPA and SEA can infer anti-symmetry, symmetry, composition, and inversion patterns.*

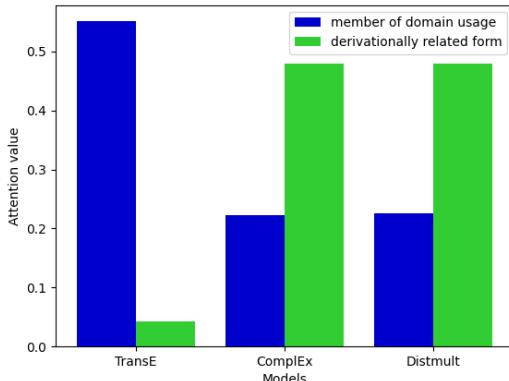


Figure 3: Comparison between the importance given by each model to a symmetric (in green) and antisymmetric (in blue) relation.¹

It is important to notice that a model can infer a pattern inherently or infer a pattern under a certain condition (see Table 1). Our model aims to take advantage of the inference power of multiple models on heterogeneous patterns with minimum certain conditions by providing attention mechanisms per relation type forming different patterns. Note that our model is not influenced by incapable models on particular patterns because the attention can be learned as zero for those models. Overall, our combined model can inherit the capabilities mentioned in Table 1 and ignore the incapability of other models which is shown in Theorem 3.2 and Theorem 3.3. Hence, if our model is executed on a dataset containing only a single pattern, we do not expect to outperform the combined models, rather than achieving competitive performance to the best model.

Proof of propositions can be found in Appendix A.1.

4 EXPERIMENTS

In this section, we conduct extensive evaluations to show the effectiveness of our proposed approach. To do so, we first introduce the utilized datasets, followed by the selected baseline for the combination and the comparison. We then present the experimental setup and hyper-parameter setting. The results and analysis are presented in three folds: comparison with the individual baselines, comparison with other combination models, and comparison with models in the Ultrahyperbolic space. Finally, we provide several analyses to show the role of attention on learning and inference over various patterns for different kinds of relations and models.

4.1 Dataset

We use the following standard benchmark for the evaluation:

- **Wordnet:** WN18RR [8] is a subset of WN18, which contains a mixture of symmetric and antisymmetric as relational patterns, and hierarchical structural patterns. *see also* and *hypernym* are examples of symmetry and hierarchy in this dataset. WN18RR contains 11 relations, 86,835 training triples, and 40,943 entities. Compared to the other datasets in KGE literature, WN18RR is considered sparse;

¹E.g. *ethics* is a form that is derivationally related to *ethicist*.

- **FreeBase:** FB15k-237 [34] is the subset of FB15k from removing leakage of inverse relations [8]. FB15k-237 is less sparse than WN18RR, and mainly contains composition patterns. It contains 237 relations, 272,115 triples, and 14,541 entities.
- **NELL:** NELL-995 [43] contains 75,492 entities and 200 relations, having ~ 22% hierarchical relations. We use a subset of NELL-995 with 100% hierarchy, created in [2].

4.2 Baseline

In this section, we aim to show experimentally that the geometric combination of several existing KGE models improve their performance. To this end, we select a subset of KGEs in Euclidean, Complex, and Hyperbolic space with different capabilities to show we can combine a wide range of models. In particular, we select a subset of TransE, DistMult, ComplEx, RotatE, AttE (only reflection), and AttH (hyperbolic projection operator) and compare our combined models against such baselines. We also compare our models with two additional state-of-the-art KGEs: in high dimension, TuckER [3], and in low dimension, MuRP [2], to show that our models can outperform models that were not combined. Furthermore, we also compare our model with a recent top model for combining several KGEs, namely MulDE [37] because it uses a similar set of KGEs for the combination, similar dimensions, and some of the benchmarks we used. Additionally, we will show that our model gets comparable performance with UltraE [42], a model on the Ultrahyperbolic space.

4.3 Experimental Setup

Evaluation Metrics. We use the popular ranking metrics [38] namely Mean Reciprocal Rank (MRR), and Hits@ k , $k = 1, 3, 10$. Given a set of test triples $\mathcal{T} = \{(h, r, t)\}$, for each test triple $p = (h, r, t)$, we compute its rank as follows: we first corrupt the head entity by replacement with all possible entities in the KG, say $e' \in \mathcal{E}$, and generate a set of candidate corrupted triples for p i.e., $p_c = \{p' = (e', r, t)\}$. We filter p_c by removing all generated candidates which are already appeared in the train, validation, and test sets, together with removing the cycle. After computing the score of the candidate triples and sorting them, we find the rank of the candidate test p , and call it r_p . The same procedure is performed for computing the right rank by corrupting the tail entity and computing the right rank. The average of the left and right ranks will be considered the final rank of the test triple. We then compute the average reciprocal of all test triples and report it as MRR. Hits@ k will be computed by reporting the percentage of the test triples ranked less than k .

Hyperparameters. The hyperparameters corresponding to our model are embedding dimension d , models to combine m , optimizer o , learning rate lr , number of negative samples n , batch size b , dtype dt , and double_neg dn . We additionally used α^2 as attention parameters (in place of α) playing the role of a simple kind of regularization mechanism (ar), to further penalize the models with less contribution in the attention. Following the common practice of KGEs, we use both low $d = 32$ and high dimensions $d = 500$ for the evaluation of our model. For the other hyperparameters, we use the following ranges $m = \{\text{TransE}, \text{DistMult}, \text{ComplEx}, \text{RotatE}, \text{AttE} (\text{only reflection})\}$, $o = \{\text{Adam}, \text{Adagrad}\}$, $lr = \{0.1, 0.05, 0.001\}$,

$n = \{-1, 50, 100, 150, 200, 250\}$ where -1 refers to full negative sampling [21], $b = \{100, 500\}$, $dt = \{\text{single}, \text{double}\}$, $ar = \{\text{yes}, \text{no}\}$, $dn = \{\text{yes}, \text{no}\}$. We also add reciprocal triples to the training set as the standard data augmentation technique [21]. The optimal hyperparameters for each dataset are specified in Appendix A.2.

Table 2: Comparison of H@1 for WN18RR relations. TE = TransE, CE = ComplEx, DM = DisMult

Relation	TE	CE	DM	SEPA
member meronym	0.144	0.095	0.030	0.162
hypernym	0.060	0.064	0.024	0.121
has part	0.105	0.099	0.029	0.125
instance hypernym	0.246	0.242	0.139	0.242
member of domain region	0.269	0.077	0.096	0.327
member of domain usage	0.271	0.188	0.062	0.271
synset domain topic of	0.289	0.136	0.070	0.329
also see	0.161	0.580	0.598	0.571
derivationally related form	0.532	0.943	0.940	0.944
similar to	0.000	1.000	1.000	1.000
verb group	0.333	0.949	0.974	0.923

4.4 Link Prediction Results And Analysis

The result of comparing SEA and SEPA to the combined models on FB15k-237, WN18RR and NELL-995-h100 are shown in Table 3 ($d = 32$) and in Table 4 ($d = 500$).

As expected, while the hyperbolic version of our combined model (SEPA) outperforms all baselines in low-dimensional settings, the Euclidean one (SEA) is the best model in high-dimensional space. Comparing SEPA and SEA, in low-dimensional space, we can see the performance improvements on WN18RR and NELL-995-h100 are much more than FB15k-237. This is due to the presence of a significant amount of hierarchical relations in WordNet and NELL compared to Freebase. We still observe SEPA outperforms SEA on FB15k-237 dataset. The main reason is that SEPA combines hyperbolic manifolds with various transformations used in queries of different models, so it is capable of capturing the mixture of structural and logical patterns in a low dimension (e.g., compositional patterns in Freebase). Even though we did not combine AttE and AttH directly, but only used reflection and the hyperbolic projection, respectively, we were still able to outperform them. Similarly, SEPA outperforms MuRP in low dimensions, and SEA outperforms TuckER in high dimensions in all metrics apart from the H@1 of FB15k-237. More details are available in Appendix A.6.

Our combination model increases the expressiveness of individual models (Proposition 3.2), having the best performance gain in low-dimensional space. Besides, our model takes advantage of the inference power of the base models with fewer constraints (Table 1) by utilizing the attention mechanism. On the other hand, in high-dimensional space, Euclidean models are proven to be fully expressive [40]. Hence, even though SEA outperforms all baselines, the performance gain is not as significant as in low-dimensions.

4.5 Further analyses

We additionally make a series of further analyses to evaluate the performance of our attention-base combination function. First, we

want to show that our model is able to increase the precision of predictions for both symmetric and antisymmetric relations. Table 2 shows the H@1 results in WN18RR, in the low-dimensional setting of SEPA, compared to the individual combined KGE. Further results on H@10 can be found in Appendix A.5. For example, if we look at the symmetric relation *derivationally related form*, we can see that the H@1 of TransE is very low when compared to the one of ComplEx and DistMult, and yet, our model was able to improve this metric. Similarly, when we look at an antisymmetric relation (e.g., *member of domain usage*) we have the opposite situation, having high performance for TransE and a lower one for ComplEx and DistMult. The intuition is that the attention base combination can effectively give more importance to the best models for the specific kind of relation involved in the query. Such intuition is reinforced in Figure 3, which shows the (averaged) attention value among the individual models for the above-mentioned relations. It shows that the attention function can effectively select the correct proportion among the models for the two different kinds of relations.

Besides, the importance of the attention function is highlighted by our ablation study, which consists of turning off the attention from our best models, SEPA at dimension 32, and SEA at dimension 500. We obtained two new versions of the models, namely **SEP** and **SE**. Tables 3, 4 show that SEPA and SEA outperform SEP and SE.

Table 5: Comparison between our proposed models and the ensemble models proposed in MulDE [37]. Values marked ‘-’ were not reported in the original paper.

Model	WN18RR			FB15k-237		
	MRR	H@1	H@10	MRR	H@1	H@10
MulDE ₁	0.481	<u>0.433</u>	0.574	<u>0.328</u>	<u>0.237</u>	<u>0.515</u>
MulDE ₂	<u>0.482</u>	0.430	0.579	-	-	-
SEPA ₁	0.481	0.441	0.562	0.332	0.243	0.509
SEPA ₂	0.491	0.448	0.572	0.344	0.252	0.528

4.6 Comparison With Ensemble Models

We further compare our models to MulDE [37], which uses 64 to 512 dimensional embeddings for the teachers, and 8 to 64 dimensional ones for the junior embedding. We selected the best version of their model, MulDE₁ having 32 and 64 as junior and teachers dimensions, respectively, and, for a fair comparison, MulDE₂, having dimension 64 for both junior and teachers embeddings. We brought our models to their setting, testing SEPA at dimension 32 (SEPA₁) and 64 (SEPA₂) for both WN18RR and FB15k-237. Table 5 shows that apart from the value of H@10 in WN18RR, our models substantially outperform such baselines, having for MRR and H@1 up to 3.46% relative improvements. Besides, we notice that when increasing the number of dimensions the performance of MulDE on H@1 in WN18RR decreases, and the ones of MRR and H@10 slightly increase. On the other hand, our models can substantially improve their performance when increasing the number of dimensions.

4.7 Comparison With Models On Ultrahyperbolic Space

Additionally, we compared our models against the best versions of UltraE [42]. Even though we did not utilize Ultrahyperbolic

Elements	Model	WN18RR				FB15k-237				NELL-995-h100			
		MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
Individual Models	TransE	0.358	0.263	0.423	0.527	0.292	0.208	0.319	0.461	0.267	0.188	0.298	0.423
	DistMult	0.383	0.370	0.385	0.405	0.299	0.209	0.330	0.477	0.279	0.207	0.308	0.419
	RotatE	0.389	0.376	0.392	0.410	0.258	0.178	0.284	0.416	0.264	0.193	0.292	0.405
	ComplEx	0.419	0.395	0.426	0.465	0.264	0.184	0.289	0.421	0.247	0.176	0.275	0.387
	AttE	0.463	<u>0.430</u>	0.474	0.528	<u>0.314</u>	<u>0.227</u>	0.343	0.489	<u>0.334</u>	<u>0.247</u>	<u>0.375</u>	<u>0.503</u>
	AttH	<u>0.468</u>	0.429	<u>0.485</u>	<u>0.539</u>	<u>0.314</u>	0.223	<u>0.346</u>	0.498	0.326	0.240	0.367	0.493
Our models	SEPA	0.481	0.441	0.496	0.562	0.332	0.243	0.363	0.509	0.346	0.261	0.385	0.508
	SEA	0.468	0.430	0.485	0.538	0.326	0.238	0.356	0.504	0.333	0.245	0.376	0.504
Ablation	SEP	0.478	0.437	0.493	0.556	0.329	0.239	0.361	0.509	0.340	0.254	0.380	0.505

Table 3: Link prediction evaluation on datasets for d=32. Best score and best baseline are in bold and underlined, respectively.

Elements	Model	WN18RR				FB15k-237				NELL-995-h100			
		MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
Individual Models	TransE	0.356	0.256	0.419	0.531	0.336	0.243	0.369	0.524	0.300	0.212	0.340	0.469
	DistMult	0.443	0.412	0.453	0.504	0.343	0.249	0.380	0.533	0.322	0.238	0.359	0.486
	RotatE	0.387	0.376	0.392	0.409	0.266	0.188	0.289	0.422	0.322	0.238	0.359	0.493
	ComplEx	0.487	0.443	0.503	0.573	0.265	0.186	0.290	0.422	0.323	0.237	0.362	0.492
	AttE	<u>0.491</u>	<u>0.444</u>	<u>0.507</u>	<u>0.583</u>	<u>0.359</u>	0.264	<u>0.395</u>	<u>0.548</u>	<u>0.377</u>	<u>0.292</u>	<u>0.419</u>	<u>0.539</u>
	AttH	0.482	0.434	0.502	0.576	0.356	0.262	0.393	0.546	0.366	0.279	0.412	0.532
Our models	SEPA	0.480	0.436	0.498	0.570	0.354	0.259	0.390	0.545	0.347	0.260	0.387	0.520
	SEA	0.500	0.454	0.518	0.591	0.360	0.264	0.398	0.549	0.384	0.294	0.432	0.554
Ablation	SE	0.495	0.448	0.513	0.587	0.353	0.259	0.389	0.542	0.381	0.292	0.427	0.548

Table 4: Link prediction evaluation on datasets for d=500.**Table 6:** Comparison between our proposed models and the best UltraE [42] in WN18RR. Best score in bold and second best underlined.

	Model	WN18RR			
		MRR	H@1	H@3	H@10
d=32	UltraE (q=4)	0.488	<u>0.440</u>	0.503	<u>0.558</u>
	SEPA	<u>0.481</u>	0.441	<u>0.496</u>	0.562
	SEA	0.466	0.425	0.482	0.542
d=500	UltraE (q=40)	0.501	<u>0.450</u>	<u>0.515</u>	0.592
	SEPA	0.480	0.436	0.498	0.570
	SEA	<u>0.500</u>	0.454	0.518	<u>0.591</u>

space as a sophisticated manifold containing several sub-manifolds, our models get competitive results to the state-of-the-art in the Ultrahyperbolic space (Table 6). In particular, SEPA gets competitive results in low-dimensions, while SEA in high-dimensions.

One may consider using our idea to integrate approaches such as [12, 42] with other baselines. However, due to their involved multiple geometric spaces, such integration will require a substantial revision of combinations of transformations and, hence, is left for future work.

5 CONCLUSION

In this paper, we propose a new approach that facilitates the combination of the query representations from a wide range of popular knowledge graph embedding models, designed in different

spaces such as Euclidean, Hyperbolic, ComplEx, etc. We presented a spherical approach together with attention to queries to capture heterogeneous logical and structural patterns. We presented a theoretical analysis to justify such characteristics in expressing and inferring patterns and provided experimental analysis on various benchmark datasets with different rates of patterns to show our models uniformly perform well in link prediction tasks on various datasets with diverse characteristics in terms of patterns. Our ablation studies, relation analysis on WN18RR and analysis of the learned attention values show our models mainly take the advantage of the best-performing models in link prediction tasks. By doing that, we achieved state-of-the-art results in Euclidean and Hyperbolic spaces.

In future work, we will combine various manifolds besides combining the queries in knowledge graph embedding. Additionally, the proposed approach could be applied to other tasks. For example, it could be possible to use an attention mechanism to combine multi-hop queries computed using different complex query answering methods [27, 28].

ACKNOWLEDGMENTS

This work has received funding from the following projects: The European Union’s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No: 860801; BMWi Servicemeister (01MK20008F); DFG - COFFEE (STA 572_15-2); and DFG Excellence Strategy in the Clusters of Excellence IntCDC at the University of Stuttgart, RP 20.

REFERENCES

- [1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *ISWC/ASWC (Lecture Notes in Computer Science, Vol. 4825)*. Springer, 722–735.
- [2] Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. Multi-relational poincaré graph embeddings. *Advances in Neural Information Processing Systems* 32 (2019), 4463–4473.
- [3] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. 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 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 5184–5193. <https://doi.org/10.18653/v1/D19-1522>
- [4] D. Beckett. 2004. RDF/XML Syntax Specification. W3C TR.
- [5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013), 2787–2795.
- [6] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 6901–6914.
- [7] S. Chang. 2018. Scaling Knowledge Access and Retrieval at Airbnb. AirBnB Medium Blog.
- [8] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *AAAI*. AAAI Press, 1811–1818.
- [9] D. Devarajan. 2017. Happy Birthday Watson Discovery. IBM Cloud Blog.
- [10] X. L. Dong, X. He, A. Kan, and et al. 2020. AutoKnow: Self-Driving Knowledge Collection for Products of Thousands of Types. *KDD* (2020), 2724–2734.
- [11] Genet Asefa Gesese, Russa Biswas, Mehwish Alam, and Harald Sack. 2021. A survey on knowledge graph embeddings with literals: Which model links better literal-ly? *Semantic Web* 12, 4 (2021), 617–647. <https://doi.org/10.3233/SW-200404>
- [12] Albert Gu, Frederic Sala, Beliz Gunel, and Christopher Ré. 2019. Learning Mixed-Curvature Representations in Product Spaces. In *ICLR (Poster)*. OpenReview.net.
- [13] F. Hamad, I. Liu, and X. Xing Zhang. 2018. Food Discovery with Uber Eats: Building a Query Understanding Engine. Uber Engineering Blog. <https://www.uber.com/en-DE/blog/uber-eats-query-understanding/>
- [14] Nicolas Heist and Heiko Paulheim. 2019. Uncovering the Semantics of Wikipedia Categories. In *ISWC (1) (Lecture Notes in Computer Science, Vol. 11778)*. Springer, 219–236.
- [15] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys (CSUR)* 54, 4 (2021), 1–37.
- [16] Wei Hu, Honglei Qiu, Jiacheng Huang, and Michel Dumontier. 2017. BioSearch: a semantic search engine for Bio2RDF. *Database J. Biol. Databases Curation* 2017 (2017), bax059. <https://doi.org/10.1093/database/bax059>
- [17] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Trans. Neural Networks Learn. Syst.* 33, 2 (2022), 494–514.
- [18] Unmesh Joshi and Jacopo Urbani. 2022. Ensemble-Based Fact Classification with Knowledge Graph Embeddings. In *ESWC (Lecture Notes in Computer Science, Vol. 13261)*. Springer, 147–164.
- [19] Seyed Mehran Kazemi and David Poole. 2018. SimplE Embedding for Link Prediction in Knowledge Graphs. In *NeurIPS*. 4289–4300.
- [20] Denis Krompaß and Volker Tresp. 2015. Ensemble solutions for link-prediction in knowledge graphs. In *Proceedings of the 2nd Workshop on Linked Data for Knowledge Discovery, Porto, Portugal*. 1–10.
- [21] Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical Tensor Decomposition for Knowledge Base Completion. In *ICML (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 2869–2878.
- [22] Mojtaba Nayyeri, Sahar Vahdati, Can Aykulu, and Jens Lehmann. 2021. 5* Knowledge Graph Embeddings with Projective Transformations. In *AAAI*. AAAI Press, 9064–9072.
- [23] Mojtaba Nayyeri, Sahar Vahdati, Emanuel Sallinger, Mirza Mohtashim Alam, Hamed Sharif Yazdi, and Jens Lehmann. 2021. Pattern-Aware and Noise-Resilient Embedding Models. In *ECIR (1) (Lecture Notes in Computer Science, Vol. 12656)*. Springer, 483–496.
- [24] Mojtaba Nayyeri, Chengjin Xu, Yadollah Yaghoobzadeh, Sahar Vahdati, Mirza Mohtashim Alam, Hamed Sharif Yazdi, and Jens Lehmann. 2021. Loss-Aware Pattern Inference: A Correction on the Wrongly Claimed Limitations of Embedding Models. In *PAKDD (3) (Lecture Notes in Computer Science, Vol. 12714)*. Springer, 77–89.
- [25] N. Fridman Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor. 2019. Industry-scale knowledge graphs: lessons and challenges. *CACM* 62, 8 (2019), 36–43.
- [26] Umair Qudus, Michael Röder, Muhammad Saleem, and Axel-Cyrille Ngonga Ngomo. 2022. HybridFC: A Hybrid Fact-Checking Approach for Knowledge Graphs. In *ISWC (Lecture Notes in Computer Science, Vol. 13489)*. Springer, 462–480.
- [27] Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over Knowledge Graphs in Vector Space Using Box Embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- [28] Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. *Advances in Neural Information Processing Systems* 33 (2020), 19716–19726.
- [29] Baoxu Shi and Tim Weninger. 2017. ProjE: Embedding Projection for Knowledge Graph Completion. In *AAAI*. AAAI Press, 1236–1242.
- [30] S. Shrivastava. 2017. Bring rich knowledge of people, places, things and local businesses to your apps. BingBlogs.
- [31] A. Singhal. 2012. Introducing the Knowledge Graph: things, not strings. Google Blog. <https://blog.google/products/search/introducing-knowledge-graph-things-not/>
- [32] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW*. ACM, 697–706.
- [33] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.
- [34] Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, CVSC 2015, Beijing, China, July 26-31, 2015*. Association for Computational Linguistics, 57–66.
- [35] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*. JMLR.org, 2071–2080.
- [36] D. Vrandecic and M. Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.
- [37] Kai Wang, Yu Liu, Qian Ma, and Quan Z Sheng. 2021. Mulde: Multi-teacher knowledge distillation for low-dimensional knowledge graph embeddings. In *Proceedings of the Web Conference 2021*. ACM / IW3C2, 1716–1726.
- [38] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.
- [39] Yinquan Wang, Yao Chen, Zhe Zhang, and Tian Wang. 2022. A Probabilistic Ensemble Approach for Knowledge Graph Embedding. *Neurocomputing* (2022).
- [40] Yanji Wang, Rainer Gemulla, and Hui Li. 2018. On multi-relational link prediction with bilinear models. In *Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press, 4227–4234.
- [41] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 28*. AAAI Press, 1112–1119.
- [42] Bo Xiong, Shichao Zhu, Mojtaba Nayyeri, Chengjin Xu, Shirui Pan, Chuan Zhou, and Steffen Staab. 2022. Ultrahyperbolic Knowledge Graph Embeddings. In *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. ACM, 2130–2139.
- [43] Wenhui Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Association for Computational Linguistics, 564–573.
- [44] Chengjin Xu, Mojtaba Nayyeri, Sahar Vahdati, and Jens Lehmann. 2021. Multiple Run Ensemble Learning with Low-Dimensional Knowledge Graph Embeddings. In *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.
- [45] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- [46] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. 2731–2741.
- [47] Xiaofei Zhou, Qianan Zhu, Ping Liu, and Li Guo. 2017. Learning knowledge embeddings by combining limit-based scoring loss. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1009–1018.

A APPENDIX

This appendix includes the proof of the propositions proposed in the paper, followed by optimal hyperparameter settings, additional related works, further explanation of Table 1, and more experimental analysis.

A.1 Proofs

Proof of Proposition 3.1. This proposition states that a vector \mathbf{a}_1 is a solution of the minimization problem if and only if \mathbf{a}_1 lies between the vector query embeddings \mathbf{q}_1 and \mathbf{q}_2 of a query q . Without loss of generality, we assume that \mathbf{q}_1 and \mathbf{q}_2 lie in the x-axis. The equivalence is proved in both directions.

- (1) If \mathbf{a}_1 is the solution of the minimization problem, we need to prove that \mathbf{a}_1 lies between \mathbf{q}_1 and \mathbf{q}_2 . Assume that \mathbf{a}_1 does not lie between \mathbf{q}_1 and \mathbf{q}_2 . By the triangle inequality, $p(\mathbf{q}_1, \mathbf{a}_1) + p(\mathbf{q}_2, \mathbf{a}_1) \geq p(\mathbf{q}_1, \mathbf{q}_2)$. Since $p(\mathbf{q}_1, \mathbf{q}_E) + p(\mathbf{q}_2, \mathbf{q}_E) = p(\mathbf{q}_1, \mathbf{q}_2)$, it follows that \mathbf{a}_1 is not the minimum. This contraction comes from assuming that \mathbf{a}_1 does not lie between \mathbf{q}_1 and \mathbf{q}_2 . Hence, \mathbf{a}_1 lies between \mathbf{q}_1 and \mathbf{q}_2 .
- (2) Let \mathbf{a}_1 be in between of \mathbf{q}_1 and \mathbf{q}_2 , and \mathbf{a}_2 be an arbitrary vector that does not lie between \mathbf{q}_1 and \mathbf{q}_2 . By the triangle inequality, $p(\mathbf{q}_1, \mathbf{a}_2) + p(\mathbf{q}_2, \mathbf{a}_2) \geq p(\mathbf{q}_1, \mathbf{q}_2)$. Since \mathbf{a}_1 lies between \mathbf{q}_1 and \mathbf{q}_2 , it follows that $p(\mathbf{q}_1, \mathbf{q}_2) = p(\mathbf{q}_1, \mathbf{a}_1) + p(\mathbf{q}_1, \mathbf{a}_1)$. Substituting $p(\mathbf{q}_1, \mathbf{q}_2)$ with the right side of the last equation in the last inequality, we conclude that $p(\mathbf{q}_1, \mathbf{a}_2) + p(\mathbf{q}_1, \mathbf{a}_2) \geq p(\mathbf{q}_1, \mathbf{a}_1) + p(\mathbf{q}_2, \mathbf{a}_1)$. Hence, \mathbf{a}_1 is a solution of the minimization problem. \square

Proof of Proposition 3.2. Here, we prove that our hyperbolic version of our model SEPA subsumes AttH. We start with the sketch of the proof on the example of AttH, and then present the complete proof in detailed steps in a general case. Because every transformation used in AttH is also used in SEPA with attention values, if we set the attention values corresponding to the transformations used in AttH, and set the other attention values to zero, then SEPA represents AttH scoring. Therefore, AttH is a special case of SEPA.

Similar points exist for ComplEx, TransE and DistMult.

Here we prove mathematically that SEPA can represent the same query to AttH. let us assume that we combine the queries \mathbf{q}_{AttH} , \mathbf{q}_{TransE} , $\mathbf{q}_{DistMult}$, and $\mathbf{q}_{ComplEx}$, which are the query representations of AttH, TransE, DistMult, and ComplEx respectively. According to Equation 12, we have $\mathbf{q}_{SEPA} = \mathbf{a}_1 \mathbf{q}_{AttH} + \mathbf{a}_2 \mathbf{q}_{TransE} + \mathbf{a}_3 \mathbf{q}_{DistMult} + \mathbf{a}_4 \mathbf{q}_{ComplEx}$. By setting relation embeddings of ComplEx and DistMult to zero, we have $\mathbf{q}_{DistMult} = \mathbf{q}_{ComplEx} = \mathbf{0}$. Now, we need to find a way to cancel the query of TransE by setting the attention value of the TransE query to zero. \mathbf{a}_2 will be close to zero if $r_{TransE} = -Mw$ (in Equation 13) ($M > 0$ is a sufficiently large number). Therefore, the query of TransE will be canceled in \mathbf{q}_{SEPA} , and we then have $\mathbf{q}_{SEPA} \approx \mathbf{a}_1 \mathbf{q}_{AttH}$. Note that \mathbf{a}_1 is close to one if w (in Equation 13) is a large vector almost parallel to \mathbf{q}_{AttH} . Therefore, we have $\mathbf{q}_{SEPA} \approx \mathbf{q}_{AttH}$. Because the query is approximated, the score is also approximated. Similarly, we can prove that SEPA can approximate the query of TransE, ComplEx, and DistMult, thus their score as well.

A similar process is also applicable for proof of subsumption between SEA and AttE, where AttE is a special case of SEA. Therefore, SEA can represent any scoring presented by AttE, and this completes the proof.

Proof of Corollary 3.3. This is a direct conclusion of Proposition 3.2, thus it is a corollary. In Proposition 3.2, we prove that SEPA and SEA subsume TransE, DistMult, RotatE, ComplEx and AttH (AttE), and also we know from [22] that if a model A subsumes a model B, then the model A can infer all patterns that the model B can infer. Therefore, SEPA and SEA can infer the patterns that TransE, DistMult, RotatE, ComplEx, and AttH (AttE) can infer. It has been already proven in [33] that the models are capable of inferring anti-symmetry, symmetry, composition, and inversion.

Table 7: Comparison of H@10 for WN18RR relations. TE = TransE, CE = ComplEx, DM = DisMult

Relation	TE	CE	DM	SEPA
member meronym	0.427	0.223	0.130	0.409
hypernym	0.197	0.124	0.065	0.277
has part	0.334	0.227	0.142	0.323
instance hypernym	0.480	0.426	0.221	0.500
member of domain region	0.417	0.229	0.154	0.481
member of domain usage	0.423	0.231	0.104	0.458
synset domain topic of	0.447	0.241	0.162	0.461
also see	0.723	0.625	0.616	0.714
derivationally related form	0.958	0.957	0.946	0.966
similar to	1.000	1.000	1.000	1.000
verb group	0.962	0.974	0.974	0.974

A.2 Best hyperparameters per model and per dataset

Table 8, 9, 10 specifies the hyperparameter lists corresponding to the WN18RR, FB15k-237 and NELL-995-h100 respectively.

A.3 Related Work: combination between machine learning models *including* KGE

We review the related work corresponding to the general approaches in machine learning, including embedding that combines different models.

DuEL [18] exploits embedding models for classifying facts to be either true or false, rather than ranking them. Starting from a tail query $(h, r, ?)$, it uses an embedding model to obtain the top k list of predicted answers and feeds different classifiers (e.g., LSTM, CNN) to label each answer as true or false. Finally, the predictions are ensembled using different techniques. A similar approach [26], also proposed an ensemble-based framework for fact-checking. Starting from a triple (h, r, t) , it runs three different methods: (1) a text-based approach, (2) a KGE model, and (3) a path-based approach. It concatenates the outputs and lets a neural network compute a final veracity score. On the other hand, in our work, we propose to combine query representations of different KGE models.

Model	WN18RR							
	m	lr	o	n	b	dt	ar	dn
SEPA (d=32)	TCD	0.001	Adam	250	500	single	yes	yes
SEPA (d=500)	TCD	0.001	Adam	250	500	single	yes	yes
SEA(d=32)	TCD	0.001	Adam	250	100	single	yes	yes
SEA(d=500)	ALL	0.001	Adam	250	100	single	no	no

Table 8: Best Hyperparameters for WN18RR. T = TransE, D = DistMult, C = ComplEx, R = RotatE, AttH(Reflection) = A. ALL = all models combined

Model	FB15k-237							
	m	lr	o	n	b	dt	ar	dn
SEPA (d=32)	TCD	0.05	Adagrad	250	500	double	yes	no
SEPA (d=500)	ALL	0.05	Adagrad	250	100	double	no	no
SEA(d=32)	ALL	0.1	Adagrad	250	500	single	no	no
SEA(d=500)	ALL	0.1	Adagrad	250	500	single	no	no

Table 9: Best Hyperparameters for FB15k-237.

Model	NELL-995-h100							
	m	lr	o	n	b	dt	ar	dn
SEPA (d=32)	ALL	0.001	Adam	250	100	single	no	no
SEPA (d=500)	ALL	0.001	Adam	250	500	single	no	no
SEA(d=32)	ALL	0.001	Adam	250	500	single	no	no
SEA(d=500)	ALL	0.001	Adam	250	500	single	yes	no

Table 10: Best Hyperparameters for NELL-995-h100.

A.4 Further Explanation of Table 1

Here, we present a further explanation of the results in Table 1, and show how to compute the constraints.

- *TransE/Symmetric*: According to the Table 1, TransE cannot model a symmetric pattern. To show this, we take a triple (h, r, t) and its symmetry (t, r, h) . To model both triples in the vector space by TransE, we need to have

$$\mathbf{h} + \mathbf{r} = \mathbf{t}, \quad \mathbf{t} + \mathbf{r} = \mathbf{h}.$$

Combining the two equations leads to $\mathbf{r} = \mathbf{0}$. This is counted as incapability for modeling symmetry by TransE because a null vector for relation embedding leads to the same embeddings for all entities connected by the relation. This is in contradiction with the assumption of embedding models to assign unique vectors to each entity in the KG.

- *TransE/AntiSymmetry*: Lets r be antisymmetry. Thus, if (h, r, t) is true, we have $\mathbf{h} + \mathbf{r} = \mathbf{t}$. This trivially leads to $\mathbf{t} + \mathbf{r} \neq \mathbf{h}$ without using further constraint. Note that we use the assumption that entity embeddings are unique in the vector space.
- *TransE/Hierarchy*: According to the Table 1, TransE cannot model hierarchical patterns. To show this, we take a small part of a tree as a hierarchical structure. We consider the root (h) and two children of the root node entity t_1, t_2 . The triples (h, r, t_1) and (h, r, t_2) are valid. To model both triples in the vector space by TransE, we need to have

$$\mathbf{h} + \mathbf{r} = \mathbf{t}_1, \quad \mathbf{h} + \mathbf{r} = \mathbf{t}_2.$$

Comparing the two equations, we conclude that $\mathbf{t}_1 = \mathbf{t}_2$. This is counted as incapability for modeling hierarchy by TransE

Table 11: Comparison with MuRP, against models of dimension 32, and TuckER, against models of dimension 500, which results were taken from [6]. Best score in bold and second best underlined.

Model	WN18RR				FB237			
	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10
MuRP	0.465	0.420	0.484	<u>0.544</u>	0.323	0.235	0.353	0.501
SEPA	0.481	0.441	0.496	0.562	0.332	0.243	0.363	0.509
SEA	0.468	0.430	0.485	0.538	0.326	0.238	0.356	0.504
TuckER	0.470	<u>0.443</u>	0.482	0.526	0.358	0.266	0.394	0.544
SEA	0.480	0.436	<u>0.498</u>	<u>0.570</u>	0.354	0.259	0.390	0.545
SEA	0.500	0.454	0.518	0.591	0.360	<u>0.264</u>	0.398	0.549

because all entities embeddings are assumed to be unique in the vector space which is not the case for hierarchy.

A similar calculation is required for other patterns/models which we do not go through because the process is similar to TransE.

Given a pattern, a model with fewer constraints has better inference capability compared to the models with more constraints. According to the Theorem 3.2 and 3.3, our model can take advantage of each model without being affected by their incapability due to using a relation-specific attention mechanism.

A.5 H@10 per relation

Table 7 presents the performance of our model and different base models considering the metric Hits@10 on each relation of the WN18RR dataset. We have a similar observation to the results on Hits@1. In most cases, our model outperforms the base models. For a few relations, our model gets slightly lower performance than the base models. We hypothesize that might be related to numerical optimization, or our model aimed to increase the overall accuracy for all relations, so for the relations with less frequencies, our model gets slightly lower performance to have overall higher performance in all relations when increase in performance of one relation leads to decrease the performance of other relation.

A.6 Comparison with TuckER and MuRP

Here, we present further analysis of our models with MuRP and TuckER. In particular, as shown in Table 11 we compared MuRP with the low-dimensional versions of our models, and TuckER with the high-dimensional ones. The reason is that MuRP is a hyperbolic model, hence its best performances are shown in low-dimensional space, while TuckER is a Euclidean one and shows the best performances in a high-dimensional space.

We observe that SEPA outperforms MuRP in all metrics, having for example a relative improvement of around 5% in the H@1 metric of the WN18RR dataset. Besides, SEA outperforms TuckER for all metrics of WN18RR and most metrics of FB15k-237. In particular, it obtains around 12% relative improvements in the H@1 of WN18RR. On the other hand, it obtains around 0.75% relative worsening in the H@1 of FB15k-237.

Overall, we observe that our models were able to outperform such baselines even though they were not included in the combination.