Fantastic Knowledge Graph Embeddings and How to Find the Right Space for Them

Mojtaba Nayyeri[1,2(✉)], Chengjin Xu[1], Sahar Vahdati[2,3], Nadezhda Vassilyeva[1], Emanuel Sallinger[3,4], Hamed Shariat Yazdi[1], and Jens Lehmann[1,5]

[1] Smart Data Analytics Group (SDA), University of Bonn, Bonn, Germany
{nayyeri,xuc,jens.lehmann}@cs.uni-bonn.de, nadya.vassilyeva@gmail.com, shariatyazdi@gmail.com
[2] InfAI Dresden Lab, Dresden, Germany
vahdati@infai.org
[3] University of Oxford, Oxford, UK
emanuel.sallinger@cs.ox.ac.uk
[4] TU Wien, Wien, Austria
[5] Fraunhofer IAIS Dresden Lab, Dresden, Germany
jens.lehmann@iais.fraunhofer.de

Abstract. During the last few years, several knowledge graph embedding models have been devised in order to handle machine learning problems for knowledge graphs. Some of the models which were proven to be capable of inferring relational patterns, such as symmetry or transitivity, show lower performance in practice than those not allowing to infer those patterns. It is often unknown what factors contribute to such performance differences among KGE models in the inference of particular patterns. We develop the concept of a solution space as a factor that has a direct influence on the practical performance of knowledge graph embedding models as well as their capability to infer relational patterns. We showcase the effect of solution space on a newly proposed model dubbed SpacE^{ss}. We describe the theoretical considerations behind the solution space and evaluate our model against state-of-the-art models on a set of standard benchmarks namely WordNet and FreeBase.

Keywords: Link prediction · Knowledge graph embedding · Relation pattern · Solution space · Knowledge completion

1 Introduction

Knowledge Graphs (KGs) have recently become a crucial part of different AI applications. In its simplest definition, a KG is a set of triples of the form (h, r, t), where h and t refer to entities and r refers to the relation between these entities. Following this structure, a lot of KGs have been published in recent years,

J. Z. Pan et al. (Eds.): ISWC 2020, LNCS 12506, pp. 438–455, 2020.
https://doi.org/10.1007/978-3-030-62419-4_25

such as Freebase [2], WordNet [14], WikiData [23], and DBpedia [11]. Although quantitatively KGs often consist of several thousand entities and relations and millions of triples, this is nowhere near enough to cover the knowledge that exists in the real world – even when restricted to a particular domain. Therefore, KGs often suffer from incompleteness.

One of the common approaches for knowledge graph completion is the *Knowledge Graph embeddings* (KGEs). KGEs assign a latent feature vector to each entity and relation in a KG. Furthermore, a scoring function is used to define the degree to which a relation between two entities is plausible. Relations between entities of a graph often follow particular relational patterns, e.g. symmetric, transitive, inverse patterns. Such patterns are generally given by logical formulas [8,18] and the ability to infer them is broadly considered as expressiveness of a KGE model [25]. However, not every KGE model is designed to infer all kinds of patterns, meaning that characteristics of the patterns are not taken into consideration by the model in the inference of implicit knowledge. Given a logical formula of the form $premise \implies conclusion$, a constraint is enforced for the plausibility of the grounding atoms involved in the conclusion when the grounding of atoms in the premise holds. For example, given a symmetric rule, $(e_l, r, e_r) \leftrightarrow (e_r, r, e_l)$, if a grounding (h, r, t) of (e_l, r, e_r) is true, then (t, r, h) is constrained to be true as well. Such a constraint should be followed by KGE models in the associated vector space measuring the correctness of triples.

Already existing models have been majorly designed and evaluated without specifically considering relational patterns. This can lead to two problems: the models are either 1.) not capable of encoding any pattern or 2.) they are only partially capable of encoding patterns, both of which can lead to wrong inferences. For example, when the relation vector is non-zero, one of the baseline KGEs dubbed TransE [3] is not able to infer symmetric relational patterns ($(h, r, t) \implies (t, r, h)$). Instead, it only infers the explicit triple (h, r, t) from such a relation and not its symmetrical triple (t, r, h) [13,26]. Yet, "the success of such a task heavily relies on the ability of modeling and inferring the patterns of (or between) the relations" [19]. The ability of encoding relational patterns and expressiveness of KGE models is a property of their *solution space* (SS), i.e., the set of all possible vectors that can be assigned to the entities and relations involved in a particular pattern. This heavily depends on multiple aspects such as *data complexity*, *model formulation*, and *embedding dimension*. Data complexity in KGs denotes the extent to which the relations between entities are interconnected. A combination of relational patterns results in more constraints which causes higher data complexity. This also enforces constraints in the associated vector space for the KGE model. For example, a cycle (sequence of nodes of a graph connected in a closed path) containing symmetric and anti-symmetric relations causes wrong inference in the RotatE model [19] (see Sect. 3).

The model formulation defines the way entity and relation vectors are optimized in the vector space to measure triple correctness. Constraints caused by patterns in the vector space are dependent on model-formulation and directly limit the solution space of any model. Given a triple (h, r, t) and its embedding

vectors $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ where r is symmetric, TransE induces the correctness of the triple as $(\mathbf{h} + \mathbf{r} \approx \mathbf{t})$ and its symmetrical counterpart as $(\mathbf{t} + \mathbf{r} \approx \mathbf{h})$. Therefore, for symmetric relation r, the solution space for the relation is one i.e. $\mathbf{r} = \mathbf{0}$.

Generally, using a higher embedding dimension (values of 1,000 [19] or even 10,000 [6] have been used) is considered as one way of increasing the solution space of a KGE model by increasing the number of possible embeddings satisfying the relational patterns. However, using a very big dimension is not always practical in large scale KGs due to the prohibitive memory requirements. In addition, there are also cases in which increasing the embedding dimension does not necessarily improve the solution space. For example, in the TransE the solution space for encoding symmetric patterns [19] is always only one independent of the embedding dimension.

In this paper, we show a novel approach towards overcoming the problems of KGE models to encode relational patterns. Our goal is to show how the model formulation can extend the *solution space* and improve the ability for encoding relational patterns. To do so, we introduce a new knowledge graph embedding model *SpacE*^{ss}[1] and show that it is capable of expressing each pattern reflexive, symmetric, and inverse patterns individually. *SpacE*^{ss} covers both translation and rotation transformations which enables it to inherit the expressive power of TransE, RotatE, and TransComplEx for encoding *composition, transitivity, equivalence, and implication* [3,16,19]. Compared to RotatE and TransComplEx, SpacE^{ss} provides a bigger solution space when encoding different relation patterns. Finally, we evaluate our model experimentally on several popular KGs to demonstrate our model's performance in practice.

2 Related Work

The aim of a KGE model is to optimize a loss function (denoted by $\mathcal{L}$) in order to embed entities and relations of a KG K into a d-dimensional vector space. Each model also defines a score function $f(h, r, t)$, or $f^r_{h,t}$, that measures the probability of correctness of a triple (h, r, t). There are several types of KGE models [24] namely distance-based (TransE, RotatE, TransComplEx), semantic-matching (HolE [17], Distmult [28], complEx [21] and TuckER [22]) and neural network-based (ConvE [4]) models. This section provides a review of popular and prominent score functions used by different embedding models that are distace-based, as they are the main focus of this work.

TransE [3] is one of the early KGE models. The core idea here is to optimize the embedding vectors in real space such that $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ holds for every valid triple (h, r, t). Intuitively, this means that a relation vector is a translation from head to tail. This restrictive assumption enables TransE to properly infer some, but not all of the relation patterns. For example, the relation vector becomes a zero vector when the model encodes a reflexive relation (as for any h, we have $\mathbf{h} + \mathbf{r} \approx \mathbf{h}$). Thus, if a relation is reflexive, it is enforced by the model to

[1] https://github.com/mojtabanayyeri/KGE-Models/tree/master/SpacESS.

be symmetric as well [10]. TransE is a starting point for a family of different embedding models such as TransR [12], TransH [27] and TransD [9,29]. These works attempted to improve expressiveness of TransE by modifying its score function, which is $f^r_{h,t} = \|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$. However, neither of these models succeeded in solving the aforementioned problem [10].

TransComplEx [16] is an extension of TransE from real space to complex space (i.e. $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathcal{C}^d$). Such a space consists of complex numbers shown as $x = Re(x) + i\, Im(x)$. $Re(x)$ and $Im(x)$ refer to the real and imaginary parts of x respectively. TransComplEx represents relation vector $\mathbf{r}$ as a translation from head $\mathbf{h}$ to the complex conjugate of the tail $\bar{\mathbf{t}}$. The score function then becomes $f^r_{h,t} = \|\mathbf{h} + \mathbf{r} - \bar{\mathbf{t}}\|$, where $\|.\|$ is the $L2$ norm of the vector whose elements are the modulus of each complex element of the vector $(\mathbf{h} + \mathbf{r} - \bar{\mathbf{t}})$. TransComplEx encodes reflexive and symmetric relational patterns as well as the ones neither reflexive nor irreflexive. However, later we show the role of the limited space considered by the model and its influences on the performance.

RotatE [19] is a very recent new model that has already garnered a lot of attention as one of the state-of-the-art KGE models. RotatE takes advantage of the Euler formula $e^{i\theta} = \cos(\theta) + i\sin(\theta)$ and requires that for every correct triple (h, r, t) $h_j r_j = t_j$ holds $\forall j \in \{0, \ldots, d\}$ where $|r_j| = \sqrt{Re(r_j)^2 + Im(r_j)^2} = 1$. Setting $|r_j|$ to 1, combined with the Euler formula, means that the model performs a rotation of the j-th element h_j of the head vector $\mathbf{h}$ by the j-th element $r_j = e^{i\theta_{r_j}}$ of a relation vector $\mathbf{r}$ to get the j-th element t_j of the tail vector $\mathbf{t}$, where θ_{r_j} is the phase of the relation r. The score function of RotatE is then defined as $f^r_{h,t} = \|\mathbf{h} \circ \mathbf{r} - \mathbf{t}\|$, where $\circ$ is the Hadamard (element-wise) product of two vectors. RotatE encodes symmetric, inverse, and composition relation patterns. However, the constraints enforced by several patterns restrict the vector space of the embeddings. This restriction indeed causes wrong inferences when the model embeds entities and relations into a vector space (this is discussed with a motivating example in Sect. 3).

To understand the core of the issue, the RotatE model takes a triple (h, r, t) in the KG and returns three vectors $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ and arrange them based on its score function. For symmetric relations, it was shown in [19, Lemma 1] that $\mathbf{r}_j = e^{i0}$ or $e^{i\pi}$, which corresponds to rotation of 0 or π for the j-th element of the relation vector. In this case, the solution space of RotatE has two values of 0 or π per each considered dimension. Therefore, each element of a symmetric relation is either 1 or -1 (e.g. $\mathbf{r} = [1, -1, \ldots, -1]$). Given a vector $\mathbf{h}$, there are only *two* possible options for each element of the tail vector $\mathbf{t}_j$: either $\mathbf{t}_j = \mathbf{h}_j$ or it is the "mirror image" of $\mathbf{h}_j$ (i.e. rotated by π). Later, we raise the attention on an important factor behind this problem explaining the *solution space* in more details. In the next section, we present this problem in an example.

3 Motivating Example

Consider a knowledge graph containing companies with four different relation types between them: two symmetric relations *in a business* (br) relationship

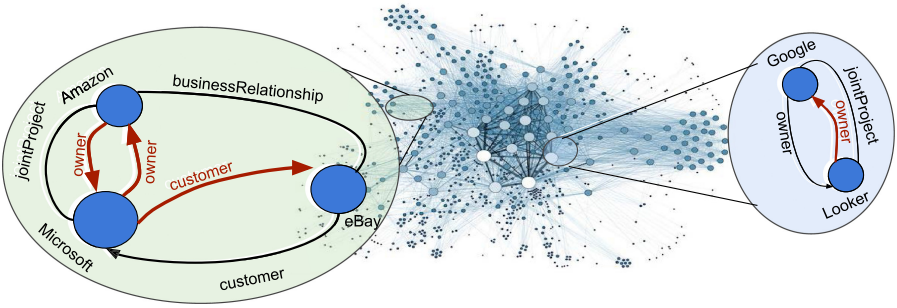


Fig. 1. Motivating Example. The RotatE embedding model infers wrong links (in red) for subgraphs with symmetric and anti-symmetric patterns. (Color figure online)

and in *joint projects* (jp) relationship and two anti-symmetric relationships *is customer of* (c) or *is owner of* (o) of another company. The triples of our concern in the KG are as follow:

$$\begin{cases} eBay \circ r_{br} = Amazon, \\ Amazon \circ r_{jp} = Microsoft, \\ eBay \circ r_c = Microsoft. \end{cases} \qquad \begin{cases} Google \circ r_{jp} = Looker, \\ Google \circ r_o = Looker, \end{cases} \tag{1}$$

In RotatE model, the symmetric relations r_{jp}, r_{br} obtain embedding vectors with elements of either -1 (rotation 0) or 1 (rotation π). The anti-symmetric relations vectors c and o get arbitrary values (except 1 and -1) [19]. Considering the left side of Eq. 1, we substitute Amazon by $eBay \circ \mathbf{r_{br}}$ in the second row. The right side of the equation remains the same. Therefore, we have:

$$\begin{cases} eBay \circ r_{br} \circ r_{jp} = Microsoft, \\ eBay \circ r_c = Microsoft. \end{cases} \qquad \begin{cases} Google \circ r_{jp} = Looker, \\ Google \circ r_o = Looker. \end{cases} \tag{2}$$

These lead to creation of extra constraints $r_{br} \circ r_{jp} = r_c$ and $r_{jp} = r_o$. We already know r_{jp}, r_{br} are symmetric and are represented in vectors with elements of -1 and 1. From the extra constraints created above, r_c is enforced to be symmetric (with elements of -1 and 1). Moreover, r_o will be equivalent with r_{jp}. Thus a set of wrong inferences (shown in Fig. 1 in red) are made by the model, such as $Amazon \circ r_o = Microsoft$, $Microsoft \circ r_o = Amazon$, $Microsoft \circ r_c = eBay$, and $Looker \circ r_o = Google$. This is especially problematic in large scale KGs, where there are many different symmetric relations and millions of entities. In our example, consider just adding the roughly 100 million ownership relations in the EU between companies. As the overwhelming majority are non-symmetric, over 90 million wrong inferences occur.

4 Our Approach

The inference of relational patterns by KGE models heavily depends on the model formulation and causes contradictions between (theoretically proven)

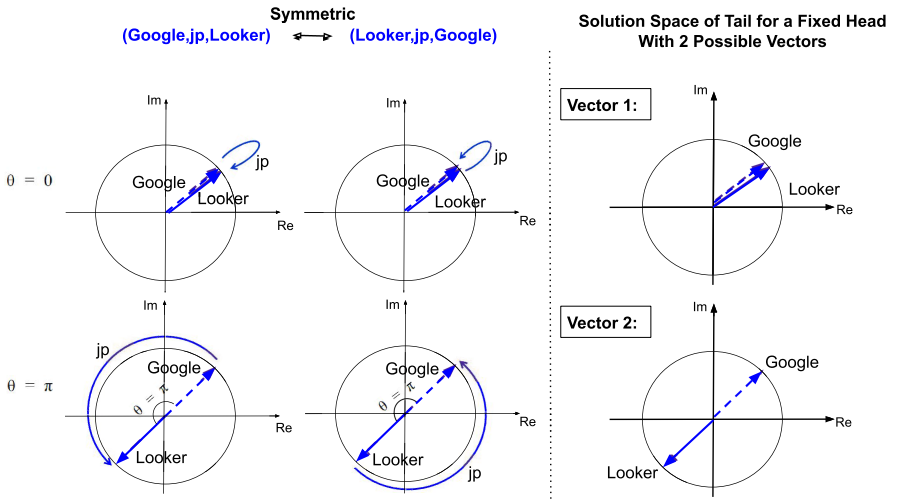


Fig. 2. Solution Space of RotatE for Symmetric. The possible vectors as solutions in SS are shown for symmetric relation of joint project (jp) in RotatE.

expressive power and (practical) performance by the models. When comparing different models, it is often unclear which factors are responsible for this difference and the expressiveness power of such models. Most work focuses on determining whether a model is capable of expressing a relational pattern or not, the most course-grained criterion possible. This course-grained analysis is a good start, but hides a lot of theoretical and practical limitations of models. A more fine-grained understanding of the capability of models is missing so far. Our focused study led us to discover a hidden factor as a cause of this issue namely *solution space*. In this section, we first describe the meaning and formulation of SS and introduce our novel KGE model namely *SpacE*^{ss} empowered by the SS concept.

4.1 Solution Space - a Cause of Expressiveness in KGEs

Initially, using formulation of RotatE model without enforcing any constraints (patterns) in the KG, the possible solutions for representing embedding vectors of each relation r is an ∞ space. However, by enforcing relational patterns for example symmetric constraint, the SS of r reduces to 2 (i.e., $\{-1, 1\}$) per each dimension (or 2^d for all dimension). This reduction in solution space causes issues (e.g. wrong inference) when additional constraints are added as shown in Eq. 1 and Fig. 2. Generally, such constraints enforced by the definition of the corresponding relation pattern over the score function formula of a KGE builds the *solution space (SS)* of the model. Conceptually, SS is the coverage of all the possible variations for the embeddings of the elements of a triple (h,r,t) in the corresponding geometric space e.g. vector space. Here, we provide the formu-

lation of SS and its variations for the considered distance-based KGE models (TransE, RotatE, and TransComplEx) and the considered relational patterns.

The *solution space* of an entity in tail position t is the set $\mathbf{S}$ of all possible vectors for embedding of tail $\mathbf{t}$ having a fixed relation vector $\mathbf{r}$ and a fixed head embedding vector $\mathbf{h}$, such that (h, r, t) is a triple in the KG, i.e. $\mathbf{S}_{t|r,h} = \{\mathbf{t} \mid (h, r, t) \in KG, \varphi_m, \varphi_{mp}, \mathbf{h} \in \mathbf{E}, \mathbf{r} \in \mathbf{R}\}$ where $\mathbf{E}, \mathbf{R}$, are embedding matrices of entities and relations in the model m. φ_m is the enforced constraint by the model formulation to show a triple (h, r, t) is correct e.g. $\varphi_{RotatE} : \mathbf{h} \circ \mathbf{r} = \mathbf{t}$. φ_{mp} is the constraint formulation obtained by the model m with regard to pattern p. For example, for rotate model and the given reflexive relation r, the formula is $\varphi_{RotatEReflexive} : \mathbf{h} \circ \mathbf{r} = \mathbf{h}, \mathbf{t} \circ \mathbf{r} = \mathbf{t}$. The SS for the *head* of a triple given fixed *relation and tail* $\mathbf{S}_{h|r,t}$ and the SS of a *relation* $\mathbf{S}_{r|h,t}$ are defined analogously. We similarly define the *relation-tail* SS: $\mathbf{S}_{r,t|h} = \{(\mathbf{r}, \mathbf{t}) \mid (h, r, t) \in KG, \varphi_m, \varphi_{mp}, \mathbf{h} \in \mathbf{E}\}$. Moreover, φ is a formulation representing the relation solution space $S_{r|t,h}$ in a vector space V if $\mathbf{S}_{r|t,h} = \{\mathbf{r} \mid (h, r, t) \in KG, \varphi_m, \varphi_{mp}, \mathbf{h}, \mathbf{t} \in \mathbf{E}\}$.

This definition holds analogously for all other solution spaces of a model for different patterns. In our company example, the SS of an element in example triple for Amazon being in joint project relationship with Microsoft is $|\mathbf{S}_{\mathsf{jp}|\mathsf{Microsoft},\mathsf{Amazon}}| = 2$ considering all possible cases by RotatE model. In Sect. 4.2, we introduce SpacE^{ss} with considerably larger SS, and show how this improves its ability to represent various relational patterns.

4.2 SpacE^{ss} - A Novel KGE Model Empowered by SS

With a systematic study of SS on already existing models, claimed capable of encoding relational patterns (RotatE, TransComplEx), we concluded that none of the existing models have a proper SS for encoding of relational patterns. The results led us to propose a new embedding model *SpacE^{ss}* considering an extended solution space (SS) compared to existing models. We show the impact of SS encoding different types of relation patterns by our model with its high expressiveness, both theoretically and empirically. The improvement in expressiveness of *SpacE^{ss}* is due the larger space for solutions than RotatE, TransE and TransComplEx.

Given a triple (h, r, t), *SpacE^{ss}* first rotates the head entity counterclockwise and the tail entity clockwise to produce embedding vectors $\mathbf{h}_{\theta^r_L}$ (left rotation) and $\mathbf{t}_{\theta^r_R}$ (right rotation) respectively. It then applies a translation corresponding to the relation vector $\mathbf{r}$ from the relation-specific rotated-head (left rotation) to the relation-specific rotated-tail (right rotation), such that

$$\varphi_{SpacE^{SS}} : \mathbf{h}_{\theta^r_L} + \mathbf{r} = \mathbf{t}_{\theta^r_R}, \tag{3}$$

where $\mathbf{h}_{\theta^r_L}$ and $\mathbf{t}_{\theta^r_R}$ are computed as rotations of $\mathbf{h}_{\theta^r_L} = \mathbf{h}\, e^{i\theta^r_L}$, $\mathbf{t}_{\theta^r_R} = \mathbf{t}\, e^{-i\theta^r_R}$, and, θ^r_L and θ^r_R are the phase vectors for head and tail rotations corresponding to the relation r. The score function that computes the degree of correctness for a triple (h, r, t) is defined as:

$$f^r_{h,t} = \|\mathbf{h}_{\theta^r_L} + \mathbf{r} - \mathbf{t}_{\theta^r_R}\|. \tag{4}$$

Let $\mathbf{h}_i$ and $\mathbf{t}_i$ be the i-th elements (i = 1, ..., d) of head and tail embedding vectors, and $\theta^r_{L_i}$ and $\theta^r_{R_i}$ be the left and right rotation vectors respectively. The rotation is performed element-wise $\mathbf{h}_{\theta^r_{L_i}} = \mathbf{h}_i e^{i\theta^r_{L_i}}$, and $\mathbf{t}_{\theta^r_{R_i}} = \mathbf{t}_i e^{-i\theta^r_{R_i}}$. Our model is sufficiently expressive to encode different relation patterns including reflexive, symmetric, transitive, inverse, implication, equivalence and composition. Note that SpacE^{ss} only uses simple operators of addition, subtraction, and multiplication over embeddings of dimension d. Therefore, the computational complexity of the model is $O(d)$ which is similar to RotatE and TransComplEx.

4.3 Formulation of SS for Distance-Based KGE Models

The capability of a model to encode a specific relational pattern can be proven through a series of steps namely: 1) the formal definition of the considered relational pattern, 2) the formulation of the score function of the considered KGE model, and 3) the triple correctness condition. The latter is used for the scoring function and requires separate calculations which will be introduced in this part.

Conditions of Triple Correctness. The correctness of triples involved in a relational pattern defines whether the pattern holds. Among the wide range of scores assigned by a KGE model to the triples, a concrete threshold γ is required for deciding the correctness of each triple in a KG. In practice, this threshold is set as a hyper-parameter of the underlying KGE model. In this work, we use simplified thresholds [16] for investigating the capability of models to encode relational patterns, for correct triples: (a) if score = 0 then any triple with non-zero score is false, and (b) if score $\leq \gamma$ then any triple with score $> \gamma$ is false. Condition (a) is the most restrictive one and follows the original formulation of translation-based models and applies on the symmetric and inverse relations of all the models in Table 1. In case, the condition (a) does not apply (for example when the SS becomes zero), condition (b) could be considered with a specific γ, and applies on reflexive patterns of all the models in Table 1.

Formulation of Solution Space (SS). Table 1 illustrates the capability of three distance-based KGE models namely SpacE^{ss}, RotatE and TransComplEx in encoding relation patterns namely reflexive, symmetric and inverse. The column *Patterns* specifies each of the investigated relation patterns. The column *Cond.* shows whether threshold condition (a) or (b) is used for the considered pattern. The Table 1 indicates whether the model is capable of encoding the specified pattern under the specified condition or not. The column *Formulation* in Table 1 lists the formula under which the models encode the corresponding relation patterns, thus giving our desired *fine-grained* analysis of the capability. For space reasons, we describe only one cell of Table 1 to show SpacE^{ss} is capable of encoding the relation pattern symmetric under the condition (a). A relation is symmetric if $\forall h, t$, $(h, r, t) \leftrightarrow (t, r, h)$ holds. In other words, if a triple (h, r, t) is positive, (t, r, h) must also be positive. According to equation (3):

Table 1. A Fine-grained (Formulation) of KGE Models Encoding Relational Patterns. Each column corresponds to a specific model which is capable of encoding a specific relational pattern presented in each row, under a possible triple correctness condition (Cond.) and capability (Formulation).

	SpacE^{ss} (φ_{mp})	**RotatE** (φ_{mp})	**TransComplEx** (φ_{mp})
Ref.	$\|\mathbf{h}_i(e^{i\theta^r_{Ri}} + e^{-i\theta^r_{Ri}}) - \mathbf{r}_i\| = \lambda$	$\|\mathbf{h}_i e^{i\theta^r_i} - \mathbf{h}_i\| = \lambda$	$\| - 2\mathbf{h}_i - \mathbf{r}\| = \lambda$
Sym.	$(e^{i\theta^r_{Li}} - e^{-i\theta^r_{Ri}}) = 0$	$r_i^2 = 1$	$Re(ri) = 0$ $Re(hi) = Re(ti)$ $Im(\mathbf{h}) + Im(\mathbf{r}) = Im(\mathbf{t})$
Inv.	$\mathbf{h}(e^{i\theta^{r_1}_{Li}} + e^{-i\theta^{r_2}_{Ri}}) + r_{1i} - r_{2i} = (e^{i\theta^{r_1}_{Ri}} - e^{-i\theta^{r_2}_{Li}})$	$r_{1i}r_{2i} = 1$	$2(Im(h_i) + Im(t_i)) + Im(r_{1i}) + Im(r_{2i}) = 0$ $Re(r)^1 + Re(r)^2 = 0$

$$\begin{cases} \mathbf{h}_{\theta^r_{Li}} + \mathbf{r}_i = \mathbf{t}_{\theta^r_{Ri}} \\ \mathbf{t}_{\theta^r_{Li}} + \mathbf{r}_i = \mathbf{h}_{\theta^r_{Ri}} \end{cases}, \tag{5}$$

after a set of derivations, the resulting condition turns to be $e^{i\theta^r_{L_i}} + e^{-i\theta^r_{R_i}} = 0$ (see Table 1 for symmetric in SpacE^{ss}). It follows that a relation r is a symmetric relation if the following equations hold for it:

$$\begin{aligned} \cos(\theta^r_{L_i}) = -\cos(\theta^r_{R_i}) = \cos(\pi - \theta^r_{R_i}), \\ \sin(\theta^r_{L_i}) = \sin(\theta^r_{R_i}) = \sin(\pi - \theta^r_{R_i}). \end{aligned} \tag{6}$$

Therefore, in order to encode symmetric relation patterns by SpacE^{ss}, the summation of the rotations on the left and right sides ($\theta^r_{Li} + \theta^r_{Ri}$) should be equal to π. There are infinitely many solutions for (6) (e.g. $\{\theta^r_{L_i} = 0, \theta^r_{R_i} = \pi\}, \{\theta^r_{L_i} = \pi/3, \theta^r_{R_i} = 2\pi/3\}, \ldots$). We use the same reasoning to obtain results for each of the models with regard to an underlying relation pattern.

Company Example for Symmetry in SpacE^{ss}. Extending SS in *SpacE^{ss}* enables it to correctly encode all the patterns existed in our company example. Due to a bigger SS, triples are properly encoded in a vector space. This enables SpacE^{ss} to return correct inferences opposite to RotatE. Figure 3 shows how SpacE^{ss} encodes the example from Fig. 1. A left and a right rotation is assigned for each relation, e.g. one out of infinite solutions could be $\theta_{L_{br}} = 45$ and $\theta_{R_{br}} = -135°$. In Sub-figure a and b, the encoding of the positive triples from the symmetric relations are shown, and Sub-figure c represents the correct encoding of non-symmetric relations. In Sub-figure d, we show that SpacE^{ss} does not infer any expected incorrect triple (wrong inferences) from non-symmetric relations r_c, r_o. More precisely, the r_c relation in Sub-figure d is forced by the model formulation to have a different direction than the relation r_c in Sub-figure c. The same applies for r_o. As it contradicts with the actual triple in Sub-figure c, the model concludes that the two triples in Sub-figure d do not hold in the vector space as correct inferences.

The correct embeddings of r_c, r_o are shown in Sub-figure c. Using these vectors, we conclude that $Microsoft + r_c \neq eBay$ and $Looker + r_o \neq Google$.

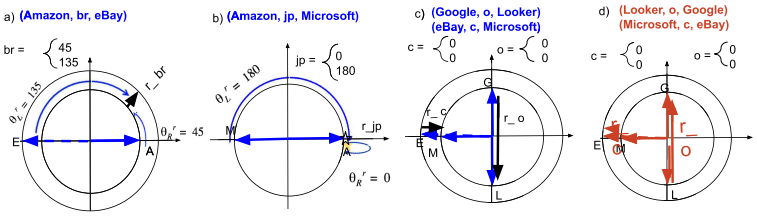


Fig. 3. Company Example in SpacE^{ss}. Encoding of the positive triples as correct inferences (blue vectors for entities and black for relations) and failure in returning wrong inferences (red vectors). (Color figure online)

Sub-figure d shows that the embeddings of r_c, r_o relations are different from their actual embeddings shown in Sub-figure c. Therefore, the model refuses the wrong inference of these symmetric relations which was not the case in RotatE.

Medium-Grained Analysis. So far, we have discussed a very coarse-grained capability, namely "yes"/"no", and a very fine-grained capability, namely the defining formula of the SS (see Table 1). It remains to show a medium-grained analysis that would allow us to easily compare different models. In Table 2, different variations of feasible solution space for three KGE models are shown which exactly gives us this medium-grained understanding. In Table 4 we visualize how theoretic FSS assumption hold in practice.

Table 2. Comparison of Solution SpacE^{ss} for Different KGE Models. An element-wise comparison of different values for SS are shown.

	SpacE^{ss}			**Rotate**			**TransComplEx**		
Ref.	$\mathbf{S}_{r\|h}$	$\mathbf{S}_{h\|r}$	-	$\mathbf{S}_{r\|h}$	$\mathbf{S}_{h\|r}$	-	$\mathbf{S}_{r\|h}$	$\mathbf{S}_{h\|r}$	-
	Inf	Inf	-	2	Inf	-	Inf	4	-
Sym.	$\mathbf{S}_{r\|h,t}$	$\mathbf{S}_{h\|r,t}$	$\mathbf{S}_{h,r\|t}$	$\mathbf{S}_{r\|h,t}$	$\mathbf{S}_{h\|r,t}$	$\mathbf{S}_{h,r\|t}$	$\mathbf{S}_{r\|h,t}$	$\mathbf{S}_{h\|r,t}$	$\mathbf{S}_{h,r\|t}$
	Inf	1	Inf	2	1	2	1	1	Inf.
Inv.	$\mathbf{S}_{r_1,r_2\|h,t}$	$\mathbf{S}_{h\|r_1,r_2,t}$	$\mathbf{S}_{h,r_1\|r_2,t}$	$\mathbf{S}_{r_1,r_2\|h,t}$	$\mathbf{S}_{h\|r_1,r_2,t}$	$\mathbf{S}_{h,r_1\|r_2,t}$	$\mathbf{S}_{r_1,r_2\|h,t}$	$\mathbf{S}_{h\|r_1,r_2,t}$	$\mathbf{S}_{h,r_1\|r_2,t}$
	Inf	1	Inf	4	1	1	1	1	1

Note that solution space allows multiple dimensions, e.g. fixing head and having freedom in relation and tail, or any of the other combinations. For each relational pattern, the solution of each possible combination of vectors is provided. For example, a combination of three possible solutions $\mathbf{S}_{r|h,t}$, $\mathbf{S}_{h|r,t}$ and $\mathbf{S}_{h,r|t}$ is considered for symmetric relations. Thus, in case $S_{r|h,t} = Inf$ if for SpacE^{ss}; the solution space for RotatE, however is 2 i.e., $S_{r|h,t} = 2$. In the same way, SpacE^{ss} provides a bigger solution space for other relation patterns compared to TransComplEx and RotatE.

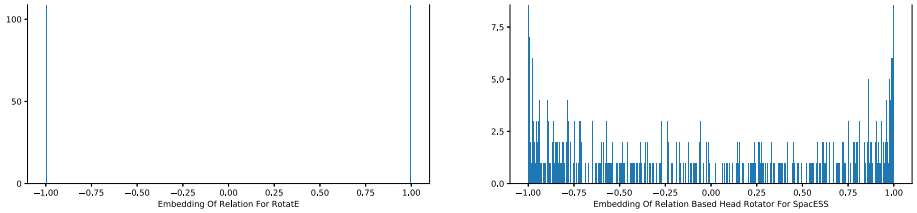


Fig. 4. Solution Space. The possible solutions for symmetric relation of "Similar To" in WN18 by RotatE (2) and SpacE^{ss} (Inf.) – compatible results in Table 2.

5 Experiments

The proposed model is evaluated on the link prediction problem using the filtered setting [3]. The task here is to predict whether a relation is likely to hold between two given entities. Here, we first generate a set of candidate triples by corrupting once the head h and once the tail entity t for each positive test triple (h, r, t). We remove any candidate triples constructed in this way if they appear in either validation, training, or the test set. Finally, we rank the remaining candidate triples against the original test triple (h, r, t). We use the standard evaluation methods: mean rank (MR), mean reciprocal rank (MRR) and hits at top N (Hits@N) for N = 1, 3, and 10 [24]. MR is the average rank of all the correct test triples; MRR is the average reciprocal rank of the correct triples and is defined as: $\sum_{j=1}^{n_t} \frac{1}{r_j}$, where r_j is the rank of the j-th (positive) test triple and n_t - the

Table 3. Evaluation 1. Results of models evaluated on FB15k and WN18. SpacE^{ss}+Pat is SpacE^{ss} model with pattern *Pat* explicitly injected. SpacE^{ss}-small is SpacE^{ss} model with 10 negative samples per one positive and dimension of 200. Results for models marked by ⋆ are reported from [19]; results for TuckER are from [1]; and results from TransComplEx are quoted from [16].

	FB15k					WN18				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE⋆	–	.463	.297	.578	.749	–	.495	.113	.888	.943
DistMult⋆	42	**.798**	–	–	.893	655	.797	–	–	.946
HolE⋆	–	.524	.402	.613	.739	–	.938	.930	.945	.949
ComplEx⋆	–	.692	.599	.759	.840	–	.941	.936	.945	.947
ConvE⋆	51	.657	.588	.723	.831	374	.943	.935	.946	.956
RotatE⋆	40	.797	**.746**	.830	.884	309	.949	.944	.952	.959
TuckER	–	.795	.741	.833	.892	–	**.953**	**.949**	**.955**	.958
TransComplEx	38	.682	–	–	.875	284	.922	–	–	.955
SpacE^{ss}	34.5	.760	.667	.836	.895	197	.946	.936	.953	**.962**
SpacE^{ss}-small	41	.732	.630	.815	.884	228	.946	.936	.953	**.962**
SpacE^{ss}+Inverse	35.3	.774	.686	.845	**.898**	**141**	.939	.921	.953	**.962**
SpacE^{ss}+Implication	35	.765	.673	.839	.896	–	–	–	–	–
SpacE^{ss}+Symmetry	36	.768	.680	.838	.894	–	–	–	–	–
SpacE^{ss}+Equality	36	.773	.687	.845	.896	–	–	–	–	–
SpacE^{ss}+All	**33.7**	.789	.713	**.851**	**.898**	–	–	–	–	–

Table 4. Evaluation 2. Results of models evaluated on FB15k-237 and WN18RR. SpacE^{ss}+Pat is SpacE^{ss} model with pattern *Pat* explicitly injected. SpacE^{ss}-small is SpacE^{ss} model with 10 negative samples per one positive and dimension of 200. Results for models marked by ⋆ are reported from [19]; results for TuckER are from [1]; and results from TransComplEx are quoted from [16].

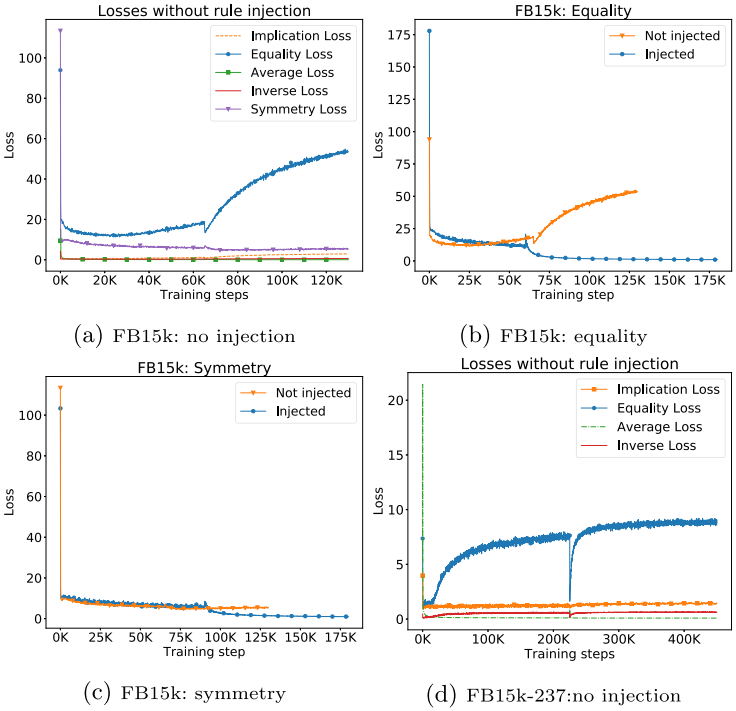
	FB15k-237					WN18RR				
	MR	MRR	HIT@1	HIT@3	HIT@10	MR	MRR	HIT@1	HIT@3	HIT@10
TransE⋆	357	.294	–	–	.465	3384	.226	–	–	.501
DistMult⋆	254	.241	.155	.263	.419	5110	.43	.39	.44	.49
ComplEx⋆	339	.247	.158	.275	.428	5261	.44	41	.46	.51
ConvE⋆	244	.325	.237	.356	.501	4187	.43	.40	.44	.52
RotatE⋆	177	.338	.241	.375	.533	3340	**.476**	.428	.492	.571
TuckER	–	**.358**	**.266**	**.394**	**.544**	–	.470	**.443**	.482	.526
TransComplEx	223	.317	–	–	.493	4081	.389	–	–	.498
SpacE^{ss}	167	.337	.238	.376	.539	**2959**	.457	.392	.488	**.583**
SpacE^{ss}-small	171	.333	.236	.369	.530	2986	.469	.412	**.493**	.577
SpacE^{ss}+Inverse	167	.340	.243	.375	.537	–	–	–	–	–
SpacE^{ss}+Implication	168	.338	.240	.374	.540	–	–	–	–	–
SpacE^{ss}+Equality	167	.337	.240	.376	.538	–	–	–	–	–
SpacE^{ss}+All	**163**	.335	.237	.372	.533	–	–	–	–	–

Table 5. Evaluation 3. Results with low Dimension equal to 10.

	FB15k					WN18				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE⋆	3350	.023	.009	.022	.048	1355	.128	.066	.138	.246
DistMult⋆	3459	.021	.010	.019	.041	753	.403	.247	.475	.742
ComplEx⋆	876	.122	**.069**	.134	.220	**709**	.453	.304	.524	.769
RotatE⋆	748.7	.050	.018	.045	.105	891	.522	.403	.591	.748
TransComplEx	511	.115	.053	.132	.239	1700	.117	.065	.123	.221
SpacE^{ss}	**393**	**.130**	**.069**	**.136**	**.246**	805	**.569**	**.451**	**.647**	**.785**
	FB15k-237					WN18RR				
	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10
TransE⋆	431	.159	.092	.170	.292	6207	.093	.009	.141	.249
DistMult⋆	432	.159	.089	.170	.302	8749	.100	.001	.180	.352
ComplEx⋆	**388**	.165	.094	.173	.314	8977	.203	.118	.266	.358
RotatE⋆	489	.132	.067	.139	.266	6358	.323	.245	.391	.430
TransComplEx	453	.154	.088	.163	.292	6645	.177	.119	.202	.291
SpacE^{ss}	390	**.183**	**.109**	**.197**	**.334**	**6115**	**.393**	**.368**	**.406**	**.437**

number of triples in the test set; and Hits@N is the percentage of the triples whose rank is equal or smaller than N.

Datasets. We evaluate our model on set of four widely used knowledge graphs: FB15k [3] FB15k-237 [20], WN18 [3], and WN18RR [4]. The relation patterns (rules) for FB15k and WN18 are adapted from [8] and [7] respectively. We only considered rules with confidence level of 0.8 or higher. The number of triples involved in symmetric, implication and inverse patterns both in FB15k and FB15k-237 are above $35k$ in each pattern category. The number of triples with inverse patterns in FB15k is above $390k$. The above statistics are based on triple



(a) FB15k: no injection (b) FB15k: equality

(c) FB15k: symmetry (d) FB15k-237:no injection

Fig. 5. Evaluation 4. Evolution of losses for relation patterns during optimization phase with/without injection: (a) shows the convergence of losses for main relation patterns in FB15K without injection. In (b), the equality patterns are shown with/without injection; (c) shows the same for symmetric patterns and (d) shows the convergence of losses for all patterns on FB15k-237 without injection.

counting, however, we follow [8] when constructing valid groundings: a grounding of a rule is considered valid if the triples appearing in its antecedent are present in the knowledge graph, while the triples in the consequent are not. Given an example pattern $p \Rightarrow q$ (e.g. inverse is $\forall X, Y : r(X, Y) \implies r'(Y, X)$), we refer to p as the antecedent and q as the consequent of the rule. Most of the groundings in both FB15k and WN18 are for inverse rule.

Experimental Setup. We train our model by using the RotatE loss [19]. We use Adagrad [5] as the optimizer of SpacE^{ss}. The hyper parameters were fine-tuned on the validation set, using grid search over the following ranges: batch size $\in$ {200, 512, 1024}, embedding dimension dim $\in$ {200, 500}, number of negative samples #neg $\in$ { 10, 20, 50 }, adversarial sampling temperature $\alpha \in$ {0.5, 1.0}, $\gamma \in$ {3, 9, 12, 18, 24 }, learning rate $lr \in$ {0.001, 0.01, 0.1 }. γ is the hyper-parameter of the losses. Embedding dimension 500 and 20 negative samples are used as best hyper-parameters for our model.

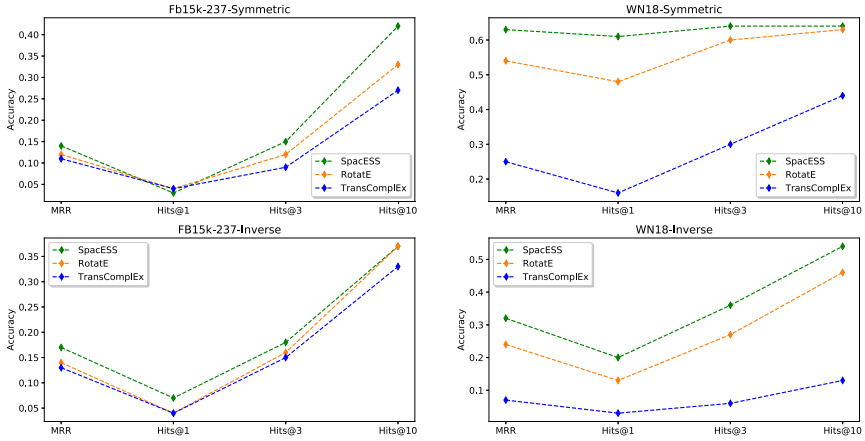


Fig. 6. Symmetric and Inverse Patterns. Practical investigation on the theories of Table 1 in low dimension 10 for FB15K-237 and 15 for WN18.

Results and Discussion. A comparison of SpacE^{ss} to TransE, DistMult, HolE, CompEx, ConvE, and RotatE is provided in Tables 3 and 4. We follow the steps introduced in [7,15] for explicit pattern injecting such that: first, the loss function for the relational patterns is computed and then the results are added to the main loss function as a regularization term. For example, in order to inject a symmetric pattern, we add the regularization term $\|f_r(h,t) - f_r(t,h)\|$. We use following definitions to compute regularization terms for the other rules: we say that $A \iff B$ holds iff $A = B$ and $A \implies B$ iff $A \leq B$; (soft) truth value (or the score) of $A \wedge B$ is computed as $A \times B$, $A \vee B$ as $A + B - A \times B$, and $\neg A$ as $1 - A$. The loss terms for $A \leq B$ and $A = B$ are computed as $Relu(A - B)$ and $\|A - B\|$, respectively. We test our model both with and without explicitly injected relational patterns. We use SpacE^{ss}+Pat ("Pat" for a specific pattern that is injected) to denote that the model uses pattern regularization (one type of rule is injected at a time). For instance, SpacE^{ss}+Inverse refers to SpacE^{ss} with inverse relations injected explicitly via regularization terms. SpacE^{ss}+All refers to the results of injecting multiple (all) patterns.

According to [19], FB15K and WN18 datasets contain a significant amount of relational patterns (inverse and symmetric). Table 2 (which is derived from 1) shows the design and development of the SpacE^{ss} model have been done with the purpose of having a bigger solution space than RotatE and TransComplEx. Therefore, in two evaluation settings (with bigger and smaller dimension), the SpacE^{ss} model achieves: a) a better performance than the other distance-based models encoding relational patterns (with same dimension), b) same accuracy (in smaller dimension) due to expanded solution space for encoding pattern e.g. inverse relations in our experiments on FB15K and WN18.

As shown in Table 3, SpacE^{ss} obtains 89.8 Hits@10 while RotatE and TransComplEx get 88.4 and 87.5 respectively. These results confirm our theoretical

discussions when SpacE^{ss} outperforms RotatE and TransComplEx on FB15K. We additionally show that even with a smaller dimension (200), the performance of SpacE^{ss} closely compete with the results of the RotatE model on FB15K and WN18 datasets. This is an additional confirmation for our expectation that by having a bigger solution space, the model (SpacE^{ss}) is enabled to: a) encode relational patterns, b) have correct inferences, c) stay in high (and obtain better) performance even with a smaller dimension. Comparing SpacE^{ss} with SpacE^{ss}+Pat, (SpacE^{ss} with injected patterns), we conclude that our model is capable of inferring the relational patterns even without explicit injection. However, we also denote that pattern injection did not have a high impact on the results of the model in this setting. This is additionally justified by tracing the convergence of loss for relation patterns shown in Fig. 5. The losses of patterns (except equality loss) converge properly by learning on data (Sub-figure 5a). Although FB15k and WN18 have testing leakage, it is still worth using these two datasets while investigating capacity of models. These datasets contain many inverse and symmetric patterns, therefore if the solution space of a model is limited, then the model cannot express those patterns and the accuracy is expected to be dropped substantially even with testing leakage.

As shown in Sub-figure 5b, injection enables the quality loss to converge. Using symmetry relation (Sub-figure 5c), we show even without injection, the model properly infers the pattern. Although with these two datasets (FB15K and WN18), our focus has been on showcasing relation patterns of type inverse and symmetric, encoding of relational patterns by SpacE^{ss} is not limited to these. This is proven by running SpacE^{ss} on FB15K-237 and WN18-RR datasets within which the inverse relational patterns have mostly been removed originally. The results are shown in Table 4 and Table 3 where our performance is closely competing with RotatE and TransComplEx. However, in comparison to TransComplEx with performance of 49.3, our model achieves a better performance of 53.9 in Hits@10. Tucker gets the state-of-the-art performance on FB15K-237. However, Tucker obtains these results by using much more parameters due to the design of its scoring function. Moreover, this performance is also due to the used boosting techniques such as data augmentation (adding reverse triples), and using 1−n scoring which is not applicable in large scale KGs. Using WN18RR, SpacE^{ss} outperforms all models considering Hits@10 and MR, even with a smaller dimension. Table 5 includes the results for dimension 10.

As said, the solution space of a model is heavily depending on the model formulation. However, one can increase it with the cost of getting high in the size of the embedding dimension. Since the evaluation in the state-of-the-art models have been done on relatively small standard KGs with big dimensions e.g. 500, 1000, the difference between models is not visible. Normally there would be two ways of highlighting the effect of solution space: 1) compare the embedding models with regard to their solution space in a very large scale of KGs e.g. millions of entities and billions of triples, 2) prototype it with toy KGs e.g. FB15k-237 in a very low dimension e.g. 5 or 10. Since the first approach is not feasible technically, we provided the results following the second way. Our extended evaluations

using very low dimensions show that for FB15k-237, RotatE has 26.6 Hits@10, however SpacE^{ss} gets 33.4 with dimension 10. With dimension 5, the difference in performance is even more visible up to 15% (RotatE shows 3% and SpacE^{ss} 19% in Hits@10) of differences (complete results have been omitted from this paper for lack of space). Figure 6 shows the results for low dimensional of 15 over WN18 and 10 over FB15K-237. For symmetric patterns of WN18 in Hits@1, we gain 13% difference over RotatE (SpacE^{ss} = 61.01, RotatE = 48.3).

6 Conclusion and Future Work

In this paper, we introduced the concept of solution space as an approach towards overcoming the expressiveness problem of embedding models. It provides a fine-grained analysis on the capability of the models to express certain patterns. We introduced the SpacE^{ss} model that is designed based on the concept of solution space. We specifically provided a theoretical demonstration of the solution space for SpacE^{ss}, RotatE and TransComplEx models on reflexive, symmetric and inversion patterns. An experimental evaluation is provided that shows the performance of SpacE^{ss} in comparison to the state-of-the-art models which are able to encode a relational pattern. The experiments are done both in high and low dimensions in order to simulate their utilization over large-scale KGs. The results of the comparisons on high dimension show the performance improvements of SpacE^{ss} influenced by the concept of solution space. This is further visible in low dimensional embedding where the experiments show a surprising drop in the performance of all the considered models including SpacE^{ss}, even on FB15K which has leakage on patterns. However, the difference of performance demonstrated by SpacE^{ss} is yet another approval on the importance of solution space. In this work, we only investigated a few of the well-known embedding models. Our future work contains analysing more models in terms of their solution space and broaden our scope to find more factors that influence the expressiveness of models. We showcased the effect of solution space considering some of the relational patterns. We plan to extensively include other pattern types.

Acknowledgements. This work is supported by the EC Horizon 2020 grant LAMBDA (GA no. 809965), the CLEOPATRA project (GA no. 812997), the Vienna Science and Technology Fund (WWTF) grant VRG18-013, and the EPSRC grant EP/M025268/1.

References

1. Balažević, I., Allen, C., Hospedales, T.M.: TuckER: tensor factorization for knowledge graph completion. arXiv preprint arXiv:1901.09590 (2019)
2. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: ACM SIGMOD, pp. 1247–1250. ACM (2008)
3. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPSm pp. 2787–2795 (2013)

4. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: AAAI (2018)
5. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. JMLR **12**(7), 2121–2159 (2011)
6. Ebisu, T., Ichise, R.: TorusE: knowledge graph embedding on a lie group. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
7. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Jointly embedding knowledge graphs and logical rules. In: EMNLP, pp. 192–202 (2016)
8. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Knowledge graph embedding with iterative guidance from soft rules. In: AAAI (2018)
9. Ji, G., He, S., Xu, L., Liu, K., Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In: ACL-IJCNLP, pp. 687–696 (2015)
10. Kazemi, S.M., Poole, D.: Simple embedding for link prediction in knowledge graphs. In: NIPS, pp. 4284–4295 (2018)
11. Lehmann, J., et al.: DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. Seman. Web J. **6**(2), 167–195 (2015). outstanding Paper Award (Best 2014 SWJ Paper)
12. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI (2015)
13. Liu, H., Wu, Y., Yang, Y.: Analogical inference for multi-relational embeddings. In: ICML, pp. 2168–2178 (2017)
14. Miller, G.A.: Wordnet: a lexical database for english. CACM **38**(11), 39–41 (1995)
15. Minervini, P., Costabello, L., Muñoz, E., Nováček, V., Vandenbussche, P.-Y.: Regularizing knowledge graph embeddings via equivalence and inversion axioms. In: Ceci, M., Hollmén, J., Todorovski, L., Vens, C., Džeroski, S. (eds.) ECML PKDD 2017. LNCS (LNAI), vol. 10534, pp. 668–683. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71249-9_40
16. Nayyeri, M., Xu, C., Yaghoobzadeh, Y., Yazdi, H.S., Lehmann, J.: On the knowledge graph completion using translation based embedding: the loss is as important as the score. arXiv Preprint arXiv:1909.00519 (2019)
17. Nickel, M., Rosasco, L., Poggio, T.: Holographic embeddings of knowledge graphs. In: AAAI (2016)
18. Nie, B., Sun, S.: Knowledge graph embedding via reasoning over entities, relations, and text. Future Gener. Comput. Syst. **91**, 426–433 (2019)
19. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)
20. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, pp. 57–66 (2015)
21. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML, pp. 2071–2080 (2016)
22. Tucker, L.R.: Some mathematical notes on three-mode factor analysis. Psychometrika **31**(3), 279–311 (1966)
23. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. CACM **57**(10), 78–85 (2014)
24. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: a survey of approaches and applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017)
25. Wang, Y., Gemulla, R., Li, H.: On multi-relational link prediction with bilinear models. In: AAAI (2018)

26. Wang, Y., Ruffinelli, D., Broscheit, S., Gemulla, R.: On evaluating embedding models for knowledge base completion. arXiv preprint arXiv:1810.07180 (2018)
27. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI (2014)
28. Yang, B., Yih, W.T., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. arXiv preprint arXiv:1412.6575 (2014)
29. Yoon, H.G., Song, H.J., Park, S.B., Park, S.Y.: A translation-based knowledge graph embedding preserving logical property of relations. In: NAACL - HLT (2016)