EmEL^{++}: Embeddings for $\mathcal{EL}^{++}$ Description Logic

Sutapa Mondal[a,c], Sumit Bhatia[b] and Raghava Mutharaju[a]

[a]*Knowledgeable Computing and Reasoning (KRaCR) Lab, IIIT-Delhi, India*
[b]*IBM Research AI, New Delhi, India*
[c]*TCS Research & Innovation, India*

Abstract

Knowledge graph (KG) embedding models have recently gained increased attention. However, most of the existing models for KG embeddings ignore the structure and characteristics of the underlying ontology. In this work, we present EmEL^{++} embeddings – an ontology-based embedding model for the $\mathcal{EL}^{++}$ description logic. EmEL^{++} maps the classes and the relations in an ontology to an n-dimensional vector space such that the relations between classes and relations in the ontology are preserved in the vector space. We evaluate the proposed embeddings on four different datasets and show that the proposed embeddings outperform the traditional knowledge graph embeddings on the subsumption reasoning task.

Keywords

Ontology, $\mathcal{EL}^{++}$, Description Logic, Geometric Embeddings

1. Introduction

Methods for learning embedding functions that map the underlying entities to a vector space have gained significant attention in recent times. Different methods for learning embedding functions try to preserve the critical properties of, and relations between, the underlying entities (such as words, concepts, documents, nodes and edges in a graph) in the n-dimensional vector space. A variety of embeddings for Knowledge Graphs [1, 2, 3, 4, 5, 6] have been proposed. These methods differ in terms of the underlying properties of the knowledge bases preserved in the vector space. For example, the TransE [1] model considers the relations in a KG as a translation operator over the entities, TransH [2] models the relations as a hyperplane in the vector space to allow for reflexive, one-to-many, many-to-one, and many-to-many relations, DistMult [3] uses Matrix Factorization based approach to bring similar entities and relations together in the vector space. However, most of the KG embedding models focus on capturing the structural properties of the graph and the interaction between the entities and *do not take into account the constraints and characteristics of the underlying ontology.* Consequently,

In A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2021 Spring Symposium on Combining Machine Learning and Knowledge Engineering (AAAI-MAKE 2021) - Stanford University, Palo Alto, California, USA, March 22-24, 2021.

sutapa.mondal@tcs.com (S. Mondal); sumitbhatia@in.ibm.com (S. Bhatia); raghava.mutharaju@iiitd.ac.in (R. Mutharaju)
0000-0003-0870-1015 (S. Mondal); 0000-0002-8146-4100 (S. Bhatia); 0000-0003-2421-3935 (R. Mutharaju)

CEUR Workshop Proceedings (CEUR-WS.org)

the embeddings produced by such methods are not suited for reasoning tasks such as classification, satisfiability and consistency checking. Kulmanov et al. [7] have recently proposed EL Embeddings (ElEm) and have described a geometric interpretation of embedding $\mathcal{EL}^{++}$ ontologies in an n-dimensional vector space (Section 3). However, the ElEm embeddings are limited in terms of the coverage of $\mathcal{EL}^{++}$ constructs as they ignore the role oriented constructs in such ontologies. Further, the use case considered by them is predicting protein-protein interactions that is modeled as a traditional link prediction task in knowledge bases. In this work, we address the limitations of ElEm embeddings and propose EmEL++ embeddings. We build upon the framework introduced by Kulmanov et al. [7] and extend it to offer more complete coverage of the $\mathcal{EL}^{++}$ semantics (Section 3.1) for performing subsumption reasoning task. Baader et al. [8] have shown that all the standard reasoning tasks in $\mathcal{EL}^{++}$ ontologies can be reduced to subsumption task. Thus, to the best of our knowledge, we present the first attempts at performing reasoning tasks by embedding ontologies in a vector space. We compare our proposed approach using four ontologies of different sizes and characteristics. Our evaluation shows that EmEL^{++} outperform the traditional KG embeddings at the reasoning task and are also able to preserve the characteristics of underlying ontologies better. We would also like to emphasize that performing reasoning in the vector space is critical as it has the potential to speed up the reasoning process significantly. As we describe in Section 4, the subsumption task in vector space involves computing distances between the source class and all the other classes in the ontology. In the worst case, this is an $O(n)$ operation where n is the number of classes. Thus, irrespective of the complexity of the underlying ontology, the subsumption task could be performed in $O(n)$ time. Further, with uses of techniques such as semantic hashing or binarized embeddings [9], the similarity based search operations can be performed in $O(1)$ time. Therefore, we believe that embedding based approaches, despite their lower accuracies than standard reasoners and no theoretical guarantees of performance, offer a promising direction of future research to develop more efficient reasoners, especially for more complex description logics such as $\mathcal{SROIQ}$ (OWL 2 DL).

2. Related Work

A wide range of methods for computing KG embeddings have been proposed. Node2Vec [10] initiated the idea of learning features for networks addressing the scalability challenge. Although their results are decent for link prediction task but they make assumptions on conditional independence of the feature space which may not hold true in real world scenarios. Eventually, this concept got popularized with KGs wherein, a fact is represented as a triple of the form (h,r,t). Semantic matching KG models exploit similarity based scoring functions and match latent semantics of entities and relations based on vector representations. For example, [4] proposed RESCAL which uses multiple matrices to represent relations among entities but scalability remains an issue. [3] proposed DistMult to overcome challenges of RESCAL. Although, DistMult is similar to RESCAL but DistMult ensures low number of parameters for relations by restricting the matrices. Later, translational based models for KG embeddings used distance based scoring functions. This technique gained most attention due to its simplicity to measure the correctness of a fact. It measures the plausibility of a fact as distance between the entities after translation carried out by relation. TransE[1], TransH [2] and TransR[6] being

different variants of the same. TransE being the most representative model, treats the relations as translations such that given a fact, relation vector r minimizes the distance between h and t in vector space. Whereas, TransH interprets relations as translating operator on a hyperplane and TransR models entities and relations in two distinct spaces, performing translation on corresponding relation space. [11] worked on a novel approach inspired by the theory of quantum logic to embed a Knowledge Base (KB) in $\mathcal{ALC}$ description logic. Existing works on ontology embedding such as Onto2vec [12] focused on using word2vec as an underlying model. Most of this work focuses on encoding the entities and relations, but they lack in handling the complex relations in an ontology. Recently, [13] pointed out the lack of expressivity in the classical approaches to model relations for KG embeddings. Moreover, their work indicates that geometric models are a better way to learn embeddings for ontologies. Further, [14] present a new approach using deep learning with knowledge based systems to emulate reasoning structure. Although, they perform experiments on a synthetic and a non-synthetic dataset, but in our work we look at multiple datasets with different characteristics and reason about their varying performances. However, [7] tries to overcome the drawbacks of KG embeddings but it does not address all the $\mathcal{EL}^{++}$ constructs that are relevant to capture the relations present in an ontology. Further, the evaluation is focused only on link prediction task.

3. Background

3.1. Embedding $\mathcal{EL}^{++}$ Ontologies in a Vector Space

A description logic signature is a tuple $\langle N_C, N_R, N_I \rangle$, where N_C, N_R, and N_I are countably infinite, mutually disjoint sets of concept names, role names, and individual names respectively. In the following discussion, $\{C, C_1, C_2, D, E, \top, \bot\} \in N_C$, $\{R, S, R_1, R_2\} \in N_R$, and $\{a, b\} \in N_I$.

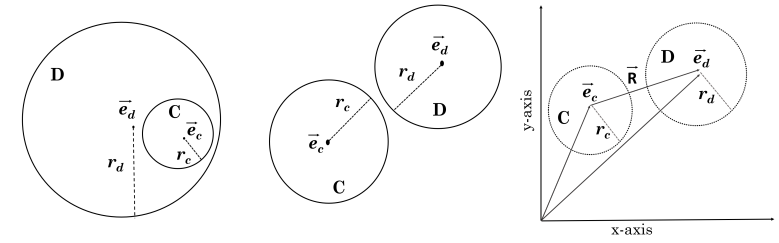
All the axioms in the $\mathcal{EL}^{++}$ description logic can be reduced to one of the normal forms [15] as follows:

1. All the TBox axioms are in one of the four normal forms ($C_1 \sqsubseteq D$, $C_1 \sqcap C_2 \sqsubseteq D$, $\exists R.C_1 \sqsubseteq D$, and $C_1 \sqsubseteq \exists R.C_2$).
2. The bottom concept can only appear on the right side of the equations, and can only appear in the first three normal forms.
3. All role inclusions are of the form $R \sqsubseteq S$ or $R_1 \circ R_2 \sqsubseteq S$.

Further, the instantiation and role assertion axioms in the ABox can be converted into TBox axioms as follows:

$$C(a) \longrightarrow \{a\} \sqsubseteq C$$
$$R(a, b) \longrightarrow \{a\} \sqsubseteq \exists R.\{b\}$$

Thus, with the above transformations, all the axioms in an $\mathcal{EL}^{++}$ ontology can be reduced to one of the normalized forms and the task of embedding ontologies in a vector space requires us to learn mapping functions for classes and relations that are part of the normal forms.



(a) Subclass Representation (b) Disjoint Representation (c) Relation Representation

Figure 1: Geometric Representation of classes and relation

3.1.1. Intuition.

Typically, for mapping the entities of interest to a vector space, requires to learn a mapping function subject to certain constraints, encoded in the form of an objective function that is optimized during the training phase. These objective functions are designed such that specific properties of the underlying entities are also retained in the vector space. For example, the word2vec [16] model for word embeddings minimizes the distance between contextually similar words, RDF2Vec [17] adapts language modeling approach to capture local information from the graph sub-structures, and TransE [1] model for KGs. Similarly, in this work we learn mapping functions that can embed $\mathcal{EL}^{++}$ ontologies in a vector space. In order to do so, we build upon and extend the framework proposed by Kulmanov et al. [7] that interprets a class in the ontology as an n-ball (defined by its radius and center) in the vector space. Let us consider two classes C and D such that $C \sqsubseteq D$. Let these two classes be represented by their respective n-balls b_c and b_d in the vector space such that $b_c : \{\vec{c}, r_c\}$ and $b_d : \{\vec{d}, r_d\}$; where $\vec{c}$ and $\vec{d}$ are the centers and r_c and r_d are the radii of the respective n-balls. Geometrically, if $C \sqsubseteq D$, the mapping function should aim to ensure that the b_c lies inside b_d (Figure1 (a)). Similarly, if C and D are disjoint, the respective n-balls should not overlap with each other in the vector space (Figure1 (b)). Further, similar to the TransE model [1], the relations in the ontology are interpreted as translations operating on the classes. More specifically, if $C \sqsubseteq \exists R.D$, the center of n-ball representing C can be moved to the center of the n-ball representing D (Figure1 (c)).

3.2. Loss Functions

With the intuitive framework described above, let us now describe the objective functions that should be optimized during the training phase to learn the mapping functions. Let $e_v : \mathbf{C} \cup \mathbf{R} \longmapsto R^n$ be the mapping function that maps each class and relation to a unique vector in the n-dimensional embedding space. For $C_i \in \mathbf{C}$, the resulting vector corresponds to the centre of the n-ball representing the class. Further, let $e_r : C \longmapsto R^+$ be the mapping function that maps each class into a non-negative real number, that represents the *radius* of the n-ball corresponding to class C. Thus, the pair (e_v, e_r) of functions represents the operations needed to *embed* an $\mathcal{EL}^{++}$ ontology into an n-dimensional space. We now describe the various loss functions to represent the different constructs in $\mathcal{EL}^{++}$. The total loss that needs to be minimized during the learning process is the sum of the individual loss functions.

3.2.1. Loss Functions for the Four Normal Forms:

As described before, the first normal form ($C \sqsubseteq D$) when embedded in a vector space can be interpreted geometrically as two n-balls, such that the n-ball corresponding to class C lies inside the n-ball corresponding to D. Hence, our mapping functions e_v and e_r should bring the centers of the two classes closer to each other, and give the sub-class a smaller radius than the super-class. The loss function presented in Equation 1 captures this intuition and penalizes the mappings that do not adhere to these constraints. Also note that in addition to the above constraints, we also add margin loss (γ) and a normalization loss that brings the centres of n-balls of all the classes on the unity sphere.

$$\mathcal{L}_{C\sqsubseteq D}(c,d) = \max\left(0, \left(\underbrace{\left\|e_v(c) - e_v(d)\right\|}_{\substack{\text{penalize if the two}\\ \text{centers are far away}}} + \underbrace{e_r(c) - e_r(d)}_{\substack{\text{penalize if sub-class}\\ \text{has larger radius}}} - \gamma\right)\right) + \left|\left\|e_v(c)\right\| - 1\right| + \left|\left\|e_v(d)\right\| - 1\right| \quad (1)$$

In the vector space, the second normal form, i.e., $C \sqcap D \sqsubseteq E$, implies that the n-ball for class E should completely engulf the area of intersection of n-balls for classes C and D. The first term in the loss function (Equation 2) imposes a penalty if the classes C and D are disjoint. The second and third terms together enforce that the center of the n-ball for class E lies in the area of intersection of n-balls for classes C and D such that it satisfies the normal form. Finally, the fourth term requires the radius of the n-ball of E to be greater than the smallest radii among n-balls of C and D.

$$\begin{aligned}\mathcal{L}_{C\sqcap D\sqsubseteq E}(c,d,e) = &\max\left(0, \left(\left\|e_v(c) - e_v(d)\right\| - e_r(c) - e_r(d) - \gamma\right)\right) + \max\left(0, \left(\left\|e_v(c) - e_v(e)\right\| - e_r(c) - \gamma\right)\right)\\ &+ \max\left(0, \left(\left\|e_v(d) - e_v(e)\right\| - e_r(d) - \gamma\right)\right) + \max\left(0, \left(\min\left(e_r(c), e_r(d)\right) - e_r(e) - \gamma\right)\right)\\ &+ \left|\left\|e_v(c) - 1\right\|\right| + \left|\left\|e_v(d) - 1\right\|\right| + \left|\left\|e_v(e) - 1\right\|\right|\end{aligned} \quad (2)$$

The first two normal forms are concerned with the mappings of classes and properties of their respective n-balls in the vector space. The next two normal forms, i.e., NF3 and NF4, involve relations and how they are associated with the classes. Recall that similar to TransE [1], we consider the relations in ontology as translations that operate on class instances. Consider the normal form $C \sqsubseteq \exists R.D$. In the vector space, C and D are represented as two n-balls b_c and b_d, respectively. If $e_v(R)$ is the vector for R in vector space, then adding $e_v(R)$ to a point in b_c should move it to a point in b_d (i.e., R translates the points in b_c to points in b_d). The following loss functions capture these semantics as expressed by the third and fourth normal forms.

$$\mathcal{L}_{C\sqsubseteq\exists R.D}(c,d,r) = \max\left(0, \left(\left\|e_v(c) + e_v(r) - e_v(d)\right\| + e_r(c) - e_r(d) - \gamma\right)\right) + \left|\left\|e_v(c)\right\| - 1\right| + \left|\left\|e_v(d)\right\| - 1\right| \quad (3)$$

$$\mathcal{L}_{\exists R.C \sqsubseteq D}(c,d,r) = \max\Big(0, \big(\|e_v(c) - e_v(r) - e_v(d)\| - e_r(c) - e_r(d) - \gamma\big)\Big) \quad + \quad \Big|\|e_v(c)\| - 1\Big| + \Big|\|e_v(d)\| - 1\Big| \tag{4}$$

3.2.2. Handling Bottom Concept ($\bot$):

Recall from the discussion in Section 3 that the bottom concept can appear only on the right hand side of the first three normal forms [8]. We now present the loss functions for each of the three special cases. The resulting first normal form $C \sqsubseteq \bot$ indicates that class C is unsatisfiable. Thus, in the vector space, we represent this constraint by reducing the radius of class C to zero. This is achieved by the following loss function.

$$\mathcal{L}_{C \sqsubseteq \bot}(c) = e_r(c) \tag{5}$$

Next, the second normal form with the bottom concept is $C \sqcap D \sqsubseteq \bot$ indicating that C and D are disjoint. In the vector space, this indicates that the n-balls of classes C and D are non-overlapping. This is captured by the following loss function.

$$\mathcal{L}_{C \sqcap D \sqsubseteq \bot}(c,d) = \max\Big(0, \big(e_r(c) + e_r(d) - \|e_v(c) - e_v(d)\| + \gamma\big)\Big) \quad + \quad \Big|\|e_v(c)\| - 1\Big| + \Big|\|e_v(d)\| - 1\Big| \tag{6}$$

Finally, the third normal form $\exists R.C \sqsubseteq \bot$ indicates that in the vector space *translating* C by R results in an unsatisfiable class. We already require the radius of unsatisfiable classes to be zero (Equation 5) and since translation does not change the radius of the original class, we have the following loss function.

$$\mathcal{L}_{\exists R.C \sqsubseteq \bot}(c,r) = e_r(c) \tag{7}$$

3.2.3. Loss Functions for Role Inclusions and Role Chains:

The role vectors in our proposed framework serve the purpose of translating one class to another class. The constraints considered until now have imposed restrictions on the role vectors based on their relations with the n-balls of the concerned classes. We now present two loss functions to capture the constraints imposed by role inclusions and role chains in the ontology. The role inclusion of $R \sqsubseteq S$ implies that the vectors $e_v(R)$ and $e_v(S)$ in the vector space should be nearby because any translation produced by R should also be producible by S plus both the vectors should be in the same direction. This intuition is captured by the following loss function represented by Equation 8. Herein, the first term is indicative of the distance that ensures the vectors $e_v(R)$ and $e_v(S)$ lie in near vicinity of each other. The second term captures the directional aspect of roles in vector space such that they tend to be in same direction.

$$\mathcal{L}_{R \sqsubseteq S}(r,s) = \max\big(0, \|e_v(s) - e_v(r)\| - \gamma\big) \quad + \quad \left|1 - \frac{e_v(r).e_v(s)}{\|e_v(r)\|\|e_v(s)\|}\right| + \quad \Big|\|e_v(r)\| - 1\Big| + \Big|\|e_v(s)\| - 1\Big| \tag{8}$$

Next, we consider the hierarchy defined by the role chain $R_1 \circ R_2 \sqsubseteq S$. In the vector space, this implies that if class C can be translated to class E by successive application of R_1 and R_2, it can

also be translated to E directly by the vector for role S while preserving the direction of role vectors. The following loss function captures this behavior represented by Equation 9.

$$\mathcal{L}_{R_1 \circ R_2 \sqsubseteq S}(r_1, r_2, s) = \max\left(0, \left\|e_v(s) - e_v(r_1) - e_v(r_2)\right\| - \gamma\right) + \left|1 - \frac{(e_v(r_1) + e_v(r_2)).e_v(s)}{\left\|(e_v(r_1) + e_v(r_2))\right\|\left\|e_v(s)\right\|}\right| + \left|\left\|e_v(r_1)\right\| - 1\right| + \left|\left\|e_v(r_2)\right\| - 1\right| + \left|\left\|e_v(s)\right\| - 1\right| \tag{9}$$

Often, negative sampling is employed during the training phase to learn better embeddings as negative samples can be easily generated to enhance the training data available. In order to incorporate negative samples in the training phase, the following loss function is employed.

$$loss_{C \not\sqsubseteq \exists R.D}(c, d, r) = \max\left(0, e_r(c) + e_r(d) - \left\|e_v(c) + e_v(r) - e_v(d)\right\| + \gamma\right) + \left|\left\|e_v(c)\right\| - 1\right| + \left|\left\|e_v(d)\right\| - 1\right| \tag{10}$$

Thus, the total loss for learning the embedding function is the sum of all the loss functions given by Equations 1- 10. Further, we also add the constraint that radius of the satisfiable classes are non-negative and penalize the total loss for learning negative radius for classes.

3.3. Training and Implementation

Given an $\mathcal{EL}^{++}$ ontology, we first normalize the ontology to generate the normal forms. These normal forms then constitute as a set of TBox statements wherein each axiom is treated as a positive sample. This normalization is performed using the OWL APIs and the APIs provided by the jCel reasoner which implements the normalization rules [18]. We then introduce negative samples using the third normal form. We randomly generate corrupted axioms following $C \sqsubseteq \exists R.D$, by replacing C or D with C' or D' such that neither $C' \sqsubseteq \exists R.D$ nor $C \sqsubseteq \exists R.D'$ are asserted axioms in the ontology. Therefore, based on the facts the training process learns ontology embedding such that the facts hold true.

The code for training of embeddings and optimization is implemented using Python and Tensorflow library, and Adam optimizer [19] is used for updating the embeddings. We start the learning process by initializing the embedding vectors for classes and relations by random values. We process the training samples in mini-batches for each of the losses defined for the normal forms along with the losses for roles, and update the embeddings depending upon the total loss i.e. the sum of all the loss functions. The update process is carried till saturation or a fixed number of epochs.

4. Experiments and Results

4.1. Datasets

We use following four commonly used and publicly available ontologies of varying size and different characteristics.

1. **SNOMED CT** [20] is one of the most comprehensive ontology of clinical terms with more than 989186 TBox statements involving 307712 classes and 60 relations.

Table 1
Different ontologies used in this work and count of different types of axioms. NFi represents the i^{th} normal form as described in Section 3.

Ontology	SNOMED	ANATOMY	GO	GALLEN
Disjoint	0	184	30	0
Role Inclusion	11	89	3	958
Role Chain	1	31	6	58
NF1	446628	122142	85480	28890
NF2	27779	2121	12131	13595
NF3	482330	152289	20324	28118
NF4	32449	2143	12129	13597

2. **Anatomy** [21] is an ontology captures linkages of different phenotypes to genes. It consists of 278883 TBox statements involving 106495 classes and 218 relations.
3. **Gene Ontology(GO)** [22] unifies the representation of gene across all species. It consists of 130094 TBox statements, 45907 classes and 16 relations.
4. **GALEN** [23] also represents clinical information. It consists of 84537 TBox statements with 24353 classes and 1010 relations.

Table 1 presents the four ontologies in the increasing order of their size (number of axioms) and highlights the differences between them in terms of the types of axioms. For instance, we note that SNOMED has 0 disjoint axioms and only 1 role chain axiom, despite being the largest amongst the four ontologies. On the other hand, GALEN, being one of the smaller ontologies, has the highest number of role inclusion (958) and role chain (58) axioms. Also, observe that ANATOMY is the only ontology considered that has the representation of all the EL constructs considered in this work.

4.2. Baselines

We consider the following commonly used knowledge graph embeddings for comparison with our proposed EmEL^{++} embeddings.

1. **TransE** [1], one of the most frequently used embedding model for knowledge graph, introduced the idea of translation based embeddings where the relations between entities is interpreted as a translation operation between the entities.
2. **TransH** [2], is an extension of TransE that better handles reflexive, one-to-many, many-to-one, and many-to-many relations. TransH considers relations as hyperplanes in the embedding space. The translation operation is then performed over the projections of entities on the hyperplane.
3. **DistMult** [3], a matrix factorization based embedding model, has been found empirically to perform well at compositional reasoning tasks.
4. **EL Embeddings (ElEm)** [7] is one of the first embedding models for the $\mathcal{EL}^{++}$ description logic based model. Our proposed model is also an extension of ElEm embeddings and enhances ElEm by introducing additional constraints for a more comprehensive coverage of $\mathcal{EL}^{++}$ description logic.

Table 2
Best performing Hyper-parameters for each model. n indicates the dimension of embedding vectors and γ is the margin loss parameter.

	EmEL^{++}		ElEm		TransE		TransH		DistMult	
	n	γ	n	γ	n	γ	n	γ	n	γ
GALEN	50	0.0	50	0.0	100	-0.1	100	-0.1	100	-0.1
GO	100	-0.1	100	-0.1	100	-0.1	100	-0.1	100	0.1
ANATOMY	200	-0.1	200	-0.1	100	-0.1	100	-0.1	100	-0.1
SNOMED CT	100	-0.1	100	-0.1	50	-0.1	50	-0.1	50	-0.1

We use the pykeen framework [24] for implementations of TransE, TransH, and DistMult embedding models. For ElEm embeddings, we used the source code provided by the authors[1]. Our implementation of EmEL^{++} is publicly available at https://github.com/kracr/EmELpp.

4.3. Experimental Protocol

For learning the embeddings by different models, we first normalize the ontologies as described in Section 3. Next, we remove 30% of the subclass relation pairs from the normalized ontology to be used for validation (20%) and testing (10%). The remaining ontology with 70% sub-class relation pairs is used as the training set for learning the embedding functions. Further, we take an inferences set that consists of the inferences drawn on the training set using a standard Elk reasoner to evaluate the performance of learned embeddings. We perform hyper-parameter tuning using the 20% validation set and report the performance of fine-tuned models on the test set. The hyper-parameters to tune for all the models are the dimensions of the embedding vectors, and the margin parameter γ. We consider $n = \{50, 100, 200\}$ and $\gamma = \{-0.1, 0, 0.1\}$ yielding nine different settings. The best performing hyper-parameters for each of the models are reported in Table 2.

4.4. Reasoning Performance of EmEL^{++}

We chose subsumption as the main task to evaluate the effectiveness of the proposed EmEL^{++} embeddings. Note that once we have embedded the ontologies in a vector space, we have to reduce all the tasks we want to accomplish to operations that can be performed in an n-dimensional space. We reduce the task of subsumption in the embedding vector space as a distance-based operation. Given a test instance of the form $C \sqsubseteq D$, we take D as our source class and rank all the other classes in the ontology in increasing order of their distance from D in the vector space. We then compare the effectiveness of different embedding models based on the rank at which C is present in the ranked list. An embedding model that successfully captures the subclass relation between the two classes should be able to assign vector representations to the two classes that are very close to each other, hence, producing a lower rank for C.

Table 3 summarizes the performance of embedding models for test and inferences set. We report and evaluate the performance using six metrics. Hits at ranks 1, 10 and 100 report the fraction of test cases for which the expected class was found within top 1, 10 and 100 ranks, respectively. A median rank of m means that for 50% of the test cases, the correct answer was

[1]https://github.com/bio-ontology-research-group/el-embeddings

Table 3
Embedding Models Ranking based Performance on Test and Inferences Data

		Test Data					*Inferences Data*				
	Metric	**TransE**	**TransH**	**DistMult**	**ElEm**	**EmEL++**	**TransE**	**TransH**	**DistMult**	**ElEm**	**EmEL++**
GALEN	**Hits@1**	0.00	0.00	0.00	0.01	**0.02**	0.00	0.00	0.00	0.06	**0.07**
	Hits@10	0.00	0.00	0.00	0.08	**0.11**	0.00	0.00	0.00	0.18	**0.23**
	Hits@100	0.00	0.00	0.00	0.15	**0.16**	0.00	0.00	0.00	0.31	**0.35**
	AUC	0.54	0.48	0.51	0.64	**0.65**	0.61	0.47	0.50	0.84	**0.85**
	Median Rank	10748	11721	12600	6658	**6623**	8536	12773	12111	750	**585**
	90th %ile Rank	21308	21825	21823	**19681**	20635	18871	21890	22029	10627	**10339**
GO	**Hits@1**	0.00	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.06	**0.07**
	Hits@10	0.00	0.00	0.00	0.08	**0.09**	0.00	0.00	0.00	0.30	**0.32**
	Hits@100	0.00	0.00	0.00	0.23	**0.24**	0.00	0.00	0.00	0.49	0.49
	AUC	0.53	0.44	0.50	0.73	**0.78**	0.83	0.44	0.50	0.86	**0.90**
	Median Rank	20079	26280	22493	4302	**2908**	3718	26015	22977	**113**	122
	90th %ile Rank	40177	41996	40425	41225	**33547**	20791	41938	40707	26090	**17527**
ANATOMY	**Hits@1**	0.00	0.00	0.00	0.04	0.04	0.00	0.00	0.00	0.26	**0.27**
	Hits@10	0.00	0.00	0.00	**0.19**	0.17	0.00	0.00	0.00	**0.71**	0.65
	Hits@100	0.01	0.00	0.00	**0.41**	0.39	0.01	0.00	0.00	**0.84**	0.79
	AUC	0.53	0.44	0.49	0.73	**0.76**	0.68	0.46	0.49	**0.97**	0.96
	Median Rank	46679	61349	52486	640	**381**	29336	58987	53690	3	3
	90th %ile Rank	**91235**	96192	93619	105111	105274	66940	93177	94823	**1501**	2557
SNOMED CT	**Hits@1**	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.08**	0.07
	Hits@10	0.00	0.00	0.00	0.03	0.03	0.00	0.00	0.00	**0.30**	0.28
	Hits@100	0.00	0.00	0.00	**0.08**	0.06	0.00	0.00	0.00	**0.42**	0.37
	AUC	0.50	0.48	0.50	0.63	**0.64**	0.56	0.48	0.50	0.81	**0.83**
	Median Rank	150876	157186	151624	**80289**	87413	125431	157284	153020	**1704**	3321
	90th %ile Rank	274465	278455	275982	277874	**261359**	253452	279666	275345	220771	**192339**

found below rank m. 90^{th} percentile rank denotes the rank value below which the correct class was found for 90% of the test cases. The first observation that we make from Table 3 is that ElEm and EmEL^{++} embeddings perform better than the three commonly used KG embeddings (TransE, TransH, and DistMult). This observation highlights the inadequacy of traditional KG embeddings that do not consider the ontological constructs and rely only on the structural properties of the underlying graph. Both ElEm and EmEL^{++} embeddings incorporate specific constraints and charactersitics of $\mathcal{EL}^{++}$ description logic, and hence, the embeddings produced by these models are better at retaining the properties of the underlying ontologies in the vector space. Next, we note from the two tables that there is no clear winner among ElEm and EmEL^{++}. While EmEL^{++} outperforms ElEm for the GALEN and GO ontologies, there is no clear winner for Anatomy and SNOMED ontologies as their performance varies. This observation is consistent with previous empirical studies comparing different link prediction methods that found that no single method outperforms across a variety of datasets [25, 26]. We speculate that this divergence in performance could be attributed to the different distributions of the types of axioms in the ontology (ref. Table 1). Over (or under) representation of certain types of axioms may lead to the optimization process giving more (or less) weight to the corresponding loss functions during the training phase. Understanding the exact mechanism behind the performance characteristics of different ontologies is a crucial and challenging area of future research.

Table 4
Accuracies achieved by the ElEm and EmEL++ embeddings in terms of geomteric interpretation of the classes in different ontologies.

	Training		*Testing*		*Inferences*	
	ElEm	EmEL^{++}	ElEm	EmEL^{++}	ElEm	EmEL^{++}
GALEN	0.27	**0.64**	0.20	**0.53**	0.27	**0.64**
GO	0.45	**0.59**	0.35	**0.44**	0.48	**0.62**
ANATOMY	0.09	**0.48**	0.07	**0.22**	0.10	**0.49**
SNOMED CT	0.24	**0.55**	0.18	**0.34**	0.22	**0.48**

4.5. Preserving Ontology Characteristics in Vector Space

Next, we compare the ElEm and EmEL^{++} in terms of retaining the underlying characteristics of ontology in vector space. Recall that both the models map the classes in an ontology to n-balls in the vector space. Further, the mapping is such that the n-ball of a super-class subsumes the n-balls of its sub-classes. Thus, for a test instance $C \sqsubseteq D$, we check that the n-ball of class C lies inside the n-ball of class D in the vector space. Note that since we have the centers and radii of the corresponding n-balls, this can be checked easily. Table 4 presents the training, testing, and the inference accuracy obtained for the two embedding models for this task. We report accuracy values, i.e., the fraction of instances where the subsumption relation between the classes was maintained in the vector space. Note that this is a much stricter criterion for even if the subclass n-ball is slightly outside the n-ball of the superclass, it will be considered a failure. We observe from Table 4 that EmEL^{++} outperforms the ElEm embeddings for all the datasets and across all settings. This indicates that EmEL^{++} embeddings are better at preserving the class relationships in the mapped vector space than ElEm embeddings.

5. Conclusions

We proposed EmEL^{++}, an embedding model for $\mathcal{EL}^{++}$ ontologies. EmEL^{++} builds upon and extends the previously proposed ElEm embeddings by focusing on role inclusions and role chains and offers a more complete coverage of $\mathcal{EL}^{++}$. Experiments with four different ontologies showed that EmEL^{++} outperforms traditional KB embeddings on the subsumption reasoning task. Further, when compared with ElEm embeddings, it is able to better preserve the underlying semantics of the ontologies in the vector space. We have also shown how to perform the subsumption reasoning task in a vector space, which is an $O(n)$ operation in the worst case. We believe this is an important capability and it offers exciting directions for future work.

References

[1] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Advances in neural information processing systems, 2013, pp. 2787–2795.

[2] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Twenty-Eighth AAAI conference on artificial intelligence, 2014.

[3] B. Yang, W.-t. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, arXiv preprint arXiv:1412.6575 (2014).
[4] M. Nickel, V. Tresp, H.-P. Kriegel, A three-way model for collective learning on multi-relational data., in: Icml, volume 11, 2011, pp. 809–816.
[5] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, International Conference on Machine Learning (ICML), 2016.
[6] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: Twenty-ninth AAAI conference on artificial intelligence, 2015.
[7] M. Kulmanov, W. Liu-Wei, Y. Yan, R. Hoehndorf, EL embeddings: geometric construction of models for the description logic el++, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, AAAI Press, 2019, pp. 6103–6109.
[8] F. Baader, S. Brandt, C. Lutz, Pushing the EL Envelope, LTCS-Report LTCS-05-01, Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, Germany, 2005. See http://lat.inf.tu-dresden.de/research/reports.html.
[9] V. Misra, S. Bhatia, Bernoulli embeddings for graphs, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[10] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.
[11] D. Garg, S. Ikbal, S. K. Srivastava, H. Vishwakarma, H. Karanam, L. V. Subramaniam, Quantum Embedding of Knowledge for Reasoning, in: Advances in Neural Information Processing Systems, 2019, pp. 5595–5605.
[12] F. Z. Smaili, X. Gao, R. Hoehndorf, Onto2vec: Joint vector-based representation of biological entities and their ontology-based annotations, Bioinformatics 34 (2018) i52–i60.
[13] Ö. Özçep, M. Leemhuis, D. Wolter, Cone semantics for logics with negation, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI, 2020, pp. 1820–1826.
[14] A. Eberhart, M. Ebrahimi, L. Zhou, C. Shimizu, P. Hitzler, Completion reasoning emulation for the description logic el+, arXiv preprint arXiv:1912.05063 (2019).
[15] F. Baader, S. Brandt, C. Lutz, Pushing the EL envelope, in: IJCAI, volume 5, 2005, pp. 364–369.
[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in neural information processing systems, 2013, pp. 3111–3119.
[17] P. Ristoski, H. Paulheim, Rdf2vec: Rdf graph embeddings for data mining, in: International Semantic Web Conference, Springer, 2016, pp. 498–514.
[18] J. Mendez, jcel: A Modular Rule-based Reasoner., in: ORE, 2012.
[19] D. P. Kingma, J. Ba, Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014), 2014.
[20] K. Donnelly, SNOMED-CT: The advanced terminology and coding system for ehealth, Studies in health technology and informatics 121 (2006) 279.
[21] C. J. Mungall, C. Torniai, G. V. Gkoutos, S. E. Lewis, M. A. Haendel, Uberon, an integrative multi-species anatomy ontology, Genome biology 13 (2012) R5.
[22] G. O. Consortium, The Gene Ontology (GO) database and informatics resource, Nucleic acids research 32 (2004) D258–D261.
[23] A. Rector, J. Rogers, P. Pole, The GALEN high level ontology (1996).
[24] M. Ali, H. Jabeen, C. T. Hoyt, J. Lehmann, The KEEN Universe, in: International Semantic Web Conference, Springer, 2019, pp. 3–18.
[25] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, Journal of the American society for information science and technology 58 (2007) 1019–1031.
[26] L. Lü, T. Zhou, Link prediction in complex networks: A survey, Physica A: statistical mechanics and its applications 390 (2011) 1150–1170.