Box-To-Box Transformations for Modeling Joint Hierarchies

Shib Sankar Dasgupta, Xiang Lorraine Li, Michael Boratko
Dongxu Zhang, Andrew McCallum
College of Information and Computer Sciences
University of Massachusetts, Amherst
{ssdasgupta,xiangl,mboratko,dongxu,mccallum}@cs.umass.edu

Abstract

Learning representations of entities and relations in structured knowledge bases is an active area of research, with much emphasis placed on choosing the appropriate geometry to capture the hierarchical structures exploited in, for example, ISA or HASPART relations. Box embeddings (Vilnis et al., 2018; Li et al., 2019; Dasgupta et al., 2020), which represent concepts as n-dimensional hyperrectangles, are capable of embedding hierarchies when training on a subset of the transitive closure. In Patel et al. (2020), the authors demonstrate that only the transitive reduction is required and further extend box embeddings to capture joint hierarchies by augmenting the graph with new nodes. While it is possible to represent joint hierarchies with this method, the parameters for each hierarchy are decoupled, making generalization between hierarchies infeasible. In this work, we introduce a learned box-to-box transformation that respects the structure of each hierarchy. We demonstrate that this not only improves the capability of modeling cross-hierarchy compositional edges but is also capable of generalizing from a subset of the transitive reduction.

1 Introduction

Representation learning for hierarchical relations is crucial in natural language processing because of the hierarchical nature of common knowledge, for example, <Bird ISA Animal> (Athiwaratkun and Wilson, 2018; Vendrov et al., 2016; Vilnis et al., 2018; Nickel and Kiela, 2017). The ISA relation represents meaningful hierarchical relationships between concepts and plays an essential role in generalization for other relations, such as the generalization of <organ PARTOF person> based on <eye PARTOF of person>, and <organ ISA eye>. The fundamental nature of the ISA relation means that it is inherently involved in a large amount of compositional reasoning involving other relations.

Modeling hierarchies is essentially the problem of modeling a *poset*, or *partially ordered set*. The task of inferring missing edges that requires learning a transitive relation, was introduced in Vendrov et al. (2016). The authors also introduce a model based on the *reverse product order* on $\mathbb{R}^n$, which essentially models concepts as *infinite cones*. Region-based representations have been effective in representing hierarchical data, as containment between regions is naturally transitive. Vilnis et al. (2018) introduced axis-aligned hyperrectangles (or *boxes*) that are provably more flexible than cones, and demonstrated state-of-the-art performance in multiple tasks.

Thus far, not as much effort has been put into modeling joint hierarchies. Patel et al. (2020) proposed to simultaneously model ISA and HASPART hierarchies from Wordnet (Miller, 1995). In order to do so, they effectively augmented the graph by duplicating the nodes to create a single massive hierarchy. Their model assigns two separate box embeddings B_{ISA} and B_{HASPART} for each node n, where these two do not share any common parameter between them, and therefore misses out on a large amount of semantic relatedness between ISA and HASPART .

In this paper we propose a box-to-box transformation which translates and dilates box representations between hierarchies. Our proposed model shares information between the ISA and HASPART hierarchies via this transformation as well as cross-hierarchy containment training objectives. We compare BOX-TRANSFORM MODEL with multiple strong baselines under different settings. We substantially outperform the prior TWO-BOX MODEL while training with only the transitive reduction (which is informally the minimal graph with the same connectivity as the original hierarchy) of both hierarchies and predicting inferred composition

Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021), pages 277–288
Bangkok, Thailand (Online), August 6, 2021. ©2021 Association for Computational Linguistics

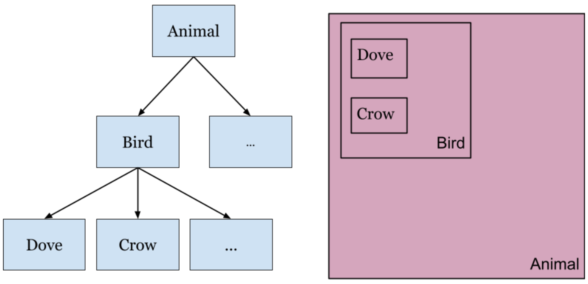


Figure 1: An example Box Embedding representation of the IsA hierarchy where

edges. As mentioned above, our model's shared learned features should allow for more generalization, and we test this by training on a subset of the transitive reduction, where we find we are able to outperform strong baselines. Finally, we perform a detailed analysis of the model's capacity to predict compositional edges and transitive closure edges, both from an overfitting and generalization standpoint, identifying subsets where further improvement is needed. The source code for our model and the dataset can be found in `https://github.com/iesl/box-to-box-transform.git`.

2 Related Work

Recent advances in representing one single hierarchy mainly fall in two categories: 1) representing hierarchies in non-Euclidian space (eg. hyperbolic space, due to the curvature's inductive bias to model tree-like structures) 2) using region-based representations instead of vectors for each node in the hierarchy (Erk, 2009). Hyperbolic space has been shown to be efficient in representing hierarchical relations, but also encounters difficulties in training (Nickel and Kiela, 2017; Ganea et al., 2018b; Chamberlain et al., 2017).

Categorization models in psychology often represent a concept as a region (Nosofsky, 1986; Smith et al., 1988; Hampton, 1991). Vilnis and McCallum (2015) and Athiwaratkun and Wilson (2018) use Gaussian distributions to embed each word in the corpus, the latter of which uses thresholded divergences which amount to region representations. Vendrov et al. (2016) and Lai and Hockenmaier (2017) make use of the reverse product order on $\mathbb{R}^n_+$, which effectively results in cone representations. Vilnis et al. (2018) further extend this cone representation to axis-aligned hyperrectangles (or *boxes*), and demonstrate state-of-the-art performance on modeling hierarchies. Various training improvement methods for box embeddings have been proposed (Li et al., 2019; Dasgupta et al., 2020), the most recent of which, *GumbelBox*, use a latent noise model where box parameters are represented via Gumbel distributions to improve on the loss landscape by making the gradient smooth for the geometric operations involved with box embeddings.

Region representations are also used for tasks which do not require modeling hierarchy. In Vilnis et al. (2018), the authors also model conditional probability distributions using box embeddings. Abboud et al. (2020) and Ren et al. (2020) take a different approach, using boxes for their capacity to contain many vectors to provide slack in the loss function when modeling knowledge base triples or representing logical queries, respectively. Ren et al. (2020) also made use of an action on boxes similar to ours, involving translation and dilation, however our work differs in both the task (i.e. representing logical queries vs. joint hierarchies) and approach, as their model represents entities using vectors and a loss function based on a box-to-vector distance. The inductive bias of hyperbolic space is also exploited to model multiple relations, Ganea et al. (2018a) learn hyperbolic transformations for multiple relations using Poincare embeddings, and show model improvement in low computational resource settings. Patel et al. (2020), which our work is most similar to, represent joint hierarchies using box embeddings. However, they represent each concept with two boxes ignoring the internal semantics of the concepts.

Modeling joint hierarchies shares some similarities with knowledge base completion, however the goals of the two settings are different. When modeling joint hierarchies you are attempting to learn simultaneous transitive relations, and potentially learn relevant compositional edges involving these relations. For knowledge base completion, on the other hand, you may be learning many different relations, and primarily seek to recover edges which were removed rather than inferring new compositional edges. Still, the models which perform knowledge base completion can be applied to this task, as the data can be viewed as knowledge base triples with only 2 relations. There have been multiple works that aim to build better knowledge representation (Bordes et al., 2013; Trouil-

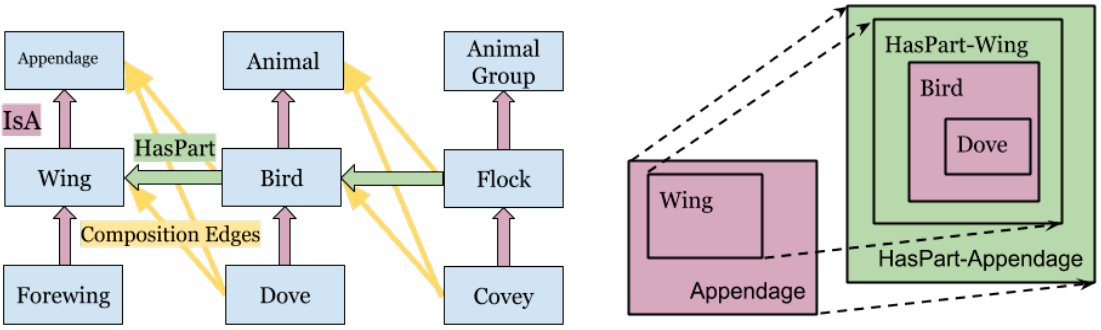


Figure 2: An overview of Box-Transform Model on joint IsA and HasPart hierarchies. Composition edges are created following certain rules and it should be correctly inferred for a well-trained model. The IsA Wing box is transformed into a HasPart Wing box representing concepts that has wings, and Bird is a subset of it. Same follows for Appendage, and the monotonicity in the IsA space is preserved in HasPart space.

lon et al., 2016; Sun et al., 2019; Balazevic et al., 2019b). Most relevant, (Chami et al., 2020; Balazevic et al., 2019a) recently proposed KG embedding methods which embeds entities in the Poincaré ball model of hyperbolic space. These models are intended to capture relational patterns present in multi-relational graphs, with a particular emphasis on hierarchical relations.

3 Background

3.1 Box Lattice Model

Introduced in (Vilnis et al., 2018), a *box lattice model* (or *box model*) is a geometric embedding which captures partial orders and lattice structure using n-dimensional hyper-rectangles. Formally, we define the set of boxes $\mathcal{B}$ in $\mathbb{R}^n$ as

$$\mathcal{B}(\mathbb{R}^n) = \{[x_1, x^1] \times \cdots \times [x_d, x^d]\}, \qquad (1)$$

where $x_i, x^j \in \mathbb{R}$, and we represent all degenerate boxes where $x_i > x^i$ with $\emptyset$. A box model for a set S is a function $\text{Box} : S \to \mathcal{B}(\mathbb{R}^n)$ which captures some desirable properties of the set S. As the name implies, the box lattice model is particularly suited to representing partial orders and lattice structures.

Definition 1 (Poset)**.** A *partially ordered set*, or *poset*, is a set P along with a relation $\preceq$ such that, for each $a, b, c \in P$, we have

1. $a \preceq a$ (reflexivity)
2. if $a \preceq b$ and $b \preceq a$ then $a = b$ (antisymmetry)
3. if $a \preceq b$ and $b \preceq c$ then $a \preceq c$ (transitivity)

Definition 2 (Lattice)**.** A *lattice* is a poset where each pair of elements have a unique upper bound called the *join*, denoted by $\wedge$, and a unique lower bound called the *meet*, denoted by $\vee$.

The authors note that there are natural geometric operations which form a lattice structure on $\mathcal{B}$:

$$\text{Box}(x) \wedge \text{Box}(y) := \prod_i [\max(x_i, y_i), \min(x^i, y^i)], \qquad (2)$$

$$\text{Box}(x) \vee \text{Box}(y) := \prod_i [\min(x_i, y_i), \max(x^i, y^i)], \qquad (3)$$

In other words, the *meet* of two boxes is the smallest containing box, and the *join* is the intersection, or $\emptyset$ if the boxes are disjoint. These geometric operations map very neatly to hierarchies, where the meet of two nodes is their closest common ancestor and the join is the closest common descendent (or $\emptyset$ if no such node exists). The ability of this model to capture lattice structure using geometric operations makes it a natural choice to embed hierarchies.

3.2 Probabilistic Box Model Training

In Vilnis et al. (2018), the authors also introduced a *probabilistic* interpretation of box embeddings and a learning method which was improved upon in Li et al. (2019) and Dasgupta et al. (2020). By using a probability measure μ on $\mathbb{R}^d$ (or by constraining the space to $[0, 1]^d$), one can calculate box volumes as $\mu(\text{Box}(X))$. The pullback of this measure yields a probability measure on S, and thus the box model can be imbued with valid probabilistic semantics. In particular, since the box space $\mathcal{B}$ is closed under intersection, we can calculate joint probabilities by computing $P(X, Y) = \mu(\mathrm{Box}(X) \wedge \mathrm{Box}(Y))$ and similarly compute conditional probabilities as

$$P(X \mid Y) = \frac{\mu(\mathrm{Box}(X) \wedge \mathrm{Box}(Y))}{\mu(\mathrm{Box}(Y))}. \tag{4}$$

The conversion from a poset or lattice structure to probabilistic semantics is accomplished by assigning conditional probabilities, namely $a \preceq b$ if and only if $P(b \mid a) = 1$. We note that the properties required of the relation $\preceq$ follow as a natural consequence of the axioms for conditional probability. Apart from providing rigor and interpretability, the calibrated probabilistic semantics also inform and facilitate the training procedure for box embeddings, which is accomplished via gradient descent using KL-divergence with respect to the aforementioned probability distribution as a loss function.

As one might expect, care must be taken to handle the case when boxes are disjoint, as there is no gradient. In Vilnis et al. (2018) the authors made use of the lattice structure to derive a lower bound on the probability, and Li et al. (2019) introduced an approximation to Gaussian convolution over the boxes which similarly handled the case of disjoint boxes. Dasgupta et al. (2020) improves this further by taking a random process perspective, ensembling over an entire family of box models. The endpoints of boxes are represented using Gumbel distributions, that is

$$\begin{aligned} \mathrm{GumbelBox}(X) &= \prod_i [X_i, X^i], \\ X_i &\sim \mathrm{MaxGumbel}(\mu_i, \beta), \\ X^i &\sim \mathrm{MinGumbel}(\mu^i, \beta), \end{aligned} \tag{5}$$

where μ, β are the location and scale parameters of the Gumbel distribution respectively. The $\mathrm{MaxGumbel}$ distribution is given by

$$f(x; \mu, \beta) = \frac{1}{\beta} \exp(-\frac{x-\mu}{\beta} - e^{-\frac{x-\mu}{\beta}}), \tag{6}$$

and the $\mathrm{MinGumbel}$ distribution given by negating x an μ. The Gumbel distribution was chosen due to it's min/max stability, making the set of Gumbel boxes closed under intersection, i.e. the intersection of two Gumbel boxes is another Gumbel box. We denote the space of all such boxes as $\mathcal{G}$. The expected volume of a Gumbel box can be efficiently calculated analytically, and in Dasgupta et al. (2020) the authors use this expected volume to calculate the conditional probabilities mentioned in equation (4). This training method leads to improved performance on many tasks, and is particularly beneficial when embedding trees, thus we will use *GumbelBox* in our setting.

3.3 Modeling Joint Hierarchies

Many existing methods have been proposed for modeling a single hierarchy, however entities are often simultaneously part of multiple hierarchies, for example *hypernymy* (i.e. IsA) and *meronomy* (i.e. HasPart). Furthermore, useful information can be shared across inferred compositional edges between the two hierarchies. For example, as shown in 2, based on <Bird,HasPart ,Wing> and <Dove,IsA ,Bird>, we can infer <Dove,HasPart ,Wing>. Due to the compositional nature of these relations, we can infer not only the per-relation transitive closure edges but also the compositional edges, i.e <Dove, HasPart , Wing>.

Formally, for two hierarchical relations r_1 and r_2, composition edges can be formulated following certain rules. In figure 2, the rules are designed as follows: for <Head,HasPart ,Tail>, < x_1, IsA , Head> represent the sub-class of Head, and <Tail, IsA , x_2 > is the super-class of Tail. Composition edges can be generated as < x_1,HasPart ,x_2 >, < x_1,HasPart ,Tail> or < Head ,HasPart ,x_2 >. These compositional edges are identified in Patel et al. (2020), where it is observed that a model which effectively captures both hierarchies should correctly predict not only over the transitive closure of each individual relation but also on these compositional edges.

4 Methods

4.1 Box-to-Box Transformation

As mentioned previously, our goal is to not only capture intra-relation transitivity, but also require the model to capture cross-hierarchy compositional edges; that is, for a set S with two partial orders $\preceq_1$, $\preceq_2$, we want a model capable of learning $(a \preceq_1 b) \wedge (b \preceq_2 c) \implies a \preceq_2 c$ and $(a \preceq_2 b) \wedge (b \preceq_1 c) \implies a \preceq_2 c$. Furthermore, we hope to do so without including these compositional edges in our training data, with the expectation that the embedding parameters capture relevant structure which allows us to recover them.

As shown in Dasgupta et al. (2020), Gumbel boxes are able to model hierarchies, we would like to benefit from this capability, particularly for modeling the IsA hierarchy, and thus we seek to learn a function $f_1 : S \to \mathcal{G}$, where

$$a \preceq_1 b \iff \frac{E[\mu(f_1(a) \cap f_1(b))]}{E[\mu(f_1(a))]} = 1. \quad (7)$$

For a given Gumbel box,

$$\begin{aligned} f(x) = \prod_{i=1}^{d} [X_i, X^i], \\ X_i \sim \mathrm{MaxGumbel}(\mu_i, \beta), \\ X^i \sim \mathrm{MinGumbel}(\mu_i + \Delta_i, \beta). \quad (8) \end{aligned}$$

where the free parameters are μ_i and Δ_i. To simultaneously model a second relation, we train a function $\varphi : \mathcal{G} \to \mathcal{G}$ such that

$$a \preceq_2 b \iff \frac{E[\mu(\varphi(f_1(a)) \cap f_1(b))]}{E[\mu(\varphi(f_1(a)))]} = 1. \quad (9)$$

For notational simplicity, we abbreviate $f_2 = \varphi \circ f_1$.

We choose the transformation φ to operate on the "min" coordinate of a Gumbel box and the "side-lengths", that is, we transform a given Gumbel box

$$\begin{aligned} f(x) = \prod_{i=1}^{d} [X_i, X^i], \\ X_i \sim \mathrm{MaxGumbel}(\mu_i, \beta), \\ X^i \sim \mathrm{MinGumbel}(\mu_i + \Delta_i, \beta). \quad (10) \end{aligned}$$

to

$$\varphi\left(\mathrm{GumbelBox}(X)\right) = \prod_{i=1}^{d} [Y_i, Y^i], \quad (11)$$

where

$$\begin{aligned} Y_i &\sim \mathrm{MaxGumbel}(\theta_i \mu_i + b_i, \beta) \\ Y^i &\sim \mathrm{MinGumbel}(\theta_i \mu_i + b_i + \mathrm{softplus}(\theta^i \Delta_i + b^i), \beta) \end{aligned}$$

and the $\theta_i, \theta^i, b_i, b^i$ are learned parameters. This effectively translates and dilates the location parameters of the Gumbel distributions which represent the "corners" of a given Gumbel box. We call this model the Box-Transform Model .

The softplus function is used here as a way to ensure the max coordinate remains larger than the min, and it also provides a simple overflow protection for the expected box volume, as might happen with side-lengths larger than one in high dimensions. While mathematically simple, this transformation allows for parameter sharing between the embedding of a concept with respect to $\preceq_1$ and with respect to $\preceq_2$. Importantly, the transformation is capable of capturing both a global translation and dilation as well as a scaled transformation of the existing learned representation, allowing the absolute position in space (which, for previous box embedding models, was irrelevant) to potentially capture relevant features of the entities.

Remark 1. The lack of a transformation on $f_1(b)$ is not an oversight. Using figure 2 as an example, if we consider the Bird box as representative of "all things which are birds", and the HasPart Wing box as the representative of "all thing which have wings", then encouraging containment of the Bird box inside the HasPart Wing box is quite natural. This conceptual motivation is precisely captured by the lack of a transformation on $f_1(b)$. This also coincides with the probabilistic semantics discussed in section 3.2, and is also the method employed by (Patel et al., 2020), where this cross-hierarchy containment objective is soley responsible for any flow of information between hierarchies in the Two-Box Model .

4.2 Connection to Two-Box Model

There are two main differences between our model and the model introduced in Patel et al. (2020), the Two-Box Model . First, the Two-Box Model preceded the Gumbel box model, and instead uses the Soft box model from (Li et al., 2019). To ensure that the benefits from our model are not conflated with the improvements from using Gumbel boxes we also train a Two-Box Model from (Patel et al., 2020) which makes use of Gumbel boxes.

Second, both models use different boxes to represent different relations, however, Two-Box Model allows both boxes to have free parameters, relying on containment between boxes representing different relations to pass information. Under the framework we have currently presented, this would be equivalent to learning two functions, f_1 and f_2, both of which have separate parameters for the min and side length of the boxes for each entity. While such a model has significant representational capacity, we would expect that it would suffer greatly from a lack of generalization. We evaluate this hypothesis by creating a second test, discussed in section 5.4, which removes edges from the transitive reduction of the training data.

5 Experiments

5.1 Dataset

We demonstrate the efficacy of BOX-TRANSFORM MODEL by using the joint hierarchy that has been created by Patel et al. (2020) from WordNet (Miller, 1995). In this dataset, *hypernymy* (ISA) and *meronymy* (HASPART) are two hierarchical relations of WordNet over noun sysnets, which are $82,114$ in total. Individually, the *hypernymy* part of the hierarchy contains $82,114$ nodes (i.e., all the synsets) with $84,363$ edges in its transitive reduction and the *meronymy* portion has $11,235$ synsets (out of $82,114$ synsets) with $9,678$ edges in its transitive reduction.

Joint Hierarchy In order to evaluate the performance on the joint hierarchy, Patel et al. (2020) created composition edges using the inter-relational semantics between *hypernymy* and *meronymy*. In particular they use the following composition rules:

$$\underbrace{\textsc{IsA} \circ \textsc{IsA} \cdots \textsc{IsA}}_{\text{0 or 1 or 2 times}} \circ \textsc{HasPart} \circ \underbrace{\textsc{IsA} \circ \textsc{IsA} \cdots \textsc{IsA}}_{\text{0 or 1 or 2 times}} = \textsc{HasPart}\,. \tag{12}$$

To illustrate from Figure 2, <*Dove* ISA *Bird*> ∧ <*Bird* HASPART *Wing*> ∧ <*Wing* ISA *Appendage*> implies that <*birds* HASPART *appendage*>. In total, $189,613$ composition edges are generated by the method described above for evaluation of the model on the joint hierarchy task. For each test/validation edge, a fixed set of negative samples of size 10 was generated by corrupting the head and tail 5 times each. The overall statistics for the dataset is provided in Table 1.

We have also created a second training dataset which further removes part of the transitive reduction to evaluate the models on their generalization capability (refer to Section 5.4 & 5.5). The dataset used for those section has different statistics and they are reported in the respective sections.

5.2 Baseline Models and Training Details

We compare BOX-TRANSFORM MODEL against geometric embedding methods as well as knowledge base completion methods. We give a brief description for each baseline below.

1. **TWO-BOX MODEL :** As mentioned in 4.2, Patel et al. (2020) extends the idea of Box embeddings (Vilnis et al., 2018; Li et al., 2019) to model joint hierarchies by defining two boxes per node, one for each relation.

2. **Order Embeddings:** (Vendrov et al., 2016) treats each concept as axis parallel cones in positive orthant. We considered two different cone parameters for each entity following the TWO-BOX MODEL (Patel et al., 2020).

3. **Poincaré Embeddings:** (Nickel and Kiela, 2017) & **Hyperbolic Entailment Cones** (Ganea et al., 2018b): Tree-structured data are best captured in hyperbolic space (Chamberlain et al., 2017). Thus in Nickel and Kiela (2017), the authors learn embedding on n-dimensional Poincaré ball. For similar reasons, Ganea et al. (2018b) uses the hyperbolic space however they extend the hyperbolic point embeddings to entailment cones. Again, for these models, two separate parameters are considered for each entity.

4. **TransE and RotatE** (Bordes et al., 2013; Sun et al., 2019): This task can be posed as knowledge base completion for a KB with only two relations. Thus we evaluate TransE and RotatE which are simple yet effective methods for knowledge base embeddings, which achieve state-of-the-art for many knowledge base embedding tasks. Unlike the TWO-BOX MODEL (Patel et al., 2020) or the other baselines, these methods have shared representation for each entity, and thus they are expected to generalise better on missing edges.

5. **Hyperbolic KG Embeddings** (Balazevic et al., 2019a; Chami et al., 2020): We also compared our method against recently proposed KG embedding methods based on hyperbolic embeddings to model hierarchical structures present in KGs. The Multi-Relational Poincaré model (MuRP) (Balazevic et al., 2019a) learns relation-specific transforms of the entities that are embedded in hyperbolic space. The RoTH (Chami et al., 2020) parameterize the relation specific transformations as hyperbolic rotation, where as the AttH (Chami et al., 2020) combines hyperbolic reflection and rotation using attention. More training details are in Appendix A.2.

Table 1: Details of the hypernymy, meronymy hierarchies and the composition edges.

	Transitive Reduction	Transitive Closure	Validation (pos/neg)	Test (pos/neg)
Hypernym	84,363	661,127	28,838/ 288,380	28,838/ 288,380
Meronym	9,678	30,333	5,164/ 51,640	5,164/ 51,640
Composite Edge	-	-	94,807/ 948,070	94,806/ 948,070

Table 2: Test F1 scores(%)of various methods for predicting the Composition edges.

Methods	F1 score
Poincaré Embeddings	43.8
Hyperbolic Entailment Cones	44.0
TransE	57.0
RotatE	51.0
Order Embeddings	68.5
MuRP	21.4
AttH	51.3
RotE	51.5
RotH	55.8
TWO-BOX MODEL (Patel et al., 2020)	68.1
TWO-BOX MODEL (with GumbelBox)	73.7
BOX-TRANSFORM MODEL	**82.2**

Table 3: Test F1 scores(%) of various methods for generalization capability.

Methods	F1 score
Poincaré Embeddings	33.5
Hyperbolic Entailment Cones	36.0
TransE	57.0
RotatE	55.0
Order Embeddings	54.5
MuRP	20.1
AttH	27.0
RotE	48.8
RotH	46.7
TWO-BOX MODEL (with GumbelBox)	58.9
BOX-TRANSFORM MODEL	**63.9**

5.3 Composition Edges from Transitive Reduction

In order to demonstrate the ability of the model to capture partially ordered (tree-like) data most embedding methods (Ganea et al., 2018b; Nickel and Kiela, 2017; Patel et al., 2020) train their model on the transitive reduction and predict on the transitive closure. For an evaluation on modeling the joint hierarchy, therefore, it is natural to train the models only on the transitive reduction of *hypernymy* and *meronymy* and evaluate on the composition edges, as done in Patel et al. (2020). We report the F1 score (with 1:10 negatives) for those edges in table 2. The threshold used for the classification is determined by maximizing the F1 score on the validation set.

From Table 2, we observe that BOX-TRANSFORM MODEL outperforms the other baselines by a significant margin. As mentioned in Patel et al. (2020) and so do we observe that in the next section 5.4 that the Poincaré embeddings and Hyperbolic entailment cones do face difficulty in learning when presented only with transitive reduction edges. However, the hyperbolic KG method Atth RoTH are able to learn the composite edges to a certain extent. The performance gain of RotH over its euclidean counterpart RotE can be attributed to its inductive bias towards modeling hierarchies. The performance of Box embedding method as proposed by Patel et al. (2020) performs at par order embedding method. However using GumbelBox formulation (Dasgupta et al., 2020), we observe significant performance boost as GumbelBox improves the local identifiability of the parameter space. Still, the capability of the BOX-TRANSFORM MODEL to benefit from shared cross-hierarchy features allows it to substantially outperform even this improved version of the TWO-BOX MODEL . This is likely due to the fact that the inductive bias provided by the transformation is more in line with the data; the model can benefit from the containments learned as a result of the ISA relation, and learn a HASPART transformation which potentially preserves these containments.

5.4 Learning from Incomplete Transitive Reduction

In Patel et al. (2020), and also in our previous experiment, we already observe that box embedding methods are highly capable of to recovering the transitive closure (in our case, composition edges) given the transitive reduction only. In this experiment, we train with even less of the transitive reduction, moving some of these edges to the test

Table 4: Single hierarchy F1 score (%) analysis on IsA and HasPart . The overall dataset is the combination of overfitting, generalization and extended generalization

	Type	Overall TC(X)	Overfitting TC(X1)	Generalization X-X1	Extended Generalization TC(X) - TC(X1) -(X-X1)
TransE	IsA	52.9	52.1	66.5	46.0
Two Box Model		47.8	58.9	19.9	22.9
Box-Transform Model		**57.3**	60.0	65.9	44.4
TransE	HasPart	**59.9**	63.0	56.1	48.3
Two Box Model		51.6	54.8	40.8	37.8
Box-Transform Model		58.8	64.2	33.4	25.4

Table 5: Joint hierarchy F1 score (%) analysis. The overall data is the combination of overfitting and generalization.

	Overall COMP(X, Y)	Overfitting COMP(X1, Y1)	Generalization COMP(X, Y) - COMP(X1, Y1)
TransE	58.8	70.1	68.6
Two Box Model	62.5	72.7	63.6
Box-Transform Model	**69.6**	86.1	70.0

set. Now, reconstruction of the closure and the composition edges require models to generalize over the missing parts of the graph. We train on 9175 *meronymy* edges and 80372 *hypernymy* edges and test/validate on an aggregated pool of 251783 edges. Please refer to the Appendix A.1 for details on dataset creation and statistics.

From Table 3, we observe that Box-Transform Model outperforms all the baseline methods by a large extent. Although the two box model is performing worse than Box-Transform Model , it is able to beat other baselines. Out of the two Knowledge base completion methods TransE performs the best and achieves comparative performance to two box model. Although the hyperbolic KG embeddings were able to perform well on the composite edges, their generalization performance is relatively lower than other KG embedding methods. We also observe that the RotE model that was under performing in composite edges, outperforms RotH by some margin in this generalization setting. We select the top three best performing methods for further analysis for each type of edges in the graph.

5.5 Performance analysis on different splits

Training on a subset of the transitive reduction showed that our model could generalize to composition edges even with the absence of essential edges to make such prediction. We further perform evaluation analysis using the same training data with the best-performed model selected by maximizing the f1 score on composition edges. We evaluate the model performance on the transitive closure for each hierarchy (IsA and HasPart), and the composition edges on the joint hierarchy.

For each single hierarchy, some edges are removed from the transitive reduction X to create the incomplete transitive reduction training data $X1$. Evaluating the transitive closure of X directly evaluates the model's performance on each hierarchy, denoted as $\text{TC}(X)$. This can be further divided into three categories: dataset that evaluates model ability to capture transitive closure of $X1$, $\text{TC}(X1)$, dataset that evaluates model generalization ability on missing edges $X - X1$, and dataset that evaluates model's extended generalization ability on $\text{TC}(X) - \text{TC}(X1)$.

Composition edges from the joint hierarchy can be analyzed the same way. $\text{Comp}(X, Y)$ represent all the composition edges in the full wordnet dataset, composed by IsA transitive reduction X and HasPart transitive reduction Y. It can be further divided into two categories: data that evaluate model overfitting ability to capture $\text{Comp}(X_1, Y_1)$ where X_1 and Y_1 is the corresponding training IsA and HasPart data in section 5.4, and data that evaluate model generalization ability on learning logical operations $\text{Comp}(X, Y) - \text{Comp}(X_1, Y_1)$. The detailed statistics on each of these splits are provided in Appendix A.4. The evaluation dataset is created by randomly creating negative examples with the pos: neg ratio 1:10. We select the top 3 best models from section 5.4, then choose the threshold that maximized the F1 score for the validation data of each split and report the test F1. As shown in table 4 and table 5, our model performs the best overall across different dataset splits. BOX-TRANSFORM MODEL performs much better on the full transitive closure of ISA , and all the composition edges. In general, BOX-TRANSFORM MODEL performs much better on transitive closure and composition edges by a large margin in all overfitting settings. TransE does better on predicting removed edges from the transitive reduction (which serves more as an analysis of the model's capability, as it is not a typical evaluation for partial order completion), however we note that our model does surprisingly well on the ISA missing edges, which we attribute to the shared semantics between the hierarchy made possible by this box-to-box transformation.

6 Conclusion

We proposed a box-to-box transformation that facilitates sharing of learned features across hierarchies when modeling joint hierarchies. We demonstrate the BOX-TRANSFORM MODEL is capable of achieving state-of-the-art performance compared with other strong baseline models when predicting compositional edges across a joint hierarchy. Furthermore, the model also outperforms other models when modeling the transitive closure of each relation independently. In the future, we aim to extend the current model from two relations to multiple relations in order to obtain more generalization from hierarchical ISA edges.

Acknowledgments

We the anonymous reviewers for their constructive feedback. This work was supported in part by the Center for Intelligent Information Retrieval and the Center for Data Science, in part by the Chan Zuckerberg Initiative, in part by the National Science Foundation under Grant No. IIS-1763618, in part by University of Southern California subcontract no. 123875727 under Office of Naval Research prime contract no. N660011924032, and in part by University of Southern California subcontract no. 89341790 under Defense Advanced Research Projects Agency prime contract no. FA8750-17-C-0106. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems NeurIPS*.

Ben Athiwaratkun and Andrew Gordon Wilson. 2018. Hierarchical density order embeddings. In *International Conference on Learning Representations*.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019a. Multi-relational poincaré graph embeddings. In *Advances in Neural Information Processing Systems*, volume 32, pages 4463–4473. Curran Associates, Inc.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019b. 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*.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Antoine Bordes, Nicolas Usunier, A. Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems*.

Benjamin Paul Chamberlain, James R. Clough, and Marc Peter Deisenroth. 2017. Neural embeddings of graphs in hyperbolic space. *13th international workshop on mining and learning from graphs held in conjunction with KDD*.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. *arXiv preprint arXiv:2005.00545*.

Shib Sankar Dasgupta, Michael Boratko, Dongxu Zhang, Luke Vilnis, Xiang Lorraine Li, and Andrew McCallum. 2020. Improving local identifiability for probabilistic box embeddings. In *Neural Information Processing Systems*.

Katrin Erk. 2009. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. 2018a. Hyperbolic neural networks. In *Advances in neural information processing systems*, pages 5345–5355.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018b. Hyperbolic entailment cones for learning hierarchical embeddings. In *International Conference on Machine Learning*.

James A Hampton. 1991. The combination of prototype concepts. *The psychology of word meanings*, pages 91–116.

Alice Lai and Julia Hockenmaier. 2017. Learning to predict denotational probabilities for modeling entailment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

Xiang Li, Luke Vilnis, Dongxu Zhang, Michael Boratko, and Andrew McCallum. 2019. Smoothing the geometry of probabilistic box embeddings. In *International Conference on Learning Representations*.

George A Miller. 1995. WordNet: a lexical database for English. *Communications of the ACM*.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Neural Information Processing Systems*.

Robert M Nosofsky. 1986. Attention, similarity, and the identification–categorization relationship. *Journal of experimental psychology: General*, 115(1):39.

Dhruvesh Patel, Shib Sankar Dasgupta, Michael Boratko, Xiang Li, Luke Vilnis, and Andrew McCallum. 2020. Representing joint hierarchies with box embeddings. *Automated Knowledge Base Construction*.

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. *International Conference on Learning Representations*.

Edward E Smith, Daniel N Osherson, Lance J Rips, and Margaret Keane. 1988. Combining prototypes: A selective modification model. *Cognitive science*, 12(4):485–527.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *International Conference on Learning Representations*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of images and language. In *International Conference on Learning Representations*.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic embedding of knowledge graphs with box lattice measures. In *Association for Computational Linguistics*.

Luke Vilnis and Andrew McCallum. 2015. Word representations via gaussian embedding. *International Conference on Learning Representations*.

A Appendix

A.1 Dataset creation steps from Section 5.4

In order to remove edges from the transitive reductions, we iterate through the transitive reduction edges of meronymy. With 0.5 probability we choose the edge for further processing. For each chosen HasPart edge, we select an outgoing IsA edge and pair them. We drop the IsA edge from the pair with 0.9 probability (the ratio of HASPART to ISA transitive reduction) and drop the HASPART edge in case the ISA is not dropped already. This procedure ensures that all the edge removals happen around the composition edges, thus, the results reflect the models true capacity to generalize well for this joint hierarchy task. We evaluate the model on the composition edges, the removed reduction edges, and the closure edges with 251783 in numbers which we split into two parts for validation and test. In Table 3, we report the F1 score on this aggregated evaluation data with 1:10 fixed true negatives.

A.2 Training Details

In our experiments, we have kept the number of parameters same across all the methods. We use 5 dimensional box embeddings for the Two Box Model (Patel et al., 2020). Since box embeddings are specified using min and side length in the same dimension. Thus we compare with 10 dimensional order embeddings, Poincaré embeddings, and hyperbolic entailment cones. However, since the above mentioned methods has two different number of parameters for each node, we use 20 dimensional vectors for RotatE, TransE to account for that. Our Box-Transform Model uses 10 dimension box embeddings for similar reason.

Hyperparameter range: We use Bayesian hypermeter optimizer with Hyperband algorithm for all the methods using the web interface (Biewald, 2020). The hyperparameter ranges are $Gumbel\beta \in [0.001, 3]$, Softplus temperature for box volume $T \in [1, 30]$, $lr \in [0.0005, 1]$, batch size $\in \{8096, 2048, 1024, 512\}$, number of negative samples $\in [2, 30]$ for all the methods. For max margin trainging we searched for the $margin \in [1, 50]$.

The best hyperparameters for our method and a few competitive baselines are provided in appropriate **config** files along with the source code. We will make the code public after the anonymity period.

In order to remove edges from the transitive reductions, we iterate through the transitive reduction edges of *meronymy*. With 0.5 probability we choose the edge for further processing. For each chosen HasPart edge, we select an outgoing IsA edge and pair them. We drop the IsA edge from the pair with 0.9 probability (the ratio of HasPart to IsA transitive reduction) and drop the HasPart edge in case the IsA is not dropped already.

This procedure ensures that all the edge removals happen around the composition edges, thus, the results reflect the models true capacity to generalize well for this joint hierarchy task. We evaluate the model on the composition edges, the removed reduction edges, and the closure edges with 251783 in numbers which we split into two parts for validation and test. In Table 3, we report the F1 score on this aggregated evaluation data with 1:10 fixed true negatives.

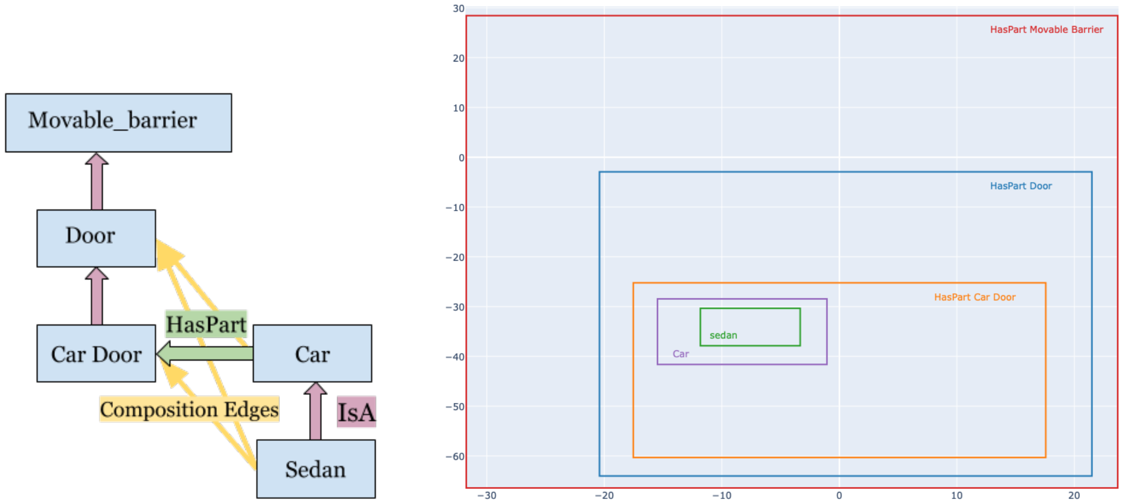
A.3 Visualization

We plot 2-dimensional box embeddings to inspect the quality of our proposed Box-Transform Model . Please refer to Figure 3. Here, we use the box embedding parameters of the best performing model from experiment 5.3 (Table 2). Note that, the model is 10 dimensional. However, for a perfectly trained model for the hierarchical tree-like data, we should observe more numbers of full containments, i.e., containment along each dimension. Thus, we pick two dimensions randomly out of the 10-d to visualize the box embeddings.

In the example in Figure 3 (next page), the facts that <Car,HasPart ,CarDoor> and <CarDoor,IsA ,Door> would enable us to infer that <Car,HasPart , Door>. This is a particular example of the compositional edges. We observe from the Figure 3 that the HasPart transformation of the ”Car Door” and ”Door” successfully encloses the IsA transformation of the ”Car”, thus our model is able infer that composition edge . All the other composite edges such as <Sedan,HasPart , CarDoor >, <Sedan,HasPart , Door> etc. can be similarly inferred from the visualization.

A.4 Details of the splits from Section 5.5

We report the performance of our method on different splits which are qualitatively different from each other. The detailed statistics of these splits can be found in Table 6 & 7.



(a) Example of Joint Hierarchy extracted from the WordNet dataset.

(b) We plot the transformed ISA box for ”Sedan” & ”Car” and transformed HASPART box for ”Door”, ”Car Door”, ”Movable Barrier” on the same space. The transformations do preserve the containment and provide an consistent assignment of box embedddings for the example on left.

Figure 3: 2-dimensional visualization of proposed Box embedding model.

Table 6: Dataset statistics for different parts of individual ISA and PARTOF hierarchy.

Hierarchy	TC(X)	TC(X1)	X-X1	TC(X) - TC(X1) - (X-X1)
IsA	61,667	51,195	3,991	6,481
HasPart	30,335	26,388	503	3,444

Table 7: Dataset statistics for different composition edges in Joint Hierarchy.

Hierarchy	Comp(X, Y)	COMP(X1, Y1)	COMP(X, Y1) - COMP(X1, Y1)
Joint Hierarchy	189,613	146,867	42,746