Concept2Box: Joint Geometric Embeddings for Learning Two-View Knowledge Graphs

Zijie Huang[1]*, Daheng Wang[2], Binxuan Huang[2], Chenwei Zhang[2], Jingbo Shang[4], Yan Liang[2], Zhengyang Wang[2], Xian Li[2], Christos Faloutsos[3], Yizhou Sun[1], Wei Wang[1]
[1]University of California, Los Angeles, CA, USA [2]Amazon.com Inc, CA, USA
[3]Carnegie Mellon University, PA, USA [4]University of California, San Diego, CA, USA
{zijiehuang,yzsun,weiwang}@cs.ucla.edu,
{dahenwan,binxuan,cwzhang,ynliang,zhengywa,xianlee}@amazon.com,
jshang@ucsd.edu,christos@cs.cmu.edu

Abstract

Knowledge graph embeddings (KGE) have been extensively studied to embed large-scale relational data for many real-world applications. Existing methods have long ignored the fact many KGs contain two fundamentally different views: high-level ontology-view concepts and fine-grained instance-view entities. They usually embed all nodes as vectors in one latent space. However, a single geometric representation fails to capture the structural differences between two views and lacks probabilistic semantics towards concepts' granularity. We propose Concept2Box, a novel approach that jointly embeds the two views of a KG using dual geometric representations. We model concepts with box embeddings, which learn the hierarchy structure and complex relations such as overlap and disjoint among them. Box volumes can be interpreted as concepts' granularity. Different from concepts, we model entities as vectors. To bridge the gap between concept box embeddings and entity vector embeddings, we propose a novel vector-to-box distance metric and learn both embeddings jointly. Experiments on both the public DBpedia KG and a newly-created industrial KG showed the effectiveness of Concept2Box.

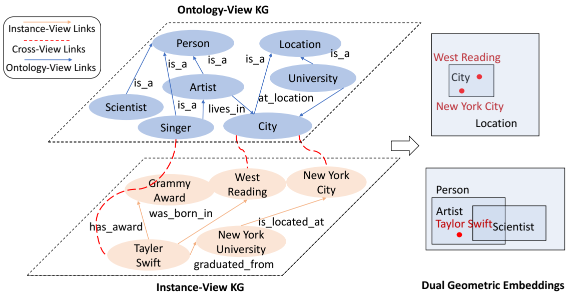

Figure 1: A two-view KG consisting of (1) an ontology-view KG with high-level concepts and meta-relations, (2) an instance-view KG with fine-grained instances and relations, and, (3) a set of cross-view links between two views of KG. Concept2Box learns dual geometric embeddings where each concept is modeled as a box in the latent space and entities are learned as point vectors.

1 Introduction

Knowledge graphs (KGs) like Freebase (Bollacker et al., 2008) and DBpedia (Lehmann et al., 2015) have motivated various knowledge-driven applications (Yasunaga et al., 2021; Lin et al., 2021). They contain structured and semantic information among nodes and relations where knowledge is instantiated as factual triples. One critical fact being ignored is that many KGs consist of two fundamentally different views as depicted in Figure 1: (1) an ontology-view component with high-level concepts and meta-relations, e.g., *("Singer", "is_a", "Artist")*, and (2) an instance-view component with specific entities and their relations, e.g., *("Taylor Swift", "has_award", "Grammy Award")*. Also, there are valuable cross-view links that connect ontological concepts with entities instantiated from them, which bridge the semantics of the two views.

Modern KGs are usually far from complete as new knowledge is rapidly emerging (Huang et al., 2022). How to automatically harvest unknown knowledge from data has been an important research area for years. Knowledge graph embedding (KGE) methods achieve promising results for modeling KGs by learning low-dimensional vector representations of nodes and relations. However, most existing methods (Bordes et al., 2013; Wang et al., 2014; Sun et al., 2019; Wang et al., 2021b) have a major limitation of only considering KGs of a single view. The valuable semantics brought by jointly modeling the two views are omitted. In contrast, modeling two-view KGs in a holistic way has a key advantage: entity embeddings provide rich information for understanding corresponding ontological concepts; and, in turn, a concept embedding

*Part of work was done during internship at Amazon.

Findings of the Association for Computational Linguistics: ACL 2023, pages 10105–10118
July 9-14, 2023

provides a high-level summary of linked entities, guiding the learning process of entities with only few relational facts.

Modeling two-view KG is non-trivial where the challenges are three-fold: First, there exists a significant structural difference between the two views, where the ontology-view KG usually forms a hierarchical structure with concept nodes of different granularity, and the instance-view KG usually follows a flat or cyclical structure (Iyer et al., 2022) with each entity associated with a specific meaning. Second, concepts often exhibit complex relations such as intersect, contain and disjoint. Traditional vector-based embeddings (Bordes et al., 2013; Sun et al., 2019) fail to adequately express them and capture interpretable probabilistic semantics to explain the granularity of concepts. Third, to bridge the knowledge from two views, effective semantic mappings from fine-grained entities to their corresponding concepts need to be carefully considered. For work that models two-view KGs, they either model concepts and entities all as points in the Euclidean space (Hao et al., 2019) and in the product manifold space (Wang et al., 2021c), which can fail to capture the structural differences among the two views and result in suboptimal performance, or they embed concepts and entities separately as points in the hyperbolic and spherical space respectively (Iyer et al., 2022), which cannot provide interpretation towards concepts' granularity.

To this end, we propose Concept2Box, a novel two-view KG embedding model for jointly learning the representations of concepts and instances using dual geometric objects. Specifically, we propose to leverage the geometric objects of boxes (i.e. axis-aligned hyperrectangle) to embed concept nodes in the ontology-view KG. This is motivated by the desirable geometric properties of boxes to represent more complex relationships among concepts such as intersect, contain, and disjoint, and also capture the hierarchy structure. For example, in Figure 1 we can see *City* is contained within *Location*, denoting it's a sub-concept of *Location*; *Artist* and *Scientist* intersects with each other, denoting they are correlated. In addition, box embeddings provide probabilistic measures of concepts' granularity based on the volumes, such as *Person* is a larger box than *Scientist*. Entities in the instance-view KG are modeled with classic vector representations to capture their specific meanings. As we model concepts and entities using boxes and vectors respectively, we further design a new metric function to measure the distance from a vector to a box in order to bridge the semantics of the two views. Based on this vector-to-box distance function, the proposed model is trained to jointly model the semantics of the instance-view KG, the ontology-view KG, and the cross-view links. Empirically, Concept2Box outperforms popular baselines on both public and our newly-created industrial recipe datasets.

Our contributions are as follows: (1) We propose a novel model for learning two-view KG representations by jointly embedding concepts and instances with different geometric objects. (2) We design a new metric function to measure the distance between concept boxes and entity vectors to bridge the two views. (3) We construct a new industrial-level recipe-related KG dataset. (4) Extensive experiments verify the effectiveness of Concept2Box.

2 Related Work

2.1 Knowledge Graph Embedding (KGE)

The majority of work on knowledge graph embeddings only considers a single view, where models learn the latent vector representations for nodes and relations and measure triple plausibility via varying score functions. Representatives include translation-based TransE (Bordes et al., 2013), TransH (Wang et al., 2014); rotation-based RotatE (Sun et al., 2019), and neural network-based ConvE (Dettmers et al., 2018). Most recently, some work employs graph neural networks (GNNs) to aggregate neighborhood information in KGs (Huang et al., 2022; Zhang et al., 2020), which have shown superior performance on graph-structured data by passing local message (Huang et al., 2020, 2021; Javari et al., 2020; Wang et al., 2021a). For work that pushes two-view or multiple KGs together (Hao et al., 2019; Huang et al., 2022), they either model all nodes as points in the Euclidean space (Hao et al., 2019; Huang et al., 2022) and in the product manifold space (Wang et al., 2021c), or separately embed entities and concepts as points in two spaces respectively (Iyer et al., 2022). Our work uses boxes and vectors for embedding concepts and entities respectively, which not only gives better performance but also provides a probabilistic explanation of concepts' granularity based on box volumes.

2.2 Geometric Representation Learning

Representing elements with more complex geometric objects than vectors is an active area of study (Chen et al., 2021). Some representative models include Poincaré embeddings (Nickel and Kiela, 2017), which embeds data into the hyperbolic space with the inductive basis of negative curvature. Hyperbolic entailment cones (Ganea et al., 2018) combine order embeddings with hyperbolic geometry, where relations are viewed as partial orders defined over nested geodesically convex cones. Elliptical distributions (Muzellec and Cuturi, 2018) model nodes as distributions and define distances based on the 2-Wasserstein metric. While they are promising for embedding hierarchical structures, they do not provide interpretable probabilistic semantics. Box embeddings not only demonstrate improved performance on modeling hierarchies but also model probabilities via box volumes to explain the granularity of concepts.

3 Preliminaries

3.1 Problem Formulation

We consider a two-view knowledge graph $\mathcal{G}$ consisting of a set of entities $\mathcal{E}$, a set of concepts $\mathcal{C}$, and a set of relations $\mathcal{R} = \mathcal{R}^{\mathcal{I}} \cup \mathcal{R}^{\mathcal{O}} \cup \mathcal{S}$, where $\mathcal{R}^{\mathcal{I}}, \mathcal{R}^{\mathcal{O}}$ are the entity-entity and concept-concept relation sets respectively. $\mathcal{S} = \{(e, c)\}$ is the cross-view entity-concept relation set with $e \in \mathcal{E}, c \in \mathcal{C}$. Entities and concepts are disjoint sets, i.e., $\mathcal{E} \cap \mathcal{C} = \emptyset$.

We denote the instance-view KG as $\mathcal{G}^{\mathcal{I}} := \{\mathcal{E}, \mathcal{R}^{\mathcal{I}}, \mathcal{T}^{\mathcal{I}}\}$ where $\mathcal{T}^{\mathcal{I}} = \{(h^{\mathcal{I}}, r^{\mathcal{I}}, t^{\mathcal{I}})\}$ are factual triples connecting head and tail entities $h^{\mathcal{I}}, t^{\mathcal{I}} \in \mathcal{E}$ with a relation $r^{\mathcal{I}} \in \mathcal{R}^{\mathcal{I}}$. The instance-view triples can be any real-world instance associations and are often of a flat structure such as *("Taylor Swift", "has_award", "Grammy Award")*. Similarly, the ontology-view KG is denoted as $\mathcal{G}^{\mathcal{O}} := \{\mathcal{C}, \mathcal{R}^{\mathcal{O}}, \mathcal{T}^{\mathcal{O}}\}$ where $\mathcal{T}^{\mathcal{O}} = \{(h^{\mathcal{O}}, r^{\mathcal{O}}, t^{\mathcal{O}})\}$ are triples connecting $h^{\mathcal{O}}, t^{\mathcal{O}} \in \mathcal{C}$ and $r^{\mathcal{O}} \in \mathcal{R}^{\mathcal{O}}$. The ontology-view triples describe the hierarchical structure among abstract concepts and their meta-relations such as *("Singer", "is_a", "Artist")*.

Given: a two-view KG $\mathcal{G}$, we want to **learn** a model which effectively embeds entities, concepts, and relations to latent representations with different geometric objects to better capture the structural differences of two views. We evaluate the quality of the learned embeddings on the KG completion task and concept linking task, described in Section 5.

3.2 Probabilistic Box Embedding

Box embeddings represent an element as a hyperrectangle characterized with two parameters (Vilnis et al., 2018): the minimum and the maximum corners $(\boldsymbol{x_m}, \boldsymbol{x_M}) \in \mathbb{R}^d$ where $x_m^i < x_M^i$ for each coordinates $i \in 1, 2, \cdots d$. Box volume is computed as the product of side-lengths $\text{Vol}(x) = \prod_{i=1}^{d} \left(x_M^i - x_m^i\right)$. Given a box space $\Omega_{\text{Box}} \subseteq \mathbb{R}^d$, the set of boxes $\mathcal{B}\left(\Omega_{\text{Box}}\right)$ is closed under intersection (Chen et al., 2021) and box volumes can be viewed as the unnormalized probabilities to interpret concepts' granularity. The intersection volume of two boxes is defined as $\text{Vol}(x \cap y) = \prod_i \max\left(\min\left(x_M^i, y_M^i\right) - \max\left(x_m^i, y_m^i\right), 0\right)$ and the conditional probability is computed as $P(x|y) = \frac{\text{Vol}(x \cap y)}{\text{Vol}(y)}$, with value between 0 and 1 (Vilnis et al., 2018).

Directly computing the condition probabilities would pose difficulties during training. This can be caused by a variety of settings like when two boxes are disjoint but should overlap (Chen et al., 2021), there would be no gradient for some parameters. To alleviate this issue, we rely on Gumbel boxes (Dasgupta et al., 2020), where the min and max corners of boxes are modeled via Gumbel distributions:

$$\begin{aligned} \text{Vol}(x) &= \prod_{i=1}^{d} \left[x_m^i, x_M^i\right] \quad \text{where} \\ x_m^i &\sim \text{GumbelMax}\left(\mu_m^i, \beta\right), \\ x_M^i &\sim \text{GumbelMin}\left(\mu_M^i, \beta\right), \end{aligned} \tag{1}$$

μ_m^i, μ_M^i are the minimum and maximum coordinates parameters; β is the global variance. Gumbel distribution provides min/max stability (Chen et al., 2021) and the expected volume is approximated as

$$\mathbb{E}[\text{Vol}(x)] \approx \prod_{i=1}^{d} \beta \log\left(1 + \exp\left(\frac{\mu_M^i - \mu_m^i}{\beta} - 2\gamma\right)\right), \tag{2}$$

where γ is the Euler–Mascheroni constant (Onoe et al., 2021). The conditional probability becomes

$$\mathbb{E}\left[P(x|y)\right] \approx \frac{\mathbb{E}[\text{Vol}(x \cap y)]}{\mathbb{E}[\text{Vol}(y)]}. \tag{3}$$

This also leads to improved learning as the noise ensembles over a large collection of boxes which allows the model to escape plateaus in the loss function (Chen et al., 2020; Onoe et al., 2021). We use this method when training box embeddings.

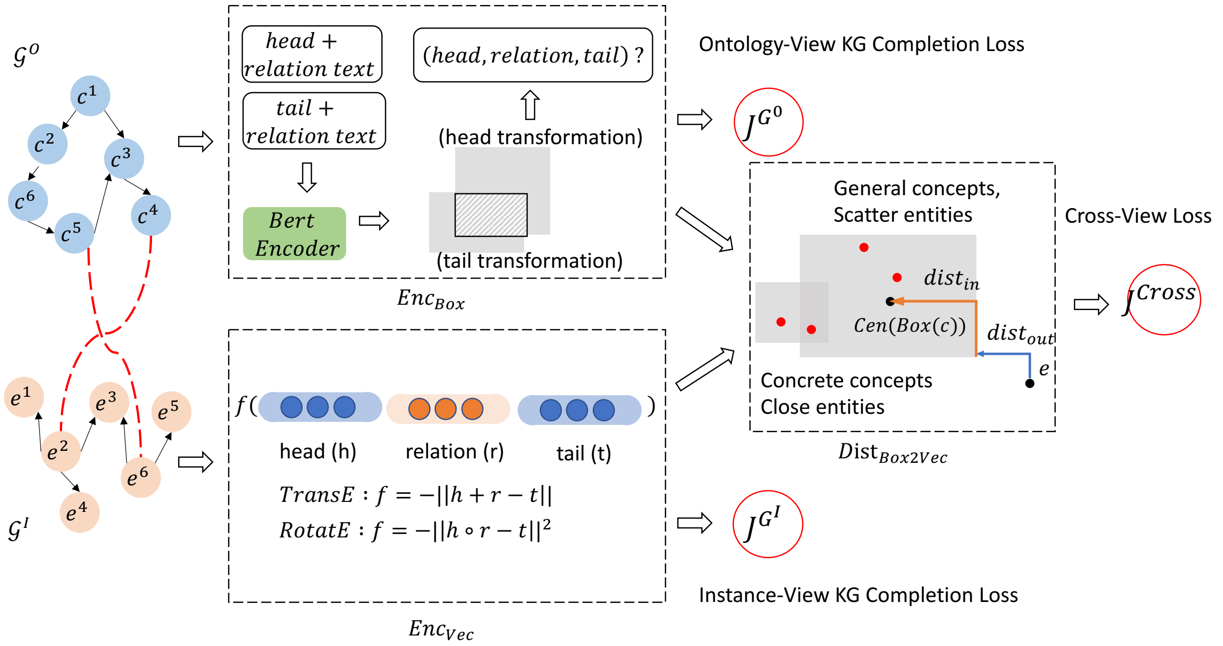

Figure 2: The overall framework of our proposed Concept2Box includes 3 modules. First, we employ probabilistic box embeddings to model the ontology-view KG, capturing the hierarchical structure and preserving the granularity of concepts (upper). Second, we model the instance-view KG by applying the vector-based KG embedding method (bottom). Third, to bridge these two views, we design a novel distance metric defined from a box to a vector (right). The model is learned by jointly optimizing three loss terms of each corresponding module.

4 Method

In this section, we introduce Concept2Box, a novel KGE model to learn two-view knowledge graph embeddings with dual geometric representations. As shown in Figure 2, Concept2Box consists of three main components: (1). The ontology-view box-based KG embedding module that captures the hierarchical structure and complex relations of concepts trained via the KG completion loss over $\mathcal{G}^{\mathcal{O}}$; (2). The instance-view vector-based KG embedding module trained via the KG completion loss over $\mathcal{G}^{\mathcal{I}}$. It has the flexibility to incorporate any off-the-shelf vector-based KGE methods (Bordes et al., 2013; Sun et al., 2019); (3). The cross-view module trained via the concept linking loss over $\mathcal{S}$. This module relies on a new distance metric for bridging the semantics between a vector and a box. We now present each component in detail.

4.1 Ontology-View Box Embedding

For the ontology-view KG, we model each concept as a Gumbel box to capture the hierarchical structure and interpret concept granularity based on the box volume. Specifically, each Gumbel box x is parameterized using a center $\mathrm{cen}(x) \in \mathbb{R}^d$ and offset $\mathrm{off}(x) \in \mathbb{R}^d_+$, and the minimum and maximum location parameters are given by $\boldsymbol{\mu}_{\mathbf{m}} = \mathrm{cen}(x) - \mathrm{off}(x)$, $\boldsymbol{\mu}_{\mathbf{M}} = \mathrm{cen}(x) + \mathrm{off}(x)$. To compute the triple score from box embeddings, we use the approximate conditional probability of observing a triple given its relation and tail, as shown

$$\phi(h^{\mathcal{O}}, r^{\mathcal{O}}, t^{\mathcal{O}}) = \frac{\mathbb{E}\big[\mathrm{Vol}(f_{r^{\mathcal{O}}}(\mathrm{Box}(h^{\mathcal{O}})) \cap f_{r^{\mathcal{O}}}(\mathrm{Box}(t^{\mathcal{O}})))\big]}{\mathbb{E}\big[\mathrm{Vol}(f_{r^{\mathcal{O}}}(\mathrm{Box}(t^{\mathcal{O}})))\big]}, \tag{4}$$

where $f_{r^{\mathcal{O}}}$ denotes the box transformation function to account for the influence of the relation $r^{\mathcal{O}}$. As shown in Figure 1, $f_{lives\_in}(Artist)$, $f_{is\_a}(Artist)$ are the transformed box embeddings of the concept *Artist* in the context of the living locations and supertypes. The bounded score ϕ should be close to 1 for positive triples and 0 for negative ones.

Previous work Chen et al. (2021) defined the transformation function f_r as shifting the original center and offset of the input box x: $\mathrm{cen}(f_r(x)) = \mathrm{cen}(x) + \boldsymbol{\tau}^{\boldsymbol{r}}$, $\mathrm{off}(f_r(x)) = \mathrm{off}(x) \circ \boldsymbol{\Delta}^{\boldsymbol{r}}$, where $\circ$ is the Hadamard product and $(\boldsymbol{\tau}^{\boldsymbol{r}}, \boldsymbol{\Delta}^{\boldsymbol{r}})$ are relation-specific parameters. To further incorporate semantic information from the text of a concept c and relation r, we propose a new transformation function as:

$$\begin{aligned}&\mathbf{h}^{[\mathrm{CLS}]} = \mathrm{BERT}\left(c_{\text{text}} + \text{'SEP'} + r_{\text{text}}\right)\\&\mathrm{cen}\left(f_r(\mathrm{Box}(c))\right), \mathrm{off}\left(f_r(\mathrm{Box}(c))\right) = f_{\text{proj}}(\mathbf{h}^{[\mathrm{CLS}]}).\end{aligned} \tag{5}$$

Specifically, we first concatenate the textual content of the concept and relation for feeding into a pre-trained language model BERT (Devlin et al., 2019) that captures the rich semantics of the text. We obtain the embedding by taking the hidden vector at the [CLS] token. Then, we employ a projection function f_{proj} to map the hidden vector $\mathbf{h}^{[\text{CLS}]} \in \mathbb{R}^{\ell}$ to the $\mathbb{R}^{2d}$ space, where ℓ is the hidden dimension of the BERT encoder, and d is the dimension of the box space. In practice, we implement f_{proj} as a simple two-layer Multi-Layer Perception (MLP) with a nonlinear activation function. We further split the hidden vector into two equal-length vectors as the center and offset of the transformed box.

Remark 1. Instead of encoding relation text first to get the shifting and scaling parameters and then conducting the box transformation as introduced in (Chen et al., 2020), we feed the concept and relation text jointly to the BERT encoder to allow interaction between them through the attention mechanism. The goal is to preserve both concept and relation semantics by understanding their textual information as a whole.

We train the ontology-view KG embedding module with the binary cross entropy loss as below:

$$\mathcal{J}^{\mathcal{G}^{\mathcal{C}}} = \sum_{\substack{(h^{\mathcal{O}},r^{\mathcal{O}},t^{\mathcal{O}})\in\mathcal{T}^{\mathcal{O}},\\ (\widetilde{h}^{\mathcal{O}},r^{\mathcal{O}},\widetilde{t}^{\mathcal{O}})\notin\mathcal{T}^{\mathcal{O}}}} |\phi(h^{\mathcal{O}},r^{\mathcal{O}},t^{\mathcal{O}})-1|^2 + \ |\phi\left(\widetilde{h}^{\mathcal{O}},r^{\mathcal{O}},\widetilde{t}^{\mathcal{O}}\right)|^2, \qquad (6)$$

where $(\widetilde{h}^{\mathcal{O}},r^{\mathcal{O}},\widetilde{t}^{\mathcal{O}})$ is a negative sampled triple obtained by replacing head or tail of the true triple (h^{O},r^{O},t^{O}). We do not use the hinge loss defined in vector-based KGE models (Bordes et al., 2013), as the triple scores are computed from the conditional probability and strictly fall within 0 and 1.

4.2 Instance-View and Cross-View Modeling

Next, we introduce learning the instance-view KG and the cross-view module for bridging two views.

Instance-View KG Modeling. As entities in the instance-view KG are fine-grained, we choose to model entities and relations as classic point vectors. We employ the standard hinge loss defined as follows to train the instance-view KGE module.

$$\mathcal{J}^{\mathcal{G}^{\mathcal{I}}} = \sum_{\substack{(h^{\mathcal{I}},r^{\mathcal{I}},t^{\mathcal{I}})\in\mathcal{T}^{\mathcal{I}}\\ (\widetilde{h}^{\mathcal{I}},r^{\mathcal{I}},\widetilde{t}^{\mathcal{I}})\notin\mathcal{T}^{\mathcal{I}}}} \begin{bmatrix} f_{\text{vec}}\left(h^{\mathcal{I}},r^{\mathcal{I}},t^{\mathcal{I}}\right) \\ -f_{\text{vec}}\left(\widetilde{h}^{\mathcal{I}},r^{\mathcal{I}},\widetilde{t}^{\mathcal{I}}\right) \\ +\gamma^{\text{KG}} \end{bmatrix}_{+}, \qquad (7)$$

where $\gamma^{\text{KG}} > 0$ is a positive margin, f_{vec} is an arbitrary vector-based KGE model such as TransE (Bordes et al., 2013) and RotatE (Sun et al., 2019). And $(\widetilde{h}^{\mathcal{I}},r^{\mathcal{I}},\widetilde{t}^{\mathcal{I}})$ is a sampled negative triple.

Cross-View KG Modeling. The goal of the cross-view module is to bridge the semantics between the entity embeddings and the concept embeddings by using the cross-view links. It forces any entity $e \in \mathcal{E}$ to be close to its corresponding concept $c \in \mathcal{C}$. As we model concepts and entities with boxes and vectors respectively, existing distance measures between boxes (or between vectors) would be inappropriate for measuring the distance between them. To tackle this challenge, we now define a new metric function to measure the distance from a vector to a box. Our model is trained to minimize the distances for the concept-entity pairs in the cross-view links based on this metric function. Specifically, given a concept c and an entity e, if we denote the minimum and maximum location parameters of the concept box as $\boldsymbol{\mu_m}, \boldsymbol{\mu_M}$, and the vector for the entity as $\boldsymbol{e}$, we define the distance function f_d as:

$$\begin{aligned} &f_d(e,c) = \text{dist}_{\text{out}}(e,c) + \alpha \cdot \text{dist}_{\text{in}}(e,c) \\ &\text{dist}_{\text{out}}(e,c) = \|\text{Max}\left(\boldsymbol{e}-\boldsymbol{\mu_M},\mathbf{0}\right) + \text{Max}\left(\boldsymbol{\mu_m}-\boldsymbol{e},\mathbf{0}\right)\|_1 \\ &\text{dist}_{\text{in}}(e,c) = \|\text{Cen}(\text{Box}(c)) - \text{Min}\left(\boldsymbol{\mu_M},\text{Max}\left(\boldsymbol{\mu_m},\boldsymbol{e}\right)\right)\|_1 \end{aligned} \qquad (8)$$

where dist_{out} corresponds to the distance between the entity and closest corner of the box; dist_{in} corresponds to the distance between the center of the box and its side or the entity itself if the entity is inside the box (Ren et al., 2020). $0 < \alpha < 1$ is a fixed scalar to balance the outside and inside distances, with the goal to punish more if the entity falls outside of the concept, and less if it is within the box but not close to the box center. An example is shown in Figure 2. The proposed vector-to-box distance metric is nonnegative and commutative.

However, there is one drawback to the above formulation: different boxes should have customized volumes reflecting their granularity. A larger box may be a more general concept (e.g., *Person*), so it is suboptimal for all paired entities to be as close as to its center, in contrast to more concrete concepts like *Singer*, as illustrated in Figure 2. Moreover, for instances that belong to multiple concepts, roughly pushing them to be equally close to the centers of relevant concepts can result in poor embedding results. Consider we have two recipe concepts *KungPao chicken* and *Curry chicken*, suppose *KungPao Chicken* is more complicated and requires more ingredients. If we roughly push cooking wine, salt, and chicken breast these shared ingredient instances to be as close as to both of their centers, it will possibly result in forcing the centers and volume of the two recipes to be very similar to each other. Therefore, we propose to associate the balancing scalar α to be related to each concept box volume. Specifically, we define $\alpha(x)$ with regard to a box x as follows:

$$\alpha(x) = \alpha \cdot \text{Sigmoid}(\frac{1}{\text{Vol}(x)}). \tag{9}$$

We multiply the original constant coefficient α with a box-specific value negatively proportional to its volume. The idea is to force entities to be closer to the box center when the box has a smaller volume.

The cross-view module is trained by minimizing the following loss with negative sampling:

$$\mathcal{J}^{\text{Cross}} = \sum_{\substack{(e,c)\in\mathcal{S} \\ (e\prime,c\prime)\notin\mathcal{S}}} \left[\sigma(f_d(e,c)) - \sigma\left(f_d\left(e',c'\right)\right) + \gamma^{\text{Cross}}\right], \tag{10}$$

where γ^{Cross} represents a fixed scalar margin.

4.3 Overall Training Objective

Now that we have presented all model components, the overall loss function is a linear combination of the instance-view and ontology-view KG completion loss, and the cross-view loss, as shown below:

$$\mathcal{J} = \mathcal{J}^{\mathcal{G}^{\mathcal{O}}} + \lambda_1 \mathcal{J}^{\mathcal{G}^{\mathcal{I}}} + \lambda_2 \mathcal{J}^{\text{Cross}}, \tag{11}$$

where $\lambda_1, \lambda_2 > 0$ are two positive hyperparameters to balance the scale of the three loss terms.

4.4 Time Complexity Analysis

The time complexity of Concept2Box is $\mathcal{O}(3d|\mathcal{T}^{\mathcal{O}}| + d|\mathcal{T}^{\mathcal{I}}| + 5d|\mathcal{S}|)$, where $\mathcal{T}^{\mathcal{O}}, \mathcal{T}^{\mathcal{I}}, \mathcal{S}$ are the ontology-view KG triples, instance-view KG triples, and cross-view set defined in Sec 3.1. Here d is the embedding dimension of vectors. The time complexity of other vector-based methods is $\mathcal{O}(d|\mathcal{T}^{\mathcal{O}}| + d|\mathcal{T}^{\mathcal{I}}| + d|\mathcal{S}|)$ which is asymptotically the same as Concept2Box.

5 Experiments

In this section, we introduce our empirical results against existing baselines to evaluate our model performance. We start with the datasets and baselines introduction and then conduct the performance evaluation. Finally, we conduct a model generalization experiment and a case study to better illustrate our model design.

5.1 Dataset and Baselines

We conducted experiments over two datasets: one public dataset from DBpedia (Hao et al., 2019), which describes general concepts and fine-grained entities crawled from DBpedia. We also created a new recipe-related dataset, where concepts are recipes, general ingredient and utensil names, etc, and the entities are specific products for cooking each recipe searched via Amazon.com, along with some selected attributes such as brand. The detailed data structure can be found in Appendix A. The statistics of the two datasets are shown in Table 1.

We compare against both single-view KGE models and two-view KGE models as follows:

•**Vector-based KGE models:**: Single-view vector-based KGE models that represent triples as vectors, including TransE (Bordes et al., 2013), RotatE (Sun et al., 2019),DistMult (Yang et al., 2015), and ComplexE (Trouillon et al., 2016).
• **KGBert** (Yao et al., 2020): Single-view KG embedding method which employs BERT to incorporate text information of triples.
• **Box4ET** (Onoe et al., 2021): Single-view KGE model that represents triples using boxes.
•**Hyperbolic GCN** (Chami et al., 2019): Single-view KGE method that represents triples in the hyperbolic space.
•**JOIE** (Hao et al., 2019): Two-view KGE method that embeds entities and concepts all as vectors.

5.2 KG Completion Performance

The KG completion (KGC) task is to predict missing triples based on the learned embeddings, which helps to facilitate the knowledge learning process in KG construction. Without loss of generality, we discuss the case of predicting missing tails, which we also refer to as a query $q = (h, r, ?t)$ (Huang et al., 2022; Chen et al., 2021). We conduct the KGC task for both views against baselines.

Evaluation Protocol. We evaluate the KGC performance using mean reciprocal ranks (MRR), accuracy (Hits@1) and the proportion of correct answers ranked within the top 10 (Hits@10). All three metrics are preferred to be higher, so as to indicate better KGC performance. For both $\mathcal{G}^{\mathcal{I}}$ and $\mathcal{G}^{\mathcal{O}}$, we randomly split the triples into three parts: 80% for training, 10% for validation and 10% for testing. During model training and testing, we utilize all cross-view links $\mathcal{S}$.

Results. The evaluation results are shown in Table 2. Firstly, by comparing the average per-

Dataset	Instance-View KG			Ontology-View KG			Cross-View Links
	# Entities	# Relations	# Triples	# Concepts	# Relations	# Triples	# Links
DBpedia	111,762	305	863,643	174	20	763	99,748
Recipe	21,457	9	200,288	32,922	5	445,632	11,474

Table 1: Statistics of the DBpedia and Recipe datasets.

	Recipe						DBpedia					
	$\mathcal{G}^{\mathcal{I}}$ KG Completion			$\mathcal{G}^{\mathcal{O}}$ KG Completion			$\mathcal{G}^{\mathcal{I}}$ KG Completion			$\mathcal{G}^{\mathcal{O}}$KG Completion		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
TransE	0.28	26.31	38.59	0.18	11.24	30.67	0.32	22.70	48.12	0.54	47.90	61.84
RotatE	0.30	28.31	40.01	0.22	15.68	35.47	0.36	29.31	54.60	0.56	49.16	68.19
DistMult	0.29	27.33	39.04	0.18	12.37	32.04	0.28	27.24	29.70	0.50	45.52	64.73
ComplexE	0.27	26.58	37.99	0.19	14.76	34.02	0.32	27.39	46.63	0.55	47.80	62.23
JOIE	0.41	29.73	60.14	0.36	22.58	53.62	0.48	35.21	72.38	0.60	52.48	79.71
Hyperbolic GCN	0.36	28.58	52.48	0.25	16.04	40.35	0.30	33.68	46.72	0.54	47.59	62.11
KG-Bert	0.30	29.92	51.32	0.24	19.04	38.78	0.39	31.10	60.46	0.63	55.69	81.07
Box4ET	0.42	28.81	59.88	0.35	23.01	53.79	0.42	33.69	68.12	0.64	55.89	81.45
Concept2Box	**0.44**	**30.33**	**62.01**	**0.37**	**23.16**	**54.92**	**0.51**	**36.52**	**73.11**	**0.65**	**56.82**	**83.01**

Table 2: Results of KG completion task. Best results are bold-faced

formance between single-view and two-view KG models, we can observe that two-view methods are able to achieve better results for the KGC task. This indicates that the intuition behind modeling two-view KGs to conduct KG completion is indeed beneficial. Secondly, by comparing vector-based methods with other geometric representation-based methods such as Hyperbolic GCN, there is a performance improvement on most metrics, indicating that simple vector representations have its limitation, especially in capturing the hierarchical structure among concepts (shown by the larger performance gap in $\mathcal{G}^{\mathcal{O}}$). Our Concept2Box is able to achieve higher performance in most cases, verifying the effectiveness of our design.

5.3 Concept Linking Performance

The concept linking task predicts the associating concepts of given entities, where each entity can be potentially mapped to multiple concepts. It tests the quality of the learned embeddings.

Evaluation Protocol. We split the cross-view links $\mathcal{S}$ into 80%, 10%, 10% for training, validation, testing respectively. We utilize all the triples in both views for training and testing. For each concept-entity pair, we rank all concept candidates for the given entity based on the distance metric from a box to a vector introduced in Sec 4.2, and report MRR, Hits@1 and Hits@3 for evaluation.

Results. The evaluation results are shown in Table 3. We can see that Concept2Box is able to achieve the highest performance across most metrics, indicating its effectiveness. By comparing with Box4ET where both entities and concepts are modeled as boxes, our Concept2Box performs better, which indicates our intuition that entities and concepts are indeed two fundamentally different types of nodes and should be modeled with different geometric objects. Finally, we observe that Box4ET, and Concept2Box are able to surpass methods that are not using box embeddings, which shows the advantage of box embeddings in learning the hierarchical structure and complex behaviors of concepts with varying granularity.

Datasets	Recipe			DBPedia		
Metrics	MRR	Acc.	Hit@3	MRR	Acc.	Hit@3
TransE	0.23	21.36	30.68	0.53	43.67	60.78
RotatE	0.25	23.01	33.34	0.72	61.48	75.67
DistMult	0.24	22.34	31.05	0.55	49.83	68.01
ComplexE	0.22	22.50	31.45	0.58	55.07	70.17
JOIE	0.57	54.58	66.39	0.85	75.58	96.02
Hyper. GCN	0.60	53.49	67.03	0.86	76.11	96.50
Box4ET	0.59	54.36	68.88	**0.88**	77.04	97.38
Concept2Box	**0.61**	**55.33**	**69.01**	0.87	**78.09**	**97.44**

Table 3: Results of concept linking task.

Abaltion Studies. To evaluate the effectiveness of our model design, we conduct an ablation study on the DBpedia dataset as shown in Table 4. First, to show the effect of joint modeling the text semantics of relations and concepts for generating the relation-transformed box embeddings as in Eqn 5, we compare with separately modeling relations as box shifting and scaling introduced in (Chen et al., 2021). The latter one gives poorer results, indicating the effectiveness of our proposed box encoder.

Next, we study the effect of the volume-based balancing scalar defined in Eqn 9. Compared with a fixed balancing scalar for all boxes, Concept2Box is able to generate better results.

	$\mathcal{G}^{\mathcal{I}}$ KG Completion			$\mathcal{G}^{\mathcal{O}}$ KG Completion			Concept Linking		
	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@3
Concept2Box	**0.506**	**36.52**	**73.11**	**0.644**	**56.82**	**83.01**	**0.874**	**78.09**	**97.44**
- relation shift&scale	0.483	34.99	72.18	0.631	55.34	81.97	0.865	77.64	96.34
- fixed distance coefficient	0.479	35.13	71.88	0.635	54.98	82.33	0.854	76.91	96.13

Table 4: Results of ablation study on DBpedia.

5.4 Model Generalization

We test our model's generalization ability by training our model on the Recipe dataset and evaluating it on the concept linking task with recipe-product pairs collected from other online resources. Specifically, in the evaluation data, all products are known entities in the Recipe dataset during training, where we can directly obtain their embeddings. For recipes, only 3% of them are observed, and for unseen recipes, we generate their box embeddings by sending their title as input to the box encoder. We divide these recipe-product pairs into two groups based on whether the recipes are known. We use the first group with known recipes to test the generalization ability in the transductive setting: When test data follows a different distribution as opposed to the training data, how does the model perform? It differs from Sec 5.3 in that the test data share the same distribution with the training data as we randomly split them from the same dataset. We use the second group with unknown recipes to test in the inductive setting: When unknown concepts are sent as input, how does the model perform?

Diversity-Aware Evaluation. To verify our learned embeddings can capture complex relations and hierarchy structure among concepts, we use diversity-aware evaluation for the generalization experiment similar to that in (Hao et al., 2020). The idea is that for each recipe, a more preferred ranking list is one consisting of products from multiple necessary ingredients, instead of the one with products all from one ingredient. Since we model recipes and ingredients as concepts, and products as entities, for each recipe, we first extract its top K ingredients (types) from $\mathcal{G}^{\mathcal{O}}$, and then for each of the K ingredient we extract the top M products (items). We set $M \times K = 120$ and let $K = 10, 12, 24$.

	Known Recipes			Unknown Recipes		
# types * # items	10*12	12*10	24*5	10*12	12*10	24*5
JOIE	68.33	69.17	68.45	57.34	57.56	56.38
Concept2Box	**71.97**	**73.33**	**69.43**	**61.32**	**66.77**	**59.68**

Table 5: H@120 with different number of item types.

Results. Table 5 shows the results of H@120 among two-view KGE methods. We can see that Concept2Box is able to achieve the best across different settings, showing its great generalization ability. Note that when changing the number of types, Concept2Box is able to bring more performance gain than JOIE. This can be understood as the hierarchy structure of concepts is well captured by the box embeddings, when properly selecting the number of types, we first narrow down the concept (recipe) to the set of related concepts (ingredients) for a better understanding, which will generate better results.

5.5 Case Study

We conduct a case study to investigate the semantic meaning of learned box embeddings of concepts. We select recipe concepts and show the top 3 concepts that have the largest intersection with them after the relation transformation of "using ingredients". As shown in Figure 3, the dominant ingredients for each recipe are at the top of the list, verifying the effectiveness of Concept2Box. Also, we found that *Egyptian lentil Soup* has a larger intersection with *Black Rice Pudding* than with *Thai Iced Tea*, since the first two are dishes with more similarities and the third one is beverage. We also examine the box volume of concepts which reflects their granularity. Among ingredient concepts, we observe that Water, Sugar, and Rice are of the largest volumes, indicating they are common ingredients shared across recipes. This observation confirms that the box volume effectively represents probabilistic semantics, which we believe to be a reason for the performance improvement.

6 Conclusion

In this paper, we propose *Concept2Box*, a novel two-view knowledge graph embedding method with dual geometric representations. We model high-level concepts as boxes to capture the hierarchical structure and complex relations among them and reflect concept granularity based on box volumes. For the instance-view KG, we model fine-grained entities as vectors and propose a novel metric function to define the distance between a

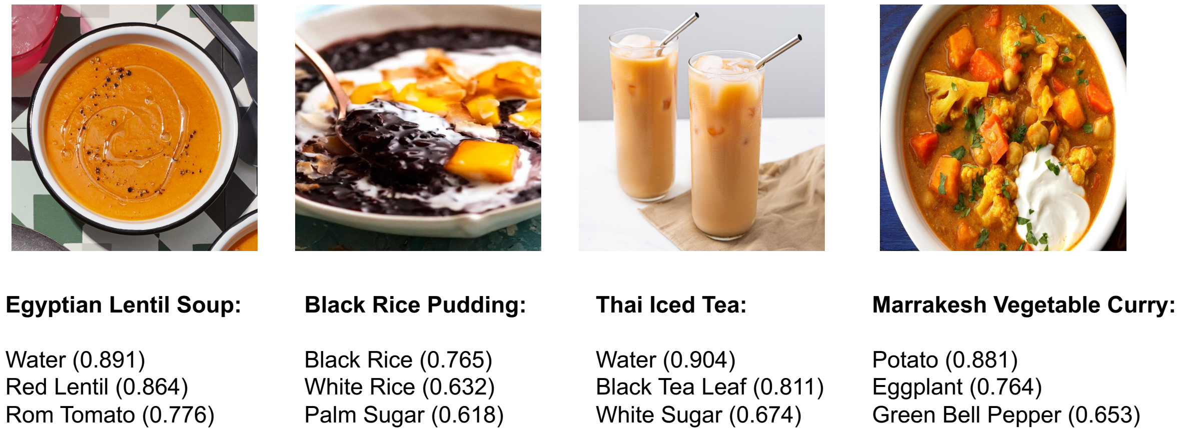

Figure 3: Case Study on the Recipe Dataset.

box to a vector to bridge the semantics of the two views. Concept2Box is jointly trained to model the instance-view KG, the ontology-view KG, and the cross-view links. Empirical experiment results on two real-world datasets including a newly-created recipe dataset verify the effectiveness of Concept2Box.

7 Limitations

Our model is developed to tackle the structural difference between the ontology and instance views of a KG. However, many modern KGs are multilingual, where different portions of a KG may not only differ in structure but also differ in the text semantics. How to jointly capture these differences remain unsolved. Also since box embeddings naturally provide interpretability to the granularity of the learned concepts, how to use the current model to discover unknown concepts from these embeddings is also challenging.

8 Ethical Impact

Our paper proposed Concept2Box, a novel two-view knowledge graph embedding model with dual geometric representations. Concept2Box neither introduces any social/ethical bias to the model nor amplifies any bias in the data. For the created recipe dataset, we did not include any customers'/sellers' personal information and only preserved information related to products' attributes and their relations. Our model implementation is built upon public libraries in Pytorch. Therefore, was e do not foresee any direct social consequences or ethical issues.

9 Acknowledgement

This work was partially supported by NSF 1829071, 2106859, 2200274, 2211557, 1937599, 2119643, 2303037, DARPA #HR00112290103/HR0011260656, NASA, Amazon Research Awards, and an NEC research award.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, page 2787–2795.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 4869–4880.

Xuelu Chen, Michael Boratko, Muhao Chen, Shib Sankar Dasgupta, Xiang Lorraine Li, and Andrew McCallum. 2021. Probabilistic box embeddings for uncertain knowledge graph reasoning. In *Proceedings of the 19th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Xuelu Chen, Muhao Chen, Changjun Fan, Ankith Uppunda, Yizhou Sun, and Carlo Zaniolo. 2020. Multilingual knowledge graph completion via ensemble knowledge transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 3227–3238.

Shib Dasgupta, Michael Boratko, Dongxu Zhang, Luke Vilnis, Xiang Li, and Andrew McCallum. 2020. Improving local identifiability in probabilistic box embeddings. *Advances in Neural Information Processing Systems*, 33:182–192.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Octavian Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic entailment cones for learning hierarchical embeddings. In *International Conference on Machine Learning*, pages 1646–1655. PMLR.

Junheng Hao, Muhao Chen, Wenchao Yu, Yizhou Sun, and Wei Wang. 2019. Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 1709–1719.

Junheng Hao, Tong Zhao, Jin Li, Xin Luna Dong, Christos Faloutsos, Yizhou Sun, and Wei Wang. 2020. P-companion: A principled framework for diversified complementary product recommendation. In *Proceedings of the 29th ACM International Conference on Information Knowledge Management*, CIKM '20, page 2517–2524.

Zijie Huang, Zheng Li, Haoming Jiang, Tianyu Cao, Hanqing Lu, Bing Yin, Karthik Subbian, Yizhou Sun, and Wei Wang. 2022. Multilingual knowledge graph completion with self-supervised adaptive graph alignment. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zijie Huang, Yizhou Sun, and Wei Wang. 2020. Learning continuous system dynamics from irregularly-sampled partial observations. In *Advances in Neural Information Processing Systems*.

Zijie Huang, Yizhou Sun, and Wei Wang. 2021. Coupled graph ode for learning interacting system dynamics. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Roshni G. Iyer, Yunsheng Bai, Wei Wang, and Yizhou Sun. 2022. Dual-geometric space embedding model for two-view knowledge graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 676–686.

Amin Javari, Zhankui He, Zijie Huang, Raj Jeetu, and Kevin Chen-Chuan Chang. 2020. Weakly supervised attention for hashtag recommendation using graph data. In *Proceedings of The Web Conference 2020*, WWW '20.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, and Soren Auer. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. In *Semantic Web*, pages 167–195.

Bill Yuchen Lin, Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Xiang Ren, and William Cohen. 2021. Differentiable open-ended commonsense reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4611–4625.

Boris Muzellec and Marco Cuturi. 2018. Generalizing point embeddings using the wasserstein space of elliptical distributions. NIPS'18, page 10258–10269.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *ACL*.

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

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

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2071–2080. JMLR.org.

Luke Vilnis, Xiang Lorraine Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic embedding of knowledge graphs with box lattice measures. In *ACL*.

Ruijie Wang, Zijie Huang, Shengzhong Liu, Huajie Shao, Dongxin Liu, Jinyang Li, Tianshi Wang, Dachun Sun, Shuochao Yao, and Tarek Abdelzaher. 2021a. Dydiff-vae: A dynamic variational framework for information diffusion prediction. In *SIGIR'21*.

Shen Wang, Xiaokai Wei, Cicero Nogueira Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew Arnold, Bing Xiang, Philip S. Yu, and Isabel F. Cruz. 2021b. Mixed-curvature multi-relational graph neural network for knowledge graph completion. In *Proceedings of the Web Conference 2021*, WWW '21, page 1761–1771.

Shen Wang, Xiaokai Wei, Cicero Nogueira Nogueira dos Santos, Zhiguo Wang, Ramesh Nallapati, Andrew Arnold, Bing Xiang, Philip S Yu, and Isabel F Cruz. 2021c. Mixed-curvature multi-relational graph neural network for knowledge graph completion. In *Proceedings of the Web Conference 2021*, pages 1761–1771.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, 28.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations (ICLR)*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2020. Kg-bert: Bert for knowledge graph completion. *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. In *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Zhao Zhang, Fuzhen Zhuang, Hengshu Zhu, Zhiping Shi, Hui Xiong, and Qing He. 2020. Relational graph neural network with hierarchical attention for knowledge graph completion. *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9612–9619.

A Data Construction

We introduce the data generation process of the Recipe dataset. The overall data structure is depicted in Fig 4. For the ontology-view KG, concepts consist of recipes, ingredients, and utensils for cooking them and their associated types. Dietary styles and regions where each recipe originated from are also considered concepts. We obtain these high-level recipe-related concepts from recipeDB [1] with some data cleaning procedures. Triples in the ontology-view KG describe the connectivity among these concepts such as (*Kungpao Chicken*, *use_ingredient*, *Chicken Breast*), (*Salt*, *is_of_ingredient_type*, *Additive*).

For the instance-view KG, entities are real ingredient and utensil products sold on Amazon.com by searching their corresponding concept names (ingredient and utensil names) as keywords. For each ingredient and utensil name, we select the top 10 matched products as entities in the instance-view KG. Afterward, we manually select 8 representative attributes such as flavor and brand, that are commonly shared by these products as additional entities in the instance-view KG, to provide more knowledge for the embedding learning process. The triples in the instance-view KG describe facts connecting a product to its attributes.

B Implementation Details.

We use Adam (Kingma and Ba, 2014) as the optimizer to train our model and use TransE (Bordes et al., 2013) as the KGE model for learning the instance-view KG, whose margin γ^{KG} is set to be 0.3. The margin for cross-view module γ^{Cross} is set to be 0.15. We set the latent dimensions for instance-view KG as 256, and the box dimensions for ontology-view KG as 512. We use a batch size of 128 and a learning rate $lr = 0.005$ during training.

Instead of directly optimizing $\mathcal{J}$ in Eqn 11, we alternately update $\mathcal{J}^{\mathcal{G}^{\mathcal{O}}}$, $\mathcal{J}^{\mathcal{G}^{\mathcal{I}}}$ and $\mathcal{J}^{\text{Cross}}$ with different learning rate. Specifically, in our implementation, we optimize with $\theta_{new} \leftarrow \theta_{old} - \eta\nabla\mathcal{J}^{\mathcal{G}^{\mathcal{O}}}$, $\theta_{new} \leftarrow \theta_{old} - (\lambda_1\eta)\nabla\mathcal{J}^{\mathcal{G}^{\mathcal{I}}}$, $\theta_{new} \leftarrow \theta_{old} - (\lambda_2\eta)\nabla\mathcal{J}^{\text{cross}}$ in consecutive steps within one epoch, where θ_{new} denotes our model parameters. The procedure is depicted in Algorithm 1.

[1] https://cosylab.iiitd.edu.in/recipedb/

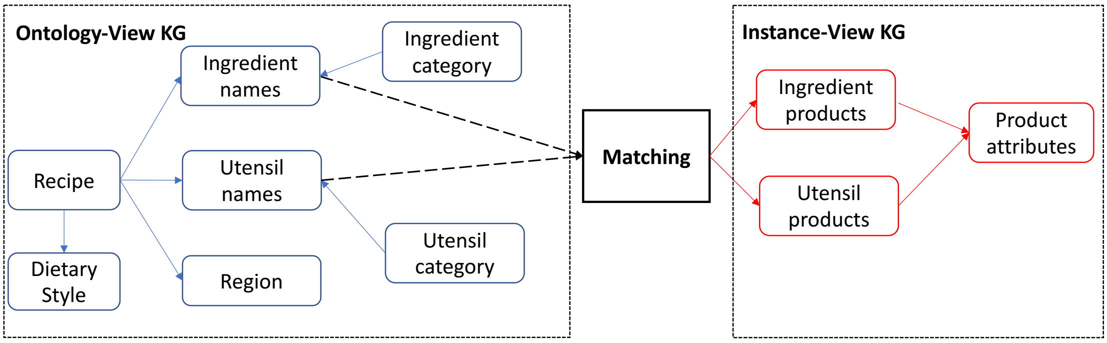

Figure 4: Recipe Dataset Structure

Algorithm 1: Concept2Box training procedure.

```
Input: A two-view KG $\mathcal{G}$. Output: Model parameters $\theta$.
while model not converged do
    //For the KGC loss of $\mathcal{G}^{\mathcal{O}}$:
    Optimize the box-based KGC loss in Eqn 6:
    $\theta_{new} \leftarrow \theta_{old} - \eta \nabla \mathcal{J}^{\mathcal{G}^{\mathcal{O}}}$
    //For the KGC loss of $\mathcal{G}^{\mathcal{I}}$:
    Optimize the vector-based KGC loss in Eqn 7:
    $\theta_{new} \leftarrow \theta_{old} - (\lambda_1 \eta) \nabla \mathcal{J}^{\mathcal{G}^{\mathcal{I}}}$
    //For the cross-view loss:
    Optimize with the cross-view loss in Eqn 10:
    $\theta_{new} \leftarrow \theta_{old} - (\lambda_2 \eta) \nabla \mathcal{J}^{\text{cross}}$
end
```

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7.

☒ A2. Did you discuss any potential risks of your work?
No risk concerns for this work.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Sec.1 for the introduction.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
No response.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.