SConE: Simplified Cone Embeddings with Symbolic Operators for Complex Logical Queries

Chau Duc Minh Nguyen and **Tim French** and **Wei Liu** and **Michael Stewart**
ARC Centre for Transforming Maintenance through Data Science,
Department of Computer Science and Software Engineering
School of Physics, Mathematics and Computing
The University of Western Australia
chau.nguyenducminh@research.uwa.edu.au
{tim.french, wei.liu, michael.stewart}@uwa.edu.au

Abstract

Geometric representation of query embeddings (using points, particles, rectangles and cones) can effectively achieve the task of answering complex logical queries expressed in first-order logic (FOL) form over knowledge graphs, allowing intuitive encodings. However, current geometric-based methods depend on the neural approach to model FOL operators (conjunction, disjunction and negation), which are not easily explainable with considerable computation cost. We overcome this challenge by introducing a symbolic modeling approach for the FOL operators, emphasizing the direct calculation of the intersection between geometric shapes, particularly sector-cones in the embedding space, to model the conjunction operator. This approach reduces the computation cost as a non-neural approach is involved in the core logic operators. Moreover, we propose to accelerate the learning in the relation projection operator using the neural approach to emphasize the essential role of this operator in all query structures. Although empirical evidence for explainability is challenging, our approach demonstrates a significant improvement in answering complex logical queries (both non-negative and negative FOL forms) over previous geometric-based models.

1 Introduction

Answering complex logical queries is a fundamental task of knowledge graphs (KGs) (Bollacker et al., 2008; Vrandečić and Krötzsch, 2014; Speer et al., 2017; Fellbaum, 2010; Lehmann et al., 2015; Mitchell et al., 2018) for various purposes of individuals and businesses. Conventional methods, such as Hartig and Heese (2007); Schmidt et al. (2010), have been well-studied on complete KGs. However, these methods face challenges in incomplete and largely-scaled KGs, as conventional methods cannot traverse graphs via missing connections. Time complexity is another challenge as it grows exponentially during the traversal process. Modern approaches, such as Hamilton et al. (2018); Ren et al. (2020), use query embeddings (QEs) methods that can answer complex logical queries without the need of path traversal in graphs. The QEs methods first transform a complex logical query into a machine-readable format: (1) converting a query in natural textual form into first-order logic (FOL) form (including conjunction $\wedge$, disjunction $\vee$, negation $\neg$ and existential quantification $\exists$ operator) and (2) decomposing it into a computation graph (including relation projection operator). For example, Fig. 1 depicts the process of turning a complex logical query "*Which universities do the Nobel Prize winners of Australian citizens work in?*" into a computation graph. This FOL query is then projected in the embedding space, required for the modeling process to learn to answer the query.

Among different approaches in representing queries in the embedding space, geometric-based approaches have had renewed interests since the work in point embeddings (Hamilton et al., 2018). Following works have expanded this approach using hyper-boxes (Ren et al., 2020), sets of points as particles (Bai et al., 2022), hyperboloids (Choudhary et al., 2021b) and 2D-cones (Zhang et al., 2021). These works commonly resort to set operators over shapes that can handle the conjunction, only a few Zhang et al. (2021) can handle the negation. Nevertheless, existing geometric-based methods depend on the neural approach to model the conjunction operator. This approach is not easily explainable, counter-intuitive and does not take full advantage of the properties that these geometric representations are intended to be used for.

We highlight in this paper the essential role the projection operator plays in all complex query embedding methods. This operator is often learned through training neural architectures together with the logical operators in an end-to-end fashion. The semantic role of this operator is to obtain a meaningful representation of a predicate (relation) in

Findings of the Association for Computational Linguistics: ACL 2023, pages 11931–11946

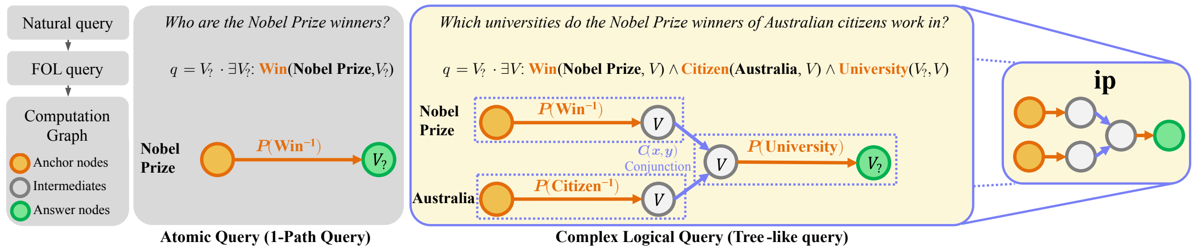

Figure 1: Comparison between an atomic query (with a Projection operator) and a complex logical query (with Projection plus Intersection operators, referred to as an ip structure, having three atomic queries and a conjunction).

FOL, which operates as a function converting a domain of input embeddings into a range of outputs. These are fundamentally different from the roles of the logical operators, in which geometric approaches can be modelled as set operators. Moreover, little work highlights the importance of the neural approach in learning the relation projection.

We address the first issue by introducing a novel symbolic operator in geometry for answering the conjunctive queries. Specifically, we directly calculate the intersection of geometric shapes in the embedding space (see an example in Fig. 2 and more details in Sec. 4.2). We use cone embeddings (Zhang et al., 2021) as a key geometric representation in our approach, since conic shapes were shown to be effective in modeling all FOL operators. By directly calculating intersection, our approach can reduce the computational cost of modeling the conjunction operator, as there is no need to incorporate expensive neural training in this logic operator, as compared with other geometric-based models (Hamilton et al., 2018; Ren et al., 2020; Choudhary et al., 2021b; Zhang et al., 2021). Further, classifying types of geometric intersection (partial, complete and none type) can improve the explainability in modeling conjunction operator (see Sec 4.2). To highlight the importance of relation projection (finding tail entities from a source entity via a relation) in complex logical query embeddings, we propose a general framework of modeling this operator, called relation projection network (RPN) (see Fig. 3). The RPN can enhance the learning in the relation projection operation, due to its high frequency and its dominance in diverse query structures (see Fig. 6).

Overall, we introduce **S**implified **Con**e **E**mbeddings (SConE) for modeling the relation projection and logical operators in complex queries. Our contributions are: (1) introducing a symbolic modeling for the conjunction operator in FOL query, (2) proposing a general framework using RPN to improve the learning of relation projection operator in both atomic and complex logical queries and (3) surpassing model performance of previous state-of-the-art geometric-based models for both non-negation and negation queries.

2 Related Work

Atomic query answering for knowledge graph completion The atomic query has a given head concept (v_h) and a relation (r), and the answering task is to find the projected tail concept (v_t). Using geometric-based methods to answer atomic queries (or path queries) without complex logical operators has been well-studied since the appearance of knowledge graph embeddings, notably, translation-based methods (Bordes et al., 2013) and rotation (Sun et al., 2019; Zhang et al., 2020a). Further, Nickel and Kiela (2017); Balažević et al. (2019) proposed hyperbolic space (non-Euclidean geometry) over a Poincaré ball while others (Gao et al., 2020) used 3D shapes. However, these models are limited in answering complex queries involving FOL logical operators (*e.g.* Fig. 1 and Fig. 6).

Complex logical query answering for multi-hop reasoning The complex logical query has atomic queries with logical operators (see Fig. 1). Different methods addressing this task are geometry-based embeddings (points Hamilton et al. (2018), boxes Ren et al. (2020), hyperboloids Choudhary et al. (2021b), cones Zhang et al. (2021), particles Bai et al. (2022), distribution-based embeddings (Ren and Leskovec, 2020; Choudhary et al., 2021a; Huang et al., 2022; Yang et al., 2022; Long et al., 2022), auxiliary enrichment methods (Hu et al., 2022) using entity and relation type knowledge, logic-based methods (Arakelyan et al., 2021; Chen et al., 2022; Zhu et al., 2022; Xu et al., 2022) using fuzzy logic to model the logical operators, neural-based methods (Kotnis et al., 2021; Liu et al., 2022; Amayuelas et al., 2022) and oth-

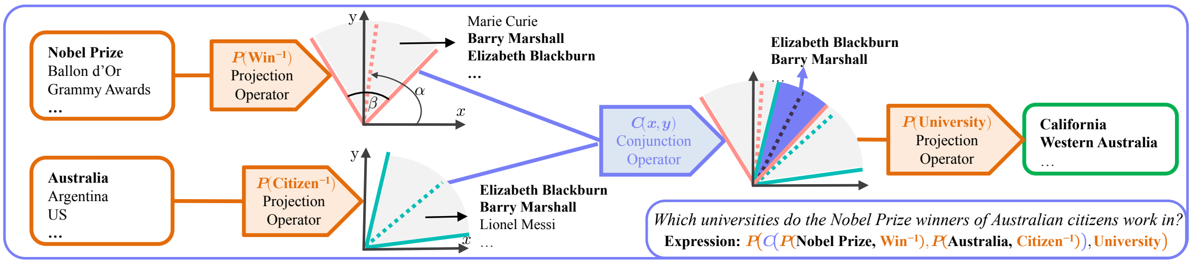

Figure 2: An example of modeling relation projection operators ($\mathcal{P}$) (using the neural approach) and conjunction operator ($\mathcal{C}$) (using the symbolic approach) of the complex logical query structure (ip) via an expression consisting of three ($\mathcal{P}$) and one ($\mathcal{C}$) (as shown in Fig. 1). Dot lines are semantic center axis (α) of sector-cones.

ers (Sun et al., 2020). Although the logic-based methods are explainable in modeling FOL operators (a learning-free), other methods such as geometric-based embeddings are challenging to interpret this process, since these rely on the neural approach to model the conjunction operator. We provide a symbolic modeling approach for handling the conjunction to improve explainability in geometric-based models.

3 Preliminaries

3.1 First-Order Logic queries over KGs

Given a set of entities ($v \in \mathcal{V}$) and a set of relations ($r \in \mathcal{R}$), a knowledge graph (KG) $\mathcal{G} = \{(v_h, r, v_t)\}$ is a set of triples, each includes a head entity (v_h), a relation (r) and a tail entity (v_t).

Given a knowledge graph, a complex FOL query is a formula consisting of: constants, quantified bound variables ($V_1, \ldots, V_n$) and free variables ($V_?$) (target), in addition to relation symbols $R(V_i, V_j)$ and logic connectives ($\exists, \wedge, \vee, \neg$). An entity of KG ($v \in \mathcal{V}$) maps to each constant or variable. Each $R(V_i, V_j)$ maps to a binary function whether a relation exists between (V_i) and (V_j). Logic connectives are conjunction ($\wedge$), disjunction ($\vee$), negation ($\neg$) and existential quantification ($\exists$)[1] (see an example of FOL query mapped to the (ip) structure in Fig. 1, and more query structures in Fig. 6). Given this example, the goal of a FOL query answering is to find the answers (or free variables) such that the formula is true.

3.2 Query Embeddings

Cone parameterization. We adopt the same definitions and propositions in Zhang et al. (2021), to define a two-dimensional sector-cone using two variables: (1) angle $\alpha \in [-\pi, \pi)$ represents the angle between the semantic center axis and the positive $x-$axis, and (2) aperture $\beta \in [0, 2\pi]$ represents the aperture of the sector cone (see an example with pink sector-cone in Fig. 2).

Query Embeddings representation. Given a complex logical query (q), we represent its embedding ($\mathbf{q}$) as a Cartesian product of two-dimensional sector-cones in the embedding space using two variables: semantic center axis $\boldsymbol{\alpha}_q \in [-\pi, \pi)^d$ and aperture $\boldsymbol{\beta}_q \in [0, 2\pi]^d$, where d is the embedding dimension. Next, given a semantic entity (v), we represent its embedding ($\mathbf{v}$) as a Cartesian product of cones embedding using semantic center axis $\boldsymbol{\alpha}_v \in [-\pi, \pi)^d$ and zero aperture defined by:

$$\mathbf{q} = (\boldsymbol{\alpha}_q, \boldsymbol{\beta}_q),\ \mathbf{v} = (\boldsymbol{\alpha}_v, \mathbf{0}) \tag{3.1}$$

3.3 Operators in First-Order Logic queries

We decompose the symbolic representation of the complex logical query (q) using a computation graph, a tree-like query (see Fig. 1). This graph has *vertexes* and *links* where each vertex represents a set of entities and each link represents a modeling process of either of two types: *relation projection* operator ($\mathcal{P}$) or any FOL operators (conjunction ($\mathcal{C}$), disjunction ($\mathcal{D}$) and negation ($\mathcal{N}$)):

- **Relation Projection (Projection)**: $\mathcal{P}(\mathbf{x}, \mathbf{r})$ computes the projection from the input ($\mathbf{x}$) as a *head* entity to the set of *tail* entities via relation ($\mathbf{r}$). Otherwise, $\mathcal{P}(\mathbf{x}, \mathbf{r}^{-1})$ computes the projection from the input ($\mathbf{x}$) as a *tail* entity to the set of *head* entities via ($\mathbf{r}$).

- **Conjunction (Intersection)**: $\mathcal{C}(\mathbf{x}_1, \mathbf{x}_2)$ computes the *intersection* of each geometric element in one set of entities ($\mathbf{x}_1$) and the corresponding element in the other entity set ($\mathbf{x}_2$).

[1]Universal quantification ($\forall$) rarely appears in the real situation, this connective is therefore excluded.

- **Disjunction (Union)**: $\mathcal{D}(\mathbf{x}_1, \mathbf{x}_2)$ computes the *union* of each geometric element in one set of entities $(\mathbf{x}_1)$ and the corresponding geometric element in the other entity set $(\mathbf{x}_2)$.

- **Negation (Complement)**: $\mathcal{N}(\mathbf{x})$ computes the *complement* of each geometric element in the set of entities $(\mathbf{x})$.

4 Modeling operators of FOL queries

We describe the modeling of relation projection (4.1) and logical operators (4.2) in a complex FOL query (q) (with its set of answer entities $\mathcal{V}_q \subset \mathcal{V}$) over knowledge graphs in the following:

4.1 Modeling relation projection

This section is to model the relation projection operator $(\mathcal{P})$ (see Sec. 3.3) for Knowledge Graph Embedding (KGE). Overall, given an atomic query $q = (v, r)$, we propose a relation projection network (RPN) with two layers: (1) first transforming the source entity using an *ensemble of multiple KGE techniques*, (2) merging the outputs at the second layer (called *entanglement layer*) to produce the sector-cones embedding of the query (see Fig. 3). We use two KGE techniques at the first layer, called *relation transformation* and *multi-layer perceptron*. As the relation projection task is similar to KGE, one can select different models in KGE such as TransE (Bordes et al., 2013) or HAKE (Zhang et al., 2020b), then adapt these into the first layer in principle.

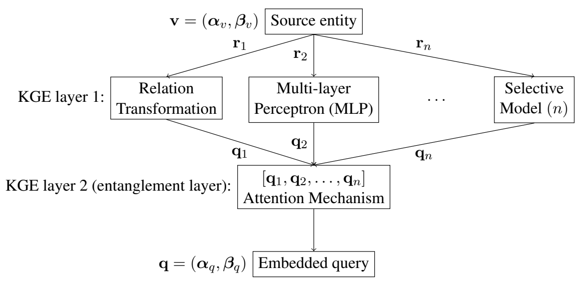

Figure 3: **(Up→Down)**: A general framework of RPN.

There are many models achieving KGE (Ji et al., 2022), the ensemble selection is therefore not restricted, but it should be efficient in model complexity and computational cost. Fundamentally, using one technique is sufficient; however, having a general framework using multiple techniques is to analyze the learning process from a broader viewpoint. We select two KGE models as a simple case to illustrate that it is possible to use multiple KGE techniques.

Relation transformation Specifically, we model an embedded relation $\mathbf{r} = (\mathbf{W}_r, \mathbf{b}_r)$ requiring for the projection operation $(\mathcal{P})$ by a neural network as in (Chen et al., 2022), where $\mathbf{W}_r$ denotes a weight matrix and $(\mathbf{b}_r)$ denotes a bias vector. We transform a source entity $\mathbf{v} = (\boldsymbol{\alpha}_v, \boldsymbol{\beta}_v)$ into an embedded query $(\mathbf{q})$ via this relation. However, as our entity representation based on sector-cones embeddings, which is different than the fuzzy sets used in (Chen et al., 2022), we add a concatenation operation of the semantic center axis $(\boldsymbol{\alpha}_v)$ and the aperture $(\boldsymbol{\beta}_v)$ to convert these into a vector $[\mathbf{v}] \in \mathbb{R}^{2d}$ as follows:

$$\mathbf{q}_t = f(\mathbf{v}) = \mathrm{LN}(\mathbf{W}_r[\mathbf{v}] + \mathbf{b}_r),$$

where LN is Layer Normalization (Ba et al., 2016). We use the basic decomposition of (Schlichtkrull et al., 2018) to define $(\mathbf{W}_r)$ and $(\mathbf{b}_r)$.

Multi-layer Perceptron (MLP) An alternative way to model the relation projection is to use MLP. We transform the entity $(\mathbf{v})$ to query $(\mathbf{q})$ via the relation $(\mathbf{r})$ by a mapping function (f) as follows:

$$\mathbf{q}_m = f(\mathbf{v}_r) = \mathrm{LN}(\mathbf{MLP}([\mathbf{v}_r])),$$

where $\mathbf{MLP} : \mathbb{R}^{2d} \rightarrow \mathbb{R}^{2d}$ is to approximately represent the mapping function $f(x)$, $\mathbf{v}_r$ is a translation embeddings of the source entity and the relation: $\mathbf{v}_r = \mathbf{v} + \mathbf{r}$. As the representation of the entity and the relation $\mathbf{r} = (\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r)$ are sector-cones embedding, we apply a concatenation operation as that in the relation transformation technique to convert $(\mathbf{v}_r)$ to the vector embedding $[\mathbf{v}_r] \in \mathbb{R}^{2d}$.

Entanglement layer After transforming the entity to the embedded query using the relation transformation and the MLP, we introduce an entanglement layer to merge the output from the first KGE layer into one output. We use attention mechanism in this layer:

$$\mathbf{q} = (\boldsymbol{\alpha}_q, \boldsymbol{\beta}_q) = s\left(\sum_i^2 \mathbf{A} \odot [\mathbf{q}_t, \mathbf{q}_m]\right),$$

where $s(x)$ is a function to split the $2d$-vector into two d-vectors, each is for the semantic center axis and the aperture embedding of the query, $\odot$ denotes Hadamard product, $[,]$ denotes an operator to stack two $2d$-vectors into a matrix in $\mathbb{R}^{2\times 2d}$ and $\mathbf{A} \in \mathbb{R}^{2\times 2d}$ is an attention matrix as follows:

$$\mathbf{A} = \mathrm{SoftMax}\left(f_a(\mathbf{q}_t, \mathbf{q}_m)\right),$$

where the SoftMax(.) function applies over the first dimension of the matrix. The $f_a(\mathbf{q}_t, \mathbf{q}_m) = \mathbf{MLP}([\mathbf{q}_t, \mathbf{q}_m])$ is attention score function. We also provide a scaling function to convert the semantic center axis $(\boldsymbol{\alpha}_q)$ and the aperture $(\boldsymbol{\beta}_q)$ into their normal range (see Appendix B.1) as that in ConE (Zhang et al., 2021).

4.2 Symbolic modeling of logical operators

In this section, we describe the modeling process of all logical operators $(\mathcal{C}, \mathcal{N}, \mathcal{D})$ using symbolic modeling only without neural-based methods, naturally making use of the geometric properties of sector-cone shapes in the embedding space. In comparison with ConE (Zhang et al., 2021), this model leveraged the neural approach to learn the conjunction $(\mathcal{C})$ while using non-neural approach to model the disjunction and negation $(\mathcal{D}, \mathcal{N})$.

Conjunction This aims to model the conjunction $\mathcal{C}(q_1, q_2)$ of any pair of conjunctive queries, each query $q_i = (\alpha_i, \beta_i)$ is in the cone embedding space. Assuming the embedding dimension $(d = 1)$ as the simplest case, each query is represented by a sector-cone (see Eq. (3.1)). As the intersection of the two conjunctive queries is also a sector-cone $q_\wedge = (\alpha_\wedge, \beta_\wedge)$; therefore, one can directly calculate this intersection from a symbolic geometric perspective as follows:

$$\beta_\wedge = \begin{cases} u_2 - l_1, \\ \beta_2, \\ 0, \\ u_1 - l_2, \\ \beta_1, \\ 0, \end{cases} \quad \alpha_\wedge = \begin{cases} u_2 - \frac{\beta_\wedge}{2}, & \text{if } c_1 \\ \alpha_2, & \text{if } c_2 \\ l_1 - \frac{|l_1 - u_2|}{2}, & \text{if } c_3 \\ u_1 - \frac{\beta_\wedge}{2}, & \text{if } c_4 \\ \alpha_1, & \text{if } c_5 \\ l_2 - \frac{|l_2 - u_1|}{2}, & \text{if } c_6 \end{cases} \tag{4.1}$$

where (u_i, l_i) is the upper and lower bound for each sector-cone $(l_i \leq \alpha_i \leq u_i)$. These calculations are that $(u_i = \alpha_i + \frac{\beta_i}{2})$ and $(l_i = \alpha_i - \frac{\beta_i}{2})$; and c_i represents each conditional scenario regarding relative position between the two sector-cones:

$$\begin{aligned} c_1 &:= (u_1 \geq u_2) \wedge (u_2 \geq l_1) \wedge (l_1 \geq l_2), \\ c_2 &:= (u_1 \geq u_2) \wedge (u_2 \geq l_2) \wedge (l_2 > l_1), \\ c_3 &:= (u_1 \geq l_1) \wedge (l_1 > u_2) \wedge (u_2 \geq l_2), \\ c_4 &:= (u_2 \geq u_1) \wedge (u_1 \geq l_2) \wedge (l_2 \geq l_1), \\ c_5 &:= (u_2 \geq u_1) \wedge (u_1 \geq l_1) \wedge (l_1 > l_2), \\ c_6 &:= (u_1 \geq l_2) \wedge (l_2 > u_1) \wedge (u_1 \geq l_1). \end{aligned} \tag{4.2}$$

Note that there are three types regarding calculating intersection of two sector-cones in Eq. (4.2): (1) partial intersection (see c_1, c_4), (2) complete intersection (see c_2, c_5) and (3) none intersection (see c_3, c_6). Figure 4 shows these cases (c_1, c_2, c_3)

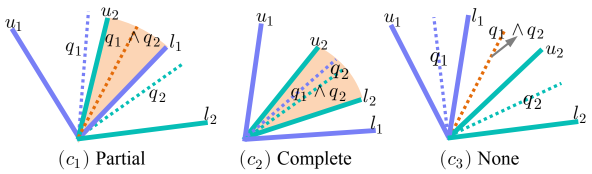

Figure 4: Different scenarios intersection of two sector-cones, purple & green region is for (q_1) & (q_2), and orange region (axis only in c_3) is for the intersection.

from one sector-cone to the other and vice versa (c_4, c_5, c_6). While the calculation of the partial and complete intersection are based on natural representation of geometric shapes, the calculation of none intersection type is based on *zero aperture* and *middle semantic axis* between the lower bound of one sector-cone and the upper bound of the other. The aperture in this situation is confidence, but the semantic axis is uncertain as it can be any axes between the mentioned bounds. We consider the middle axis as a special case for none intersection type (see further details in the following Eq. 4.3).

In general, to compute the intersection of (k) conjunctive queries, assuming this computation satisfies the *associative* and/or *commutative* law for logic, we compute the intersection $\mathcal{C}(q_i, q_{i+1})$ of the first two arbitrary conjunctive queries, then compute the intersection of $\mathcal{C}(q_i, q_{i+1})$ and the next conjunctive query (q_{i+2}) to produce $\mathcal{C}(\mathcal{C}(q_i, q_{i+1}), q_{i+2})$, and iterate this process until reaching the final conjunctive query (q_k).

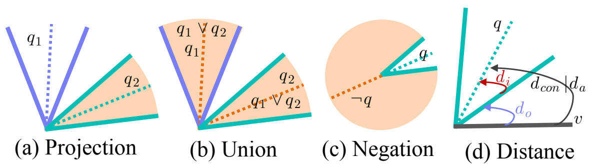

Figure 5: Operators over two sector-cones.

Conjunction: Weight semantic axis for none type intersection In the conjunction $\mathcal{C}(q_1, q_2)$ with none type intersection (c_3, c_6), the equality of axis intersection $(\alpha_\wedge)$ of two sector-cones embeddings can be any axes between the upper bound of one sector-cone and the lower bound of the other sector-cone. In general, the equality to calculate $(\alpha_\wedge)$ in the cone embedding space (see Eq. 4.1) for the none type intersection is shown below:

$$\alpha_\wedge = \begin{cases} \delta l_1 + (1-\delta)u_2, & \text{if } c_3, \\ \delta l_2 + (1-\delta)u_1, & \text{if } c_6, \end{cases} \tag{4.3}$$

where $(\delta \in [0,1])$ is a hyper-parameter to control the spatial location of axis intersection regarding that of the two mentioned bounds. Notice that, let $(\delta = 0.5)$ as in Eq. (4.1), the semantic center axis intersection $(\alpha_\wedge)$ is in the middle between the two bounds as a special case of equality in Eq. (4.3) (see Appendix F for further analysis).

Negation This aims to model the negation $\mathcal{N}(\mathbf{q})$, called $\mathbf{q}_\neg = (\boldsymbol{\alpha}_\neg, \boldsymbol{\beta}_\neg)$ of the embedded query $\mathbf{q} = (\boldsymbol{\alpha}, \boldsymbol{\beta})$. In the cone embedding space, the semantic center axis of $(\mathbf{q}_\neg)$ should be in opposite direction via the $O-$axis regarding that of the $(\mathbf{q})$ (see Fig. 5 (c)). In terms of the aperture, the summation of both apertures of $(\mathbf{q}_\neg)$ and $(\mathbf{q})$ should be close to $(2\boldsymbol{\pi})$ as follows:

$$\boldsymbol{\alpha}_\neg = \begin{cases} \boldsymbol{\alpha} - \boldsymbol{\pi}, \text{ if } (\boldsymbol{\alpha} \geq 0) \\ \boldsymbol{\alpha} + \boldsymbol{\pi}, \text{ if } (\boldsymbol{\alpha} < 0) \end{cases} \quad \boldsymbol{\beta}_\neg = 2\boldsymbol{\pi} - \boldsymbol{\beta}.$$

Disjunction Similar to cone embedding Zhang et al. (2021), we adapt the DNF technique in Ren et al. (2020) to represent the disjunction operation $\mathcal{D}(q_1, q_2)$ as disjunction of conjunctive queries (see Fig 5 (b)). Hence, we can leverage $(\mathcal{C}, \mathcal{N})$ operators above to have a set of embeddings of the conjunctive queries. Those entities nearest to any of these conjunctive queries in the cone embedding space are considered to be the answers (see the aggregated distance score in Eq. (4.5)).

4.3 Optimization

Distance score function We define a distance score function $d(\mathbf{v}, \mathbf{q})$ of the embedding between: the expected entity $\mathbf{v} = (\boldsymbol{\alpha}, \mathbf{0})$ and the query $\mathbf{q} = (\boldsymbol{\alpha}_q, \boldsymbol{\beta}_q)$ (as stated in Sec. 3.2). We use two distance types: (d_{con}) for conjunctive queries and (d_{dis}) for disjunctive queries as (Ren et al., 2020; Zhang et al., 2021). In (d_{con}), there are three terms: an outside distance (d_o), an inside distance (d_i) and a separated axis distance (d_a) (see Fig. 5 (d)), which are defined by:

$$d_{con}(\mathbf{v}, \mathbf{q}) = (1-\psi)(d_o + \lambda d_i) + \psi d_a, \tag{4.4}$$

where $\lambda \in (0,1)$ is to encourage $(\mathbf{v})$ to be covered by the sector-cone embedding $(\mathbf{q})$. The hyper-parameter (ψ) is to weight the effect of the outside with inside distance and the separated axis distance (see Appendix C for more details). To calculate (d_{dis}), we use the DNF technique in Ren et al. (2020), which obtains the minimum distance in embeddings between: an expected entity and each conjunctive query in DNF, over a (k) number of conjunctive queries:

$$d_{dis}(\mathbf{v}, \mathbf{q}) = \min\{d_{con}(\mathbf{v}, \mathbf{q}_i)_{i:1\to k}\}, \tag{4.5}$$

Loss function During the optimization process, we use the negative sampling loss $(\mathcal{L})$ (Mikolov et al., 2013a,b) as that in Ren and Leskovec (2020): $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2$, where $\mathcal{L}_1 = -\log\sigma(\gamma - d(\mathbf{v}, \mathbf{q}))$ involves a minimization of the distance $d(\mathbf{v}, \mathbf{q})$ for a positive answer entity $(v \in \mathcal{V}_q)$, and $\mathcal{L}_2 = -\frac{1}{n}\sum_i^n \log\sigma(d(\mathbf{v}', \mathbf{q}) - \gamma)$ involves a maximization of the distance $d(\mathbf{v}', \mathbf{q})$ for a number (n) of negative answer entities $(v'_{i:1\to n} \notin \mathcal{V}_q)$; $\sigma(x)$ is the activation function (*e.g.* sigmoid) and (γ) is a positive margin as hyper-parameter.

5 Experiments

5.1 Experimental setups

Multi-hop Reasoning (MHR) or Complex Logical Query Answering task Given an arbitrary complex FOL query, when traversing the incomplete KGs, *non-trivial* answers cannot be returned directly. The MHR task aims to find these answers. We evaluate our approach on there datasets: FB15k (Bollacker et al., 2008), FB15k-237 (Toutanova and Chen, 2015) and NELL995 (Xiong et al., 2017), following the preprocessing in BetaE (Ren and Leskovec, 2020). We follow the training protocol of previous works (Ren and Leskovec, 2020; Zhang et al., 2021), using 10 query syntaxes (non-negation 1p/2p/3p/2i/3i and negation 2in/3in/inp/pni/pin) for the training. We use these 10 syntaxes plus 4 unseen syntaxes (ip/up/2u/pi) for the evaluating process (see Fig. 6). An example of the (1p) query is (v, r_1) *i.e.* (`Wesleyan_University`, `major_field_of_study`), while (2p) or (3p) query corresponds to (v, r_1, r_2) or (v, r_1, r_2, r_3).

Evaluation Protocol Following the evaluation protocol in (Ren et al., 2020), given a query, we split its answers into two sets: easy answers and hard answers. The former is for those entities that can be reached on the training/validation graph through symbolic approach in graph traversing. The latter is for those that can be predicted using query embedding models, or the reasoning process

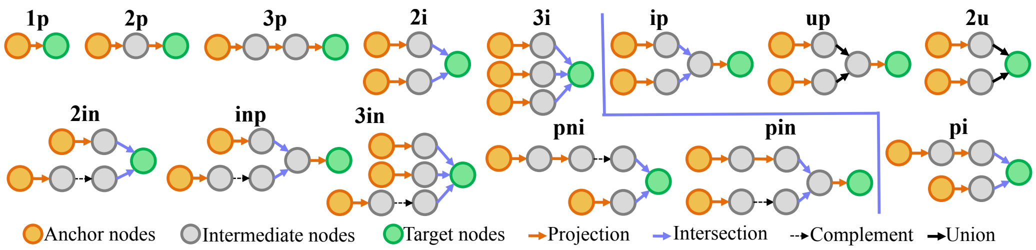

Figure 6: **(Left):** training queries. **(Left) & (Right):** those queries are involved in the evaluation process.

Dataset	Model	AVG$_p$	AVG$_n$	1p	2p	3p	2i	3i	(ip)	(pi)	(2u)	(up)	2in	3in	inp	pin	pni
FB15k	GQE	28.2	-	53.9	15.5	11.1	40.2	52.4	19.4	27.5	22.3	11.7	-	-	-	-	-
	Q2B	40.1	-	70.5	23.0	15.1	61.2	71.8	28.7	41.8	37.7	19.0	-	-	-	-	-
	Q2P	46.8	**16.4**	**82.6**	30.8	25.5	65.1	74.7	34.9	49.5	32.1	26.2	**21.9**	**20.8**	12.5	8.9	**17.1**
	ConE	49.8	14.8	73.3	33.8	29.2	64.4	73.7	35.7	50.9	55.7	31.4	17.9	18.7	12.5	**9.8**	15.1
	SConE	**53.0**	16.0	80.8	**38.2**	**30.7**	**67.0**	**75.1**	**41.7**	**52.1**	**57.1**	**34.6**	20.5	19.5	**14.5**	9.2	16.1
FB15k (237)	GQE	16.6	-	35.2	7.4	5.5	23.6	35.7	10.9	16.7	8.4	5.8	-	-	-	-	-
	Q2B	21.1	-	41.3	9.9	7.2	31.1	45.4	13.3	21.9	11.9	8.1	-	-	-	-	-
	Q2P	21.9	6.0	-	-	-	-	-	-	-	-	-	4.4	9.7	7.5	4.6	3.8
	ConE	23.4	5.9	41.8	12.8	**11.0**	32.6	**47.3**	14.0	**25.5**	14.5	**10.8**	5.4	8.6	7.8	**4.0**	3.6
	SConE	**24.1**	**6.7**	**44.2**	**13.0**	10.7	**33.8**	47.0	**17.0**	25.1	**15.5**	10.7	**6.9**	**10.6**	**7.9**	**4.0**	**4.3**
NELL (995)	GQE	18.7	-	33.1	12.1	9.9	27.3	35.1	14.5	18.5	8.5	9.0	-	-	-	-	-
	Q2B	23.6	-	42.7	14.5	11.7	34.7	45.8	17.4	23.2	12.0	10.7	-	-	-	-	-
	Q2P	25.5	6.0	-	-	-	-	-	-	-	-	-	5.1	7.4	10.2	3.3	3.4
	ConE	27.2	6.4	53.1	16.1	13.9	40.0	**50.8**	17.5	26.3	15.3	11.3	5.7	**8.1**	10.8	**3.5**	3.9
	SConE	**30.4**	**6.7**	**58.2**	**20.5**	**17.0**	**41.8**	50.7	**22.9**	**28.6**	**18.8**	**15.5**	**6.2**	8.0	**11.8**	**3.5**	**4.2**

Table 1: The average **MRR** (%) of geometric-based embedding models in FOL queries: AVG$_p$ is for EPFO queries while AVG$_n$ is for negation queries. The results of (GQE, Q2B, ConE) are taken from (Zhang et al., 2021). Union queries ($2u/up$) are in DNF forms, (ip/pi/2u/up) queries are not involved in the training process.

performs on hard answers. We use the mean reciprocal rank (MRR) metrics, computing the ranking of each hard answer against all non-answer entities, to measure the performance of models.

Baselines We use four recent geometric-based embedding models as baselines: GQE (Hamilton et al., 2018), Query2Box (Q2B) (Ren et al., 2020), Query2Particles (Q2P) (Bai et al., 2022) and ConE (Zhang et al., 2021), and obtain their results from ConE and Q2P. We also compare these results with state-of-the-art models based on fuzzy logic (see Appendix E.2).

5.2 Results

Existential Positive First-order (EPFO) queries Overall, the average MRR in all EPFO queries without negation (**AVG$_p$**) of SConE[2] significantly outperform all geometric-based baselines using the three datasets, particularly more than that of ConE by nearly 12% using the NELL995 dataset (see Table 1). For queries (1p/2p/3p/2i/3i) involving in the training process, most of the average MRR in each of these query structure (11 out of 15 metrics) significantly surpass baselines. Specially, in the (2p) query, around 26% gain of the MRR in SConE over that in ConE observes in the NELL995 dataset. With regard to queries (ip/pi/2u/up) that are not involved in the training process, the model performance of SConE also shows a significant increase of MRR, compared to that of ConE (10 out of 12 metrics), which suggests an improvement in the ability of zero-shot learning for these queries (please see Appendix E.1 for error bars of the main results).

Negation queries Overall, the average MRR in negation queries (**AVG$_n$**) of SConE is significantly higher than that of ConE by closely 14% using the FB15k-237 dataset (see Table 1); even though there is no difference in the modeling of negation operator in both models. This can be due to the effect of using RPN to enrich learning in the atomic query structure (1p), which dominantly involves in all negation queries (see Fig. 6 Bottom-Left and Sec. 5.3 for further ablation study in the RPN).

[2]Source code is available at https://github.com/nlp-tlp/scone

5.3 Ablation study - Sensitivity analysis

We conduct experiments for ablation study w.r.t. two situations: relation projection networks and geometric intersection types; and for sensitivity analysis w.r.t. two hyper-parameters: distance weight (ψ) and embedding dimensions (d) as follows:

Relation projection network Table 2 shows the average MRR on FOL queries in general and several specific EPFO queries (*e.g.* 1p/2i/ip/2u) of the test set w.r.t. different RPNs: (1) MLP only, (2) relation transformation only and (3) MLP with relation transformation and attention mechanism for the entanglement layer (see Sec. 4.1). Overall, the third scenario observes the highest model's performance of answering complex logical queries over the other scenarios. We attribute this observation to the advantage of enhancing the learning for atomic query (1p) using the RPN, resulting in an improvement in model performance in total. Specifically, an increase from 42.5 (using MLP only) to 44.2 (using both MLP and relation transformation) in the average MRR (%) of 1p query can lead to an increase in that of other query structures (2i, ip, 2u). This is because the atomic query involves in all structures and in the early stage during the decomposition process via the computation graph for each structure (see Fig. 6). Moreover, SConE with MLP only ($d = 800$, around 20.0M parameters) and symbolic modeling for logical operators uses less parameters than those in ConE ($d = 800$, around 23.9M parameters reported in Long et al. (2022) Appendix B), but both models achieve similar performance (see Table 1).

Projection	AVG$_p$	AVG$_n$	1p	2i	ip	2u	#Params
MLP only	22.4	6.5	42.5	30.9	14.5	14.6	11.3M
MLP only (*)	23.2	**6.8**	43.1	31.9	16.0	15.1	20.0M
Rtrans only	23.1	6.5	44.0	32.3	15.7	14.9	25.0M
Rtrans + MLP + Attention	**24.1**	6.6	**44.2**	**33.6**	**17.0**	**15.3**	**31.2M**

Table 2: Effect of projection network on average **MRR** (%) fixing $d = 400$ using the FB15k-237 dataset. (*) is for $d = 800$, (M) is million.

Geometric intersection of sector-cones Table 3 shows the model's performance of answering complex logical queries using different types of intersection between sector-cones. The implementation of using individual intersection type is to see the impact of each intersection type on model performance, compared with that using all conditional intersection types. Geometrically, there are three types (None, Complete, Partial) of intersection for any pair of two conjunctive queries as illustrated in Figure 4. Using one type of intersection calculation individually will result in misrepresentation for the other two. For example, assuming all query pairs have None intersection, *i.e.* only using (c_3) and (c_6) (see Eq. 4.2) for intersection calculation, we will miss the opportunity to capture Complete and Partial overlap correctly. Table 3 confirms our intuition, the model performs best when all intersection types are considered. Notice that the calculation of intersection of sector-cones neglects the neural-based approach, but the model is able to efficiently learn to answer intersection queries as that in ConE, particularly when using partial intersection only. It is arguable that the learning process focuses on the atomic query which are also involved in conjunctive queries.

Intersection		AVG$_p$	AVG$_n$	2i	3i	ip	pi
None	c_3, c_6	17.6	4.3	20.4	29.8	10.3	18.1
Complete	c_2, c_5	22.8	6.2	32.0	45.4	14.9	24.2
Partial	c_1, c_4	23.6	5.6	33.5	**48.0**	15.5	**25.2**
All	$c_1, c_2, c_3, c_4, c_5, c_6$	**24.1**	**6.6**	**33.6**	47.3	**17.0**	24.9

Table 3: Effect of intersection types for sector-cones on average **MRR** (%) (see Eq. (4.2)) using the FB15k-237 dataset.

Weight distance Table 4 (Left) shows the average MRR on FOL queries of the test set w.r.t. different weights (ψ) of distances (see Eq. (4.4)), ranging from zero (no axis distance but having inside and outside distance), half of one (equally having axis distance with inside and outside distance) to one (having axis distance but no inside and no outside distance). The model performance increases from $(\psi = 0)$ to $(\psi = 1)$, which suggests that there is an effect of using the axis distance (d_a). The performance reaches its peak when (ψ) is set to one as maximum, which suggests that the model can learn to answer complex logical queries using the axis distance only (without inside and outside distance). However, we theorize that the inside and outside distance should be involved during the training process. This is to improve the explainability of cone embeddings, where those entities inside the sector-cone are expected to be answers of the query. In this situation, the aperture plays a role in covering answer entities. Thus, to keep all distance types during the optimization process, we set $0 < \psi = 0.9 < 1$ for the main results (see Table 1).

Table 4: **(Left:)** Effect of weight distance (ψ) (fixing $d = 400$) and **(Right:)** Effect of embedding dimension (d) (fixing $\psi = 0.9$) on average **MRR** (%) using the FB15k-237 dataset. (M) is million.

SConE	AVG$_p$	AVG$_n$
$\psi = 0.0$	23.1	5.5
$\psi = 0.1$	23.2	5.6
$\psi = 0.5$	23.9	6.0
$\psi = 0.9$	**24.1**	6.6
$\psi = 1.0$	**24.1**	**6.7**

SConE	AVG$_p$	AVG$_n$	#Params
$d = 64$	17.0	4.2	4.5M
$d = 128$	20.9	5.6	7.4M
$d = 256$	23.3	6.4	16.3M
$d = 400$	24.1	6.6	31.2M
$d = 512$	**24.4**	**6.8**	**46.3M**

Embedding dimension Table 4 (Right) shows the average MRR on FOL queries of the test set w.r.t. different embedding dimension. The model's performance increases from using small ($d = 64$) to medium ($d = 512$) embedding dimension. This observation suggests that there is a significant effect of this hyper-parameter on the average MRR (of both EPFO and negation queries). Additionally, in the case $d = 256$ (with around 16.3M parameters), SConE uses less 30% in the number of parameters than those in ConE ($d = 800$ with around 23.9M parameters), but both achieves similar average MRR using the same FB15k-237 dataset.

6 Conclusions

We have provided a symbolic modeling for logical operators, particularly computing geometric intersection of sector-cones for modeling the conjunction. In addition, we highlighted the importance of the projection operator by introducing a relation projection network using neural-based approach, to strengthen the learning in atomic queries involved in all FOL query syntaxes. Our neural-symbolic approach using geometric embeddings significantly outperforms state-of-the-art geometric-based models in both EPFO and negation queries.

Limitations

Although our geometric embedding approach can handle a complete set of basic FOL operators (existential quantification, conjunction, disjunction and negation), the modeling of negation operator cannot narrow down the predicted answers to relevant topics of atomic queries. For example, one can expect the answers of this negation question/query "*List Argentina players who are not Lionel Messi in World Cup 2022?*" to be any teammates of Lionel Messi (*i.e.* 2in query structure). However, the current model is designed to return all elements in the entire entity set except for Lionel Messi, which have redundant objects (*e.g.* trees, music, houses). This is a common limitation not only in geometric-based models but in others using fuzzy sets representation. This is due to the fact that the modeling of negation operator is assumed to be the complement set of a questionable entity w.r.t. the entire entity set. Our hypothesis is that the expected answers should be narrowed into the complement set w.r.t. a sub-topic of relevant entity set.

In addition, when apertures of two sector-cones are obtuse angles, the current calculation of partial intersection cannot correctly model the conjunction operator. This special case is inevitable in a system using geometric representation that is closed under negation and conjunction, but not for disjunction (see Appendix A.2 for further details).

Ethics Statement

The ability of models to answer complex logical queries is achievable to reason about knowledge graphs. Due to model's uncertainty, one potential negative impact of this task is the out-of-control in automatic reasoning over open large-scale knowledge graphs, where there are diverse source of information. Some of which though can be missed from KGs due to incompleteness or private purposes, but they can be possibly reasoned using the query embedding methods.

Acknowledgements

This research is supported by the Australian Research Council through the Centre for Transforming Maintenance through Data Science (grant number IC180100030), funded by the Australian Government. Wei Liu acknowledges the support from ARC Discovery Projects DP150102405. Further, the authors would like to thank all the anonymous reviewers for their insightful feedback.

References

Alfonso Amayuelas, Shuai Zhang, Susie Xi Rao, and Ce Zhang. 2022. Neural methods for logical reasoning over knowledge graphs. In *International Conference on Learning Representations (ICLR)*.

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. 2021. Complex query answering with neural link predictors. In *International Conference on Learning Representations (ICLR)*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Jiaxin Bai, Zihao Wang, Hongming Zhang, and Yangqiu Song. 2022. Query2Particles: Knowledge graph reasoning with particle embeddings. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2703–2714, Seattle, United States. Association for Computational Linguistics.

Ivana Balažević, Carl Allen, and Timothy Hospedales. 2019. Multi-relational poincaré graph embeddings. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, NIPS'19, Red Hook, NY, USA. Curran Associates Inc.

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, New York, NY, USA. Association for Computing Machinery.

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*, NIPS'13, page 2787–2795, Red Hook, NY, USA. Curran Associates Inc.

Xuelu Chen, Ziniu Hu, and Yizhou Sun. 2022. Fuzzy logic based logical query answering on knowledge graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):3939–3948.

Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan Reddy. 2021a. Probabilistic entity representation model for reasoning over knowledge graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 23440–23451. Curran Associates, Inc.

Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K Reddy. 2021b. Self-supervised hyperboloid representations from logical queries over knowledge graphs. In *Proceedings of the Web Conference*, WWW '21, page 1373–1384, New York, NY, USA. Association for Computing Machinery.

Christiane Fellbaum. 2010. *WordNet*, pages 231–243. Springer Netherlands, Dordrecht.

Chang Gao, Chengjie Sun, Lili Shan, Lei Lin, and Mingjiang Wang. 2020. Rotate3d: Representing relations as rotations in three-dimensional space for knowledge graph embedding. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 385–394, New York, NY, USA. Association for Computing Machinery.

William L. Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. 2018. Embedding logical queries on knowledge graphs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 2030–2041, Red Hook, NY, USA. Curran Associates Inc.

Olaf Hartig and Ralf Heese. 2007. The sparql query graph model for query optimization. In *The Semantic Web: Research and Applications*, pages 564–578, Berlin, Heidelberg. Springer Berlin Heidelberg.

Zhiwei Hu, Victor Gutierrez Basulto, Zhiliang Xiang, Xiaoli Li, Ru Li, and Jeff Z. Pan. 2022. Type-aware embeddings for multi-hop reasoning over knowledge graphs. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 3078–3084. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Zijian Huang, Meng-Fen Chiang, and Wang-Chien Lee. 2022. Line: Logical query reasoning over hierarchical knowledge graphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 615–625, New York, NY, USA. Association for Computing Machinery.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Bhushan Kotnis, Carolin Lawrence, and Mathias Niepert. 2021. Answering complex queries in knowledge graphs with bidirectional sequence encoders. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):4968–4977.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Xiao Liu, Shiyu Zhao, Kai Su, Yukuo Cen, Jiezhong Qiu, Mengdi Zhang, Wei Wu, Yuxiao Dong, and Jie Tang. 2022. Mask and reason: Pre-training knowledge graph transformers for complex logical queries. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 1120–1130, New York, NY, USA. Association for Computing Machinery.

Xiao Long, Liansheng Zhuang, Li Aodi, Shafei Wang, and Houqiang Li. 2022. Neural-based mixture probabilistic query embedding for answering FOL queries on knowledge graphs. In *Proceedings of the*

2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 3001–3013, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6341–6350, Red Hook, NY, USA. Curran Associates Inc.

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 (ICLR)*.

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

Michael Schmidt, Michael Meier, and Georg Lausen. 2010. Foundations of sparql query optimization. In *Proceedings of the 13th International Conference on Database Theory*, ICDT '10, page 4–33, New York, NY, USA. Association for Computing Machinery.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4444–4451. AAAI Press.

Haitian Sun, Andrew O. Arnold, Tania Bedrax-Weiss, Fernando Pereira, and William W. Cohen. 2020. Faithful embeddings for knowledge base queries. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 564–573, Copenhagen, Denmark. Association for Computational Linguistics.

Zezhong Xu, Wen Zhang, Peng Ye, Hui Chen, and Huajun Chen. 2022. Neural-symbolic entangled framework for complex query answering. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Dong Yang, Peijun Qing, Yang Li, Haonan Lu, and Xiaodong Lin. 2022. GammaE: Gamma embeddings for logical queries on knowledge graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 745–760, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhanqiu Zhang, Jianyu Cai, and Jie Wang. 2020a. Duality-induced regularizer for tensor factorization based knowledge graph completion. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020b. Learning hierarchy-aware knowledge graph embeddings for link prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):3065–3072.

Zhanqiu Zhang, Jie Wang, Jiajun Chen, Shuiwang Ji, and Feng Wu. 2021. Cone: Cone embeddings for multi-hop reasoning over knowledge graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34.

Zhaocheng Zhu, Mikhail Galkin, Zuobai Zhang, and Jian Tang. 2022. Neural-symbolic models for logical queries on knowledge graphs. In *International Conference on Machine Learning (ICML)*.

A Modeling logical operators - None intersection type

A.1 Comments on partial and complete type of intersection

Note that the condition (c_1) for partial intersection type has a special case when these equalities holds $(u_2 = l_1)$ and $(l_1 = l_2)$. In this situation, the partial type (c_1) can be considered as the complete type (c_2). Hence, these types of intersection can be used interchangeably. Similarly, we can interchangeably use the calculation of the partial and complete type for the condition (c_4, c_5). This is due to the fact that the calculation of these intersection types become as follows:

$$\beta_\wedge = \begin{cases} 0, \\ \beta_2, \\ 0, \\ \beta_1 \end{cases} \quad \alpha_\wedge = \begin{cases} u_2, & \text{if } c_1, \\ \alpha_2, & \text{if } c_2, \\ u_1, & \text{if } c_4, \\ \alpha_1, & \text{if } c_5. \end{cases}$$

In terms of the cases (c_1, c_2), notice that $(u_2 = l_1 = l_2)$ or $(u_2 = l_2)$, hence $(0 = \beta_2)$, and $(u_2 = \alpha_2 + \frac{\beta_2}{2})$ or $(u_2 = \alpha_2)$. Similar explanation is for the cases (c_4, c_5).

A.2 Special case of partial intersection

In case of the aperture of two sector-cones are both obtuse angles as shown in Fig. 7, the calculation of partial intersection under conditions (c_1, c_4) cannot exactly model the conjunction operator. Specially, the calculation of $(\alpha_\wedge, \beta_\wedge)$ for conjunction operator in Sec. 4.2 will return $(\alpha_\wedge, \beta_\wedge)$ of the right intersection sector-cone, but will ignore $(\alpha_\wedge, \beta_\wedge)$ of the left intersection sector-cone in this special case. Our hypothesis is that this special case is inevitable if a system using geometric representation, such as conic shapes, is closed under negation and conjunction, but not disjunction. This limitation can be addressed by providing a refined approach for calculating the partial intersection when both apertures are obtuse angles. We leave this approach as an extension for future work.

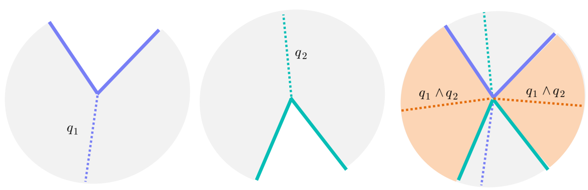

Figure 7: A special case of partial intersection of two sector-cones.

B Further details in section modeling operators of FOL queries

B.1 Scaling function

Continuing Sec. 4.1, after obtaining the output from entanglement layer as $\mathbf{q} = (\boldsymbol{\alpha}, \boldsymbol{\beta})$, we scale the semantic center axis and the aperture into their normal ranges $[-\pi, \pi)$ and $[0, 2\pi]$ respectively, defined in Sec. 3.2. The final embedded query $\mathbf{q} = (\boldsymbol{\alpha}', \boldsymbol{\beta}')$ is as follows:

$$\boldsymbol{\alpha}' = \pi \tanh(\lambda_1 \boldsymbol{\alpha}), \tag{B.1}$$

$$\boldsymbol{\beta}' = \pi \tanh(\lambda_2 \boldsymbol{\beta}) + \pi, \tag{B.2}$$

where (λ_1, λ_2) are scaling hyper-parameters.

Stats		FB15k	FB15k-237	NELL995
Entities		14,951	14,505	63,361
Relations		1,345	237	200
Triples (Edges)	Train	483,142	272,115	114,213
	Valid	50,000	17,526	14,324
	Test	59,071	20,438	14,267
	Total	592,213	310,079	142,804

Table 5: Statistics of three datasets, reported from Ren and Leskovec (2020)

Split	Query syntaxes	FB15k	FB15k-237	NELL995
Train	1p/2p/3p/2i/3i	273,710	149,689	107,982
	2in/3in/inp/pin/pni	27,371	14,968	10,798
Valid	1p	59,097	20,101	16,927
	Others (Each)	8,000	5,000	4,000
Test	1p	67,016	22,812	17,034
	Others (Each)	8,000	5,000	4,000

Table 6: Statistics of query structures preprocessed by Ren and Leskovec (2020).

C Distance score functions

Continuing the calculation of distance score function in Sec. 4.3, the calculations of the outside with inside distances and axis distance are as follows:

$$d_o = \Big\|\min\{d_l, d_u\}\Big\|_1, \; d_i = \Big\|\min\{d_\alpha, d_\beta\}\Big\|_1,$$
$$d_a = \Big\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_q\Big\|_1, \tag{C.1}$$

where $||\cdot||_1$ denotes the L_1 norm, the upper bound $(\mathbf{u} = \boldsymbol{\alpha}_q + \frac{\boldsymbol{\beta}_q}{2})$ and the lower bound $(\mathbf{l} = \boldsymbol{\alpha}_q - \frac{\boldsymbol{\beta}_q}{2})$ are of the query $(\mathbf{q})$; $d_l = |1 - \cos(\boldsymbol{\alpha} - \mathbf{l})|$ and $d_u = |1 - \cos(\boldsymbol{\alpha} - \mathbf{u})|$ is the lower and upper bound outside distance respectively, $d_\alpha = |1 - \cos(\boldsymbol{\alpha} - \boldsymbol{\alpha}_q)|$ and $d_\beta = |1 - \cos(\frac{\boldsymbol{\beta}_q}{2})|$ is the axis and the aperture inside distance respectively.

Dataset	d **embed dim**	b **batch size**	n **negative sampling**	m **max steps**	γ	l **learning rate**	ψ **weight distance**	λ_1	λ_2	λ
FB15k	400	512	128	450k	30	0.00005	0.9	1.0	2.0	0.02
FB15k-237	400	512	128	350k	20	0.00005	0.9	1.0	2.0	0.02
NELL995	400	512	128	350k	20	0.00005	0.9	1.0	2.0	0.02

Table 7: Found hyper-parameters for the main results.

Dataset	**AVG**$_p$	**AVG**$_n$	**1p**	**2p**	**3p**	**2i**	**3i**	**ip**	**pi**	**2u**	**up**	**2in**	**3in**	**inp**	**pin**	**pni**
FB15k	53.0 $\pm$ 0.3	16.0 $\pm$ 0.1	80.8 $\pm$ 0.3	38.2 $\pm$ 0.3	30.7 $\pm$ 0.2	67.0 $\pm$ 0.3	75.1 $\pm$ 0.2	41.7 $\pm$ 0.5	52.1 $\pm$ 0.3	57.1 $\pm$ 0.5	34.6 $\pm$ 0.4	20.5 $\pm$ 0.1	19.5 $\pm$ 0.1	14.5 $\pm$ 0.04	9.2 $\pm$ 0.2	16.1 $\pm$ 0.2
FB15k-237	24.1 $\pm$ 0.1	6.7 $\pm$ 0.05	44.2 $\pm$ 0.1	13.0 $\pm$ 0.1	10.7 $\pm$ 0.1	33.8 $\pm$ 0.2	47.0 $\pm$ 0.2	17.0 $\pm$ 0.1	25.1 $\pm$ 0.2	15.5 $\pm$ 0.2	10.7 $\pm$ 0.2	6.9 $\pm$ 0.1	10.6 $\pm$ 0.2	7.9 $\pm$ 0.1	4.0 $\pm$ 0.04	4.3 $\pm$ 0.04
NELL995	30.4 $\pm$ 0.2	6.7 $\pm$ 0.1	58.2 $\pm$ 0.1	20.5 $\pm$ 0.2	17.0 $\pm$ 0.2	41.8 $\pm$ 0.3	50.7 $\pm$ 0.2	22.9 $\pm$ 0.4	28.6 $\pm$ 0.3	18.8 $\pm$ 0.1	15.5 $\pm$ 0.3	6.2 $\pm$ 0.1	8.0 $\pm$ 0.1	11.8 $\pm$ 0.1	3.5 $\pm$ 0.04	4.2 $\pm$ 0.1

Table 8: Error bars of **MRR** (%) for the main results of SConE. ($\pm$) is for standard deviation.

D Experimental setups

D.1 Datasets

Following experimental settings in Ren and Leskovec (2020) for the training and evaluation process, which pre-processed datasets (FB15k Bollacker et al. (2008), FB15k-237 Toutanova and Chen (2015) and NELL995 Xiong et al. (2017)) and publicly available at this link [3] , Table 5 shows statistics of these datasets regarding the number of entities, the number of relations and the number of triples. In addition, Table 6 shows the number of queries in different structures for the training/validation/test set.

D.2 Training and evaluation settings: Further details, hyper-parameters and error bars

Following the original work in (Ren and Leskovec, 2020) [4], we implement all experiments using Pytorch as Deep Learning framework under Python. For each experiment, we conduct it on a single NVIDIA Tesla V100 GPU, in the UWA Kaya High Performance Computing (HPC) cluster. For the settings in the relation projection network, we use a three-layer **MLP** using 1600 dimension for hidden layers and Rectified Linear Units (ReLU) for the activation function. Further, following the found hyper-parameters in ConE (Zhang et al., 2021): $\lambda_1 = 1.0$, $\lambda_2 = 2.0$, $\lambda = 0.02$, batch size $b = 512$ and negative sampling size $n = 128$, we use these in all experiments. With regard to other hyper-parameters, we search for the best performance in MRR. Specifically, (γ) involving in the loss function is in {20, 30}, the learning rate (l) is in $\{1e^{-4}, 5e^{-5}\}$. Table 7 shows found hyper-parameters of the main results in Table 1. For obtaining error bars of the main results, we run the model five times, each uses different random seed in $\{0, 10, 100, 1000, 10000\}$ (see Table 8 for further details).

[3]https://github.com/snap-stanford/KGReasoning

[4]https://github.com/snap-stanford/KGReasoning,licensed under the MIT License

E Additional results

E.1 Error bars for the main results

Table 8 shows error bars of the average MRR (in percentage) for the main results of SConE reported in Table 1 (see random seed settings as described in Appendix. D.2). We compute the standard deviation (std) of results from five experiments using each of the three dataset (FB15k, FB15k-237 and NELL995). Overall, the error bar of the average MRR is low in all query structures and in average for EPFO and negation queries, which demonstrates the stability of model performance.

E.2 Comparison results with fuzzy logic-based models

Table 9 shows comparison in the average MRR (in percentage) of SConE with that of other fuzzy logic-based models. In the FB15k-237 dataset, GNN-QE achieves state-of-the-art performance of answering complex logical queries. Compared to other models (ENeSy and FuzzQE), the performance of SConE is nearly to that of these logic-based models. With regard to the NELL995 dataset, although SConE achieves the lowest performance in answering negation queries, the performance of SConE reaches its highest in answering non-

Dataset	Model	AVG$_p$	AVG$_n$	1p	2p	3p	2i	3i	ip	pi	2u	up	2in	3in	inp	pin	pni
FB15k-237	CQD-CO	21.8	-	**46.7**	9.5	6.3	31.2	40.6	16.0	23.6	14.5	8.2	-	-	-	-	-
	CQD-Beam	22.3	-	**46.7**	11.6	8.0	31.2	40.6	18.7	21.2	14.6	8.4	-	-	-	-	-
	FuzzQE	24.2	8.5	42.2	13.3	10.2	33.0	47.3	18.9	26.2	15.6	10.8	9.7	12.6	7.8	5.8	6.6
	ENeSy	24.5	8.5	44.7	11.7	8.6	34.8	50.4	**19.7**	27.6	14.2	8.4	**10.1**	10.4	7.6	6.1	**8.1**
	GNN-QE	**26.8**	**10.2**	42.8	**14.7**	**11.8**	**38.3**	**54.1**	18.9	**31.1**	**16.2**	**13.4**	10.0	**16.8**	**9.3**	**7.2**	7.8
	SConE	24.1	6.7	44.2	13.0	10.7	33.8	47.0	17.0	25.1	15.5	10.7	6.9	10.6	7.9	4.0	4.3
NELL995	CQD-CO	28.8	-	**60.4**	17.8	12.7	39.3	46.6	22.0	30.1	17.3	13.2	-	-	-	-	-
	CQD-Beam	28.6	-	**60.4**	**20.6**	11.6	39.3	46.6	23.9	25.4	17.5	12.2	-	-	-	-	-
	GNN-QE	28.9	9.7	53.3	18.9	14.9	**42.4**	**52.5**	18.9	**30.8**	15.9	12.6	9.9	**14.6**	11.4	6.3	6.3
	FuzzQE	29.3	8.0	58.1	19.3	15.7	39.8	50.3	21.8	28.1	17.3	13.7	8.3	10.2	11.5	4.6	5.4
	ENeSy	29.4	**9.8**	59.0	18.0	14.0	39.6	49.8	**24.8**	29.8	16.4	13.1	**11.3**	8.5	11.6	**8.6**	**8.8**
	SConE	**30.4**	6.7	58.2	20.5	**17.0**	41.8	50.7	22.9	28.6	**18.8**	**15.5**	6.2	8.0	**11.8**	3.5	4.2

Table 9: Comparison the average **MRR** (%) of SCone with that of logic-based embedding models (CQD-CO, CQD-Beam, FuzzQE, ENeSy, GNN-QE). Union queries ($2u/up$) are in DNF forms. Results of CQD-CO, CQD-Beam, GNN-QE are taken from (Zhu et al., 2022).

Delta	AVG	AVG$_p$	AVG$_n$	2i	3i	ip	pi	2in	3in	inp	pin	pni
$\delta = 0.0$	17.8	**24.4**	5.9	34.5	48.6	16.8	26.6	5.3	9.3	7.9	3.6	3.6
$\delta = 0.1$	**17.9**	**24.4**	6.1	34.3	48.2	17.0	26.2	5.6	9.7	7.8	3.8	3.8
$\delta = 0.5$	**17.9**	24.1	**6.8**	33.9	47.0	16.9	25.1	6.9	10.8	7.9	4.0	4.4
$\delta = 0.9$	17.8	24.3	6.2	33.9	47.7	16.8	26.1	6.0	9.8	7.8	3.8	3.8
$\delta = 1.0$	17.8	24.3	6.1	34.1	47.6	16.8	26.3	5.8	9.7	7.8	3.7	3.7

Table 10: The effect of weight axis (δ) for the none type of intersection queries on average **MRR** (%) (see Eq. (4.3)).

negation queries among other logic-based models. Particularly, there is a significant improvement in the average MRR regarding union queries, compared to that in other models.

F Further sensitivity analysis - Weight of semantic axis for none intersection type

Table 10 shows the performance of SConE w.r.t. different weights ($\delta \in [0, 1]$) of semantic axis in the case of none type intersection (see Eq. (4.3)). We conduct five experiments using different weights in $\{0.0, 0.1, 0.5, 0.9, 1.0\}$. When ($\delta = 0.0$) or ($\delta = 1.0$), the semantic axis of an intersection query corresponds to the lower bound of one sector-cone or the upper bound of the other sector-cone. When ($\delta = 0.1$) or ($\delta = 0.9$), the spatial position of this semantic axis is close the lower bound of one sector-cone or the upper bound of the other sector-cone respectively. In a special case when ($\delta = 0.5$), the semantic axis is in the middle of the two bounds.

Overall, the average MRR (**AVG**) of SConE for both non-negation and negation queries is similar from one to another in all different weights (δ). However, there is a slight difference between **AVG**$_p$ for non-negation queries and **AVG**$_n$ for negation queries. When ($\delta = 0.1$), SConE achieves the highest **AVG**$_p$ but not for **AVG**$_n$. In contrast, when ($\delta = 0.5$), SConE achieves the highest **AVG**$_n$ but not for **AVG**$_p$. Since there is a slight difference in **AVG**$_p$ using ($\delta = 0.1$) and ($\delta = 0.5$) but there is highly difference in **AVG**$_n$ using these weights, we select the special case with ($\delta = 0.5$) or the middle semantic axis of intersection query and report the main results. Further, we observe that there is no significant difference in the average MRR (**AVG**) of model performance for both non-negation and negation queries (see second column of Table 10). Thus, any semantic axes (or cones) between the two mentioned bounds can be considered as the semantic axis of intersection query. Note that the aperture of intersection query in the case of none intersection type is equivalent to zero.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section Limitations (after Section 6 Conclusion), Appendix A.3

☑ A2. Did you discuss any potential risks of your work?
Section Ethics Statement (after Section Limitations)

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

☒ 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?

Section 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5.3, Appendix D.2

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?
Section 5.1, Appendix D.1, D.2

☑ 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?
Appendix D.2, Appendix E.1

☐ 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.)?
Not applicable. Appendix D.2

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.