Generating $\mathcal{SROI}^-$ Ontologies via Knowledge Graph Query Embedding Learning

Yunjie He[ab,*], **Daniel Hernandez**[a], **Mojtaba Nayyeri**[a], **Bo Xiong**[a], **Yuqicheng Zhu**[ab], **Evgeny Kharlamov**[bd] **and**
Steffen Staab[ac]

[a]University of Stuttgart
[b]Bosch Center for Artificial Intelligence
[c]University of Southampton
[d]University of Oslo
ORCID (Yunjie He): https://orcid.org/0009-0005-4461-2863, ORCID (Daniel Hernandez):
https://orcid.org/0000-0002-7896-0875, ORCID (Steffen Staab): https://orcid.org/0000-0002-0780-4154

Abstract. Query embedding approaches answer complex logical queries over incomplete knowledge graphs (KGs) by computing and operating on low-dimensional vector representations of entities, relations, and queries. However, current query embedding models heavily rely on excessively parameterized neural networks and cannot explain the knowledge learned from the graph. We propose a novel query embedding method, AConE, which explains the knowledge learned from the graph in the form of $\mathcal{SROI}^-$ description logic axioms while being more parameter-efficient than most existing approaches. AConE associates queries to a $\mathcal{SROI}^-$ description logic concept. Every $\mathcal{SROI}^-$ concept is embedded as a cone in complex vector space, and each $\mathcal{SROI}^-$ relation is embedded as a transformation that rotates and scales cones. We show theoretically that AConE can learn $\mathcal{SROI}^-$ axioms, and defines an algebra whose operations correspond one-to-one to $\mathcal{SROI}^-$ description logic concept constructs. Our empirical study on multiple query datasets shows that AConE achieves superior results over previous baselines with fewer parameters. Notably on the WN18RR dataset, AConE achieves significant improvement over baseline models. We provide comprehensive analyses showing that the capability to represent axioms positively impacts the results of query answering.

1 Introduction

Knowledge Graphs (KGs) such as Wikidata [21], Freebase [4], and YAGO [18] represent real-world facts as sets of *triples* of the form (s, p, o) which encode atomic assertions as $p(s, o)$. Graph database engines can store and query KGs efficiently using query languages such as SPARQL [1] that can express a variety of queries that result of combining atomic queries called *triple patterns*.

The first step of querying KGs is answering triple patterns with the stored triples. However, when querying incomplete KGs, some triples are not explicitly available in the triple store and thus are not included in the answers, nor into the intermediary results of an answer. To provide plausible answers beyond what is known, these missing triples must be inferred. Figure 1 shows an example, where the available triples (L.Messi, playsFor, ArgentinaNFT), and
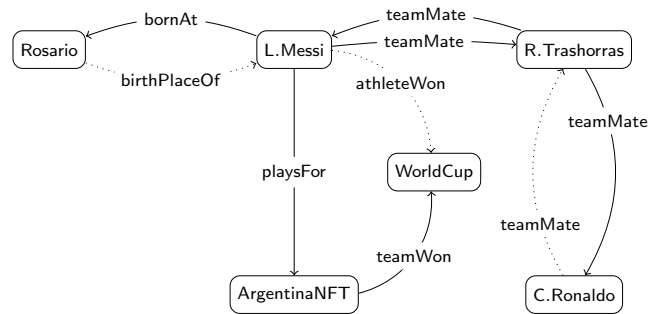
Figure 1: An example of an incomplete KG. The nodes represent entities, the edges with solid lines represent known atomic statements, whereas the edges with dotted lines represent missing atomic statements that must be inferred.

(ArgentinaNFT, teamWon, WorldCup) should suggest that triple (L.Messi, athleteWon, WorldCup) is a missing triple. KG embedding methods can predict these missing triples [5, 27, 13] by learning how to embed entities and relations into vector representations, which can be points or more complex geometric objects. These methods use these embeddings to answer triple patterns by computing plausibility scores by applying geometric operations.

Query embedding methods [16, 29, 15] go beyond querying triple patterns. They provide plausible answers to queries that combine triple patterns into first-order logic queries with logical connectives (e.g., negation ($\neg$), conjunction ($\wedge$), and disjunction ($\vee$)). However, current query embedding approaches are restricted to queries with a single unquantified variable and are called *tree-form queries* because their *computation graph* is a tree [17]. The tree-form queries correspond to the $\mathcal{SROI}^-$ description logic concepts that do not include the symbols $\top$ (the concept for all entities) nor $\bot$ (the concept for no elements), nor concept names (e.g., Athlete or Team), but nominals (i.e., concepts with a unique element like $\{$C.Ronaldo$\}$).

Example 1. *The query asking for the birthplaces of the athletes who have won either the World Cup or the Europe Cup, but do not play in the same team as* C.Ronaldo *can be expressed as the* $\mathcal{SROI}^-$

concept

$$C \equiv \exists \mathsf{birthPlaceOf}.(\neg(\exists \mathsf{teamMate}^-.\{\mathsf{C.Ronaldo}\}) \sqcap$$
$$(\exists \mathsf{athleteWon}.\{\mathsf{WorldCup}\} \sqcup$$
$$\exists \mathsf{athleteWon}.\{\mathsf{EuroCup}\})),$$

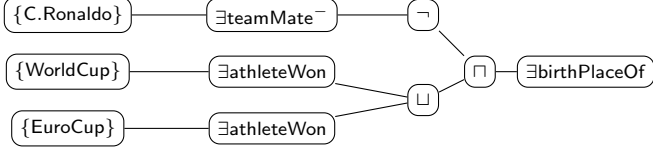*whose computation graph is depicted in Figure 2.*

Figure 2: Computation graph for the concept C in Example 1.

Query embedding methods [16, 29, 15] learn a model M to predict the answers to queries by embedding entities and relations as geometrical objects of a vector space. These geometrical objects are used to recursively compute geometrical objects for all nodes in the computation graph of the query. Finally, the query embeddings are compared with the entity embeddings of the candidate answers to the query by using a similarity function that represents the plausibility of an answer.

The quality of query embedding methods depends on their ability to represent the *logical patterns* satisfied by a knowledge graph. For example, some relations are symmetric (e.g., teamMate) while others are antisymmetric (e.g., playsFor); some relations are the inverse of other relations (e.g., bornAt and birthPlaceOf) and some relations may be composed by others (e.g., if an athlete x playsFor a team y and y teamWon a cup z, then x athleteWon z). As these logical patterns influence the interplay of entities and relations, several works have studied their effect on KG embeddings [19, 12, 24] and demonstrated that an embedding's ability to support them improves its link prediction quality. For query embeddings, similar support of logical patterns in the embedding space is still lacking.

Given a model M of query embedding method, we can distinguish two ways in which M can exploit a logical pattern P. The first is the ability to *predict* answers that are entailed by P. The second is the ability of model M to *explain* such predictions by inferring P as an axiom that can be obtained from the geometrical relations between the objects in the embedding space.

Although query embedding methods have achieved great success in predicting query results over incomplete data, they fail to explain the learned knowledge because, to achieve better results, they endow geometric operations with neural network operations that impede explainability. For example, the method BetaE [15] represents queries q with multidimensional beta distributions $\boldsymbol{q}$, and the set of answers to q in M, denoted $[\![q]\!]_M$, consists of the entities e such that $\boldsymbol{e} \in \boldsymbol{q}$. A query $q(y) = \exists x(q_1(x) \wedge r(x,y))$, extending a given query $q_1(x)$ with a relation r, is embedded as a beta distribution $\boldsymbol{q}$ that results from applying a neural network over the distribution $\boldsymbol{q_1}$ for query q_1. The use of neural networks to represent relations hinders the inference of axioms like playsFor ∘ teamWon ⊑ athleteWon. On the other hand, BoxEL [23] and Box²EL [9] embed description logic concepts, but are limited to reduced description logics ($\mathcal{EL}$ and $\mathcal{EL}^{++}$).

In this paper, we propose a novel query embedding model, AConE, that can explain several logical patterns expressed as $\mathcal{SROI}^-$ axioms learned for query answering tasks. Being more parameter-efficient than most existing approaches [29, 7, 30, 16], AConE re-duces the dependency on neural networks by translating logical operators to a simpler algebraic structure. To achieve this explainability, AConE embeds each $\mathcal{SROI}^-$ concept as a multidimensional cone in the complex vector space, and relations as scaling and rotations of cones. Then, each logical operator is translated into geometric operations in embedding space. By doing so, our method can generate $\mathcal{SROI}^-$ ontologies by learning query embeddings.

In summary, this paper makes the following contributions:

1. We formalize the notions of tree-form logical query in terms of $\mathcal{SROI}^-$ concepts, and logical pattern in terms of $\mathcal{SROI}^-$ axioms. We propose an algebra of cones in the complex plane, and theoretically identify the subset of the $\mathcal{SROI}^-$ axioms that can be represented with the cone algebra models (Section 4).
2. We present criteria where a multicone embedding expresses six different logical patterns, namely role containment, composition, transitivity, inverse, symmetry, and asymmetry (Section 5).
3. We propose a novel method, AConE, which embeds $\mathcal{SROI}^-$ concepts as cones in the complex plane (Section 6). This technique allows us to leverage the rotation operator through Euler's formula while maintaining the geometric representation of concepts and concept operators to allow the explanation of logical patterns as $\mathcal{SROI}^-$ axioms that the embedding explicitly encodes.
4. We show that modeling cones in the complex plane, coupled with the rotation operator as a complex product, is more parameter-efficient than modeling cones in the 2D real space with multi-layer neural operators (Section 8).
5. We create new datasets and dataset splits (Section 7.3) to conduct a more detailed analysis of the influence of patterns in complex query answering, providing a finer-grained evaluation, and more in-depth insights into the often-overlooked problem of pattern inference in complex query answering.
6. Our experiments show that AConE outperforms state-of-the-art baselines that represent query regions using vectors, geometries, or distributions, using geometric operations to model logical operations one-to-one (Section 7.2).

2 Related Work

Logical patterns in knowledge graphs. KG embedding methods aim to learn KG representations that capture latent structural and logical patterns [5, 27, 20, 19, 22]. In particular, RotatE [19] captures a broad range of logical patterns, such as symmetry, inversion, and composition, among others. While these KG embedding methods excel at predicting links, they cannot answer first-order logical queries. BoxEL [23] and Box²EL [9] embed description logic concepts, but are limited to reduced description logics ($\mathcal{EL}$ and $\mathcal{EL}^{++}$).

Query answering. Path-based [26, 11], neural [7, 16, 29, 15, 10], and neural-symbolic [2, 31, 30] methods have been developed to answer (subsets of) queries. Among these methods, geometric and probabilistic query embedding approaches [7, 16, 29, 15] provide an effective way to answer tree-form queries over incomplete and noisy KGs. This is done by representing entity sets as geometric objects or probability distributions, such as boxes [16], cones [29], or Beta distribution [15], and performing neural logical operations directly on them. The Graph Query Embedding (GQEs) [7] was first proposed to answer only conjunctive queries via modeling the query q as single vector $\boldsymbol{q}$ through neural translational operators. However, modeling a query as a single vector limits the model's expressiveness in modeling multiple entities. Query2Box [16] remedies this flaw by modeling entities as points within boxes. This allows Query2Box to

predict the intersection of entity sets as the intersection of boxes in vector space. ConE [29] was proposed as the first geometry-based query embedding method that can handle negation via embedding the set of entities (query embedding) as cones in Euclidean space.

All of the above query embedding methods commonly apply multi-layer perceptron networks for selecting answer entities of atomic queries by relation and performing logical operations. Such methods suffer from two problems. Firstly, their ability to capture logical patterns in KGs remains unclear due to the limited explainability of neural networks. Secondly, a large number of parameters need to be trained for an outstanding model performance. Our proposed method, AConE, overcome these issues by providing a one-to-one mapping between logical and geometrical operators.

3 Preliminaries

The Description Logic $\mathcal{SROI}^-$. We next present the standard $\mathcal{SROI}^-$ syntax and semantics and assume standard semantics as defined in Baader et al. [3]. For these definitions, we assume three pairwise disjoint sets **C**, **R**, and **I**, whose elements are respectively called *concept names*, *relation names*, and *individual names*.

Definition 1 ($\mathcal{SROI}^-$ Concept Descriptions). $\mathcal{SROI}^-$ *concept descriptions C and relation descriptions R are defined by the following grammar*

$$C ::= \top \mid A \mid \{a\} \mid \neg C \mid C \sqcap C \mid \exists R.C$$
$$R ::= r \mid r^-$$

where the symbol $\top$ is a special concept name, and symbols A, a, and r stand for concept names, individual names, and relation names, respectively. Concept descriptions $\{a\}$ are called nominals.

Given two concept descriptions C and D, the expression $C \sqsubseteq D$ is a concept-axiom. *Given the relation descriptions $R, S, R_1, \ldots, R_n$ (with $n \geq 1$), the expressions $\rho_1 \circ \cdots \circ \rho_{n-1} \sqsubseteq \rho_n$, $\mathsf{Disj}(R, S)$, $\mathsf{Trans}(R)$, $\mathsf{Ref}(R)$, $\mathsf{Irref}(R)$, $\mathsf{Sym}(R)$, and $\mathsf{Asym}(R)$ are relation-axioms. Given two individual names $a, b \in \mathbf{I}$, a concept description C and a relation description ρ, $a(C)$ is a* concept-assertion *and $\rho(a, b)$ is a* relation-assertion.

We write $C \equiv D$ as an abbreviation for two axioms $C \sqsubseteq D$ and $D \sqsubseteq C$, and likewise for $\rho_1 \equiv \rho_2$. We write $\bot$, $C \sqcup D$, $\forall \rho.C$ as abbreviations for $\neg \top$, $\neg(\neg C \sqcap \neg D)$ and $\neg \exists \rho. \neg C$, respectively.

An $\mathcal{SROI}^-$ *knowledge base* (or *ontology*) $\mathcal{K}$ is a triple $(\mathcal{R}, \mathcal{T}, \mathcal{A})$ where $\mathcal{R}$ is a finite set of relation-axioms called the *RBox*, $\mathcal{T}$ is a finite set of concept-axioms called the *TBox*, and $\mathcal{A}$ is a finite set of assertions called the *ABox*.

An *interpretation* $\mathcal{I}$ is a pair $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a set $\Delta^{\mathcal{I}}$, called the *domain*, and a function $\cdot^{\mathcal{I}}$ such that we have for each individual name $a \in \mathbf{I}$, an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$; for each concept name $A \in \mathbf{C}$, a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$; and for each relation name $r \in \mathbf{R}$, a relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a *model* of a knowledge base $\mathcal{K}$ if and only if all the axioms and assertions in $\mathcal{K}$ are satisfied according to the standard semantics defined in Baader et al. [3]. Given two knowledge bases $\mathcal{K}_1$ and $\mathcal{K}_2$, $\mathcal{K}_1$ *entails* $\mathcal{K}_2$, denoted $\mathcal{K}_1 \models \mathcal{K}_2$, if and only if every model $\mathcal{I}$ of $\mathcal{K}_1$ is also a model of $\mathcal{K}_2$.

Knowledge graphs and queries. We next define knowledge graphs in terms of $\mathcal{SROI}^-$ knowledge bases, and tree-form queries in terms of $\mathcal{SROI}^-$ concepts.

Definition 2 (Knowledge Graph). *A knowledge graph G is a $\mathcal{SROI}^-$ knowledge base $\mathcal{K}$ whose RBox is empty, and its $\mathcal{T}$ contains a unique axiom $\top \sqsubseteq \{a_1\} \sqcup \cdots \sqcup \{a_n\}$, where $\{a_1, \ldots, a_n\}$ is the set of all individuals names occurring in the ABox. This axiom is called* domain-closure assumption.

Definition 3 (Tree-form query). *Given a knowledge graph G, and an individual name x that does not occur in G, a tree-form query q is an assertion $C(x)$ where C is a $\mathcal{SROI}^-$ concept description. The answers to query q, are all individuals a occurring in G such that $G \models C(a)$.*

Informally, the query answering task over incomplete data consists of predicting answers to a query q over a knowledge graph pattern G given only a subset $G' \subset G$. Knowledge graph embeddings to generalize the knowledge on G' to predict answers on G.

4 $\mathcal{SROI}^-$ concepts and Cone Algebra

In this section, we present an algebra of cones in the complex plane, called the Cone Algebra, we show the correspondence between this algebra and $\mathcal{SROI}^-$ concepts, and we identify the subset of the $\mathcal{SROI}^-$ axioms that are expressible with this algebra.

Definition 4 (Cone). *A cone $\mathsf{C}(\alpha, \beta)$ is a region in the complex plane $\mathbb{C}$ determined by a pair of angles $(\alpha, \beta) \in \mathbb{R}^2$ as follows:*

$$\mathsf{C}(\alpha, \beta) = \{e^{i\theta} : \alpha \leq \theta \text{ and } \theta \leq \beta\}.$$

The empty cone, *denoted $\mathsf{C}_\perp$, is the cone such that $\alpha > \beta$. A singleton cone with angle α, denoted C_α, is a cone such that $\alpha = \beta$. A proper cone, denoted $\mathsf{C}_{\alpha \to \beta}$, is a cone where $\alpha + 2\pi > \beta$. A full cone, denoted $\mathsf{C}_\top$, is the cone such that $\alpha + 2\pi \leq \beta$.*

Notice that $e^{i\theta} = e^{i\theta + 2k\pi}$, for every natural number k. Thus, the same cone can be determined with multiple combination of angles.

Notice that the intersection is not closed on the set of cones. Indeed, the region $\mathsf{C}(0, \frac{3}{2}\pi) \cap \mathsf{C}(\pi, \frac{5}{2}\pi)$ is not a cone. This hinders the use of cones to represent $\mathcal{SROI}^-$ concepts. To overcome this limitation, we can embed concepts on sets of cones.

Definition 5 (Multicone algebra). *A multicone $\mathsf{MC}(\mathcal{C})$ is a region determined by a set of cones $\mathcal{C}$ as follows:*

$$\mathsf{MC}(\mathcal{C}) = \bigcup_{\mathsf{C}(\alpha, \beta) \in \mathcal{C}} \mathsf{C}(\alpha, \beta)$$

The multicone algebra *is the algebra over the set of multicones defined by the binary operations $\cap$ and $\cup$ that are defined as the set operations over the multicon regions.*

Proposition 1. *The multicone algebra is a field with sum $\cup$ and multiplication $\cap$, where the identity of $\cup$ is $\mathsf{MC}_\perp = \mathsf{MC}(\{\mathsf{C}_\perp\})$, the identity $\cap$ is $\mathsf{MC}_\top = \mathsf{MC}(\{\mathsf{C}_\top\})$, and the additive and multiplicative inverse of a multicone $\mathsf{MC}(\mathcal{C})$, denoted $\neg \mathsf{MC}(\mathcal{C})$, is the multicone $\mathsf{MC}_\top \setminus \mathsf{MC}(\mathcal{C})$.*

Definition 6 (Rotation algebra). *Given a triple $(\theta, \gamma, \delta) \in \mathbb{R}^3$ where $0 \leq \gamma \leq 2\pi$, a rotation $\mathsf{R}\langle \theta, \gamma, \delta \rangle$ is a function that maps every singleton and proper cone $\mathsf{C}(\alpha, \beta)$ to the cone $\mathsf{C}(\alpha', \beta')$ such that*

$$\beta' + \alpha' = \theta(\beta + \alpha), \qquad \beta' - \alpha' = \gamma(\beta - \alpha) + \delta,$$

and that maps the empty and the full cone to themselves:

$$\mathsf{R}\langle \theta, \gamma, \delta \rangle(\mathsf{C}_\perp) = \mathsf{C}_\perp, \qquad \mathsf{R}\langle \theta, \gamma, \delta \rangle(\mathsf{C}_\top) = \mathsf{C}_\top.$$

Table 1: Multicone Embedding Semantics.

Concept	Semantics
$\top$	$MC_\top$
$\{a\}$	$MC(\{C_\theta\})$ where $a^\mathcal{E} = e^{i\theta}$
$C \sqcap D$	$C^\mathcal{E} \cap D^\mathcal{E}$
$\neg C$	$MC_\top \setminus C^\mathcal{E}$
$\exists R.C$	$R^\mathcal{E}(C^\mathcal{E})$
r^-	$(r^\mathcal{E})^-$
$R \circ S$	$R^\mathcal{E} \circ S^\mathcal{E}$
$A(a)$	$a^\mathcal{E} \in A^\mathcal{E}$
$R(a,b)$	$(a^\mathcal{I}, b^\mathcal{E}) \in R^\mathcal{E}$
$C \sqsubseteq D$	$C^\mathcal{E} \subseteq D^\mathcal{E}$
$R_1 \circ \cdots \circ R_n \sqsubseteq S$	$(R_1 \circ \cdots \circ R_n)^\mathcal{E} \subseteq S^\mathcal{E}$
$\mathsf{Disj}(R, S)$	$R^\mathcal{E} \cap S^\mathcal{E} = \emptyset$
$\mathsf{Trans}(R)$	$(R \circ R)^\mathcal{E} \subseteq R^\mathcal{E}$
$\mathsf{Ref}(R)$	$\{(a,a) \mid a \in \mathbf{I}\} \subseteq R^\mathcal{E}$
$\mathsf{Irref}(R)$	$\{(a,a) \mid a \in \mathbf{I}\} \cap R^\mathcal{E} = \emptyset$
$\mathsf{Sym}(R)$	for all $a, b \in \mathbf{I}$, if $(a,b) \in R^\mathcal{E}$ then $(b,a) \in R^\mathcal{E}$
$\mathsf{Asym}(R)$	for all $a, b \in \mathbf{I}$, if $(a,b) \in R^\mathcal{E}$ then $(b,a) \notin R^\mathcal{E}$

The rotation parameters θ, γ, and δ are called the rotation angle *and the* aperture factor*, and the* aperture adding*. We call* aperture-multiplicative *and* aperture-additive *rotations to rotations of the respective forms $R\langle\theta, \gamma, 0\rangle$ and $R\langle\theta, 1, \delta\rangle$.*

We call rotation algebra *to be the algebraic structure whose ground set is the set of rotations over cones, and has a binary operation $\circ$ denoting function composition (i.e., $(f \circ g)(x) = g(f(x))$).*

We write $R\langle\theta, \gamma, \delta\rangle^- = R\langle-\theta, \frac{1}{\gamma}, -\delta\rangle$, and with a slight abuse of notation, given a multicone $MC(\mathcal{C})$, we write

$$R\langle\theta, \gamma, \delta\rangle(MC(\mathcal{C})) = MC(\{R\langle\theta, \gamma, \delta\rangle(C(\alpha, \beta)) : C(\alpha, \beta) \in \mathcal{C}\}).$$

Definition 7 (Multicone Embedding). *A multicone embedding is a function $\mathcal{E}$ that maps each individual name $a \in \mathbf{I}$ to a complex number $a^\mathcal{E} = e^{i\theta} \in C_\top$, each concept name $A \in \mathbf{C}$ to a multicone $A^\mathcal{E}$, and each relation name $r \in \mathbf{R}$ to a rotation $r^\mathcal{E} = R\langle\theta, \gamma, \delta\rangle$. The cone embedding of a knowledge graph G is a cone embedding restricted to the individual names and relation names occurring in knowledge graph G.*

Given a multicone embedding $\mathcal{E}$, Table 1 defines the multicone $C^\mathcal{E}$ corresponding to a $\mathcal{SROI}^-$ concept description C. The definition of the multicone semantics for relations, assertions and axioms is straightforward. For example, the embedding of $R \circ S$ is the relation $\{(a,c) : (a,b) \in R^\mathcal{E}, (b,c) \in S^\mathcal{E}\}$, and an $\mathcal{E} \models C(a)$ if and only if $a^\mathcal{E} \in C^\mathcal{E}$, $\mathcal{E} \models R(a,b)$ if and only if $(a^\mathcal{E}, b^\mathcal{E}) \in R^\mathcal{E}$, and $C \models D$ if and only if $C^\mathcal{E} \subseteq D^\mathcal{E}$.

The full cone. Since $C_\top$ is an absorbing element for the rotation algebra, multicone embeddings infer wrong axioms $\top \sqsubseteq \exists R.\top$.

The rotation commutativity. In certain cases, the order in which rotations are applied does not affect the resulting cone.

Proposition 2. *Given two aperture-multiplicative rotations $R\langle\theta_1, \gamma_1, 0\rangle$ and $R\langle\theta_2, \gamma_2, 0\rangle$ such that $\gamma_1 \leq 1$ and $\gamma_2 \leq 1$, then*

$$R\langle\theta_1, \gamma_1, 0\rangle \circ R\langle\theta_2, \gamma_2, 0\rangle = R\langle\theta_2, \gamma_2, 0\rangle \circ R\langle\theta_1, \gamma_1, 0\rangle.$$

The issue with the commutativity is that if two relation names r and s are embedded with aperture-multiplicative rotations $R\langle\theta_r, \gamma_r, 0\rangle$ and $R\langle\theta_s, \gamma_s, 0\rangle$ with $\gamma_r \leq 1$ and $\gamma_s \leq 1$, then we can infer that $r \circ s \equiv s \circ r$. So, the commutativity of these rotations may introduce a bias into the models that can lead to wrong predictions.

This bias favoring commutativity does not hold if $\gamma > 1$, as the following counter example shows.

$$(R\langle 0, 2, 0\rangle \circ R\langle 0, \tfrac{1}{2}, 0\rangle)(C(0, \pi)) = C_\top,$$
$$(R\langle 0, \tfrac{1}{2}, 0\rangle \circ R\langle 0, 2, 0\rangle)(C(0, \pi)) = C(0, \pi).$$

However, rotations with $\gamma > 1$ generate larger cones (sometimes full cones), and thus can lead to a reduced prediction accuracy.

Aperture-additive rotations are not commutativity because, for every $\delta \neq 0$, we can find a cone C such that $R\langle\theta, 1, \delta\rangle(C)$ is either $C_\top$ or $C_\perp$ (which are absorbing elements). However, under certain conditions, commutativity holds. For example, if u, v, and w are three positive real numbers with $u + v \leq w$, and $R_1 = R\langle\theta_1, 1, u\rangle$, $R_2 = R\langle\theta_2, 1, v\rangle$, $R_3 = R\langle\theta_1, 1, -u\rangle$, $R_4 = R\langle\theta_2, 1, -v\rangle$, then

$$(R_1 \circ R_2)(C(0, 2\pi - w)) = (R_2 \circ R_1)(C(0, 2\pi - w)),$$
$$(R_3 \circ R_4)(C(0, w)) = (R_4 \circ R_3)(C(0, w)).$$

Under these circumstances, there is a bias that favored the inference of axioms $\exists R_1(\exists R_2.C) \equiv \exists R_2(\exists R_1.C)$. Hence, both aperture-multiplicative and aperture-additive rotations are biased.

Distributivity of the existential over the disjunction. To guarantee that the operations are closed, we needed to extend the embeddings to multicones. However, multicones do not satisfy a basic property. The full multicone $MC_\top$ can be understood as two possible multicones, namely $MC(\{C_\top\})$ and $MC(\{C(0, \pi), C(\pi, 2pi)\})$. However, the application of the rotation $R^\mathcal{E} = R\langle\theta, \frac{1}{2}, 0\rangle$ returns different multicones for both representations of the multicone. Since a multicone $MC(\{C_1, C_2\})$ is equal to $C_1 \cup C_2$, the tautology $\exists R.(C \sqcup D) \equiv \exists R.C \sqcup \exists R.D$ does not hold in the embedding space.

Proposition 3. *The axiom $\exists R.(C \sqcup D) \equiv \exists R.C \sqcup \exists R.D$ holds in multicone embeddings for rotations $R\langle\theta, \gamma, \delta\rangle$ if $\gamma \geq 1$ and $\delta \geq 0$.*

5 Expressing Logical Patterns

In this section, we present how logical patterns are expressed in the multicone embedding space. Given the relation names $r_1, r_2, r_3 \in \mathbf{R}$, correspond to the following $\mathcal{SROI}^-$ axioms:

$r_1 \sqsubseteq r_2$	(role containment),	$r_1 \circ r_2 \sqsubseteq r_3$	(composition),
$\mathsf{Trans}(r_1)$	(transitivity),	$r_1 \equiv r_2^-$	(inverse),
$\mathsf{Sym}(r_1)$	(symmetry),	$\mathsf{Asym}(r_1)$	(asymmetry).

Role containment pattern. Given two rotations $R^\mathcal{E} = R\langle\theta_1, \gamma_1, 0\rangle$ and $S^\mathcal{E} = R\langle\theta_2, \gamma_2, 0\rangle$ with $\theta_1 \neq \theta_2$ and $\gamma_1 \neq \gamma_2$, it never hold that $R^\mathcal{E} \sqsubseteq S^\mathcal{E}$. Indeed, we can always consider a cone with a sufficiently small aperture angle (e.g., a singleton cone) to show that this inclusion does not hold. On the other hand, for rotations $R^\mathcal{E} = R\langle\theta_1, 1, \delta_1\rangle$ and $S^\mathcal{E} = R\langle\theta_2, 1, \delta_2\rangle$ the inclusion can hold, even with distinct values for θ_1 and θ_2.

Proposition 4. *Given two rotations $R^\mathcal{E} = R\langle\theta_1, 1, \delta_1\rangle$ and $S^\mathcal{E} = R\langle\theta_2, 1, \delta_2\rangle$, the axiom $R \sqsubseteq S$ holds if $\theta_2 + \frac{\delta_2}{2} \geq \theta_1 + \frac{\delta_1}{2}$ and $\theta_2 - \frac{\delta_2}{2} \leq \theta_1 - \frac{\delta_1}{2}$*

Proposition 5. *Given two rotations $R^\mathcal{E} = R\langle\theta_1, \gamma_1, \delta_1\rangle$ and $S^\mathcal{E} = R\langle\theta_2, \gamma_2, \delta_2\rangle$, the axiom $R \sqsubseteq S$ holds if $\gamma_1 \leq \gamma_2$ and $\delta_1 \leq \delta_2$.*

Composition and transitivity patterns. The rules for composition and transitivity are a corollary of Proposition 4 and Proposition 5. For space limitations we omit the conditions for the Transitivity pattern. The conditions can be obtained from equivalence of axioms $\mathsf{Trans}(R)$ and $R \circ R \sqsubseteq R$.

Corollary 6. *Given three rotations* $R_1^{\mathcal{E}} = \mathsf{R}\langle\theta_1, 1, \delta_1\rangle$, $R_2^{\mathcal{E}} = \mathsf{R}\langle\theta_2, 1, \delta_2\rangle$, *and* $R_3^{\mathcal{E}} = \mathsf{R}\langle\theta_3, 1, \delta_3\rangle$, *the axiom* $R_1 \circ R_2 \sqsubseteq R_3$ *holds if* $|\theta_2 - (\theta_1 + \theta_2)| \leq \delta_3 - (\delta_1 + \delta_2)$, $\delta_1 \geq 0$, $\delta_2 \geq 0$, *and* $\delta_3 \geq 0$.

Corollary 7. *Given* $R_1^{\mathcal{E}} = \mathsf{R}\langle\theta_1, \gamma_1, \delta_1\rangle$, $R_2^{\mathcal{E}} = \mathsf{R}\langle\theta_2, \gamma_2, \delta_2\rangle$, *and* $R_3^{\mathcal{E}} = \mathsf{R}\langle\theta_3, \gamma_3, \delta_3\rangle$, *the axiom* $R_1 \circ R_2 \sqsubseteq R_3$ *holds if* $\gamma_1\gamma_2 \leq \gamma_3$, $\delta_1 + \delta_2 \leq \delta_3 0$, $\delta_1 \geq 0$, *and* $\delta_2 \geq 0$.

Inverse pattern. Because inverses cannot generally found, we cannot define inverse patterns for relations. However, inverse patterns still can be inferred under the circumstances where the commutativity is hold, which is discussed in Section 4.

Symmetry and asymmetry patterns. The symmetry pattern holds on rotations that applied twice over a singleton cone produce the same singleton cone. This happens when two rotations complete the circle. Similarly, the asymmetry pattern holds when two rotations do not coincide with a circle.

Proposition 8. *Given a rotation* $R^{\mathcal{E}} = \mathsf{R}\langle\theta, \gamma, \delta\rangle$ *with* $\gamma \geq 1$ *and* $\rho \geq 0$, *if there is a natural number* k *such that* $2\theta + \frac{\gamma\rho+\rho}{2} \geq 2k\pi$ *and* $2\theta - \frac{\gamma\rho+\rho}{2} \leq 2k\pi$, *then the axiom* $\mathsf{Sym}(r)$ *holds. Otherwise, the axiom* $\mathsf{Asym}(r)$ *holds.*

6 Tree-form Query Answering with AConE

To accommodate the learning of logical patterns in query answering over KGs, we propose a new model, AConE, which embeds KGs according to the multicone embedding (Definition 7). Our approach distinguishes itself from ConE [29], which represents cone embeddings using two-dimensional vectors in real space. We reframe these embeddings in the complex plane. By doing so, we introduce an inductive bias that facilitates the learning of logical patterns. This bias is achieved by combining the embeddings with relational rotations on cones, enhancing the model's ability to capture complex relationships and patterns. In AConE, the embedding of a tree-form query $q = C(x)$ is parameterized by a pair $\boldsymbol{q} = (\boldsymbol{h_U}, \boldsymbol{h_L}) \in \mathbb{C}^d$, where d is the embedding dimension, $|\boldsymbol{h_U}|_2 = 1$, $|\boldsymbol{h_L}|_2 = 1$ with $|\cdot|_2$ being the L2 norm. The vectors $\boldsymbol{h_U}$ and $\boldsymbol{h_L}$ represent the counterclockwise upper and lower boundaries of the cone, such that

$$\boldsymbol{h_U} = e^{i\boldsymbol{\theta_U}} = e^{i\left(\boldsymbol{\theta_{ax}} + \frac{\boldsymbol{\theta_{ap}}}{2}\right)}, \quad \boldsymbol{h_L} = e^{i\boldsymbol{\theta_U}} = e^{i\left(\boldsymbol{\theta_{ax}} - \frac{\boldsymbol{\theta_{ap}}}{2}\right)}, \quad (1)$$

where $\boldsymbol{\theta_{ax}} \in [-\pi, \pi)^d$ represents the angle of the symmetry axis of the cone and $\boldsymbol{\theta_{ap}} \in [0, 2\pi]^d$ represents the cone aperture.

For each index j with $1 \leq j \leq d$, we write $[\boldsymbol{v}]_j$ for the value of the j-th component of a vector $\boldsymbol{v}$. The i-th component of $\boldsymbol{q}$ represents the proper cone $\mathsf{C}([\boldsymbol{\theta_L}]_j, [\boldsymbol{\theta_U}]_j)$ if $[\boldsymbol{\theta_{ap}}]_j > 0$ and $[\boldsymbol{\theta_{ap}}]_j < 2\pi$, the full cone $\mathsf{C_\top}$ if $[\boldsymbol{\theta_{ap}}]_j = 2\pi$, and the singleton cone $\mathsf{C}_{[\boldsymbol{\theta_L}]_j}$ if $\boldsymbol{\theta_{ap}} = 0$. Notice that we do not provide an encoding for the empty cone because interesting queries are satisfiable. The queries are thus modeled as multidimensional cones according to Definition 7, and each entity answering the query is modeled as a vector $\boldsymbol{h}^* = e^{i\boldsymbol{\theta}^*}$ in the cone parametrized with the query embedding.
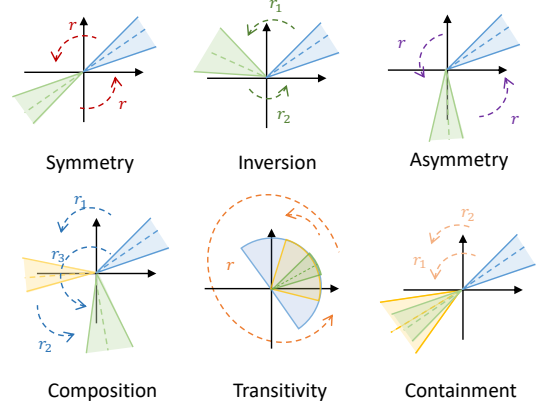
Figure 3: Logical patterns captured by AConE in a single dimension

6.1 Geometric Operators

To answer tree-form queries, AConE translates the concept constructors $\exists$, $\sqcap$, $\sqcup$, and $\neg$ into corresponding geometric operators, i.e. *relational rotation* $\mathcal{P}_\exists$, *intersection* $\mathcal{P}_\sqcap$, *union* $\mathcal{P}_\sqcup$, and *negation* $\mathcal{P}_\neg$, in the complex plane. AConE derives the final query embedding by executing these geometric operators along the computation graph of the query. We next describe these geometric operators.

Relational Rotation. Given a set of entities $\mathcal{S} \subset \mathbf{I}$ and a relation $r \in \mathbf{R}$, the transformation operator for r selects the entities $\mathcal{S}' = \{e' \in \mathbf{I} : G \models r(e, e'), e \in \mathcal{S}\}$. To this end, we model r with a vector $\boldsymbol{r} = (\boldsymbol{r_U}, \boldsymbol{r_L}) \in \mathbb{C}^{2 \times d}$ encoding a counterclockwise rotation on query embeddings about the complex plane origin such that $|\boldsymbol{r_U}| = 1$ and $|\boldsymbol{r_L}| = 1$. The rotation $\boldsymbol{r}$ transforms a query embedding $\boldsymbol{q} = (\boldsymbol{h_U}, \boldsymbol{h_L})$ into a query embedding $\mathcal{P}_\exists(\boldsymbol{q}, \boldsymbol{r}) = \boldsymbol{q}' = (\boldsymbol{h_U}', \boldsymbol{h_L}')$ where

$$\boldsymbol{h_U}' = \boldsymbol{h_U} \circ \boldsymbol{r_U}, \qquad \boldsymbol{h_L}' = \boldsymbol{h_L} \circ \boldsymbol{r_L}, \qquad (2)$$

where $\circ$ is the Hadamard (element-wise) product (i.e., for each component j, $[\boldsymbol{h_U}]'_j = [\boldsymbol{h_U}]_j \cdot [\boldsymbol{r_U}]_i$ and $[\boldsymbol{h_L}]'_j = [\boldsymbol{h_L}]_j \cdot [\boldsymbol{r_L}]_i$). If a pair $([\boldsymbol{r_U}]_j, [\boldsymbol{r_L}]_j)$ is seen as a cone, with axis angle $[\boldsymbol{\theta_{ax,r}}]_j$ and aperture angle $[\boldsymbol{\theta_{ap,r}}]_j$, then the rotation corresponds to the aperture-additive rotation $\mathsf{R}\langle[\boldsymbol{\theta_{ax,r}}]_j, 1, [\boldsymbol{\theta_{ap,r}}]_j\rangle$ (see Definition 6). That is, the axis and aperture angles of $\boldsymbol{q}'$ are $\boldsymbol{\theta_{ax}}' = \boldsymbol{\theta_{ax}} + \boldsymbol{\theta_{ax,r}}$, and $\boldsymbol{\theta_{ap}}' = \boldsymbol{\theta_{ap}} + \boldsymbol{\theta_{ap,r}}$. Figure 3 illustrates how does AConE model the logical patterns described in Section 5 through the example on a single dimension.

Intersection. The intersection of a set of query embeddings $Q = \{\boldsymbol{q_1}, ..., \boldsymbol{q_n}\}$, denoted $\mathcal{P}_\sqcap(Q) = \boldsymbol{q}'$ is defined with the permutation-invariant functions SemanticAverage and CardMin [29], which calculate the center $\boldsymbol{\theta_{ax}}'$ and the aperture $\boldsymbol{\theta_{ap}}'$ of the resulting cone encoded by $\boldsymbol{q}'$ as follows:

$$\begin{aligned} \boldsymbol{\theta_{ax}}' &= \text{SemanticAverage}(\{\boldsymbol{q_k}\}_{k=1}^n), \\ \boldsymbol{\theta_{ap}}' &= \text{CardMin}(\{\boldsymbol{q_k}\}_{j=1}^n). \end{aligned} \qquad (3)$$

SemanticAverage is expected to approximate the axis of the cone resulting from the intersection of the input cones. CardMin predicts the aperture $\boldsymbol{\theta_{ap}}'$ of the intersection set such that $[\boldsymbol{\theta_{ap}}']_j$ should be no larger than the aperture of any cone $\boldsymbol{q_i} \in Q$, since the intersection set is the subset of all input entity sets. Both functions, SemanticAverage and CardMin, use a neural network to improve the results. We extend the details of them in Appendix D.

Union. Given a set of query embeddings $Q = \{\boldsymbol{q_1}, \ldots, \boldsymbol{q_n}\}$, the union operator $\mathcal{P}_{\sqcup}(Q)$ returns the set Q. Intuitively, for each k where $1 \le k \le d$, the set Q encodes a multicone embedding for the query $(q_1 \sqcup \cdots \sqcup q_n)(x)$, which represents the multicone $\mathsf{MC}(\mathsf{C}_1, \ldots, \mathsf{C}_n)$ where C_i is the cone encoded in $[\boldsymbol{q_i}]_k$.

Notice that multicones cannot be further used as input of other operations because all AConE operations are applied on cones. To compute queries, we thus follow [16], translating queries into disjunctive normal form, so we only perform the disjunction operator in the last step in the computation graph.

Negation. The negation of a cone $\boldsymbol{q} = (\boldsymbol{h_U}, \boldsymbol{h_L})$, denoted $\mathcal{P}_{\neg}(\boldsymbol{q})$, is the cone $\boldsymbol{q'}$ that contains the entities in the complement of $\boldsymbol{q}$, that is, $\boldsymbol{q'} = (\boldsymbol{h_L}, \boldsymbol{h_U})$.

6.2 Optimization

Learning Objective. Given a set of training samples, our goal is to minimize the distance between the query embedding $\boldsymbol{q}$ and the answer entity vector $\boldsymbol{h^*}$, while maximizing the distance between $\boldsymbol{q}$ and its negative samples. Thus, we define our training objective as

$$L = -\log \sigma(\gamma - \mathrm{d}_{\mathrm{comb}}(\boldsymbol{h^*}, \boldsymbol{q})) - \frac{1}{k} \sum_{i=1}^{k} \log \sigma(d(\boldsymbol{h_i'}, \boldsymbol{q}) - \gamma), \quad (4)$$

where $\mathrm{d}_{\mathrm{comb}}$ is the *combined distance* defined below, γ is a margin, $\boldsymbol{h^*}$ is a positive entity, and $\boldsymbol{h_i'}$ is the i-th negative entity, k is the number of negative samples, and σ represents the sigmoid function.

Combined Distance. Inspired by [29, 16], the distance between $\boldsymbol{q}$ and $\boldsymbol{h^*}$ is defined as a combination of inside and outside distances, $d_{com}(\boldsymbol{q}, \boldsymbol{h^*}) = \mathrm{d}_{\mathrm{o}}(\boldsymbol{q}, \boldsymbol{h^*}) + \lambda \, \mathrm{d}_{\mathrm{i}}(\boldsymbol{q}, \boldsymbol{h^*})$:

$$\mathrm{d}_{\mathrm{o}}(\boldsymbol{q}, \boldsymbol{h^*}) = \min\{\|\boldsymbol{h_U} - \boldsymbol{h^*}\|_1, \|\boldsymbol{h_L} - \boldsymbol{h^*}\|_1\}, \quad (5)$$

$$\mathrm{d}_{\mathrm{i}}(\boldsymbol{q}, \boldsymbol{h^*}) = \min\{\|\boldsymbol{h_{ax}} - \boldsymbol{h^*}\|_1, \|\boldsymbol{h_U} - \boldsymbol{h_{ax}}\|_1\}, \quad (6)$$

where $\| \cdot \|_1$ is the L_1 norm, $\boldsymbol{h_{ax}}$ represents the cone center, and $\lambda \in (0, 1)$. Note that $\mathrm{d}_{\mathrm{comb}}$ can only be used for measuring the distance between a single query embedding and an answer entity vector. Since we represent the disjunctive queries in Disjunctive Normal Form as a set of query embeddings Q, the distance between the answer vector and such set of embeddings is the minimum distance:

$$\mathrm{d}_{\mathrm{comb}}(Q, \boldsymbol{h^*}) = \min\{\mathrm{d}_{\mathrm{comb}}(\boldsymbol{q}, \boldsymbol{h^*}) : \boldsymbol{q} \in Q\}. \quad (7)$$

7 Experiments and Analysis

In this section, we answer the following research questions with experimental statistics and corresponding case analyses. **RQ1**: how well does AConE improve query answering on incomplete KGs? **RQ2**: How is the improvement of results related to the capturing of logical patterns?

7.1 Experimental Setup

Dataset and Query Structure. For a fair comparison with the baseline models, we use the same query structures and logical query datasets from NELL-QA [26] and WN18RR-QA [8], and the open-sourced framework created by [15] for logical query answering tasks. Figure 4 illustrates all query structures in experiments. We trained our model using 10 specific query structures and tested it on all 14 query structures to evaluate the generalization ability of our model regarding unseen query structures. More experimental details can be found in the supplementary file.
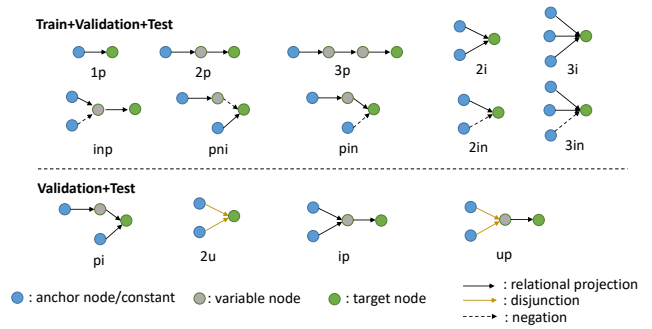
Figure 4: Fourteen types of queries used in the experiments. p represents an edge to another entity labelled with a relation name r (i.e., an operation $\exists r$), i represents the intersection ($\sqcap$), u represents the disjunction ($\sqcup$), and n represents the negation ($\neg$).

Evaluation Metrics. We use Mean Reciprocal Rank (MRR) as the evaluation metric. Given a sample of queries Q, MRR represents the average of the reciprocal ranks of results, $\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}$.

7.2 RQ1: how well does AConE improve query answering on incomplete KGs?

We evaluate AConE on two logical query-answering benchmark datasets that include a variety of complex logical patterns: NELL-QA created by [15] and WN18RR-QA by [8]. AConE is compared with various query embedding models that also define query regions as vectors, geometries, or distributions, including GQE [7], Query2Box (Q2B) [16], BetaE [15], ConE [29], and LinE [8]. To ensure a fair comparison, we indeed referenced the results of all baseline models from corresponding papers. In the case of ConE's performance on the WN18RR dataset, we conducted our own experiments due to the absence of previously reported results. This involved rerunning ConE with an optimized search for hyperparameters to generate the relevant data.

Main Results. Table 2 summarizes the performance of all methods on answering various query types. Compared with baselines that can only model queries without negation. Overall, AConE (aperture-additive) outperforms baseline methods on the majority of query types in Figure 4 while achieving competitive results on the others. Specifically, AConE consistently achieves improvements on non-negation queries on all datasets. Furthermore, our model brings notable improvement over baseline models on the WN18RR dataset, where the average accuracy of AConE is 18.35% higher than that of the baseline models.

On the other hand, we observe that AConE performs closely to the baseline models on answering query types involving negation on dataset NELL, in spite of its consistently good performance on WN18RR. This may be due to two factors: firstly, handling negation is a challenging research question for complex logical query-answering tasks. When it comes to negation queries, all the current models show inferior performance compared to their performance on non-neg queries. Secondly, modeling negation as the complement leads to bias in prediction and high uncertainty in a large number of answers for negation queries. Thus, we leave the task of further improving our model on negation queries for future research.

Table 2: MRR results (%) of AConE, LinE (the results of LinE are only available on WN18RR-QA), ConE, BETAE, Q2B, and GQE on answering tree-form (∃, ∧, ∨, ¬) queries on datasets NELL-QA, and WN18RR. The best statistic is highlighted in bold, while the second best is highlighted in underline.

Dataset	Model	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
WN18RR-QA	GQE	18.0	4.5	2.7	19.3	23.9	9.9	10.6	2.3	3.5	-	-	-	-	-
	Q2B	22.4	4.6	2.3	25.6	41.2	13.2	11.0	2.9	3.4	-	-	-	-	-
	BetaE	44.1	9.8	3.8	57.2	76.2	32.6	17.9	7.5	5.3	12.7	59.9	5.1	4.0	7.4
	LinE	45.1	12.3	6.7	47.1	67.1	24.8	14.7	8.4	6.9	12.5	60.8	7.3	5.2	_7.7_
	ConE	_46.8_	_14.5_	_9.3_	_59.0_	_83.9_	_33.6_	_18.7_	_10.0_	_9.8_	_13.9_	_61.8_	_10.6_	_7.3_	7.6
	AConE	**50.9**	**17.6**	**9.9**	**70.5**	**89.0**	**38.9**	**29.6**	**18.4**	**14.0**	**18.3**	**70.1**	**13.4**	**7.6**	**10.4**
NELL-QA	GQE	33.1	12.1	9.9	27.3	35.1	18.5	14.5	8.5	9.0	-	-	-	-	-
	Q2B	42.7	14.5	11.7	34.7	45.8	23.2	17.4	12.0	10.7	-	-	-	-	-
	BetaE	53.0	13.0	11.4	37.6	47.5	24.1	14.3	12.2	8.5	5.1	_7.8_	_10.0_	3.1	3.5
	ConE	_53.1_	_16.1_	_13.9_	_40.0_	_50.8_	**26.3**	_17.5_	_15.3_	_11.3_	**5.7**	**8.1**	**10.8**	**3.5**	**3.9**
	AConE	**54.5**	**17.7**	**14.4**	**41.9**	**53.0**	_26.1_	**20.7**	**16.5**	**12.8**	_5.2_	7.7	9.4	_3.2_	_3.7_

Table 3: Statistics on the percentage of queries that involve various logical patterns in the NELL test dataset

Relation	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
Symmetry	20.3	29.7	38.6	25.3	29.9	34.4	31.9	31.2	34.6	27.6	26.2	38.6	37.1	36.7
Inversion	26.9	41.3	51.0	42.0	51.9	51.9	46.0	43.2	43.8	42.0	56.1	51.0	52.0	54.3
Composition	21.6	35.2	43.4	31.5	41.7	42.7	40.1	35.8	40.9	34.7	42.5	43.1	43.0	43.6
Containment	37.9	51.5	63.1	50.9	59.1	54.3	48.35	48.9	49.32	48.7	50.3	60.7	60.6	58.1
Transitivity	3.1	2.8	3.3	2.9	3.7	3.2	2.5	3.3	2.9	3.7	3.6	8.4	4.2	3.95

7.3 RQ2: How is the improvement of results related to the capturing of logical patterns?

To investigate the influence of logical patterns learning on the query answering model, we provide an ablation analysis considering both model and data:

Model Perspective. We compared the performance of three AConE's variants: AConE (Base), AConE (aperture-multiplicative) and AConE (aperture-additive). They utilize alternative relational rotating transformation strategies for capturing the logical patterns while keeping other modules unchanged. Note that AConE (Base) represents AConE model with any neural relational transformation module. This configuration operates similarly to ConE in terms of its foundational approach. Consequently, we used the results of ConE as a direct representation of "AConE (Base)". This decision was based on their operational similarity, ensuring a fair and coherent presentation of our findings.

**Table 4**: Ablation study of AConE on NELL dataset.

Model	1p	2p	3p	2i	3i	pi	ip	2u	up
AConE (Base)	53.1	16.1	13.9	40.0	50.8	26.3	17.5	15.3	11.3
AConE (Ap-Mul)	51.3	16.6	13.8	38.4	48.4	18.9	19.5	14.7	12.1
AConE (Ap-Add)	**54.5**	**17.7**	**14.4**	41.9	53.0	26.1	**20.7**	16.5	12.8

As Table 4 shows, the performance variation among AConE's variants highlights the impact of relation transformation on the model's ability to capture logical patterns. In addition, the outperformance of AConE confirms the efficiency of logical patterns in logical query reasoning tasks.

Data Perspective. To better study the specific impact of AConE on queries involving logical patterns, more in-depth analysis is made on the query answering dataset NELL. We categorize the test dataset into five categories based on the relations involved in the queries. For each query in subgroups [1] *Inverse*, *Symmetry*, *Composition*, *Containment* and *Transitivity*, there is at least one relation encompassing the corresponding logical pattern. Table 3 provides more details about

---

[1] Note that *Asymmetry* is not included here because such type of relations are missing from the selected rules mined from NELL.

these subgroups of queries and the classification process is elaborated in Appendix B. The category *Others* corresponds to queries that do not involve any of these logical patterns. Figure 5 shows the average performances of AConE and neural baseline model (ConE) on these subgroups. It is observed that AConE outperforms the neural baseline model on queries that had logical patterns, especially inverse relations. However, AConE does not generalize as well to queries that were not influenced by logical patterns compared to the baseline model. This analysis supports our hypothesis that AConE is better suited for capturing logical patterns in tree-form query answering tasks than purely neural models, with fewer model parameters and a simpler model structure, which is elaborated in section 8.
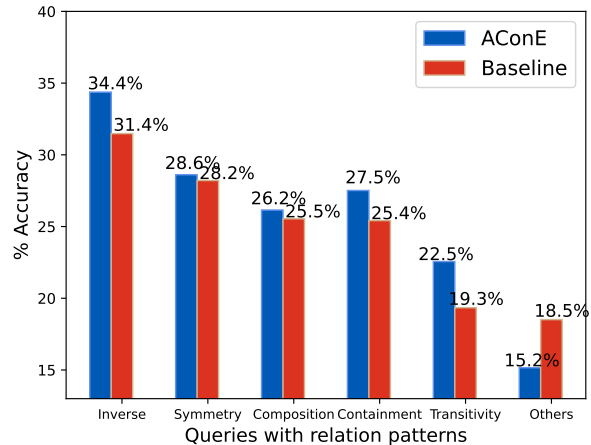
Figure 5: Average performances of AConE and Baseline model over query subgroups with different logical patterns.

8 Analysis on Model Parameters

The acquisition of logical patterns not only enhances the model performance but also alleviates the reliance on excessively-parameterized neural networks in existing methods. Table 5 summarizes the number of parameters and the average performance of our method AConE and other baseline models on non-negation queries which can be handled by all methods. It shows that AConE has the

second-fewest number of parameters among these models, though it achieves much better performances than the baseline models.

Table 5: Number of parameters in AConE and other baseline models.

Model	AVG MRR	Parameters
BetaE[15]	28.26	57,574,000
GQE[7]	10.52	52,290,400
ConE[29]	31.73	44,010,401
Q2B[16]	14.06	26,306,000
Ours	37.64	36,325,601

9 Conclusion and Future Work

In this work, we investigate the important yet understudied impact of logical pattern inference on the query answering task. We propose a new embedding method, the multicone embedding, and study its algebraic structure and capability to express $\mathcal{SROI}^-$ ontologies. We show some limitations of the method, like breaking the tautology $\exists R.(C \sqcap D) \equiv \exists R.C \sqcup \exists R.D$, the bias favoring the commutativity, and the (general) non-existence of inverse rotations.

Our practical method, AConE, is motivated by the multicone algebra, but it has differences. First, multicones consisting of multiple cones are difficult to manage due to the unconstrained number of parameters they require. Hence, we limit the representation of queries to single multicones (i.e., cones). Since this simplification makes the cone intersection no longer closed, we define it as a neural operation returning a single cone. We limit rotations to be aperture-additive because they outperform aperture-multiplicative rotations (see Table 4).

Although the experiment datasets had no concept names, multi-cone embeddings support concept names. We use these datasets to make a fair comparison with the existing methods, which do not support concept names. Since most real datasets include concept names, and concept names are a key component of knowledge representation, we plan to extend our research to datasets with concept names. Furthermore, we think that AConE can be extended for knowledge graphs including also non-empty RBoxes and TBoxes.

Ethics Statement

The authors declare that we have no conflicts of interest. This article does not contain any studies involving business and personal data.

Acknowledgements

References

[1] SPARQL 1.1 Query Language. Technical report, W3C, 2013. URL http://www.w3.org/TR/sparql11-query.

[2] E. Arakelyan, D. Daza, P. Minervini, and M. Cochez. Complex query answering with neural link predictors. In *ICLR*. OpenReview.net, 2021.

[3] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003. ISBN 0-521-78176-0.

[4] K. D. Bollacker, C. Evans, P. K. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In J. T. Wang, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM, 2008. doi: 10.1145/1376616.1376746. URL https://doi.org/10.1145/1376616.1376746.

[5] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

[6] L. Galárraga, C. Teflioudi, K. Hose, and F. M. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6): 707–730, 2015.

[7] W. L. Hamilton, P. Bajaj, M. Zitnik, D. Jurafsky, and J. Leskovec. Embedding logical queries on knowledge graphs. In *NeurIPS*, pages 2030–2041, 2018.

[8] Z. Huang, M. Chiang, and W. Lee. Line: Logical query reasoning over hierarchical knowledge graphs. In *KDD*, pages 615–625. ACM, 2022.

[9] M. Jackermeier, J. Chen, and I. Horrocks. Box^{2}el: Concept and role box embeddings for the description logic EL++. *CoRR*, abs/2301.11118, 2023. doi: 10.48550/ARXIV.2301.11118. URL https://doi.org/10.48550/arXiv.2301.11118.

[10] B. Kotnis, C. Lawrence, and M. Niepert. Answering complex queries in knowledge graphs with bidirectional sequence encoders. In *AAAI*, pages 4968–4977. AAAI Press, 2021.

[11] X. V. Lin, R. Socher, and C. Xiong. Multi-hop knowledge graph reasoning with reward shaping. In *EMNLP*, pages 3243–3253. Association for Computational Linguistics, 2018.

[12] M. Nayyeri, C. Xu, Y. Yaghoobzadeh, S. Vahdati, M. M. Alam, H. S. Yazdi, and J. Lehmann. Loss-aware pattern inference: A correction on the wrongly claimed limitations of embedding models. In *PAKDD (3)*, volume 12714 of *Lecture Notes in Computer Science*, pages 77–89. Springer, 2021.

[13] M. Nickel, V. Tresp, and H. Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, pages 809–816. Omnipress, 2011.

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[15] H. Ren and J. Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *NeurIPS*, 2020.

[16] H. Ren, W. Hu, and J. Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *ICLR*. OpenReview.net, 2020.

[17] H. Ren, M. Galkin, M. Cochez, Z. Zhu, and J. Leskovec. Neural graph reasoning: Complex logical query answering meets graph databases, 2023.

[18] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy, editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 697–706. ACM, 2007. doi: 10.1145/1242572.1242667. URL https://doi.org/10.1145/1242572.1242667.

[19] Z. Sun, Z. Deng, J. Nie, and J. Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR (Poster)*. OpenReview.net, 2019.

[20] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016.

[21] D. Vrandecic and M. Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, 2014.

[22] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119. AAAI Press, 2014.

[23] B. Xiong, N. Potyka, T. Tran, M. Nayyeri, and S. Staab. Faithful embeddings for $\mathcal{EL}^{++}$ knowledge bases. In *ISWC*, volume 13489 of *Lecture Notes in Computer Science*, pages 22–38. Springer, 2022.

[24] B. Xiong, S. Zhu, M. Nayyeri, C. Xu, S. Pan, C. Zhou, and S. Staab. Ultrahyperbolic knowledge graph embeddings. In *KDD*, pages 2130–2139. ACM, 2022.

[25] W. Xiong, T. Hoang, and W. Y. Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pages 564–573. Association for Computational Linguistics, 2017.

[26] W. Xiong, T. Hoang, and W. Y. Wang. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1060. URL https://aclanthology.org/D17-1060.

[27] B. Yang, W. Yih, X. He, J. Gao, and L. Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR (Poster)*, 2015.

[28] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov, and A. Smola. Deep sets, 2017. URL https://arxiv.org/abs/1703.06114.

[29] Z. Zhang, J. Wang, J. Chen, S. Ji, and F. Wu. Cone: Cone embeddings for multi-hop reasoning over knowledge graphs. In *NeurIPS*, pages 19172–19183, 2021.

[30] Z. Zhu, M. Galkin, Z. Zhang, and J. Tang. Neural-symbolic models for logical queries on knowledge graphs. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 27454–27478. PMLR, 2022.

[31] Z. Zhu, M. Galkin, Z. Zhang, and J. Tang. Neural-symbolic models for logical queries on knowledge graphs. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 27454–27478. PMLR, 2022.

A   Experimental Details

The source code of AConE, the benchmark datasets, and the instructions for the reproduction of our experimental results are attached as zip files.

Description on Benchmark Datasets   For a fair comparison with the baseline models, we use the same query structures and logical query datasets from NELL-QA [26] and WN18RR-QA [8], and the open-sourced framework created by [15] for logical query answering tasks. Table 6 summarizes the statistics of benchmark datasets.

Table 6: Statistics of benchmark datasets.

Dataset	Training		Validation		Testing	
	1p/2p/3p/2i/3i	Neg	1p	others	1p	others
NELL	107,980	10,798	16,927	4,000	17,034	4,000
WN18RR	517,545	51,750	5,202	13,000	5,356	13,000

Hyperparameters and Computational Resources.   All of our experiments are implemented in Pytorch [14] framework and run on four Nvidia A100 GPU cards. For hyperparameters search, we performed a grid search of learning rates in $\{5 \times 10^{-5}, 10^{-4}, 5 \times 10^{-4}\}$, the batch size in $\{256, 512, 1024\}$, the negative sample sizes in $\{128, 64\}$, the regularization coefficient λ in $\{0.02, 0.05, 0.08, 0.1\}$ and the margin γ in $\{20, 30, 40, 50\}$. The best hyperparameters are shown in Table 7.

Table 7: Hyperparameters found by grid search. d is the embedding dimension, b is the batch size, n is the negative sampling size, γ is the margin in loss, l is the learning rate, λ is the regularization parameter in the distance function.

Dataset	d	b	n	γ	l	λ
NELL	800	512	128	20	1×10^{-4}	0.02
WN18RR	800	512	128	20	1×10^{-4}	0.02

B   Classification Process of Test Queries

This paragraph describes the process for identifying subgroups of test queries with different relation patterns. We first mined rules from the triples-only dataset NELL995 [25] using the rule mining tool AMIE [6]. The rules that are significantly supported by a large amount of data and have body coverages greater than 0.2 are selected. These rules are then categorized into six types: symmetry, inversion, composition, containment, transitivity and other. Each category includes a list of relations that encompass relation patterns. Using the selected relations, the test dataset of the logical query dataset NELL [15] is traversed, and queries are classified into six subgroups: symmetry, inversion, composition, containment, transitivity and other. A query is classified into a subgroup based on the presence of at least one relation from the corresponding relation subgroup. The other subgroup does not include any of these patterns.

C   Model Robustness Check

In this section, we check the model robustness of AConE by running AConE 10 times with different random seeds. The mean performance and standard deviations are reported in Table 8. The slight variances in the results indicate that AConE is robust to the different initializations of model parameters and our reported improvements are not due to randomness.

D Intersection Operator

In this section, we explain the details of two important components of the intersection operator, **SemanticAverage**$(\cdot)$ and **CardMin**$(\cdot)$

SemanticAverage$(\cdot)$ is expected to compute the semantic center $\boldsymbol{\theta}'_{ax}$ of the input $\{(\boldsymbol{\theta}_{j,ax}, \boldsymbol{\theta}_{j,ap})\}_{j=1}^{n}$. Specifically, the computation process is provided as:

$$[\mathbf{x}; \mathbf{y}] = \sum_{i=1}^{n} [\mathbf{a}_j \circ \cos(\boldsymbol{\theta}_{j,ax}); \mathbf{a}_j \circ \sin(\boldsymbol{\theta}_{j,ax})],$$
$$\boldsymbol{\theta}'_{ax} = \mathbf{Arg}(\mathbf{x}, \mathbf{y}), \tag{8}$$

where cos and sin represent element-wise cosine and sine functions. $\mathbf{Arg}(\cdot)$ computes the argument given $\mathbf{x}$ and $\mathbf{y}$. $\mathbf{a}_j \in \mathbb{R}^d$ are attention weights such that

$$[\mathbf{a}_j]_k = \frac{\exp([\mathbf{MLP}([\boldsymbol{\theta}_{j,ax} - \boldsymbol{\theta}_{j,ap/2}; \boldsymbol{\theta}_{j,ax} + \boldsymbol{\theta}_{j,ap/2}])]_k)}{\sum_{m=1}^{n} \exp([\mathbf{MLP}([\boldsymbol{\theta}_{m,ax} - \boldsymbol{\theta}_{m,ap/2}; \boldsymbol{\theta}_{m,ax} + \boldsymbol{\theta}_{m,ap/2}])]_k)}, \tag{9}$$

where $\mathbf{MLP} : \mathbb{R}^{2d} \to \mathbb{R}^d$ is a multi-layer perceptron network, $[\cdot; \cdot]$ is the concatenation of two vectors.

CardMin$(\cdot)$ predicts the aperture $\boldsymbol{\theta}'_{ap}$ of the intersection set such that $[\boldsymbol{\theta}'_{ap}]_i$ should be no larger than any $\boldsymbol{\theta}^i_{j,ap}$, since the intersection set is the subset of all input entity sets.

$$[\theta'_{ap}]_i = \min\{\theta^i_{1,ap}, \ldots, \theta^i_{n,ap}\} \cdot \sigma([\mathbf{DeepSets}(\{(\boldsymbol{\theta}_{j,ax}, \boldsymbol{\theta}_{j,ap})\}_{j=1}^{n})]_i) \tag{10}$$

where $\mathbf{DeepSets}(\{(\boldsymbol{\theta}_{j,ax}, \boldsymbol{\theta}_{j,ap})\}_{j=1}^{n})$ [28] is given by

$$\mathbf{MLP}(\tfrac{1}{n} \sum_{j=1}^{n} \mathbf{MLP}([\boldsymbol{\theta}_{j,ax} - \boldsymbol{\theta}_{j,ap/2}; \boldsymbol{\theta}_{j,ax} + \boldsymbol{\theta}_{j,ap/2}])).$$

Table 8: AConE's MRR mean values and standard variances (%) on answering tree-form ($\exists$, $\wedge$, $\vee$, $\neg$) queries

Dataset	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
NELL	54.5±	17.7±	14.4±	41.9±	53.0±	26.1±	20.7±	16.5±	12.8±	5.2±	7.7±	9.4±	3.2±	3.7±
	0.097	0.152	0.192	0.130	0.038	0.084	0.146	0.112	0.137	0.012	0.079	0.154	0.012	0.097