Extending Complex Logical Queries on Uncertain Knowledge Graphs

Anonymous ACL submission

Abstract

The study of machine learning-based logical query answering enables reasoning with large-scale and incomplete knowledge graphs. This paper advances this area of research by addressing the uncertainty inherent in knowledge. While the uncertain nature of knowledge is widely recognized in the real world, it does not align seamlessly with the first-order logic that underpins existing studies. To bridge this gap, we explore the soft queries on uncertain knowledge, inspired by the framework of soft constraint programming. We propose a neural symbolic approach that incorporates both forward inference and backward calibration to answer soft queries on large-scale, incomplete, and uncertain knowledge graphs. Theoretical discussions demonstrate that our method avoids catastrophic cascading errors in the forward inference while maintaining the same complexity as state-of-the-art symbolic methods for complex logical queries. Empirical results validate the superior performance of our backward calibration compared to extended query embedding methods and neural symbolic approaches.

1 Introduction

Representing and reasoning with factual knowledge are essential functionalities of artificial intelligence systems. As a powerful way of knowledge representation, Knowledge Graphs (KGs) (Miller, 1995; Suchanek et al., 2007; Vrandečić and Krötzsch, 2014) use nodes to represent entities and edges to encode the relations between entities. Recently, Complex Query Answering (CQA) over KGs has attracted considerable attention because this task requires multi-hop logical reasoning over KGs and supports many applications (Ren et al., 2023). This task requires answering the existential First Order Logic (FOL) query, involving existential quantification ($\exists$), conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$). While answering FOL queries has been extensively researched by database community (Riesen et al., 2010; Hartig and Heese, 2007), such studies overlook the **incompleteness** of most KGs. Consequently, conventional graph traversal methods for *relational database queries* may neglect certain answers due to the missing links of KGs. In recent studies on *complex logical queries on knowledge graphs*, the generalizability of machine learning models is leveraged to predict the missing links of observed KGs and conduct first-order logic reasoning (Ren and Leskovec, 2020; Arakelyan et al., 2021; Liu et al., 2021; Wang et al., 2023b). This combination of machine learning and logic enables further possibilities in data management (Ren et al., 2023).

The uncertainty of knowledge is widely observed ranging from daily events (Zhang et al., 2020) to complex biological systems (Szklarczyk et al., 2023). *To represent the uncertain knowledge*, confidence values p are associated with facts to augment the KG (Carlson et al., 2010; Speer et al., 2017; Szklarczyk et al., 2023), known as the uncertain KG. As exemplified in the right of Figure 1, which illustrates the Job Candidates uncertain KG, confidence values are used to quantify the degree of practice level associated with specific skills. Uncertainty is prevalent in existing KGs primarily because most are constructed using machine learning models, where the predicted likelihood of relational facts inherently introduces uncertainty. Notable examples include O*NET (Pai and Costabello, 2021), STRING (Szklarczyk et al., 2023), ConceptNet (Speer et al., 2017), and AESR (Zhang et al., 2020). To address the incompleteness of uncertain KGs, recent studies estimate the confidence values of missing facts with the generalization power of ML models (Chen et al., 2019; Pai and Costabello, 2021).

To reason with uncertainty, many extensions of first-order logic have been made to cope with the uncertainty in knowledge representation systems formally (Adams, 1996). It is noteworthy to men-

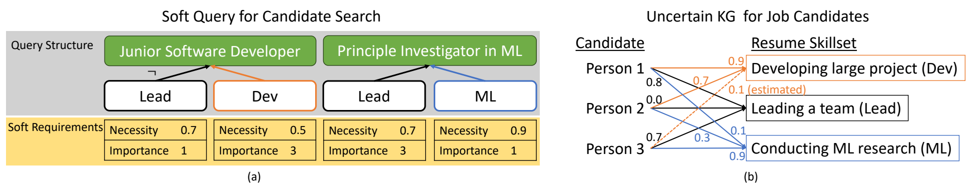

Figure 1: **(a)** Examples of two soft queries in the candidate search procedure. The soft queries introduced in this paper are jointly defined by first-order logic and soft requirements. In particular, soft requirements (necessity and importance) are introduced to characterize fine-grained decision-making preferences, distinguishing them from first-order queries. **(b)** Incomplete uncertain KG for to what extent a candidate possesses a skill. Solid lines indicate the observed knowledge, while dashed lines indicate the unobserved data. Values indicate confidence level, where the higher value indicates the fact is more likely to be true.

Table 1: Comparison of different problem settings. FO: First Order, EFO: Existential First Order.

Problem Settings	Language of reasoning	Uncertainty of Knowledge	Unobserved Knowledge
Relational database	FO	-	-
Probablistic database	FO	Confidence value p	-
Open world probablistic database	FO	Confidence value p	Uniformed p_u for all unobserved facts
Complex logical queries on KG	EFO	-	ML generalization to unobserved facts
Soft queries on uncertain KG (Ours)	EFO + Soft requirements	Confidence value p	ML generalization to unobserved facts and confidence p

tion that the study of *probabilistic databases* also extends the relational databases with a confidence value $p \in [0, 1]$ (Cavallo and Pittarelli, 1987; Suciu et al., 2022). From a machine learning perspective, however, previous studies in *open world probabilistic databases* are limited from two aspects: (1) They assume uniform uncertainty for all unobserved knowledge (Ceylan et al., 2021), leading to weaker characterization of the incomplete knowledge without the generalization. (2) They focus on first-order logic, which might be insufficient to describe practical reasoning processes with uncertainty. To address the limitations, we extend the complex query answering to uncertain KG and propose soft queries combining query structure and soft requirements, as shown in the left of Figure 1.

This paper studies the machine learning method for reasoning with incomplete and uncertain knowledge, advancing previous studies in symbolic probabilistic databases. Its contribution is threefold.

Contribution 1: A novel and practical setting. We propose a novel setting of Soft Queries on Uncertain KG (SQUK). Our setting extends the previous setting of *complex logical queries on KGs* in two ways: (1) For the incomplete knowledge base, SQUK extends the incomplete KG to incomplete and *uncertain* KG. (2) For the language describing reasoning, SQUK extends first-order language to *uncertainty-aware* soft queries with soft requirements, which are motivated by real-world reasoning with uncertainty and the establishment in soft constraint programming (Schiex, 1992; Rossi et al., 2006). We also introduce the formal definition in Section 3. The comparison of SQUK against other settings is detailed in Table 1.

Contribution 2: ML method for soft queries. We bridge machine learning and SQUK by proposing Soft Reasoning with calibrated Confidence values (SRC), which uses Uncertain Knowledge Graph Embeddings (UKGEs) to tackle the unobserved information and achieves the same computational complexity as the state-of-the-art inference algorithms (Bai et al., 2023; Yin et al., 2024). The error analysis is also conducted for SRC given the error bound function, characterizing how the performance is affected by UKGEs and the query structures. Based on our analysis, we suggest calibrating the confidence by debiasing and learning, which further boosts the performance of SRC.

Contribution 3: Extensive empirical studies. We also conduct extensive empirical studies to benchmark the performance of a broad spectrum of methods under the UKGE (Chen et al., 2019) settings. The calibrated SRC is compared against baselines including Query Embedding (Ren and Leskovec, 2020) methods with Number Embeddings (QE+NE) (Vaswani et al., 2017) and symbolic search method (Yin et al., 2024). In particular, we compared the differences between QE+NE and SRC under various soft query settings, demonstrating the advantage of SRC. We also make a fair comparison with large language models on annotated soft queries in a natural language setting.

We highlight the uniqueness of the SQUK setting by examining the differences between uncertain KGs and KGs, comparing soft queries with logical queries, and addressing the challenges posed by two versions of incomplete knowledge. These differences are also summarized in Table 1. Additionally, we present the related work concerning complex logical query answering and uncertain knowledge graph embedding in Appendix A.

2 Background

Uncertain KGs enhance traditional KGs by augmenting each triple fact with a confidence value, thereby facilitating the modeling of uncertain knowledge, which is particularly useful in various domains. Figure 1 illustrates uncertainty in job backgrounds. We formally define an uncertain knowledge graph as a set of knowledge as follows:

Definition 1 (Uncertain knowledge graph)**.** *Let $\mathcal{E}$ be the set of entities and $\mathcal{R}$ be the set of relations, an uncertain knowledge graph $\mathcal{G}$ is a set of quadruple $\{(s_i, r_i, o_i, p_i)\}$, where $s_i, o_i \in \mathcal{E}$ are entities, $r_i \in \mathcal{E}$ is relation and $p_i \in [0,1]$*[1] *represents the confidence value for the relation fact (s_i, r_i, o_i). This confidence value p_i indicates the degree of certainty regarding the truth of the fact.*

Following the closed-world assumption (Reiter, 1981) and treating all unobserved facts as false, we can derive the weight graph form for uncertain KG and represent it with the confidence function $P : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \mapsto [0,1]$ as follows:

$$P(h_i, r_i, t_i) = \begin{cases} p_i & (h_i, r_i, t_i, p_i) \in \mathcal{G}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Uncertain KGs also suffer incomplete issues (Chen et al., 2019, 2021a), with observed knowledge representing only a small portion of the total facts. The assumption that all unseen relational facts are false is inappropriate in real-world scenarios. To address this challenge, previous research on uncertain KGs has proposed a machine learning task to predict the confidence scores of these unseen relational facts (Chen et al., 2019, 2021a). Typically, the observed knowledge in uncertain KGs is split into three nested sets of facts, where $\mathcal{G}_{\text{train}} \subsetneq \mathcal{G}_{\text{valid}} \subsetneq \mathcal{G}_{\text{test}}$. The training set $\mathcal{G}_{\text{train}}$ is used to train the model, while the validation and test sets are used to evaluate its performance in predicting the confidence scores of unseen facts.

Uncertain Knowledge Graph Embeddings (UKGEs) (Chen et al., 2019, 2021a) have been the mainstream methods for predicting unseen relational facts in uncertain KGs, as they learn low-dimensional representations that effectively capture the semantics between relations and facts, demonstrating strong generalizability. UKGEs are trained on partial facts $\mathcal{G}_{\text{train}}$ and approximate the confidence function $\mathcal{P}$ deriving from complete facts, defined as the following confidence function:

Definition 2. *An UKGE parameterizes a differentiable confidence function $\hat{P} : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \mapsto [0,1]$.*

In practice, obtaining complete facts is challenging, so P_{test} induced by $\mathcal{G}_{\text{test}}$ are usually substituted for $\mathcal{P}$, which adhere to previous approaches (Chen et al., 2019; Pai and Costabello, 2021). We present the connection between this setting and the open-world assumption in Appendix E.

3 Soft Queries

The uncertainty inherent in KGs can be modeled using confidence values for each knowledge. However, current complex logical queries are defined on a boolean basis [2], which is not compatible with uncertain KGs. This uncertainty necessitates new definitions for logical operations and answer sets. In this section, we introduce the definition of our extended soft queries.

3.1 Syntax and semantic

Definition 3 (Syntax of soft queries)**.** *Soft queries are the disjunction of soft conjunctive queries ϕ_i:*

$$\Phi(y) = \phi_1(y) \ovee \cdots \ovee \phi_q(y), \quad (2)$$

where y is the free variable. Each $\phi_i(y)$ is the conjunction of the soft atomic formula:

$$\phi_i(y) = \exists x_1, \ldots, x_n . a_{i1} \owedge \cdots \owedge a_{ij}, i = 1, ..., q, \quad (3)$$

where $x_1, \ldots, x_n$ represent existentially quantified variables. Each a_i is a soft atomic formula of the form (h, r, t, α, β) or its negation $\neg(h, r, t, \alpha, \beta)$. Here, r denotes the relation, h and t can be either an entity in $\mathcal{E}$ or a variable in $\{y, x_1, ..., x_n\}$. α represents the necessity value, and β represents the importance value. The $\owedge$ and $\ovee$ represent soft conjunction and disjunction operation.

Definition 4 (Substitution)**.** *For a soft query involving variables, the substitution replaces all occurrences of the variable x (or y) with any entity $s \in \mathcal{E}$ simultaneously, denoted as s/x (or s/y).*

[1]The values of uncertain KGs also indicate the strength or importance. For simplify, previous work (Pai and Costabello, 2021) normalized the range of values into the interval $[0,1]$.

[2]For details on logical queries, please refer to Appendix D.

We denote $\phi(s)$ for the result of substituting s for the free variable y. When all variables in the soft query ϕ have been substituted, we refer to it as the substituted query. Next, we define the semantics of the soft queries, starting with the soft atomic formula. Specially, the soft atomic formula involves two novelty concepts: α necessity and β importance, which are inspired form soft Constraint Satisfaction Problems (CSPs) (Rossi et al., 2006) to manipulate the uncertainty of facts. We introduce the related work of soft CSPs in Appendix F.

1. The α necessity component draws inspiration from possibilistic CSPs (Schiex, 1992) and is designed to capture **necessity criteria**. It serves the purpose of filtering out unnecessary constraints and involves a thresholding operation. The thresholding operation $[p]_\alpha$ is defined as follows:

$$[p]_\alpha = \begin{cases} p & p \geq \alpha, \\ \textcircled{0} & \text{otherwise.} \end{cases} \tag{4}$$

2. The β importance component is influenced by weighted CSPs (Bistarelli et al., 1999) to describe **preference**, which is the weight employed to adjust the relative significance of different conditions.

Definition 5 (Semantic of soft queries)**.** *Given a semiring $(\mathbb{R}^+, \oplus, \otimes, \textcircled{0})$ over $\mathbb{R}^+$, the confidence function P induced by an uncertain knowledge graph $\mathcal{G}$, and a soft query ϕ, let s and o be entities in $\mathcal{E}$. The confidence value $U(\phi, P)$ is recursively defined as follows:*

1. *If ϕ is the substituted soft atomic query (s, r, o, α, β), then $U(\phi, P) = \beta[P(s, r, o)]_\alpha$;*
2. *If ϕ is the negation of the substituted soft atomic $\neg(s, r, o, \alpha, \beta)$, then $U(\phi, P) = \beta[1 - P(s, r, o)]_\alpha$;*
3. *If $\phi = \exists x_i \psi(y; x_i)$ is the soft query involving existentially quantified variables, then $U(\phi, P) = \oplus_{s \in \mathcal{E}} U(\phi(y; s/x_i), P)$;*
4. *If ϕ is the conjunctive query $(\phi_1 \textcircled{\wedge} \phi_2)$, then $U(\phi, P) = U(\phi_1, P) \otimes U(\phi_2, P)$.*
5. *If Φ is the disjunctive query $(\Phi_1 \textcircled{\vee} \Phi_2)$, then $U(\Phi, P) = U(\Phi_1, P) \oplus U(\Phi_2, P)$.*

To align with the semantics of the confidence value, we instantiate the semiring as $(\otimes, \oplus, \textcircled{0}) = (+, \max, -\infty)$. We discuss the utilized semiring of the previous soft CSP setting in Appendix F. With semantics, we can compute the utility of entity s for the soft query ϕ. The utility of all entities can be conveniently represented as a vector.

Definition 6 (Utility vector)**.** *Given a confidence function P induced by an uncertain knowledge base and a soft query ϕ, the utility vector of soft query Φ, denoted as $\mathbf{u} \in \mathbb{R}^{|\mathcal{E}|}$, is defined as:*

$$\mathbf{u}_i = U(\Phi(s_i/y), P), \tag{5}$$

where s_i denotes the entity indexed by i.

3.2 Example to explain the soft queries, as well as the necessity and importance

Soft queries are a powerful tool for modeling candidate searches across various job positions, enabling nuanced assessments of qualifications. To illustrate this, we present an example of using soft queries to model candidate searches for two roles: Junior Software Developer (JSD) and Principal Investigator (PI) in machine learning. Let HAS denote the relation describing a candidate's possession of a skill, while LEAD, DEV, and ML represent leadership, development, and machine learning skills, respectively. Both roles require leadership skills, but the Principal Investigator places a significantly higher emphasis on leadership compared to the Junior Software Developer. This distinction is captured by the importance parameter β, which assigns greater weight to leadership in the query for the Principal Investigator. Additionally, the necessity parameter α serves as a threshold to filter out candidates who lack the required skills. The two roles can be modeled using the following soft queries, as further explained in Figure 1:

$$\phi_{\text{JSD}}(y) = \neg(y, \textsc{Has}, \textsc{Lead}, 0.7, 1)) \textcircled{\wedge} (y, \textsc{Has}, \textsc{Dev}, 0.5, 3),$$
$$\phi_{\text{PI}}(y) = (y, \textsc{Has}, \textsc{Lead}, 0.7, 3) \textcircled{\wedge} (y, \textsc{Has}, \textsc{ML}, 0.9, 1)).$$

Beyond recruitment, soft queries provide an effective and tailored strategy for modeling logical queries on uncertain KGs. This capability can extend the application of logical queries to uncertain KGs, enabling use cases such as product recommendation (Bai et al., 2024b) and understanding user intentions (Bai et al., 2024a).

3.3 Soft query graph and utility vector

Definition 7 (Soft query graph)**.** *Given a soft query $\phi(y; x_1, ..., x_n) = \exists x_1, ..., x_n.a_1 \textcircled{\wedge} \cdots \textcircled{\wedge} a_m$, the soft query graph G_ϕ is defined by tuples induced by soft atomic formulas or its negation: $G_\phi = \{(h_i, r_i, t_i, \alpha_i, \beta_i, \textsc{Neg}_i)\}_{i=1}^m$, where $(h_i, r_i, t_i, \alpha_i, \beta_i, \textsc{Neg}_i))$ is induced by $(h_i, r_i, t_i, \alpha_i, \beta_i)$ or $\neg(h_i, r_i, t_i, \alpha_i, \beta_i)$. $\textsc{Neg}_i$ is the bool variable indicating if a_i is negated.*

If h (or t) is a variable, we say the corresponding node in G_ϕ is a *variable node*. $V(G_\phi)$ indicates the set of all variable nodes in G_ϕ. If h (or t) is an entity, we say the corresponding node is an **constant node**. The **leaf node** is a node that is only connected to one other node in the query graph. Compared to the operation tree in complex logical queries (Ren and Leskovec, 2020), the soft query graph can model any conjunctive soft queries.

4 Methodology

In this section, we propose Soft Reasoning with calibrated Confidence values (SRC) to facilitate reasoning with various query structures and soft requirements. SRC is a symbolic reasoning method that utilizes UKGE to provide confidence values. Since UKGE inevitably has prediction errors, we present a mild assumption regarding the UKGE error bound in Equation (7). Our error analysis over SRC indicates that the inference error is manageable as the complexity of the query structure increases. To further reduce this error, we introduce two orthogonal calibration strategies: *Debiasing* (D) and *Learning* (L).

4.1 Forward inference

The main paper discusses soft queries in which the query graphs are acyclic simple graphs. Cases with the complete case (cycles and self-loops) are detailed in Appendix H.

Given the soft query ϕ, SRC efficiently derives the utility vector $U(\phi, \hat{P})$ based on the confidence function $\hat{P}$ approximated by UKGE. The core idea is to progressively prune the edges of the soft query graph while preserving the constraints of the remaining edges, ensuring that the final utility vector remains unchanged. State vectors are used to record the constraints of the pruned edges during the inference process. Specifically, each variable node $z \in G_\phi$ is described by a *state vector* $C_z \in \mathbb{R}^{|\mathcal{E}|}$. The notation $(G_\phi, \{C_z : z \in V(G_\phi)\})$ denotes a soft query graph with state vectors.

We define equivalent transformations as T:

$$T(G_\phi, \{C_z : z \in V(G_\phi)\}) = (G_\psi, \{C'_z : z \in V(G_\psi)\}), \quad (6)$$

where G_ψ is a subgraph of G_ϕ (with at least one edge eliminated), C'_z is the updated state vector, and T guarantees the utility vector $\hat{\mathbf{u}}$ unchanged.

Two lemmas are presented to induce two equivalent transformations, denoted as T_e and T_l, respectively. The proof can refer to Appendix H.

Algorithm 1 SRC (simple acyclic case)

```
Require: Input soft query graph $G_\phi$ and initialize the state vectors $\{C_z\}$.
Ensure: Output utility vector $\hat{\mathbf{u}}(G_\phi, \{C_z\})$.
  $(G_\phi, \{C_z\}) \leftarrow$ RemoveConstNode$(G_\phi, \{C_z\})$
  while There exists a leaf node do
    $(G_\phi, \{C_z\}) \leftarrow$ RemoveLeafNode$(G_\phi, \{C_z\})$
  end while
  Get the utility vector by retrieving $C_y$.
```

Lemma 1. *For each constant node in G_ϕ, an $O(|\mathcal{E}|)$ transformation T_c exists to remove it.*

Realization of the equivalent transformation T_c induces a function RemoveConstNode.

Lemma 2. *For each leaf node in G_ϕ, an $O(|\mathcal{E}|^2)$ transformation T_l exists to remove it.*

Realization of the equivalent transformation T_l induces a function RemoveLeafNode.

The above two transformations are constructive and can be applied to remove edges once the corresponding nodes are found. Until the soft query graph only contains a free variable node, the state vector of the free variable C_y is the desired utility vector. The procedure of SRC for acyclic soft query graphs is presented in Algorithm 1 and one toy example of the execution is visualized in Figure 3. We first remove constant nodes, as these are commonly found and can be easily eliminated using Lemma 1. Subsequently, we identify and remove leaf nodes step by step, according to the Lemma 2, noting that the leaf node always exists for acyclic queries.

Complexity analysis. We begin with a rough estimation of the complexity. The space complexity of the inference algorithm is $O(|\mathcal{R}||\mathcal{E}|^2)$. Let n_e denote the number of edges involving existential variables and n_r represent the remaining edges. For acyclic queries, the time complexity is $O(n_e|\mathcal{E}|^2 + n_r|\mathcal{E}|)$. We further reduce complexity by leveraging sparsity, a natural characteristic of knowledge graphs. For space complexity, only non-zero values or those exceeding a threshold δ_1 are stored and the average sparsity ratio of our constructed neural matrices is approximately 3.4%, reducing storage requirements by 97%. For time complexity, sparsity can be used to accelerate computation. Removing a leaf node x typically involves $O(E^2)$ operations, but by considering only rows where C_x is non-zero, this reduces to $O(\mathcal{E} \cdot (|C_x > \delta_2|))$, where $|C_x > \delta_2|$ represents the

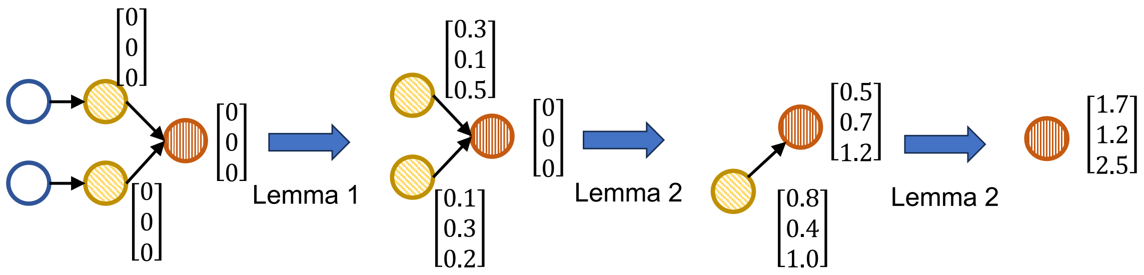

Figure 2: A toy model illustrating the process of the SRC. Each variable node is assigned a state vector, which is updated through the algorithm as edges are removed. The final state vector of the free variable is the desired.

non-zero number of C_x. Thus, the time complexity becomes $O(n_e\mathcal{E} \cdot \max_x(|C_x > \delta_2|))$, where n_e is the number of edges involving existential variables.

4.2 Error analysis

The current model still exhibits significant prediction errors, with the mean absolute error on CN15K generally reaching around 0.2 (Chen et al., 2021a,b). To facilitate the error analysis of our proposed search algorithm, we introduce the error bound $\varepsilon(\delta)$ as the following:

Definition 8. *Let ε be a function that maps the error δ to a tail probability which can uniformly bounds the error of $\hat{P}$ for some kind of norm:*

$$\Pr\left(\max_{(s,r,o)} \|\hat{P}(s,r,o) - \mathcal{P}(s,r,o)\| > \delta\right) < \varepsilon(\delta). \quad (7)$$

We note that $\varepsilon(\delta) < 1$ is guaranteed for all δ and $\varepsilon(\delta)$ decreases monotonically with δ, though a better link predictor provides a tighter $\varepsilon(\delta)$. This definition provides a practical and flexible approach for analyzing the errors over uncertain KGs. We look into the error of each soft atomic query:

Theorem 1. *For any soft atomic query $\psi = (h, r, y, \alpha, \beta)$, let the uniform inference error be*

$$\max_{\psi, s\in\mathcal{E}} \|U(\psi(s/y), \hat{P}) - U(\psi(s/y), \mathcal{P})\| = \epsilon(\alpha, \beta)$$

Then we estimate the distribution of $\epsilon(\alpha,\beta)$ by the uniform error-bound $\varepsilon(\delta)$ provided in Equation (7) and assume the probability density function of $\mathcal{P}$ is $f(\xi)$:

$$\Pr\left(\epsilon(\alpha,\beta) > \delta\right) < \varepsilon(\frac{\delta}{\beta}) + (1 - \varepsilon(\frac{\delta}{\beta})) \int_0^1 \varepsilon(|\alpha - \xi|) f(\xi) \mathrm{d}\xi.$$

Moreover, the numerical stability is guaranteed:

Theorem 2. *For a soft conjunctive query $\phi = \exists x_1, ..., x_n.a_1 \owedge \cdots \owedge a_m$, where $a_i = (h_i, r_i, t_i, \alpha_i, \beta_i)$, and any entity $s \in \mathcal{E}$, the error accumulated is at most linear:*

$$\|U(\phi(s), \hat{P}) - U(\phi(s), \mathcal{P})\| \leq \Sigma_{i=1}^m \epsilon(\alpha_i, \beta_i). \quad (8)$$

This conclusion ensures that there is no catastrophic cascading error in our forward inference algorithm. The proof of all the above theorems can refer to Appendix N.

4.3 Two calibration strategies

Debiasing. The confidence function $\hat{P}$ of UKGE is biased towards zero. We propose a debiasing strategy for the inference. We modify the soft requirements α as $\alpha - \Delta_\alpha$. We can see that this simple debiasing strategy improves performance. SRC with this strategy is denoted as SRC(D).

Learning. The pre-trained UKGE is not optimal for SRC in the incomplete uncertain KGs. We propose the *calibration by learning (L)* strategy by learning the calibrated confidence function. Specifically, we calibrate the confidence function $\hat{P}_c$ by learnable affine transformation (Arakelyan et al., 2023) as following:

$$\hat{P}_c(s,r,o) = \hat{P}(s,r,o)(1 + \rho_\theta(s,r,o)) + \lambda_\theta(s,r,o), \quad (9)$$

$$[\rho_\theta(s,r,o), \lambda_\theta(s,r,o)] = \sum_{j\in\{s,r,o\}} (W_j \mathbf{e}_j + b_j).$$

Here, $\rho_\theta(s,r,o)$ and $\lambda_\theta(s,r,o)$ are the affine parameters. $\mathbf{e}_j \in \mathbb{R}^d$ represents the embedding for entity or relation $j \in \{s,r,o\}$ with embedding dimension d. Meanwhile, $W_j \in \mathbb{R}^{2\times d}$ and $b_j \in \mathbb{R}^2$ are the learnable parameters associated with j.

As implied by both Theorem 1 and Theorem 2, the error of SRC is rooted in the error bound of UKGE. As we can see from Equation (8) the error bound is governed by both ε and the integral of ε over the domain $[0, \max(\alpha, 1-\alpha)]$. An important implication is that when $\alpha = 0$, the integral of ε will be fully $[0,1]$[3]. Therefore, our theoretical analysis motivates the goal of calibration as the minimization of the mean squared error between the predicted utility and the observed utility of answers:

$$\mathcal{L} = \sum_{\mathbf{u}(s)>0} (U(\phi(s), \hat{P}_c) - U(\phi(s), \mathcal{P}))^2, \quad (10)$$

[3]The case of $\alpha = 1$ ruled out almost all uncertain cases, which is not applicable in differentiable learning.

where $U(\phi(s), \hat{P}_c)$ represents the predicted utility vector of a soft query ϕ according to Definition 6. SRC with this strategy is denoted as SRC(L).

Notably, we only need to train the calibration transformation in the cases of $\alpha = 0$, achieving a simpler training strategy but better generalization capability when compared to the QE+NE baselines, as will be presented in Appendix B.

5 SQUK Dataset Construction

We provide a brief overview of dataset construction and the details can refer to Appendix K.

5.1 Useful queries and evaluation protocols

The validation/test uncertain knowledge graph incorporates new facts that will update the utility vectors of specific soft queries. Only these particular queries are considered meaningful and included in the evaluation. The evaluation of soft queries not only considers recall but also accounts for the values of the recalled answers. Therefore, we adopt metrics from the learning-to-rank framework (Liu et al., 2009) as our evaluation protocol, which includes Mean Average Precision (MAP), Normalized Discounted Cumulative Gain (NDCG), Spearman's rank correlation coefficient (ρ), and Kendall's rank correlation coefficient (τ).

We choose 1P, 2P, 2I, 2IN, and 2IL as train query types and add 3IN, INP, IP, 2M, and IM as validation/test query types. The training query types encompass basic operations, allowing us to evaluate the ability of machine learning methods to generalize to commonly used unseen query types (Yin et al., 2024). We visualize the structure of these types in Figure 4. Additionally, Table 8 shows the statistics of training queries of our dataset.

Table 2: The statistics of train queries.

KG	1P	2P	2I	2IN	2IL
PPI5k	9,724	9,750	9,754	1,500	1,500
O*NET20k	18,266	18,300	18,300	1,850	1,850
CN15k	52,887	52,900	52,900	5,300	5,300

5.2 Uncertain KGs

We utilize three standard uncertain KGs: CN15k (Chen et al., 2019) for encompassing commonsense, PPI5k (Chen et al., 2019) for biology, and O*NET20K (Pai and Costabello, 2021) for employment domains. These uncertain KGs are noisy and incomplete, requiring the ML models to predict the confidence values.

5.3 Soft requirements

For the α parameter, we establish connections with the percentile value of the relation to represent the necessity value effectively. We assign specific percentiles to different necessity levels: the 25th, 50th, and 75th percentiles correspond to "low", "normal", and "high" necessity criteria, respectively. We ensure that a "zero" requirement is assigned when the necessity criteria reaches 0. We also introduce a hybrid strategy that randomly selects necessity values, enabling a comprehensive evaluation.

For the β importance setting, we employ two strategies: "equal" and "random". Under the "equal" strategy, all importance values are assigned an importance value of 1.0. In contrast, the "random" strategy introduces variability by assigning random decimal numbers between 0 and 1 to represent the importance of each soft atomic formula.

6 Experiments

In this section, we empirically explore how to answer soft queries. We mainly compare our method with generalized SoTA CQA models on SQUK dataset, including commonly used query embedding models and advanced symbolic search methods. Additionally, we evaluate the performance of advanced commercial LLMs on soft queries with clear natural language descriptions. The implementation details of these experiments are in Appendix G. We also conduct the ablation study regarding the distribution and impact of two parameters α and β on both kinds of approaches, which is presented in Appendix B due to page limitations. We also provide the qualitative analysis of the experiments in Appendix C.

6.1 Main results

Baselines We select two mainstream CQA methods as baselines: query embedding and symbolic search. Specifically, we focus on two classical query embedding methods: LogicE (Luus et al., 2021) and ConE (Zhang et al., 2021). To enable soft queries, we incorporate the relation projection network with Number Embedding (NE) and adjust the loss function accordingly. [4] The forward inference of our method, SRC, is directly generalized from SoTA symbolic methods FIT (Yin et al., 2024), which serve as baselines for search methods.

Models analysis. The main results are presented in Table 3, demonstrating that our proposed method

[4]Detailed information can be found in Appendix J.

Table 3: Result of answering soft queries. Logic+NE and ConE+NE refer to the query embedding with number embedding extensions. SRC is our inference method, and SRC(D), SRC(L), and SRC(D+L) are explained in Section 4.3. The four metrics are all higher, indicating better performance.

Uncertain KG	Models	τ													AVG. ρ	AVG. MAP	AVG. NDCG
		1P	2P	2I	2IN	2IL	2M	2U	3IN	IP	IM	INP	UP	AVG.			
CN15k	LogicE+NE	9.1	-1.5	4.8	6.0	18.3	5.1	-14.1	3.5	-2.4	6.4	-0.9	9.6	4.8	5.8	7.0	11.2
	ConE+NE	5.3	4.3	3.5	6.3	18.4	6.9	20.5	2.9	1.8	10.4	1.7	14.9	8.1	10.0	7.7	13.2
	SRC	15.0	2.4	-0.0	2.1	10.7	9.2	25.5	-2.0	-9.0	7.9	-4.4	13.0	5.9	8.9	9.2	15.5
	SRC(D)	16.6	11.8	-0.6	6.9	10.9	11.7	34.4	0.1	5.2	12.5	4.7	24.2	11.5	15.1	12.9	21.8
	SRC(L)	15.8	11.8	-0.4	2.4	11.0	12.4	32.3	-0.8	3.6	11.1	1.1	22.8	10.3	13.7	12.6	21.1
	SRC(D+L)	15.6	13.4	-0.3	5.2	11.2	13.5	36.8	-0.4	8.2	12.2	4.8	28.2	**12.4**	**16.2**	**13.7**	**23.2**
PPI5k	LogicE+NE	20.5	22.6	17.1	10.4	24.4	20.0	30.4	9.1	12.4	14.1	-2.6	32.9	14.8	20.7	8.0	16.4
	ConE+NE	29.2	42.5	26.4	20.7	32.6	33.9	35.9	16.6	36.6	29.4	22.5	42.2	30.7	40.8	44.1	49.2
	SRC	66.6	70.9	49.9	42.7	71.0	42.9	71.6	32.7	65.6	37.1	57.2	70.4	56.5	66.7	68.4	70.7
	SRC(D)	66.7	68.7	53.1	44.9	73.1	46.7	73.3	37.7	63.7	41.3	56.9	69.9	58.0	68.5	64.0	69.8
	SRC(L)	66.8	71.7	52.7	43.4	72.8	43.8	72.7	34.4	66.6	38.3	58.0	71.4	57.7	67.8	**69.8**	**71.6**
	SRC(D+L)	66.9	69.0	53.5	45.1	73.5	46.9	73.4	38.1	63.8	41.6	57.1	69.9	**58.2**	**68.7**	64.1	70.1
O*NET20k	LogicE+NE	6.3	9.5	43.5	3.9	36.6	9.7	15.3	8.3	11.1	8.8	3.8	-9.8	13.8	18.5	3.5	6.4
	ConE+NE	30.8	41.9	57.0	21.8	46.0	37.7	49.7	48.0	22.7	21.7	14.2	53.1	36.8	47.2	27.5	38.7
	SRC	72.0	54.9	68.6	67.6	67.3	36.9	76.0	59.2	47.6	29.1	48.9	52.4	57.3	65.3	27.1	41.3
	SRC(D)	71.7	55.2	74.3	67.7	70.9	49.3	80.1	65.0	52.7	44.7	48.9	55.4	61.8	70.2	26.6	41.5
	SRC(L)	71.6	56.7	69.8	66.8	68.4	38.2	77.6	59.9	51.3	32.1	49.8	55.4	58.7	66.5	**27.6**	**41.8**
	SRC(D+L)	71.7	55.6	74.3	67.5	71.0	49.7	80.2	65.0	52.9	45.2	49.2	55.9	**61.9**	**70.5**	26.7	41.7

significantly outperforms both query embedding methods and directly generalized symbolic methods. By leveraging the two calibration strategies, debiasing (D) and learning (L), as explained in Section 4.3, our method achieves superior results across most KGs and metrics on average.

Query structure analysis. Although ConE performs well on some trained query types, it struggles with newly emerged query types and those involving negation, such as INP and IM. In contrast, our method exhibits excellent performance across the majority of query types, demonstrating robust combinatorial generalization capabilities on complex queries. Our method particularly excels in handling challenging query types that involve existential variables, such as 2P, 2M, IM, and INP, highlighting its advantages in these scenarios.

6.2 The comparison with LLMs

Table 4: The accuracy of manually annotated queries.

Model	Llama3.3 70B	Gemini-1.5-pro	GPT-3.5-turbo	GPT-4-preview	SRC
Accuracy	42.6	37.1	34.3	37.8	**48.9**

We devise an evaluation framework to assess the performance of LLMs, benchmarking their powerful reasoning abilities over uncertain knowledge. To ensure fairness of the comparison, we consider the queries sampled from CN15k and we choose four candidate answers for each query. These queries have also been manually filtered and labeled to ensure clearness and correctness. We describe the syntax and semantics of soft queries using natural language, prompting LLMs to select the most suitable answer. The details on this setting construction can refer to Appendix M.

The results, shown in Table 4, indicate that even the simple symbolic SRC achieves significantly higher accuracy compared to Llama3.3 70B, Gemini-1.5-pro, GPT-3.5-turbo, and GPT-4-preview. This demonstrates that large language models (LLMs) struggle with complex arithmetic operations involving uncertain values of knowledge. Our evaluation is fair, as the required uncertain knowledge is derived from well-known commonsense KGs ConceptNet, and the logical operations are expressed in natural language. Nevertheless, even advanced commercial LLMs struggle to select the highest-scoring answer. This further emphasizes the difficulties presented by the proposed soft queries and highlights the ongoing need for the development of symbolic approaches.

7 Conclusion

In this paper, we introduce a novel setting, soft queries on uncertain KGs, which further extends complex logical queries on KG. The soft queries consider the incompleteness of large-scale uncertain KGs and require the incorporation of ML methods to estimate scores for new relational linking while handling semiring algebraic structures. Our proposed soft queries also propose the soft requirements inspired by soft constraint satisfaction problems to control the uncertainty of knowledge. To facilitate the research of soft queries, we construct a soft query answering dataset consisting of three uncertain KGs. Furthermore, we propose a new neural-symbolic approach with both forward inference and backward calibration. Both theoretical analysis and experimental results demonstrate that our method has satisfactory performance.

8 Limitation

Soft queries extend complex logical queries over Knowledge Graphs (KGs) by incorporating soft requirements within uncertain KGs. However, the scope of the proposed soft queries is limited, as it primarily focuses on conjunctive queries. While conjunctive queries form the foundation of complex logical queries, this restriction may hinder the expressiveness and applicability of the proposed soft queries. Furthermore, the dataset does not include cyclic queries, which are NP-complete, even though their complexity can be more easily addressed.

9 Potential Impact

Soft queries have the potential to perpetuate existing biases present in the underlying knowledge graphs. If these graphs contain skewed or discriminatory information, the results generated by soft queries may reflect and amplify these biases, leading to unfair outcomes in applications such as hiring, credit scoring, or law enforcement. This raises significant ethical concerns about fairness, as marginalized groups may be disproportionately affected by biased query results, resulting in systemic inequality.

The utilization of soft queries to extract information from knowledge graphs can pose serious privacy risks. If queries access sensitive personal data without proper safeguards, there is a potential for unauthorized disclosures that violate individuals' privacy rights. This concern is heightened in contexts where the data might be used for profiling or surveillance, making it imperative to establish robust privacy protections and ethical guidelines to ensure that individuals' information is handled responsibly and transparently.

References

Ernest Wilcox Adams. 1996. A primer of probability logic.

Alfonso Amayuelas, Shuai Zhang, Xi Susie Rao, and Ce Zhang. 2021. Neural Methods for Logical Reasoning over Knowledge Graphs.

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. 2021. Complex Query Answering with Neural Link Predictors. In *International Conference on Learning Representations*.

Erik Arakelyan, Pasquale Minervini, and Isabelle Augenstein. 2023. Adapting Neural Link Predictors for Complex Query Answering. *arXiv preprint*. ArXiv:2301.12313 [cs].

Jiaxin Bai, Chen Luo, Zheng Li, Qingyu Yin, and Yangqiu Song. 2024a. Understanding inter-session intentions via complex logical reasoning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 71–82.

Jiaxin Bai, Yicheng Wang, Tianshi Zheng, Yue Guo, Xin Liu, and Yangqiu Song. 2024b. Advancing abductive reasoning in knowledge graphs through complex logical hypothesis generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1312–1329.

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.

Yushi Bai, Xin Lv, Juanzi Li, and Lei Hou. 2023. Answering Complex Logical Queries on Knowledge Graphs via Query Computation Tree Optimization. In *Proceedings of the 40th International Conference on Machine Learning*, pages 1472–1491. PMLR. ISSN: 2640-3498.

Stefano Bistarelli, Ugo Montanari, Francesca Rossi, Thomas Schiex, Gérard Verfaillie, and Hélene Fargier. 1999. Semiring-based csps and valued csps: Frameworks, properties, and comparison. *Constraints*, 4:199–240.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam Hruschka, and Tom Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 24, pages 1306–1313.

Roger Cavallo and Michael Pittarelli. 1987. The theory of probabilistic databases. In *VLDB*, volume 87, pages 1–4.

Ismail Ilkan Ceylan, Adnan Darwiche, and Guy Van den Broeck. 2021. Open-world probabilistic databases: Semantics, algorithms, complexity. *Artificial Intelligence*, 295:103474.

Jianshu Chen. 2023. Learning Language Representations with Logical Inductive Bias. *arXiv preprint*. ArXiv:2302.09458 [cs].

Xuelu Chen, Michael Boratko, Muhao Chen, Shib Sankar Dasgupta, Xiang Lorraine Li, and

Andrew McCallum. 2021a. Probabilistic box embeddings for uncertain knowledge graph reasoning. *arXiv preprint arXiv:2104.04597*.

Xuelu Chen, Muhao Chen, Weijia Shi, Yizhou Sun, and Carlo Zaniolo. 2019. Embedding uncertain knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3363–3370.

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

Zhu-Mu Chen, Mi-Yen Yeh, and Tei-Wei Kuo. 2021b. Passleaf: A pool-based semi-supervised learning framework for uncertain knowledge graph embedding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4019–4026.

Nurendra Choudhary, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan Reddy. 2021. Probabilistic entity representation model for reasoning over knowledge graphs. *Advances in Neural Information Processing Systems*, 34:23440–23451.

Will Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. 2018. Embedding logical queries on knowledge graphs. *Advances in neural information processing systems*, 31.

Olaf Hartig and Ralf Heese. 2007. The sparql query graph model for query optimization. In *European Semantic Web Conference*, pages 564–578. Springer.

Shuang Liang. 2023. Knowledge graph embedding based on graph neural network. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 3908–3912.

Lihui Liu, Boxin Du, Heng Ji, ChengXiang Zhai, and Hanghang Tong. 2021. Neural-Answering Logical Queries on Knowledge Graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1087–1097.

Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331.

Francois Luus, Prithviraj Sen, Pavan Kapanipathi, Ryan Riegel, Ndivhuwo Makondo, Thabang Lebese, and Alexander Gray. 2021. Logic embeddings for complex query answering. *arXiv preprint arXiv:2103.00418*.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41.

Sumit Pai and Luca Costabello. 2021. Learning embeddings from knowledge graphs with numeric edge attributes. *arXiv preprint arXiv:2105.08683*.

Raymond Reiter. 1981. On closed world data bases. In *Readings in artificial intelligence*, pages 119–140. Elsevier.

Raymond Reiter. 1986. A sound and sometimes complete query evaluation algorithm for relational databases with null values. *Journal of the ACM (JACM)*, 33(2):349–370.

H Ren, W Hu, and J Leskovec. 2020. Query2box: Reasoning Over Knowledge Graphs In Vector Space Using Box Embeddings. In *International Conference on Learning Representations (ICLR)*.

Hongyu Ren, Mikhail Galkin, Michael Cochez, Zhaocheng Zhu, and Jure Leskovec. 2023. Neural Graph Reasoning: Complex Logical Query Answering Meets Graph Databases. *arXiv preprint*. ArXiv:2303.14617 [cs].

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. *Advances in Neural Information Processing Systems*, 33:19716–19726.

Kaspar Riesen, Xiaoyi Jiang, and Horst Bunke. 2010. Exact and inexact graph matching: Methodology and applications. *Managing and mining graph data*, pages 217–247.

Francesca Rossi, Peter van Beek, and Toby Walsh. 2006. *Handbook of Constraint Programming*. Elsevier Science Inc., USA.

Thomas Schiex. 1992. Possibilistic constraint satisfaction problems or "how to handle soft constraints?". In *Uncertainty in Artificial Intelligence*, pages 268–275. Elsevier.

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.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. 2022. *Probabilistic databases*. Springer Nature.

Damian Szklarczyk, Rebecca Kirsch, Mikaela Koutrouli, Katerina Nastou, Farrokh Mehryary, Radja Hachilif, Annika L Gable, Tao Fang, Nadezhda T Doncheva, Sampo Pyysalo, et al. 2023. The string database in 2023: protein–protein association networks and functional enrichment analyses for any sequenced genome of interest. *Nucleic acids research*, 51(D1):D638–D646.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85. Publisher: ACM New York, NY, USA.

Zihao Wang, Weizhi Fei, Hang Yin, Yangqiu Song, Ginny Y Wong, and Simon See. 2023a. Wasserstein-Fisher-Rao Embedding: Logical Query Embeddings with Local Comparison and Global Transport. *arXiv preprint arXiv:2305.04034*.

Zihao Wang, Yangqiu Song, Ginny Wong, and Simon See. 2023b. Logical Message Passing Networks with One-hop Inference on Atomic Formulas. In *The Eleventh International Conference on Learning Representations*.

Zihao Wang, Hang Yin, and Yangqiu Song. 2021. Benchmarking the Combinatorial Generalizability of Complex Query Answering on Knowledge Graphs. *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 1.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Dong Yang, Peijun Qing, Yang Li, Haonan Lu, and Xiaodong Lin. 2022a. GammaE: Gamma Embeddings for Logical Queries on Knowledge Graphs. *arXiv preprint*. ArXiv:2210.15578 [cs].

Haotong Yang, Zhouchen Lin, and Muhan Zhang. 2022b. Rethinking knowledge graph evaluation under the open-world assumption. *Advances in Neural Information Processing Systems*, 35:8374–8385.

Hang Yin, Zihao Wang, and Yangqiu Song. 2024. Rethinking existential first order queries and their inference on knowledge graphs. In *The Twelfth International Conference on Learning Representations*.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020. Aser: A large-scale eventuality knowledge graph. In *Proceedings of the web conference 2020*, pages 201–211.

Zhanqiu Zhang, Jie Wang, Jiajun Chen, Shuiwang Ji, and Feng Wu. 2021. Cone: Cone embeddings for multi-hop reasoning over knowledge graphs. *Advances in Neural Information Processing Systems*, 34:19172–19183.

Zhaocheng Zhu, Mikhail Galkin, Zuobai Zhang, and Jian Tang. 2022. Neural-Symbolic Models for Logical Queries on Knowledge Graphs. *arXiv preprint arXiv:2205.10128*.

Appendix

A Related Work

A.1 Complex logical queries

Answering complex logical queries over knowledge graphs is naturally extended from link prediction and aims to handle queries with complex conditions beyond simple link queries. This task gradually grows by extending the scope of complex logical queries, ranging from conjunctive queries (Hamilton et al., 2018) to Existential Positive First-Order (EPFO) queries (Ren et al., 2020), Existential First-Order (EFO) queries (Ren and Leskovec, 2020), real Existential First-Order queries (Yin et al., 2024). The primary method is query embedding, which maps queries and entities to a low-dimensional space. The form of embedding has been well investigated, such as vectors (Hamilton et al., 2018; Chen et al., 2022; Bai et al., 2022), geometric regions (Ren et al., 2020; Zhang et al., 2021), and probabilistic distributions (Ren and Leskovec, 2020; Choudhary et al., 2021; Yang et al., 2022a; Wang et al., 2023a). These methods not only explore knowledge graphs embedding but also leverage neural logical operators to generate the embedding of complex logical queries.

There are also neural-symbolic models to answer complex logical queries. Gradient optimization techniques were employed to estimate the embedding existential variables (Amayuelas et al., 2021; Arakelyan et al., 2023). Graph neural network (Zhu et al., 2022) was adapted to execute relational projects and use logical operations over fuzzy sets to deal with more complex queries. Efficient search algorithms based on link predictor over knowledge graphs were presented (Yin et al., 2024; Bai et al., 2023). While symbolic methods demonstrate good performance and offer interpretability for intermediate variables, they often struggle to scale with larger graphs due to their high computational complexity.

Many other models and datasets are proposed to enable answering queries with good performance and additional features, see the comprehensive survey (Ren et al., 2023). However, to the best of our knowledge, there is currently no existing query framework specifically designed for uncertain knowledge graphs.

A.2 Uncertain knowledge graph embedding

Uncertain knowledge graph embedding methods aim to map entities and relations into low-dimensional space, enabling the prediction of unknown link information along with confidence values. There are two primary research directions in this field.

The first line of research focuses on predicting the confidence score of uncertain relation facts. UKGE (Chen et al., 2019) was the pioneering effort to model triple plausibility as the activated product of these embedding vectors. UKGE incorporates soft probabilistic logic rules to provide the plausibility of unseen facts. Building upon this, BEUrRE (Chen et al., 2021a) utilizes complex geometric boxes with probabilistic semantics to represent entities and achieve better performance. Semi-supervised learning was applied (Chen et al., 2021b)to predict the associated confidence scores of positive and negative samples. And Graph neural networks were used (**?**Liang, 2023) to represent and predict uncertain knowledge graphs.

The other line of research aims to address link prediction on uncertain knowledge graphs by fitting the likelihood of uncertain facts. To adjust the similar task, FocusE (Pai and Costabello, 2021) was introduced, an additional layer to the knowledge graph embedding architecture. They provide variants of classical embedding methods such as TransE (Bordes et al., 2013), DistMult (Yang et al., 2015), and ComplEx (Trouillon et al., 2016).

B Ablation study: The impact of soft requirements

The two parameters, α and β play a crucial role in controlling soft constraints, thus we construct settings with varying values. Specifically, we select "zero"(Z) and "random"(R) for α, and "equal"(E) and "random"(R) for β, as explained in Section 5.3. Moreover, we sample 12 query types from O*NET20k KG. The detailed construction is in Appendix L. For ConE, we train it on each setting and test it across all settings. As for SRC, we directly test it on all settings.

Table 5: The mean NDCG of varing α and β.

Model	Train	Test				AVG.
		Z+E	Z+R	N+E	N+R	
ConE+NE	Z+E	**39.3**	35.6	31.9	31.5	34.6
	Z+R	39.0	**42.2**	27.8	28.7	34.4
	N+E	26.6	23.6	46.3	44.1	35.2
	N+R	14.0	7.5	43.5	42.2	26.8
SRC	-	34.3	37.2	**46.7**	**45.8**	**41.0**

The results in Table 5 demonstrate that SRC outperforms ConE trained on four different settings in terms of average scores. In the "Z+E" and "Z+R" settings, although ConE achieves higher scores when trained and tested within the same setting, its performance considerably declines when generalizing to other settings. Our method SRC exhibits consistent performance across various settings due to its strong generalization by theoretical foundations in Section 4.2.

C Qualitative analysis

We select the following query for the LLM to conduct a qualitative analysis: " Soft query: (bread, is related to, f_1, 0.3, 0.9) $\wedge$ (rice, is related to, f_1, 0.7, 0.3). Four candidate entities: basic, wheat, white, and food. " Among the four candidates, basic and wheat are weakly related to the two soft constraints. While white and food exhibit higher relevance, the constraints related to bread carry greater importance. Consequently, the final answer is bread. However, some LLMs may often select rice due to a tendency to hallucinate the concept of white bread. Such errors highlight the challenges in accurately interpreting soft constraints and the need for more robust mechanisms to handle nuanced queries.

D Logical queries on knowledge graphs

Definition 9 (Knowledge graphs)**.** *Let $\mathcal{E}$ be the set of entities and $\mathcal{R}$ be the set of relations. A knowledge graph is a set of triples $\mathcal{G} = \{(s_i, r_i, o_i)\}$, where $s_i, o_i \in \mathcal{E}$ are entities and $r_i \in \mathcal{E}$ is relation.*

The fundamental challenge of knowledge graphs lies in dealing with the Open World Assumption (OWA). Unlike the Closed World Assumption (CWA), which considers only observed triples as facts, OWA acknowledges that unobserved triples may also be valid.

The study of logical queries on KG considers the Existential First-Order (EFO) queries, usually with one free variable (Ren and Leskovec, 2020; Wang et al., 2021; Yin et al., 2024).

Definition 10 (Syntax of existential first-order queries)**.** *The disjunctive normal form of an existential first-order query Γ is:*

$$\Gamma(y) = \gamma_1(y) \vee \cdots \vee \gamma_q(y), \tag{11}$$

where y is the variable to be answered. Each $\gamma_i(y)$ is a conjunctive query that is expanded as

$$\gamma_i(y) = \exists x_1, \ldots, x_n . a_{i1} \wedge \cdots \wedge a_{im_i}, i = 1, ..., q, \tag{12}$$

where $x_1, \ldots, x_n$ are existentially quantified variables, each $a_{ij} = r(h,t)$ or $a_{ij} = \neg r(h,t), j = 1, ..., m_i$ is an atomic query, r is the relation, h and t are either an entity in $\mathcal{E}$ or a variable in $\{y, x_1, ..., x_n\}$.

Definition 11. *$\Gamma(s/y)$ denotes the substitution of the entity s for the variable y.*

When all free variables are substituted, **a query** $\Gamma(y)$ is transformed into **a sentence** $\Gamma(s/y)$. Given $\mathcal{G}$, answering a query $\Gamma(y)$ means finding all substitutions, such that the sentence $\Gamma(s/y)$ is entailed by $\mathcal{G}$, i.e., $\mathcal{G} \models \Gamma(s/y)$. The answer set is defined as

Definition 12 (The Answer Set of first-order 1uery)**.** *The answer set of an first-order query is defined by*

$$\mathcal{A}[\Phi(y)] = \{a \in \mathcal{E} | \ \Phi(a) \text{ is True}\} \tag{13}$$

The answers in the answer set derived from $\mathcal{G}_{\text{train}}$ is easy answers. hard answers are the answers in the set difference between the answers from $\mathcal{G}_{\text{valid}}$ and $\mathcal{G}_{\text{train}}$ (Wang et al., 2021; Ren et al., 2023). The traditional graph-matching methods can not find the answers introduced by new facts (Riesen et al., 2010). Thus, we should develop new methods.

D.1 ML-based method for logical queries on KG

Recent works are dedicated to introducing ML methods, i.e., knowledge graph embeddings, to generalize from $\mathcal{G}$ to $\widehat{\mathcal{G}}$, so that it approximates $\mathcal{G}_{\text{test}}$.

Query Embeddings (QE). Query embedding methods generally map the query Γ as an embedding (Ren and Leskovec, 2020; Liu et al., 2021; Wang et al., 2023b). One dominant way is to "translate" the procedure of solving logical formulas in Equation (11) into the set operations, such as set projection, intersection, union, and complement. Then, the neural networks are designed to model such set operations in the embedding space. We can see that direct modeling of set operations is incompatible with the concept of necessity and importance introduced in Section 3.

Inference methods. Other methods solve the open world query answering methods in a two-step approach (Arakelyan et al., 2021; Bai et al., 2023; Yin et al., 2024). In the first step, the pre-trained knowledge graph embedding estimates the $\mathcal{G}_{\text{test}}$. Then, the answers are derived by the fuzzy logic inference or optimization. These methods rely on the standard logic calculation and cannot be directly applied to our SQUK setting introduced in Section 3.

E Connection with Open World Assumption

Evaluating queries over deductive question-answering systems generally follows either the closed-world assumption (CWA) or the open-world assumption (OWA) (Reiter, 1981). Under CWA, only known facts are considered true, whereas OWA assumes that the absence of knowledge does not imply falsity (Reiter, 1986). Since knowledge graphs (KGs) are often incomplete, knowledge graph completion (KGC) has been proposed to address this challenge. In KGC, the observed knowledge in KGs is divided into three nested subsets: $\mathcal{G}_{\text{train}} \subset \mathcal{G}_{\text{valid}} \subset \mathcal{G}_{\text{test}}$. The KGC model is trained on the G_{train} subset and evaluated on the G_{valid} subset to assess its performance. For complex logical query answering, the evaluation considers "hard answers," defined as the set difference between the answers from G_{valid} and G_{train} (Wang et al., 2021; Ren et al., 2023). This approach assesses the model's ability to handle incomplete knowledge and make inferences beyond the observed facts in the training set. The evaluation of complex logical query answering extends beyond CWA but does not fully align with OWA. The same applies to the evaluation of soft query answering. Evaluating under the Open-World Assumption represents a promising avenue for future work (Yang et al., 2022b).

F Constraint Satisfaction Problem and Soft Constraint Satisfaction Problems

Constraint Satisfaction Problems (CSP) is a mathematical question defined as a set of objects whose state must satisfy several constraints or limitations. Each instance of it can be represented as a triple (Z, D, C), where $Z = (z_1, \cdots, z_n)$ is a finite tuple of n variables, $D = (D_1, \cdots, D_n)$ is the tuple of the domains corresponding to variables in Z, and $C = \{(C_1^1, C_1^2), \cdots, (C_t^1, C_t^2)\}$ is the finite set of t constraints. D_i is the domain of z_i, and $C_i^1 \subset Z$ is the scope of the i-th constraint and C_i^2 specifies how the assignments allowed by this constraint. In general definitions, the constraints in classical Constraint Satisfaction Problems are hard, meaning that none of them can be violated.

Many problems can be viewed as CSP, which include workforce scheduling and the toy 8-queens problem. Conjunctive queries are a special case to be reduced as CSP under open-world assumptions if we set the constant variable's domain as itself, set the domain of the existential variable and free variable as the entity set $\mathcal{E}$ of knowledge graphs, and treat atomic formula or its negation as binary constraint by knowledge graph.

Soft Constraint Satisfaction Problems

Though CSP is a very powerful formulation, it fails when real-life problems need to describe the preference of constraint. It usually returns null answers for problems with many constraints, which are

Table 6: Different specific soft CSP frameworks modeled as c-semirings.

Semiring	E	$\times_s$	$+_s$
Classical	$\{T, F\}$	$\wedge$	$\vee$
Fuzzy	$[0, 1]$	min	max
k-weighted	$\{0, \dots, k\}$	$+$	min
Probabilistic	$[0, 1]$	xy	max
Valued	E	$\oplus$	min_v

called over-constraints. To tackle the above weakness, many versions of Soft Constraint Satisfaction Problems (SCSP) are developed, such as fuzzy CSP, weighted CSP, and probabilistic CSP, which all follow a common semiring structure, where two semiring operations $\times_s, +_s$ are utilized to model constraint projection and combination respectively. Based on this theoretical background, we propose soft queries based on SCSP. The proposed soft queries will have the advantages of SCSPs and can handle the numeric facts representing uncertainty.

G Details of Implementation

Our experiments are run on the Nvidia V100-32G.

G.1 adaptive scoring

Let E_s, E_r, and E_t be the embedding vectors of entity s, relation r, and entity t respectively. We parameterize the adaptive scoring calibration using the following learnable affine transformations:

$$\rho_\theta(s, r, o) = W_1^1 E_s + b_1^1 + W_2^1 E_r + b_2^1 + W_3^1 E_t + b_3^1, \tag{14}$$

$$\lambda_\theta(s, r, o) = W_1^2 E_s + b_1^2 + W_2^2 E_r + b_2^2 + W_3^2 E_t + b_3^2, \tag{15}$$

where $\{W_j^i, b_j^i\}$ for $1 \leq i \leq 2$ and $1 \leq j \leq 3$ are the learnable parameters.

G.2 Uncertain knowledge graph embedding

We reproduce previous results (Chen et al., 2019, 2021a) and use the same embedding dimension. We search the other parameters including the learning rate from $\{1e-3, 5e-4, 1e-4\}$ and regularization term λ from $\{0.1, 0.01, 0.05\}$.

G.3 Query embedding with number embedding

We follow the same hyperparameter of origin paper (Luus et al., 2021; Zhang et al., 2021) but search the learning rate and margin. The embedding dimension of number embedding is the same as the dimension of entity embedding.

G.4 Two strategies of calibration

For learning strategy, we search learning rate from $\{5e-4, 5e-5, 1e-5\}$. For Debiasing strategy, we search ϵ from $\{0.05, 0.1, 0.15\}$.

H Details of Soft Reasoning with Calibrated Confidence Values

H.1 Uncertain value definition

By indexing all entities and relations, we represent s, r, and o as integers. Confidence score prediction over uncertain knowledge graphs can be conceptualized by the neural link predictor as $|\mathcal{R}|$ relation matrices $\hat{P}_r \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$, where $\hat{P}_r(s, o)$ is the predicted score of fact triple (s, r, o) and n is the number of entities. We denote $U_r(s)$ as a vector formed by the elements of the s-th row. The symbol $+$ also denotes element-wise plus operation when used in vector-vector or matrix-matrix operations. We also define two new plus operations in matrix-vector operations.

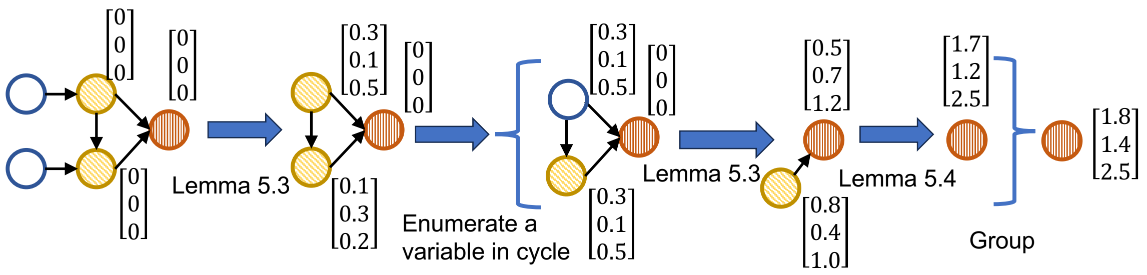

Figure 3: A toy model to present the process of SRC algorithm.

Definition 13. *Given a matrix* $\mathbf{M_1} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$ *and a vector* $\mathbf{b} \in \mathbb{R}^{|\mathcal{E}|}$*, We define two new addition operations: column-wise addition and row-wise addition as following:*

$$\mathbf{M_2} = \mathbf{M_1} +_r \mathbf{b}, \mathbf{M_2}(s,o) = \mathbf{M_1}(s,o) + \mathbf{b}(s), \tag{16}$$

$$\mathbf{M_2} = \mathbf{M_1} +_c \mathbf{b}, \mathbf{M_2}(s,o) = \mathbf{M_1}(s,o) + \mathbf{b}(o). \tag{17}$$

Definition 14 (Membership function)**.** *Given a soft query and a variable* x*,* $\mu(x, C_x)$ *is a membership function to check the current confidence value, where* $\hat{U}(\mu(s/x, C_x)) = C_x(s)$*.*

H.2 Details of the inference of SRC

Since the effect α and β are equivalent to modifying the uncertain values, we will focus on explaining how the method works in the basic scenario where $\alpha = 0.0$ and $\beta = 1.0$. Our goal is to infer the utility vector $\hat{U}[\phi(y)]$ by estimating the confidence value $\hat{U}[\phi](o) = [\hat{P} \models_s \phi(o/y)]$, for all $o \in \mathcal{E}$. In the following content, we will cut the query graph step by step while recording any lost information on the remaining nodes. After removing nodes from the soft query ϕ, we denote the remaining sub-query graph as ψ.

H.2.1 Step 1. Initialization.

We initialize each variable x as a candidate state vector C with all zero elements, denoted as $C = \mathbf{0} \in \mathbb{R}^{|\mathcal{E}|}$ This vector records the candidates and their corresponding values. Throughout the algorithmic process, the vectors C are updated iteratively, ultimately yielding the final answers represented by the resulting vector C_y.

H.2.2 Step 2. Remove constant nodes.

For constant nodes in a query graph, we can easily remove their whole edges and update the information to connected nodes by the following lemma.

Lemma 3. *For the constant nodes in a soft query, there exists an* $O(|\mathcal{E}|)$ *transformation* T_c *to remove them.*

Proof. Without loss of generality, we consider the situation that a constant node with entity e connects an existential variable x that there is only one positive edge from e to x, one negative edge from e to x, and one positive edge from x to e. To simplify, we also denote e as the related grounded entity.

$$\begin{aligned}\hat{U}[\phi](o) &= \hat{U}(\exists x.\mu(x, C_x) \wedge r_1(e,x) \wedge \neg r_2(e,x) \wedge r_3(x,e) \wedge \psi(o/y)) \\ &= \max_{s\in\mathcal{E}}[C_x(s) + \hat{P}_{r_1}(e,s) + (1 - \hat{P}_{r_2})(e,s) \\ &+ \hat{P}_{r_3}^T(e,s) + \hat{U}(\psi(o/y; s/x))] \\ &= \max_{s\in\mathcal{E}}[C'_x(s) + \hat{U}(\psi(o/y; s/x))],\end{aligned} \tag{18}$$

where $C'_x = C_x + \hat{P}_{r_1}(e) + (\mathbf{1} - \hat{P}_{r_2})(e) + \hat{P}_{r_3}^T(e)$ is updated candidate vector and $\mathbf{1} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$ is all one matrix. The situation where a constant node connects a free variable node is similar and even easier to handle. □

In the above derivation, we retain the value of answers by updating the candidate state vector of an existential variable.

H.2.3 Step 3. Remove self-loop edges.

Lemma 4. *For a soft query having self-loop edges, there exists an $O(|\mathcal{E}|)$ transformation T_c to remove self-loop edges.*

Proof. Without loss of generality, we consider the situation in which an existential variable x contains one positive loop.

$$\begin{aligned}\hat{U}[\phi](o) &= \hat{U}(\exists x.\mu(x, C_x) \wedge r(x, x) \wedge \psi(o/y; x)) \\ &= \max_{s \in \mathcal{E}}[C_x(s) + \hat{P}_r(s, s) + \hat{U}(\psi(o/y; s/x))] \\ &= \max_{s \in \mathcal{E}}[C'_x(s) + \hat{U}(\psi(o/y; s/x))],\end{aligned}$$

where $C'_x = C_x + diag(\hat{P}_r)$ and $diag(\hat{P}_r)$ is a vector formed by the diagonal elements of matrix $\hat{P}_r$. $\square$

Previous research (Yin et al., 2024) has demonstrated the difficulty of sampling high-quality self-loop queries due to the rarity of self-loop relations in real-life knowledge graphs. Similarly, in our specific uncertain knowledge graph, it is challenging to sample meaningful queries.

H.2.4 Step 4. Remove leaf nodes.

The leaf node u, which is only connected to one other node v in the soft query graph. After removing the constant nodes, if the query graph contains no circles, we can get the utility vector by removing the leaf nodes step by step. Next, we present how to handle leaf nodes by the three lemmas.

Lemma 5. *If the leaf node u is an existential variable x and v is the free variable y, there exists an $O(|\mathcal{E}|^2)$ transformation T_l to shrink its graph.*

$$\begin{aligned}\hat{U}[\phi](o) &= \hat{U}(\exists x.\mu(x, C_x) \wedge r(x, o) \wedge \mu(o, C_y) \wedge \psi(o/y)) \\ &= \max_{s \in \mathcal{E}}[C_x(s) + \hat{P}_r(s, o) + C_y(o) + \hat{U}(\psi(o/y))] \\ &= \max_{s \in \mathcal{E}}[C_x(s) + \hat{P}_r(s, o) + C_y(o)] + \hat{U}(\psi(o/y)) \\ &= C'_y(o) + \hat{U}(\psi(o/y)),\end{aligned}$$

where $C'_y(o) = \max_{s \in \mathcal{E}} M_1(s, o)$ and $M_1 = [(\hat{P}_r +_c C_y) +_r C_x]$.

Lemma 6. *If the leaf node u is a free variable y and v is the existential variable x, we can first solve the subgraph obtained by removing u, and then remove v.*

Proof. Denote ψ is the subgraph obtained by removing u, we treat y as the free variable to get its utility vector $\hat{U}(\psi(y))$. Then we can remove u by the following.

$$\begin{aligned}\hat{U}[\phi](o) &= \hat{U}(\mu(o, C_y) \wedge [\exists x.r(x, o) \wedge \psi(x)]) \\ &= C_y(o) + \hat{U}(\exists x.r(x, o) \wedge \psi(x)) \\ &= C_y(o) + \max_{s \in \mathcal{E}}[\hat{P}_r(s, o) + \hat{U}(\psi(o))] \\ &= C_y(o) + \max_{s \in \mathcal{E}} M_2(s, o), \qquad (19)\end{aligned}$$

where $M_2 = \hat{P}_r +_c \hat{U}[\psi]$. $\square$

Lemma 7. *If the leaf node u and its connected node v both are the existential variable, we can remove the leaf node u when the existential quantifier is maximization.*

Proof. It will be difficult when trying to cut the leaf node x_1,

$$\begin{aligned}\hat{U}[\phi](o) &= \hat{U}(\exists x_1, x_2.\mu(x_1, C_{x_1}) \wedge r(x_1, x_2) \wedge \mu(x_2, C_{x_2}) \wedge \psi(o)) \\ &= \max_{s_1\in\mathcal{E}, s_2\in\mathcal{E}}[C_{x_1}(s_1) + \hat{P}_r(s_1, s_2) + C_{x_2}(s_2) + \hat{U}(\psi(o, s_2/x_2))] \\ &= \max_{s_2\in\mathcal{E}}\max_{s_1\in\mathcal{E}}[C_{x_1}(s_1) + \hat{P}_r(s_1, s_2) + C_{x_2}(s_2) + \hat{U}(\psi(o, s_2/x_2))]. \end{aligned} \tag{20}$$

While the existential quantifier is maximization, it is noteworthy that for any $s_1, s_2, o \in \mathcal{E}$,

$$\begin{aligned}\max_{s_1\in\mathcal{E}}[M_3(s_1, s_2) + \hat{U}(\psi(o; s_2/x_2))] &= \max_{s_1\in\mathcal{E}}[M_3(s_1, s_2)] + \hat{U}(\psi(o; s_2/x_2)) \\ &= C'_{x_2}(s_2) + \hat{U}(\psi(o; s_2/x_2)), \end{aligned} \tag{21}$$

where $M_3 = (\hat{P}_r +_r C_{x_1}) +_c C_{x_2}$ and $C'_{x_2}(s_2) = \max_{s_1\in\mathcal{E}}[M_3(s_1, s_2)]$.

Therefore, we can remove x_1 by updating x_2 as follows,

$$\hat{U}[\phi](o) = \max_{s_2\in\mathcal{E}}[C'_{x_2}(s_2) + \hat{U}(\psi(o/y; s_2/x_2))]. \tag{22}$$

□

Combining the above three lemmas, we can step by step find a leaf node and remove it when the query graph has no cycles.

Lemma 8. *If the soft query contains no circles, we can get the utility vector by removing leaf nodes when the existential quantifier is maximization.*

H.2.5 Step 5. Enumerate on the cycle.

To the best of our knowledge, the only precise approach for addressing cyclic queries is performing enumeration over one existential node involved in the cycle, which reads $\hat{U}(\exists x.\phi(o/y; x)) = \max_{s\in\mathcal{E}} \hat{U}(\phi(o/y; s/x))$. Then, we apply Step 4 to remove this fixed existential variable since this variable is equivalent to the constant variable. The query graph breaks this cycle and becomes smaller. The remaining query can be solved by applying Step 4. When solving cyclic queries, the time complexity of this algorithm is exponential.

H.2.6 Step 6. Getting the utility vector.

Following the aforementioned steps, the query graph will only contain the free node y, resulting in the formula $\mu(y, C_y)$. By definition, the desired utility vector will be C_y, which provides the confidence values of all the candidate entities.

I Uncertain Knowledge Graph Embeddings

We introduce the backbone models for uncertain knowledge graph embedding. The results of changing the backbone are presented in Table 7.

UKGE (Chen, 2023) is a vector embedding model designed for uncertain knowledge graphs. It has been tested on three tasks: confidence prediction, relational fact ranking, and relational fact classification. To address the sparsity issue in the graph, UKGE utilizes probabilistic soft logic, allowing for the inclusion of additional unseen relational facts during training.

BEUrRE (Chen et al., 2021a) is a probabilistic box embedding model that has been evaluated on two tasks: confidence prediction and relational fact ranking. This model represents each entity as a box and captures relations between two entities through affine transformations applied to the head and tail entity boxes.

J Float Embedding for query Embedding

To enable query embedding methods to handle soft requirements in soft queries, we employ floating-point encoding to map floating-point numbers into vectors. These vectors are then added to the relation projection in the query embedding method.

Table 7: The results of answering complex soft queries with different backbone models.

Models	Metrics	1P	2P	2I	2IN	2IL	2M	2U	3IN	IP	IM	INP	UP	AVG
							CN15k							
SRC (UKGE)	MAP	20.6	7.1	7.9	14.4	14.6	3.3	14.7	7.7	6.2	3.2	4.2	6.5	9.2
	NDCG	27.7	15.4	10.1	23.8	22.6	7.9	24.3	10.4	11.1	6.9	10.5	14.9	15.5
	ρ	21.9	6.4	0.2	5.9	14.3	11.2	32.0	-1.5	-7.0	8.4	-2.3	17.4	8.9
	τ	15.0	2.4	-0.0	2.1	10.7	9.2	25.5	-2.0	-9.0	7.9	-4.4	13.0	5.9
SRC (BEUrRE)	MAP	32.2	11.5	13.2	25.9	29.2	3.9	26.6	13.9	12.8	5.3	8.4	11.6	16.2
	NDCG	41.5	21.3	15.3	37.4	39.7	9.7	40.5	17.6	19.0	10.4	17.1	22.8	24.3
	ρ	25.7	13.9	-2.3	11.6	19.0	17.4	33.5	-0.3	-0.4	21.3	5.2	21.5	13.8
	τ	18.7	8.7	-3.5	7.7	14.8	14.7	27.2	-1.9	-3.6	20.3	1.8	16.3	10.1
							PPI5k							
SRC (BEUrRE)	MAP	78.2	78.4	72.9	67.0	72.0	52.0	73.7	58.6	76.4	54.2	69.2	67.8	68.4
	NDCG	80.8	78.2	77.3	68.7	78.0	52.3	79.9	62.9	76.2	52.4	69.4	72.4	70.7
	ρ	77.7	82.3	57.9	50.9	80.6	54.3	83.0	38.8	76.2	47.6	68.3	82.2	66.7
	τ	66.6	70.9	49.9	42.7	71.0	42.9	71.6	32.7	65.6	37.1	57.2	70.4	56.5
SRC (BEUrRE)	MAP	71.0	67.8	63.3	56.3	63.6	51.8	66.3	49.4	67.7	52.6	57.7	58.8	60.5
	NDCG	74.8	70.5	68.7	61.5	70.0	49.7	74.2	55.5	68.5	49.1	61.8	66.5	64.2
	ρ	72.7	76.4	51.6	47.1	73.9	51.3	77.3	39.0	68.5	43.2	63.2	75.0	61.6
	τ	59.7	62.3	41.8	37.7	61.9	39.4	64.2	32.0	55.8	32.6	50.6	61.5	50.0
							O*NET20k							
SRC (BEUrRE)	MAP	24.9	6.4	70.6	26.0	63.3	5.6	32.8	68.5	7.7	7.3	5.2	7.7	27.1
	NDCG	44.9	19.5	80.4	46.1	74.0	17.2	57.2	76.3	19.2	17.1	17.7	24.6	41.3
	ρ	77.9	64.8	79.1	73.7	79.6	43.7	83.3	69.4	53.8	33.3	58.0	60.7	65.3
	τ	72.0	54.9	68.6	67.6	67.3	36.9	76.0	59.2	47.6	29.1	48.9	52.4	57.3
SRC (BEUrRE)	MAP	29.0	8.3	65.2	29.6	55.2	7.2	32.3	62.1	9.9	8.8	6.9	9.6	27.1
	NDCG	53.4	29.2	78.8	52.4	71.0	21.5	60.2	72.7	26.5	20.4	25.5	34.1	45.8
	ρ	69.0	59.1	78.1	63.6	76.4	40.1	78.3	66.8	47.6	30.4	51.2	57.2	60.2
	τ	58.4	47.1	66.6	53.6	62.8	32.0	67.4	55.9	39.2	24.9	40.5	46.8	49.9

J.1 Float embedding

We consider the sinusoidal encoding $g : \mathbf{R} \to R^d$ introduced in Transformer (Vaswani et al., 2017) and map the values of α and β into vector embedding, which can be formulated as:

$$g(v_i) = \begin{cases} sin(v_i/1000^{i/(2k)}) & i = 2k, \\ cos(v_i/1000^{i/(2k)}) & i = 2k+1, \end{cases} \tag{23}$$

where d is the embedding dimension.

J.2 Modified relation projection

The query embedding methods usually learn a Multi-Layer Perceptron (MLP) for each relation r, which reads as:

$$S' = \mathbf{MLP}_r(S), \tag{24}$$

where S is an embedding. Furthermore, the modified relation projection can be expressed as:

$$S' = \mathbf{MLP}_r(S + g(\alpha) + g(\beta)) \tag{25}$$

Here, $g(\alpha)$ and $g(\beta)$ are the embedding of soft requirements α and β, respectively. By incorporating $g(\alpha)$ and $g(\beta)$ into the relation project net, we enhance the representation of relation projection to better capture the soft requirement.

K Details in the Main Dataset Construction

K.1 Uncertain knowledge graphs

We sample soft queries from three standard uncertain knowledge graphs[5], covering diverse domains such as common sense knowledge, bioinformatics, and the employment domain.

CN15k (Chen et al., 2019) is a subset of the ConceptNet (Speer et al., 2017), a semantic network aimed at comprehending connections between words and phrases.

[5]We leave the exploration of Nl27k (Chen et al., 2019) for future work as it involves an inductive setting where the valid/test graphs contain unseen relations.

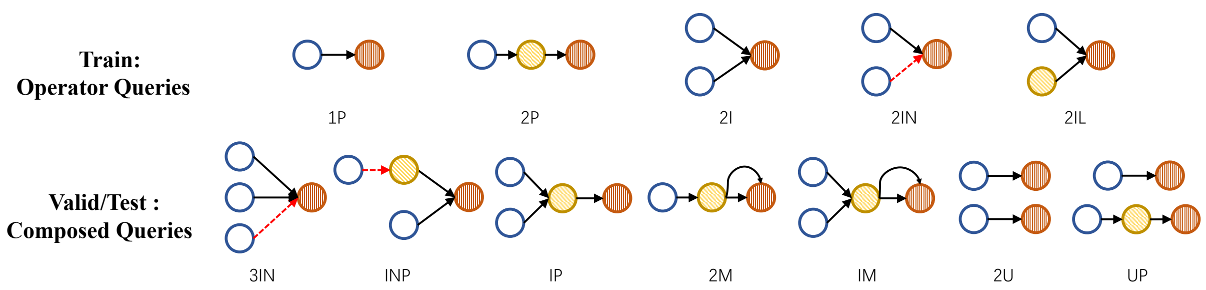

Figure 4: Query structures of query types. The white, yellow, and red circles represent constant, existential, and free nodes, respectively. The negative atomic formulas are represented by red edges, while atomic formulas are represented by black edges. Like the previous naming convention (Ren and Leskovec, 2020; Yin et al., 2024), we use ”P” for projection, “I” for intersection, “N” for negation, “M” for multi-edge, and “L” for existential leaf.

Table 8: The statistics of valid/test queries on the main dataset. Different query types have the same number in given uncertain knowledge graph.

KG	CN15k	PPI5k	O*NET20k
valid	3,000	2,000	2,000
test	3,000	2,000	2,000

PPI5k (Chen et al., 2019) is a subset of STRING (Szklarczyk et al., 2023), which illustrates protein-protein association networks collected from organisms. It assigns probabilities to the intersections among proteins.

O*NET20K (Pai and Costabello, 2021) is a subset of O*NET, a dataset that describes labeled binary relations between job descriptions and skills. The associated values are to evaluate the importance of the link within the triple.

We present the statistics of these three uncertain knowledge graphs in Table **??**.

K.2 Soft requirements

In the main experiment, we aim for the sampled data to allow the machine learning methods to generalize to various scenarios of soft requirements. Therefore, for α necessity, we employ a hybrid strategy, randomly selecting from four modes (“zero”, “low”, “normal”, “high”) for each query. As for β importance, we utilize a random strategy, randomly choosing a decimal value for different atomic soft formulas in each query.

K.3 Query types

The goal of our proposed dataset is to represent the family of existential first-order soft queries systematically. However, including too many query formulas poses challenges in analyzing and evaluating. We select query graphs including operations as train queries and unitize composed query graphs to evaluate the combinatorial generalization.

For training queries, we choose 1P, 2P, 2I, and 2IN, which include soft operators. Additionally, we select 2P for chain queries (Hamilton et al., 2018), 2M for multi-edge graphs (Yin et al., 2024), and 2IL for graphs containing ungrounded anchors (Yin et al., 2024). More complex soft query graphs can be generated from these basic graphs. We present the statistics of sampled queries in Table 8 and Tabel 9.

Table 9: The statistics of train queries on the main dataset.

KG	1P	2P	2I	2IN	2IL
CN15k	52887	52900	52900	5300	5300
PPI5k	9724	9750	9754	1500	1500
PPI5k	18266	18300	18300	1850	1850

K.4 Evaluation protocol

The open world assumption in uncertain knowledge graphs not only establishes new links between entities but can also potentially refine the values of existing triples. As more observed facts become available, the answers to soft queries not only increase in number but also undergo modifications in terms of their priority. Therefore, to evaluate the relevance judgment, we select several popular metrics commonly used in information retrieval (Liu et al., 2009), including MAP, DCG, NDCG, and Kendall's tau.

For each q, we denote the set of answers as $\mathcal{A}$, where $a_i \in \mathcal{A}$ represents the i-th answer based on its score, and $r(a), a \in \mathcal{A}$ denotes the predicted ranking of answer $a \in \mathcal{A}$. Our objective is to focus on the precision of answers and the associated predicted ranking information.

Mean Average Precision (MAP): To define MAP, we first introduce Precision at a given position, defined as:

$$\mathbf{P@k}(q) = |\{a \in \mathcal{A} | r(a) \geq k\}|/k. \tag{26}$$

Then, Average Precision is defined as follows:

$$\mathbf{AP}(q) = (\sum_{k=1}^{|\mathcal{A}|} \mathbf{P@k}(q) \cdot l_k)/|\mathcal{A}|, \tag{27}$$

where l_k is a binary judgment indicating the relevance of the answer at the kth position. Mean Average Precision is the average AP value across all test queries.

Discounted Cumulative Gain (DCG): To calculate the DCG, we utilize the Reciprocal Rank as a relative score for the answers:

$$R(a_i) = 1/r(a_i). \tag{28}$$

To incorporate the ranking position, we introduce an explicit position discount factor η_i. The DCG is then computed as:

$$\mathbf{DCG@k}(q) = \sum_{i=1}^{k} R(a_i)\eta(i), \tag{29}$$

where $\eta(i)$ is commonly expressed as $\eta(i) = 1/\log_2(i+1)$.

Normalized Discounted Cumulative Gain (NDCG): By normalizing the ideal Discounted Cumulative Gain denoted as Z_k, we obtain NDCG:

$$\mathbf{NDCG@k}(q) = \mathbf{DCG@k}(q)/Z_k. \tag{30}$$

Kendall's tau: Kendall's tau is a statistical measure that quantifies the correspondence between two rankings. Values close to 1 indicate strong agreement, while values close to -1 indicate strong disagreement.

L Details in Varying Soft Requirements Setting

In this setting, we aim to test the model's generalization ability on soft requirements under different strategies. We have chosen a pair of different strategies for each of the two parameters, resulting in a total of four groups. Specifically, we selected "zero" and "normal" for parameter "a," and "equal" and "normal" for parameter "b." Detailed statistics for this setting can be found in the table below. For each group, we follow the procedure and sample train, valid, and test queries. We present the additional results in Table 10.

M Details of Large Language Model Evaluation Setting

Since the entities and relations in CN15k are English words and phrases, the queries sampled from CN15k can be well understood by LLM. In our evaluation, we manually marked and removed meaningless queries. To facilitate our testing process, we selected only four candidate entities for each query. These four candidate entities have large distinctions in terms of scores. To avoid unexpected situations, we manually checked all chosen queries and confirmed that their correct answers could be selected without ambiguity. We present the total numbers and types of selected queries in Table 11.

Table 10: The additional results of varying soft requirements

Mode	Z+E		Z+R		N+E		N+R	
Metric	NDCG	τ	NDCG	τ	NDCG	τ	NDCG	τ
ConE Z+E	39.3	8.0	35.6	11.6	31.9	30.4	31.5	28.9
ConE Z+R	39.0	6.7	42.2	14.9	27.8	29.0	28.7	30.3
ConE N+E	26.6	32.8	23.6	27.4	46.3	38.7	44.1	36.7
ConE N+R	14.0	24.7	7.5	14.4	43.5	38.7	42.2	38.6
SRC	34.3	47.8	37.2	41.5	46.7	51.8	45.8	49.9

Table 11: The number of annotated queries.

Query type	1P	2P	2I	2IN	2IL	2M	3IN	IP	IM	SUM
Number	100	50	100	50	15	10	15	15	15	370

We articulate the syntax and semantics of our proposed soft queries by using clear natural language. The provided prompts are presented in the subsequent subsection. Through combining queries with prompts, we enable LLM to select the most appropriate answer.

M.1 Prompt

```
# Background
Given a soft query containing a free variable f, there are four candidate entities for
this variable and you need to choose the best of them to satisfy the soft query most. In
order to do so, you can first compute the confidence value to see whether the chosen
candidate entity satisfies the soft query after substituting the free variable with a
given candidate entity. After computing the confidence values for each corresponding
candidate entity, you need to pick the entity that leads to the highest confidence value.
The detailed steps are described below:

## Definition of Soft Atomic Constraint:
A soft atomic constraint c, a.k.a. a soft atomic formula or its negation, is in the form
of (h, r, t, \alpha, \beta) or \\neg (h, r, t, \alpha, \beta).

### Notation Description:
In each constraint, we have four different types of variables.
1. r is a relation;
2. h and t are two terms. Each term represents an entity or a variable whose values
are entities. And free variable is a term.
3. \alpha is called the necessity value, which represents the minimal requirement
of the uncertainty degree of this constraint. It can be any decimal between 0.0 and
1.0. If the confidence value is less than the necessity value, the constraint is
not satisfied, and thus the final confidence value becomes negative infinity;
4. \beta represents the priority of this constraint and can be any decimal
between 0.0 and 1.0.

### Confidence Value of Soft Constraints V(c):
1. The triple (h,r,t) comes from the relation fact in the knowledge graph. In our
```

```
setting, the relation fact r(h,t) is not boolean, and it has a confidence value
in the range [0.0,1.0] according to its plausibility. When the constraint is
negative, you need to first estimate r(h,t)—the confidence value of r(h,t)—and
use 1-r(h,t) as the final confidence value for this negative constraint.
2. \alpha is the threshold in our filter function, working as follows:
f(v, \alpha) = v,           if  v \geq \alpha,
                              -\infty,    if  v < \alpha.
3. \beta is a coefficient.

Thus, the final equation becomes:
V(h, r, t, \alpha, \beta) = \beta \times f(r(h,t), \alpha),
V(\\neg (h, r, t, \alpha, \beta)) = \beta \times f(1-r(h,t), \alpha).

## Definition of Conjunctive Queries \phi:
Soft conjunctive queries are composed of soft constraints.

### Notation Description:
Conjunctive query \phi = c_1 \land \dots \land c_n, where c_i is a soft constraint.

### Confidence Value of Soft Conjunctive Queries V(\phi) :
First, you need to compute the confidence values of all soft constraints in
this soft conjunctive query. Then, you can simply sum up these confidence
values as the final confidence value of the soft conjunctive query as follows:
V(\phi) = \sum_i V(c_i).

If the conjunctive query has an existential variable e, you should find an
entity to replace it and then compute the confidence value of this query.

# Output Format
Please output your response in the JSON format, where the first element
is the best candidate entity among the four options, and the second element
is your explanation for your choice.

# Question
Soft query:(h_1,r_1,f,\alpha_1,\beta_1) \land (h_2,r_2,f,\alpha_2,\beta_2)
Four candidate entities: s_1, s_2, s_3, s_4

Please return the best candidate for f1 to satisfy the above soft query.
```

N Proof for Error Analysis

N.1 Proof of Theorem 1

Firstly, we give the proof of Theorem 1:

Proof. Consider the atomic query $\psi = (a, (\alpha, \beta))$. Firstly, we consider whether the soft atomic query is positive or negative.

Let us assume $a = r(y, o)$, with y be the only free variable

Then for arbitrary r, h, t, α, β:

$$\Pr\left(\|\hat{U}[\psi](s) - \mathcal{U}[\psi](s)\| > \delta)\right) = \Pr\left(\beta\|[\hat{P}(s,r,o)]_\alpha - [\mathcal{P}(s,r,o)]_\alpha\| > \delta\right) \tag{31}$$

$$= \Pr\left(\|[\hat{P}(s,r,o)]_\alpha - [\mathcal{P}(s,r,o)]_\alpha\| > \frac{\delta}{\beta}\right) \tag{32}$$

We note that even if $a = \neg r(y, o)$, the result is the same:

$$\Pr\left(\|\hat{U}[\psi](s) - \mathcal{U}[\psi](s)\| > \delta)\right) = \Pr\left(\beta\|[1 - \hat{P}(s,r,o)]_\alpha + [\mathcal{P}(s,r,o)]_\alpha - 1\| > \delta\right)$$
$$= \Pr\left(\|[\hat{P}(s,r,o)]_\alpha - [\mathcal{P}(s,r,o)]_\alpha\| > \frac{\delta}{\beta}\right)$$

For convenience, we write $\hat{x}, x$ as the abbreviation of $\hat{P}(s,r,o), \mathcal{P}(s,r,o)$, correspondingly, then the initial formula becomes:

$$\begin{aligned}
&\Pr\left(\|\hat{x} - x\| > \frac{\delta}{\beta}\right)\\
&= \Pr\left(\|\hat{x} - x\| > \frac{\delta}{\beta} \mid \hat{x} > \alpha, x > \alpha\right)\Pr(\hat{x}, x > \alpha) + \Pr\left(\|\hat{x}\| > \frac{\delta}{\beta} \mid \hat{x} > \alpha, x < \alpha\right)\Pr\left(\hat{x} > \alpha, x < \alpha\right)\\
&+ \Pr\left(\|x\| > \frac{\delta}{\beta} \mid \hat{x} < \alpha, x > \alpha\right)\Pr(\hat{x} < \alpha, x > \alpha) + \Pr\left(\|0\| > \frac{\delta}{\beta} \mid \hat{x} < \alpha, x < \alpha\right)\Pr(\hat{x} < \alpha, x < \alpha)\\
&\le \varepsilon(\frac{\delta}{\beta})\Pr(\hat{x} > \alpha, x > \alpha) + \Pr(\hat{x} > \alpha, x < \alpha) + \Pr(\hat{x} < \alpha, x > \alpha)\\
&= \varepsilon(\frac{\delta}{\beta}) + (1 - \varepsilon(\frac{\delta}{\beta}))[\Pr(\hat{x} > \alpha, x < \alpha) + \Pr(\hat{x} < \alpha, x > \alpha)] - \varepsilon(\frac{\delta}{\beta})\Pr(\hat{x} < \alpha, x < \alpha)\\
&\le \varepsilon(\frac{\delta}{\beta}) + (1 - \varepsilon(\frac{\delta}{\beta}))[\Pr(\hat{x} > \alpha, x < \alpha) + \Pr(\hat{x} < \alpha, x > \alpha)]
\end{aligned}$$

Moreover, we use the Total Probability Theorem once again and assume $f(\xi)$ as the probability density function of x:

$$\begin{aligned}
&\Pr(\hat{x} > \alpha, x < \alpha) + \Pr(\hat{x} < \alpha, x > \alpha)\\
&= \int_0^1 \Pr(\hat{x} > \alpha \mid x = \xi < \alpha) f(\xi)\mathrm{d}\xi + \int_0^1 \Pr(\hat{x} < \alpha \mid x = \xi > \alpha) f(\xi)\mathrm{d}\xi\\
&= \int_0^\alpha \Pr(\hat{x} > \alpha \mid x = \xi) f(\xi)\mathrm{d}\xi + \int_\alpha^1 \Pr(\hat{x} < \alpha \mid x = \xi) f(\xi)\mathrm{d}\xi
\end{aligned}$$

By noting that if $\hat{x} > \alpha$ while $x = \xi < \alpha$, it must have $|\hat{x} - x| > \alpha - \xi$, we know that:

$$\Pr(\hat{x} < \alpha \mid x = \xi > \alpha) \le \Pr(|\hat{x} - x| > \alpha - \xi)$$

Therefore

$$\begin{aligned}
&\int_0^\alpha \Pr(\hat{x} > \alpha \mid x = \xi) f(\xi)\mathrm{d}\xi + \int_\alpha^1 \Pr(\hat{x} < \alpha \mid x = \xi) f(\xi)\mathrm{d}\xi\\
&\le \int_0^\alpha \Pr(|\hat{x} - x| > \alpha - \xi) f(\xi)\mathrm{d}\xi + \int_\alpha^1 \Pr(|\hat{x} - x| > \xi - \alpha) f(\xi)\mathrm{d}\xi\\
&\le \int_0^\alpha \varepsilon(\alpha - \xi) f(\xi)\mathrm{d}\xi + \int_\alpha^1 \varepsilon(\xi - \alpha) f(\xi)\mathrm{d}\xi\\
&= \int_0^1 \varepsilon(|\alpha - \xi|) f(\xi)\mathrm{d}\xi
\end{aligned}$$

Therefore we finish the proof.

□

N.2 Proof of Theorem 2

Then for Theorem 2, consider the query $\phi = \exists x_1, ..., x_n.\psi_1 \owedge \cdots \owedge \psi_m$, where $\psi_i = (a_i, (\alpha_i, \beta_i))$, the final error should be no more than a linear combination of each soft atomic query:

Proof.

$$
\begin{aligned}
&\|\hat{U}[\phi](s) - \mathcal{U}[\phi](s)\| \\
&= \| \max_{x_1=s_1,\cdots,x_n=s_n} (\hat{U}[\psi_1](s) + \cdots + \hat{U}[\psi_m](s)) - \max_{x_1=s_1,\cdots,x_n=s_n} (\mathcal{U}[\psi_1](s) + \cdots + \mathcal{U}[\psi_m](s)) \| \\
&\leq \| \max_{x_1=s_1,\cdots,x_n=s_n} \Big([\hat{U}[\psi_1](s) - \mathcal{U}[\psi_1](s)] + \cdots + [\hat{U}[\psi_m](s) - \mathcal{U}[\psi_m](s)] \Big) \| \\
&\leq \max_{x_1=s_1,\cdots,x_n=s_n} \Big(\|\hat{U}[\psi_1](s) - \mathcal{U}[\psi_1](s)\| + \cdots + \|\hat{U}[\psi_m](s) - \mathcal{U}[\psi_m](s)\| \Big) \\
&\leq \Sigma_{i=1}^{m} \epsilon(\alpha_i, \beta_i)
\end{aligned}
$$

The final line relies on the definition of ϵ, which gives the upper bound of error that only depends on α, β. □