A Foundation Model for Zero-shot Logical Query Reasoning

Mikhail Galkin[1], **Jincheng Zhou**[2,*], **Bruno Ribeiro**[2], **Jian Tang**[3,4,5], **Zhaocheng Zhu**[3,6]
[1]Intel AI Lab, [2]Purdue University, [3]Mila - Québec AI Institute,
[4]HEC Montréal, [5]CIFAR AI Chair [6]Université de Montréal

Abstract

Complex logical query answering (CLQA) in knowledge graphs (KGs) goes beyond simple KG completion and aims at answering compositional queries comprised of multiple projections and logical operations. Existing CLQA methods that learn parameters bound to certain entity or relation vocabularies can only be applied to the graph they are trained on which requires substantial training time before being deployed on a new graph. Here we present ULTRAQUERY, the first foundation model for inductive reasoning that can zero-shot answer logical queries on *any* KG. The core idea of ULTRAQUERY is to derive both projections and logical operations as vocabulary-independent functions which generalize to new entities and relations in any KG. With the projection operation initialized from a pre-trained inductive KG completion model, ULTRAQUERY can solve CLQA on any KG after finetuning on a single dataset. Experimenting on 23 datasets, ULTRAQUERY in the zero-shot inference mode shows competitive or better query answering performance than best available baselines and sets a new state of the art on 15 of them.

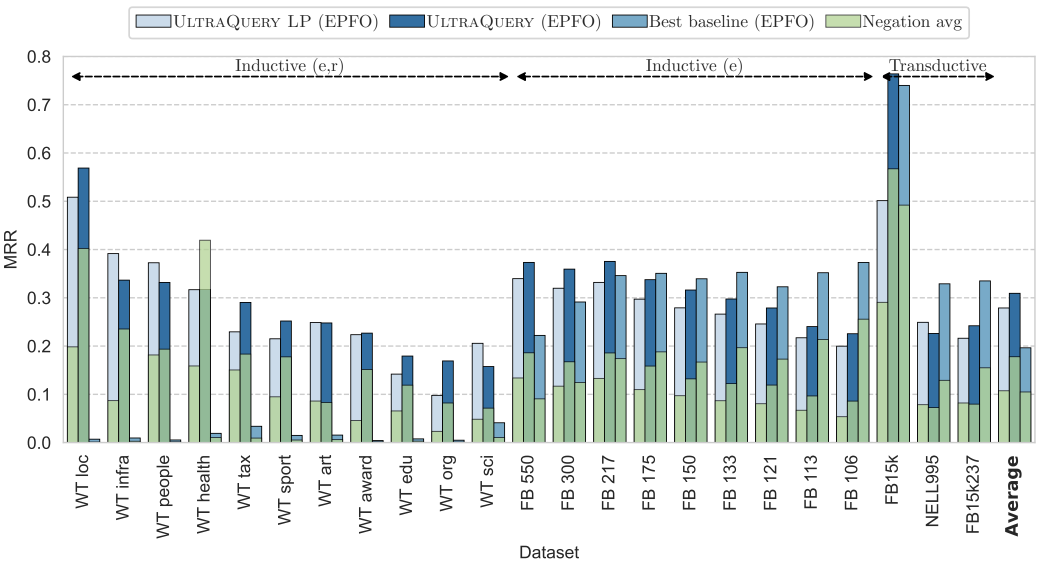


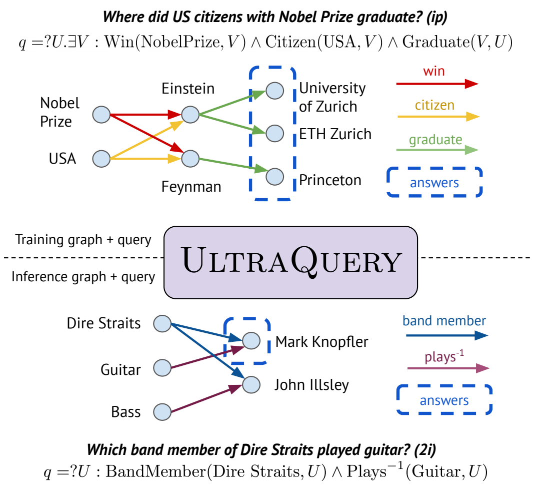
Figure 1: Zero-shot query answering performance (MRR, higher is better) of a single ULTRAQUERY model trained on one FB15k237 queries dataset compared to the best available baselines and ablated ULTRAQUERY LP on 23 datasets. *EPFO* is the average of 9 query types with $(\wedge, \vee)$ operators, *Negation* is the average of 5 query types with the negation operator $(\neg)$. On average, a single ULTRAQUERY model outperforms the best baselines trained specifically on each dataset. More results are presented in Table 2 and Appendix C.

[*]Work done during the internship at Intel. Code: https://github.com/DeepGraphLearning/ULTRA

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

1 Introduction

Complex logical query answering (CLQA) generalizes simple knowledge graph (KG) completion to more complex, compositional queries with logical operators such as intersection ($\wedge$), union ($\vee$), and negation ($\neg$). Such queries are expressed in a subset of first-order logic (FOL) where existentially quantified ($\exists$) *variables* and given *constants* comprise *relation projections* (or *atoms*), and logical operators combine projections into a logical query (graph pattern). A typical example of a logical query [27] is presented in Figure 2: $?U.\exists V : \texttt{Win}(\texttt{NobelPrize}, V) \wedge \texttt{Citizen}(\texttt{USA}, V) \wedge \texttt{Graduate}(V, U)$ where $\texttt{Win}()$ is a relation projection, $\texttt{NobelPrize}$ is a constant, and V is an existentially quantified variable.



Where did US citizens with Nobel Prize graduate? (ip)
$q = ?U.\exists V : \text{Win}(\text{NobelPrize}, V) \wedge \text{Citizen}(\text{USA}, V) \wedge \text{Graduate}(V, U)$
Nobel Prize
USA
Einstein
Feynman
University of Zurich
ETH Zurich
Princeton
win
citizen
graduate
answers
Training graph + query
Inference graph + query
UltraQuery
Dire Straits
Guitar
Bass
Mark Knopfler
John Illsley
band member
plays^{-1}
answers
Which band member of Dire Straits played guitar? (2i)
$q = ?U : \text{BandMember}(\text{Dire Straits}, U) \wedge \text{Plays}^{-1}(\text{Guitar}, U)$

Figure 2: The inductive logical query answering setup where training and inference graphs (and queries) have different entity and relation vocabularies. We propose a single model (ULTRAQUERY) that zero-shot generalizes to query answering on any graph with new entity or relation vocabulary at inference time.

Due to the incompleteness of most KGs, these logical queries cannot be directly solved by graph traversal algorithms. Consequently, CLQA methods have to deal with missing edges when modeling the projection operators. The vast majority of existing CLQA methods [27, 26, 25, 3, 5] predict missing edges by learning graph-specific entity and relation embeddings making such approaches transductive and not generalizable to other KGs. A few approaches [43, 14, 17] are able to generalize query answering to new nodes at inference time but still need a fixed relation vocabulary.

In this work, we focus on the hardest inductive generalization setup where queries and underlying graphs at inference time are completely different from the training graph, *i.e.*, both entities and relations are new. Furthermore, we aim at performing CLQA in the *zero-shot* setting with one single model. That is, instead of finetuning a model on each target dataset, we seek to design a unified approach that generalizes to any KG and query at inference time. For example, in Figure 2, the training graph describes academic entities with relations `Win`, `Citizen`, `Graduate`[2] whereas the inference graph describes music entities with relations `Band Member` and `Plays`. The query against the inference graph $?U : \texttt{BandMember}(\texttt{Dire Straits}, U) \wedge \texttt{Plays}^{-1}(\texttt{Guitar}, U)$ involves both new entities and relations and, to the best of our knowledge, cannot be tackled by any existing CLQA method that learns a fixed set of entities or relation embeddings from the training graph.

Contributions. Our contributions are two-fold. First, none of the existing CLQA methods can generalize to query answering over new arbitrary KGs with new entities and relations at inference time. We bridge this gap by leveraging the recent progress in inductive KG reasoning [15, 16] and devise ULTRAQUERY, the first foundation model for CLQA that generalizes to logical queries on any arbitrary KG with any entity and relation vocabulary in the zero-shot fashion without relying on any external node or edge features. ULTRAQUERY parameterizes the projection operator by an inductive graph neural network (GNN) and implements non-parametric logical operators with fuzzy logics [31]. The pre-trained projection operator [15] does not learn any graph-specific entity nor relation embeddings thanks to the generalizable meta-graph representation of relation interactions, and therefore enables zero-shot generalization to any KG.

Second, in the absence of existing datasets for our inductive generalization setup, we curate a novel suite of 11 inductive query answering datasets where graphs and queries at inference time have new entity and relation vocabularies. Experimentally, we train a single ULTRAQUERY model on one dataset and probe on other 22 transductive and inductive datasets. Averaged across the datasets, a single ULTRAQUERY model outperforms by 50% (relative MRR) the best reported baselines in the literature (often tailored to specific graphs) on both EPFO queries and queries with negation.

[2]We assume the presence of respective inverse relations $\texttt{r}^{-1}$.

2 Related Work

Complex Logical Query Answering. To the best of our knowledge, there is no existing approach for generalizable and inductive query answering where the method is required to deal with new entitis and relations at inference time.

Due to the necessity of learning entity and relation embeddings, the vast majority of existing methods like GQE [19], BetaE [25], ConE [40], MPQE [10] (and many more from the survey by Ren et al. [27]) are transductive-only and tailored for a specific set of entities and relations. Among them, CQD [3] and QTO [5] are inference-only query answering engines that execute logical operators with non-parametric fuzzy logic operators (*e.g.*, product logic) but still require pre-trained entity and relation embedding matrices to execute relation projections (link prediction steps). We refer the interested reader to the comprehensive survey by Ren et al. [27] that covers query answering theory, a taxonomy of approaches, datasets, and open challenges.

A few models [43, 14, 17] generalize only to new entities by modeling entities as a function of relation embeddings. Gebhart et al. [17] apply the idea of cellular sheaves and harmonic extension to translation-based embedding models to answer conjunctive queries (without unions and negations). NodePiece-QE [14] trains an inductive entity encoder (based on the fixed vocabulary of relations) that is able to reconstruct entity embeddings of the new graph and then apply non-parametric engines like CQD to answer queries against new entities. The most effective inductive (entity) approach is GNN-QE [43, 14] that parameterizes each entity as a function of the relational structure between the query constants and the entity itself. However, all these works rely on a fixed relation vocabulary and cannot generalize to KGs with new relations at test time. In contrast, our model uses inductive relation projection and inductive logical operations that enable zero-shot generalization to any new KG with any entity and relation vocabulary without any specific training.

Table 1: Comparison with existing CLQA approaches. *Ind.* denotes inductive generalization to new entities *(e)* and relations *(r)*. ULTRAQUERY is the first inductive method the generalizes to queries over new entities and relations at inference time.

Method	Ind. e	Ind. r	Ind. Logical Ops
Query2Box [26], BetaE [25]	✗	✗	Parametric, ✗
CQD [3], FuzzQE [9], QTO [5]	✗	✗	Fuzzy, ✓
GNN-QE [43], NodePiece-QE [14]	✓	✗	Fuzzy, ✓
ULTRAQUERY **(this work)**	✓	✓	Fuzzy, ✓

Inductive Knowledge Graph Completion. In CLQA, KG completion methods execute the projection operator and are mainly responsible for predicting missing links in incomplete graphs during query execution. Inductive KG completion is usually categorized [8] into two branches: (i) inductive entity (inductive (e)) approaches have a fixed set of relations and only generalize to new entities, for example, to different subgraphs of one larger KG with one set of relations; and (ii) inductive entity and relation (inductive (e, r)) approaches that do not rely on any fixed set of entities and relations and generalize to any new KG with arbitrary new sets of entities and relations.

Up until recently, the majority of existing approaches belonged to the inductive (e) family (*e.g.*, GraIL [29], NBFNet [42], RED-GNN [38], NodePiece [13], A*Net [44], AdaProp [39]) that generalizes only to new entities as their featurization strategies are based on learnable relation embeddings.

Recently, the more generalizable inductive (e, r) family started getting more attention, *e.g.*, with RMPI [18], InGram [22], ULTRA [15], and the theory of *double equivariance* introduced by Gao et al. [16] followed by ISDEA and MTDEA [41] models. In this work, we employ ULTRA to obtain transferable graph representations and execute the projection operator with link prediction over any arbitrary KG without input features. Extending our model with additional input features is possible (although deriving a single fixed-width model for graphs with arbitrary input space is highly non-trivial) and we leave it for future work.

3 Preliminaries and Problem Definition

We introduce the basic concepts pertaining to logical query answering and KGs largely following the existing literature [14, 27, 15].

Knowledge Graphs and Inductive Setup. Given a finite set of entities $\mathcal{V}$ (nodes), a finite set of relations $\mathcal{R}$ (edge types), and a set of triples (edges) $\mathcal{E} = (\mathcal{V} \times \mathcal{R} \times \mathcal{V})$, a knowledge graph $\mathcal{G}$ is a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$. The simplest *transductive* setup dictates that the graph at training time

$\mathcal{G}_{train} = (\mathcal{V}_{train}, \mathcal{R}_{train}, \mathcal{E}_{train})$ and the graph at inference (validation or test) time $\mathcal{G}_{inf} = (\mathcal{V}_{inf}, \mathcal{R}_{inf}, \mathcal{E}_{inf})$ are the same, *i.e.*, $\mathcal{G}_{train} = \mathcal{G}_{inf}$. By default, we assume that the inference graph $\mathcal{G}_{inf}$ is an incomplete part of a larger, non observable graph $\hat{\mathcal{G}_{inf}}$ with missing triples to be predicted at inference time. In the *inductive* setup, in the general case, the training and inference graphs are different, $\mathcal{G}_{train} \neq \mathcal{G}_{inf}$. In the easier inductive entity (*inductive* (e)) setup tackled by most of the KG completion literature, the relation set $\mathcal{R}$ is fixed and shared between training and inference graphs, *i.e.*, $\mathcal{G}_{train} = (\mathcal{V}_{train}, \mathcal{R}, \mathcal{E}_{train})$ and $\mathcal{G}_{inf} = (\mathcal{V}_{inf}, \mathcal{R}, \mathcal{E}_{inf})$. The inference graph can be an extension of the training graph if $\mathcal{V}_{train} \subseteq \mathcal{V}_{inf}$ or be a separate disjoint graph (with the same set of relations) if $\mathcal{V}_{train} \cap \mathcal{V}_{inf} = \varnothing$. In CLQA, the former setup with the extended training graph at inference is tackled by InductiveQE approaches [14].

In the hardest inductive entity and relation (inductive (e, r)) case, both entities and relations sets are different, *i.e.*, $\mathcal{V}_{train} \cap \mathcal{V}_{inf} = \varnothing$ and $\mathcal{R}_{train} \cap \mathcal{R}_{inf} = \varnothing$. In CLQA, there is no existing approach tackling this case and our proposed ULTRAQUERY is the first one to do so.

First-Order Logic Queries. Applied to KGs, a first-order logic (FOL) query q is a formula that consists of constants $\texttt{Con}$ ($\texttt{Con} \subseteq \mathcal{V}$), variables $\texttt{Var}$ ($\texttt{Var} \subseteq \mathcal{V}$, existentially quantified), relation *projections* $R(a, b)$ denoting a binary function over constants or variables, and logic symbols ($\exists, \wedge, \vee, \neg$). The answers $A_{\mathcal{G}}(q)$ to the query q are assignments of variables in a formula such that the instantiated query formula is a subgraph of the complete, non observable graph $\hat{\mathcal{G}}$. Answers are denoted as *easy* if they are reachable by graph traversal over the incomplete graph $\mathcal{G}$ and denoted as *hard* if at least one edge from the non observable, complete graph $\hat{\mathcal{G}}$ has to be predicted during query execution.

For example, in Figure 2, a query *Which band member of Dire Straits played guitar?* is expressed in the logical form as $?U : \texttt{BandMember}(\texttt{Dire Straits}, U) \wedge \texttt{Plays}^{-1}(\texttt{Guitar}, U)$ as an *intersection* query. Here, U is a projected target variable, `Dire Straits` and `Guitar` are constants, `BandMember` and `Plays` are *relation projections* where $\texttt{Plays}^{-1}$ denotes the inverse of the relation `Plays`. The task of CLQA is to predict bindings (mappings between entities and variables) of the target variable, *e.g.*, for the example query the answer set is a single entity $\mathcal{A}_q = \{(U, \texttt{Mark Knopfler})\}$ and we treat this answer as an *easy* answer as it is reachable by traversing the edges of the given graph. In practice, however, we measure the performance of CLQA approaches on *hard* answers.

Inductive Query Answering. In the transductive CLQA setup, the training and inference graphs are the same and share the same set of entities and relations, *i.e.*, $\mathcal{G}_{train} = \mathcal{G}_{inf}$ meaning that inference queries operate on the same graph, the same set of constants `Con` and relations. This allows query answering models to learn hardcoded entity and relation embeddings at the same time losing the capabilities to generalize to new graphs at test time.

In the inductive entity (e) setup considered in Galkin et al. [14], the inference graph extends the training graph $\mathcal{G}_{train} \subset \mathcal{G}_{inf}$ but the set of relations is still fixed. Therefore, the proposed models are still bound to a certain hardcoded set of relations and cannot generalize to any arbitrary KG.

In this work, we lift all the restrictions on the training and inference graphs' vocabularies and consider the most general, inductive (e, r) case when $\mathcal{G}_{inf} \neq \mathcal{G}_{train}$ and the inference graph might contain a completely different set of entities and relation types. Furthermore, missing links still have to be predicted in the inference graphs to reach *hard* answers.

Labeling Trick GNNs. Labeling tricks (as coined by Zhang et al. [37]) are featurization strategies in graphs for breaking symmetries in node representations which are particularly pronounced in link prediction and KG completion tasks. In the presence of such node symmetries (*automorphisms*), classical uni- and multi-relational GNN encoders [21, 33, 32] assign different *automorphic* nodes the same feature making them indistinguishable for downstream tasks. In multi-relational graphs, NBFNet [42] and A*Net [44] apply a labeling trick by using the indicator function $\textsc{Indicator}(h, v, r)$ that puts a query vector $\boldsymbol{r}$ on a head node h and puts the zeros vector on other nodes v. The indicator function does not require entity embeddings and such models can naturally generalize to new entities (while the set of relation types is still fixed). Theoretically, such a labeling strategy learns *conditional node representations* and is provably more powerful [20] than node-level GNN encoders. In CLQA, only GNN-QE [43] applies NBFNet as a projection operator making it the only approach generalizable to the inductive (e) setup [14] with new nodes at inference time. This work leverages labeling trick GNNs to generalize CLQA to arbitrary KGs with any entity and relation vocabulary.

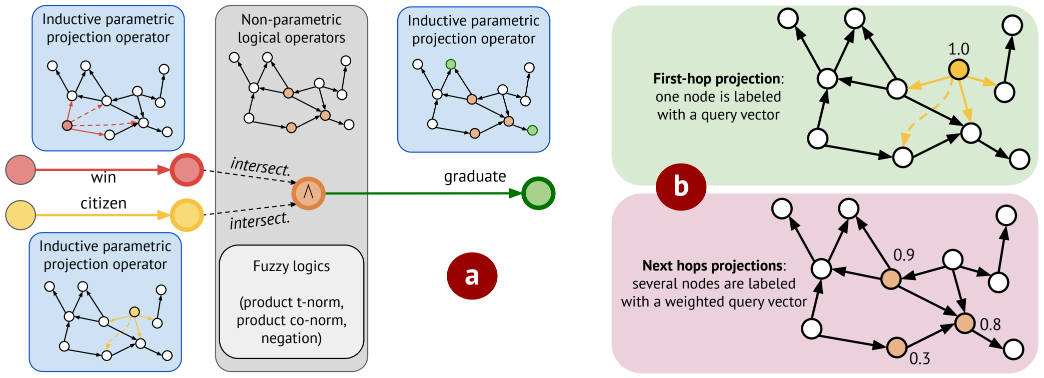

Inductive parametric projection operator
Non-parametric logical operators
Inductive parametric projection operator
win
citizen
intersect.
intersect.
graduate
Inductive parametric projection operator
Fuzzy logics
(product t-norm, product co-norm, negation)
a
b
First-hop projection: one node is labeled with a query vector
1.0
Next hops projections: several nodes are labeled with a weighted query vector
0.9
0.8
0.3

Figure 3: (**a**) Example of *ip* query answering with ULTRAQUERY: the inductive parametric projection operator (Section 4.1) executes relation projections on any graph and returns a scalar score for each entity; the scores are aggregated by non-parametric logical operators (Section 4.2) implemented with fuzzy logics. Intermediate scores are used for weighted initializion of relation projections on the next hop. **(b)** The multi-source propagation issue with a pre-trained link predictor for relation projection: pre-training on *1p* link prediction is done in the single-source labeling mode (top) where only one query node is labeled with a non-zero vector; complex queries at later intermediate hops might have several plausible sources with non-zero initial weights (bottom) where a pre-trained operator fails.

4 Method

We aim at designing a single foundation model for CLQA on any KG in the zero-shot fashion, *i.e.*, without training on a target graph. In the CLQA literature [19, 26, 25, 3, 43], it is common to break down query execution into a *relation projection* to traverse graph edges and predict missing links, and *logical operators* that model conjunction, disjunction, and union. The main challenge boils down to designing inductive projection and logical operators suitable for any entity and relation vocabulary.

4.1 Inductive Relation Projection

The vast majority of CLQA methods are inherently transductive and implement relation projections as functions over entity and relation embeddings fixed to a certain KG vocabulary, *e.g.*, with scoring functions from KG completion methods [19, 3, 5], geometric functions [26, 40], or pure neural methods [2, 34]. The only method inductive to new entities [43] learns relation embeddings and uses those as a labeling trick (Section 3) for a GNN that implements the projection operator.

As fixed relation embeddings do not transfer to new KGs with new relations, we adapt ULTRA [15], an inductive approach that builds relation representations dynamically using the invariance of *relation interactions*, as the backbone of the relation projection operator thanks to its good zero-shot performance on simple KG completion tasks across a variety of graphs. ULTRA leverages theoretical findings in multi-relational link prediction [6, 20] and learns relation representations from a *meta-graph* of relation interactions[3]. The meta-graph includes four learnable edge types or meta-relations (*head-to-tail*, *tail-to-head*, *head-to-head*, *tail-to-tail*) which are independent from KG's relation vocabulary and therefore transfer across any graph. Practically, given a graph $\mathcal{G}$ and projection query $(h, r, ?)$, ULTRA employs labeling trick GNNs on two levels. First, it builds a meta-graph $\mathcal{G}_r$ of relation interactions (a graph of relations where each node is a unique edge type in $\mathcal{G}$) and applies a labeling trick to initialize the query node r. Note that $|\mathcal{R}| \ll |\mathcal{E}|$, the number of unique relations is much smaller than number of entities in any KG, so processing this graph of relations introduces a rather marginal computational overhead. Running a message passing GNN over $\mathcal{G}_r$ results in *conditional relation representation* which are used as initial edge type features in the second, entity-level GNN. There, a starting node h is initialized with a query vector from the obtained relation representations and running another GNN over the entity graph (with a final sigmoid readout) returns a scalar score in $[0, 1]$ representing a probability of each node to be a tail of a query $(h, r, ?)$.

The only learnable parameters in ULTRA are four meta-relations for the graph of relations and GNN weights. The four meta-relations represent structural patterns and can be mined from any

[3]The meta-graph can be efficiently obtained from any KG.

multi-relational KG independent of their entity and relation vocabulary. GNN weights are optimized during pre-training. Since the model does not rely on any KG-specific entity or relation vocabulary, a single pre-trained ULTRA model can be used as a zero-shot relation projection operator on any KG. Figure 3(a) illustrates the *intersection-projection* query execution process where each projection step is tackled by the same inductive projection operator with initialization depending on the start anchor node or intermediate variables.

The multi-source propagation issue. While it is tempting to leverage ULTRA pre-trained on multiple KG datasets for relation projection, there is a substantial distribution shift (Figure 3(b)) between KG completion and CLQA. Specifically, KG completion is a special case of relation projection where the input always contains a single node. By comparison, in multi-hop complex queries, several likely nodes might have high intermediate scores and will be labeled with non-zero vectors leading to the *multiple sources* propagation mode where a pre-trained operator is likely to fail. To alleviate the issue, we experimentally study two strategies: (1) short fine-tuning of the pre-trained projection operator on complex queries (used in the main ULTRAQUERY model), or (2) use the frozen pre-trained operator and threshold intermediate scores setting all scores below $0 < k < 1$ to zero (denoted as ULTRAQUERY LP). The insight is to limit the propagation to one or a few source nodes, thereby reducing the discrepancy between training and test distributions.

4.2 Inductive Logical Operations

Learnable logical operators parameterized by neural nets in many CLQA approaches [19, 26, 40, 2] fit a particular embedding space and are not transferable. Instead, we resort to differentiable but non-parametric *fuzzy logics* [31] that implement logical operators as algebraic operations (*t-norms* for conjunction and *t-conorms* for disjunction) in a bounded space $[0, 1]$ and are used in several neuro-symbolic CLQA approaches [3, 43, 4, 5, 36]. ULTRAQUERY employs fuzzy logical operators over *fuzzy sets* $\boldsymbol{x} \in [0, 1]^{|\mathcal{V}|}$ as the relation projection operator assigns a scalar in range $[0, 1]$ for each entity in a graph. The choice of a fuzzy logic is often a hyperparameter although van Krieken et al. [31] show that the *product logic* is the most stable. In product logic, given two fuzzy sets $\boldsymbol{x}, \boldsymbol{y}$, conjunction is element-wise multiplication $\boldsymbol{x} \odot \boldsymbol{y}$ and disjunction is $\boldsymbol{x} + \boldsymbol{y} - \boldsymbol{x} \odot \boldsymbol{y}$. Negation is often implemented as $\boldsymbol{1} - \boldsymbol{x}$ where $\boldsymbol{1}$ is the *universe* vector of all ones. For second- and later i-th hop projections, we obtain initial node states $\boldsymbol{h}_v$ by weighting a query vector $\boldsymbol{r}_i$ with their probability score x_v from the fuzzy set of a previous step: $\boldsymbol{h}_v = x_v \boldsymbol{r}_i$.

4.3 Training

Following existing works [25, 43], ULTRAQUERY is trained on complex queries to minimize the binary cross entropy loss

$$\mathcal{L} = -\frac{1}{|\mathcal{A}_q|} \sum_{a \in \mathcal{A}_q} \log p(a|q) - \frac{1}{|\mathcal{V} \backslash \mathcal{A}_q|} \sum_{a' \in \mathcal{V} \backslash \mathcal{A}_q} \log(1 - p(a'|q)) \tag{1}$$

where $\mathcal{A}_q$ is the answer to the query q and $p(a|q)$ is the probability of entity a in the final output fuzzy set. ULTRAQUERY LP uses a frozen checkpoint from KG completion and is not trained on complex logical queries.

5 Experiments

Our experiments focus on the following research questions: (1) How does a single ULTRAQUERY model perform in the zero-shot inference mode on unseen graphs and queries compared to the baselines? (2) Does ULTRAQUERY retain the quality metrics like *faithfullness* and identify easy answers reachable by traversal? (3) How does the multi-source propagation issue affect the performance?

5.1 Setup and Datasets

Datasets. We employ 23 different CLQA datasets each with 14 standard query types and its own underlying KG with different sets of entities and relations. Following Section 3, we categorize the datasets into three groups (more statistics of the datasets and queries are provided in Appendix A):

- *Transductive* (3 datasets) where training and inference graphs are the same ($\mathcal{G}_{train} = \mathcal{G}_{inf}$) and test queries cover the same set of entities and relations: FB15k237, NELL995 and FB15k all from Ren and Leskovec [25] with at most 100 answers per query.
- *Inductive entity* (e) (9 datasets) from Galkin et al. [14] where inference graphs extend training graphs ($\mathcal{G}_{train} \subset \mathcal{G}_{inf}$) being up to 550% larger in the number of entities. The set of relations is fixed in each training graph and does not change at inference making the setup inductive with respect to the entities. Training queries might have more true answers in the extended inference graph.
- *Inductive entity and relation* (e, r) (11 datasets): we sampled a novel suite of WikiTopics-QA datasets due to the absence of standard benchmarks evaluating the hardest inductive setup where inference graphs have both new entities and relations ($\mathcal{G}_{train} \neq \mathcal{G}_{inf}$). The source graphs were adopted from the WikiTopics datasets [16], we follow the *BetaE setting* when sampling 14 query types with at most 100 answers. More details on the dataset creation procedure are in Appendix A.

Implementation and Training. ULTRAQUERY was trained on one FB15k237 dataset with complex queries for 10,000 steps with batch size of 32 on 4 RTX 3090 GPUs for 2 hours (8 GPU-hours in total). We initialize the model weights with an available checkpoint of ULTRA reported in Galkin et al. [15]. Following the standard setup in the literature, we train the model on 10 query types and evaluate on all 14 patterns. We employ *product t-norm* and *t-conorm* as non-parametric fuzzy logic operators to implement conjunction ($\wedge$) and disjunction ($\vee$), respectively, and use a simple $1 - x$ negation. For the ablation study, ULTRAQUERY LP uses the same frozen checkpoint (pre-trained on simple *1p* link prediction) with scores thresholding to alleviate the multi-source propagation issue (Section 4.1). More details on all hyperparameters are available in Appendix B.

Evaluation Protocol. As we train an ULTRAQUERY model only on one FB15k237 dataset and run zero-shot inference on other 22 graphs, the inference mode on those is *inductive* (e, r) since their entity and relation vocabularies are all different from the training set.

As common in the literature [25, 27], the answer set of each query is split into *easy* and *hard* answers. Easy answers are reachable by graph traversal and do not require inferring missing links whereas hard answers are those that involve at least one edge to be predicted at inference. In the rank-based evaluation, we only consider ranks of *hard* answers and filter out easy ones and report filtered Mean Reciprocal Rank (MRR) and Hits@10 as main performance metrics.

Other qualitative metrics include: (1) *faithfullness* [28], *i.e.*, the ability to recover *easy* answers reachable by graph traversal. Here, we follow the setup in Galkin et al. [14] and measure the performance of training queries on larger inference graphs where the same queries might have new true answers; (2) the ROC AUC score to estimate whether a model ranks easy answers higher than hard answers – we compute ROC AUC over *unfiltered* scores of easy answers as positive labels and hard answers as negative. (3) Mean Absolute Percentage Error (MAPE) [43] between the number of answers extracted from model's predictions and the number of ground truth answers (easy and hard combined) to estimate whether CLQA models can predict the cardinality of the answer set.

Baselines. In transductive and inductive (e) datasets, we compare a single ULTRAQUERY model with the best reported models trained end-to-end on each graph (denoted as *Best baseline* in the experiments): QTO [5] for 3 transductive datasets (FB15k237, FB15k, and NELL995) and GNN-QE [14] for 9 inductive (e) datasets. While a single ULTRAQUERY model has 177k parameters, the baselines are several orders of magnitude larger with a parameters count depending on the number of entities and relations, *e.g.*, a QTO model on FB15k237 has 30M parameters due to having $2000d$ entity and relation embeddings, and GNN-QE on a reference FB 175% inductive (e) dataset has 2M parameters. For a newly sampled suite of 11 inductive (e, r) datasets, we compare against the edge-type heuristic baseline introduced in Galkin et al. [14]. The heuristic selects the candidate nodes with the same incoming relation as the last hop of the query. More details on the baselines are reported in Appendix B

5.2 Main Experiment: Zero-shot Query Answering

In the main experiment, we measure the zero-shot query answering performance of ULTRAQUERY trained on a fraction of complex queries of one FB15k237 dataset. Figure 1 and Table 2 illustrate the comparison with the best available baselines and ablated ULTRAQUERY LP model on 23 datasets split into three categories (transductive, inductive (e), and inductive (e, r)). For each dataset, we

Table 2: Zero-shot inference results of UltraQuery and ablated UltraQuery LP on 23 datasets compared to the best reported baselines. UltraQuery was trained on one transductive FB15k237 dataset, UltraQuery LP was only pre-trained on KG completion and uses scores thresholding. The *no thrs.* version does not use any thresholding of intermediate scores (Section 4.1). The best baselines are trainable on each transductive and inductive (e) dataset, and the non-parametric heuristic baseline on inductive (e, r) datasets.

Model	**Inductive** (e, r) (11 datasets)				**Inductive** (e) (9 datasets)				**Transductive** (3 datasets)				**Total Average** (23 datasets)			
	EPFO avg		neg avg		EPFO avg		neg avg		EPFO avg		neg avg		EPFO avg		neg avg	
	MRR	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**
Best baseline	0.014	0.029	0.004	0.007	**0.328**	**0.469**	**0.176**	**0.297**	**0.468**	**0.603**	**0.259**	**0.409**	0.196	0.276	0.105	0.173
UltraQuery 0-shot	**0.280**	0.380	**0.193**	**0.288**	0.312	**0.467**	0.139	0.262	0.411	0.517	0.240	0.352	**0.309**	**0.432**	**0.178**	**0.286**
UltraQuery LP 0-shot	0.268	**0.409**	0.104	0.181	0.277	0.441	0.098	0.191	0.322	0.476	0.150	0.263	0.279	0.430	0.107	0.195
UltraQuery LP no thrs.	0.227	0.331	0.080	0.138	0.246	0.390	0.085	0.167	0.281	0.417	0.127	0.223	0.242	0.367	0.088	0.161

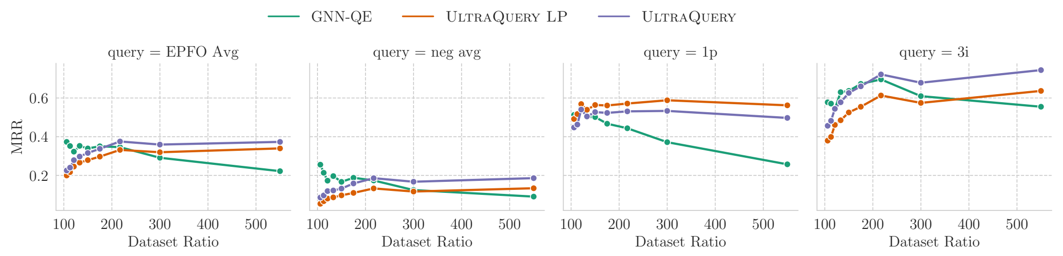


Figure 4: Mitigation of the multi-source message passing issue (Section 4) with UltraQuery: while UltraQuery LP (pre-trained only on 1p link prediction) does reach higher 1p query performance (center right), it underperforms on negation queries (center left). UltraQuery adapts to the multi-source message passing scheme and trades a fraction of 1p query performance for better averaged EPFO, *e.g.*, on the *3i* query (right), and negation queries performance. More results are in Appendix C.

measure the average MRR on 9 EPFO queries with projection, intersection, and union operators, and 5 negation queries with the negation operator, respectively.

Averaged across 23 datasets, UltraQuery outperforms available baselines by relative 50% in terms of MRR and Hits@10 on EPFO and 70% on negation queries (*e.g.*, 0.31 vs 0.20 MRR on EPFO queries and 0.178 vs 0.105 on negation queries). The largest gains are achieved on the hardest inductive (e, r) datasets where the heuristic baseline is not able to cope with the task. On inductive (e) datasets, UltraQuery outperforms the trainable SOTA GNN-QE model on larger inductive inference graphs and performs competitively on smaller inductive versions. On transductive benchmarks, UltraQuery lags behind the SOTA QTO model which is expected and can be attributed to the sheer model size difference (177k of UltraQuery vs 30M of QTO) and the computationally expensive brute-force approach of QTO that materializes the whole $(\mathcal{V} \times \mathcal{V} \times \mathcal{R})$ 3D tensor of scores of all possible triples. Pre-computing such tensors on three datasets takes considerable space and time, *e.g.*, 8 hours for FB15k with heavy sparsification settings to fit onto a 24 GB GPU. Still, UltraQuery outperforms a much larger QTO model on the FB15k dataset on both EPFO and negation queries. The graph behind the NELL995 dataset is a collection of disconnected components which is disadvantageous for GNNs.

We note a decent performance of UltraQuery LP trained only on simple *1p* link prediction and imbued with score thresholding to alleviate the multi-source message passing issue described in Section 4.1. Having a deeper look at other qualitative metrics in the following section, we reveal more sites where the issue incurs negative effects.

5.3 Analysis

Here, we study four aspects of model performance: the effect of the multi-source message passing issue mentioned in Section 4.1, the ability to recover answers achievable by edge traversal (*faithfullness*), the ability to rank easy answers higher than hard answers, and the ability to estimate the cardinality of the answer set.

The multi-source message passing effect. The pre-trained Ultra checkpoint used in UltraQuery LP is tailored for singe-source message passing and struggles in the CLQA setup on later hops with several initialized nodes (Table 2). Training UltraQuery on complex queries alleviates this issue as

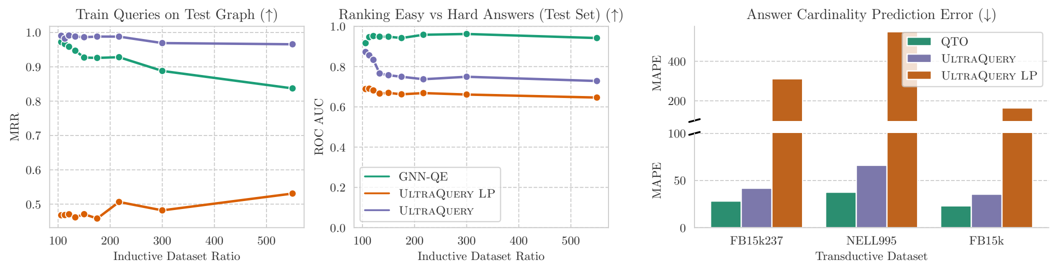


Figure 5: Qualitative analysis on 9 inductive (e) and 3 transductive datasets averaged across all 14 query types. **Faithfullness, MRR (left):** UltraQuery successfully finds easy answers in larger inference graphs and outperforms trained GNN-QE baselines. **Ranking of easy vs hard answers, ROC AUC (center):** zero-shot inference methods slightly lag behind trainable GNN-QE due to assigning higher scores to hard answers. **Cardinality Prediction, MAPE (right):** UltraQuery is comparable to a much larger trainable baseline QTO. In all cases, UltraQuery LP is significantly inferior to the main model.

shown in Figure 4, *i.e.*, while *1p* performance of UltraQuery LP is higher, the overall performance on EPFO and negative queries is lacking. In contrast, UltraQuery trades a fraction of *1p* single-source performance to a much better performance on negative queries (about $2\times$ improvement) and better performance on many EPFO queries, for example, on *3i* queries. Besides that, we note that the zero-shot performance of both UltraQuery models does not deteriorate from the increased size of the inference graph compared to the baseline GNN-QE.

Recovering easy answers on any graph. *Faithfullness* [28] is the ability of a CLQA model to return *easy* query answers, *i.e.*, the answers reachable by edge traversal in the graph without predicting missing edges. While faithfullness is a common problem for many CLQA models, Figure 5 demonstrates that UltraQuery almost perfectly recovers easy answers on any graph size even in the zero-shot inference regime in contrast to the best baseline. Simple score thresholding does not help UltraQuery LP to deal with complex queries as all easy intermediate nodes have high scores above the threshold and the multi-source is more pronounced.

Ranking easy and hard answers. A reasonable CLQA model is likely to score easy answers higher than hard ones that require inferring missing links [14]. Measuring that with ROC AUC (Figure 5), UltraQuery is behind the baseline due to less pronounced decision boundaries (overlapping distributions of scores) between easy and hard answers' scores. Still, due to scores filtering when computing ranking metrics, this fact does not have a direct negative impact on the overall performance.

Estimating the answer set cardinality. Neural-symbolic models like GNN-QE and QTO have the advantage of estimating the cardinality of the answer set based on the final scores without additional supervision. As shown in Figure 5, UltraQuery is comparable to the larger and trainable QTO baseline on FB15k237 (on which the model was trained) as well as on other datasets in the zero-shot inference regime. Since cardinality estimation is based on score thresholding, UltraQuery LP is susceptible to the multi-source propagation issue with many nodes having a high score and is not able to deliver a comparable performance.

Varying the number of graphs in training. Figure 6 and Table 3 report the inductive inference CLQA performance depending on the number of KGs in the training mixture. The original UltraQuery was trained on queries from the FB15k237. In order to maintain the zero-shot inductive inference setup on 11 inductive (e, r) and 9 inductive (e) datasets, we trained new model versions on the rest of BetaE datasets, that is, UltraQuery 2G combines FB15k237 and NELL995 queries (trained for 20k steps), UltraQuery 3G combines FB15k273, NELL995, and FB15k queries (trained for 30k steps). The most noticeable improvement of 2G and 3G versions is the increased MRR and Hits@10 on EPFO queries (9 query

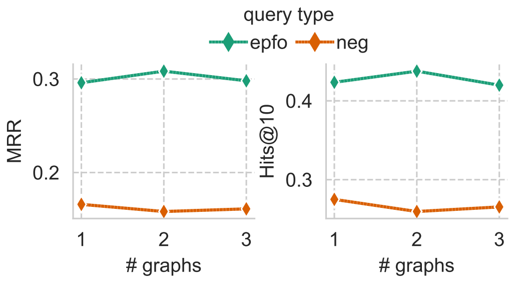


Figure 6: Average MRR (left) and Hits@10 (right) of 9 inductive (e) and 11 inductive (e, r) CLQA datasets for EPFO and negation queries depending on the number of graphs in the training mix.

Table 3: Zero-shot inference results (on 20 inductive datasets) of ULTRAQUERY trained on 1, 2, and 3 datasets, respectively. The biggest gains of the 2G model are **in bold**.

Model	**Inductive** (e, r) (11 datasets)				**Inductive** (e) (9 datasets)				**Total Average** (20 datasets)			
	EPFO avg		neg avg		EPFO avg		neg avg		EPFO avg		neg avg	
	MRR	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**	**MRR**	**H@10**
ULTRAQUERY 1G	0.280	0.380	0.193	0.288	0.312	0.467	0.139	0.262	0.296	0.423	0.166	0.275
ULTRAQUERY 2G	**0.310**	**0.413**	0.187	0.275	0.307	0.463	0.130	0.244	0.308	0.438	0.158	0.260
ULTRAQUERY 3G	0.304	0.402	0.195	0.292	0.292	0.438	0.127	0.239	0.298	0.420	0.161	0.265

types) on 11 inductive (e, r) datasets yielding about 10% gains. On 9 inductive (e) datasets the performance is either on par with the 1G version or a bit lower. Averaged across 20 datasets, the 2G version exhibits the best EPFO performance at the cost of slightly reduced negation query performance.

6 Discussion and Future Work

Limitations. The only parameterized component of ULTRAQUERY is the projection operator and, therefore, limitations and improvement opportunities stem from the projection operator [15] and its interplay with the multi-hop query answering framework. For instance, new mechanisms of tackling the multi-source propagation, better pre-training strategies, and scaling might positively impact the zero-shot CLQA performance. The support for very large KGs could be further improved by adopting more scalable entity-level GNN predictors like A*Net [44] or AdaProp [39] which have been shown to scale to graphs of millions of nodes. We are optimistic that ULTRAQUERY could scale to such graphs when integrated with those models.

Conclusion and Future Work. We presented ULTRAQUERY, the first foundation model for inductive zero-shot complex logical query answering on any KG that combines a parameterized, inductive projection operator with non-parametric logical operators. Alleviating the multi-source message propagation issue is the key to adapt pre-trained projection operators into the multi-hop query answering framework. ULTRAQUERY performs comparably to or better than strong baselines trained specifically on each graph and at the same time retains key qualitative features like faithfullness and answer cardinality estimation. Having a single query answering model working on any KG, the scope for future work is vast as highlighted by Ren et al. [27] and includes, for example, better theoretical understanding of logical expressiveness bounds, supporting more query patterns beyond simple trees [35, 36], queries without anchor nodes [7], hyper-relational queries [1], queries with numerical literals [11], or temporal queries [23].

Impact Statement. We do not envision direct ethical or societal consequences of this work. Still, models capable of zero-shot inference on any graph might be applied to domains other than those designed by the authors. Positive impacts include saving compute resources and reducing carbon footprint of training specific models tailored for each graph.

Acknowledgements

This project is supported by Intel-Mila partnership program, the Natural Sciences and Engineering Research Council (NSERC) Discovery Grant, the Canada CIFAR AI Chair Program, collaboration grants between Microsoft Research and Mila, Samsung Electronics Co., Ltd., Amazon Faculty Research Award, Tencent AI Lab Rhino-Bird Gift Fund and a NRC Collaborative R&D Project (AI4D-CORE-06). This project was also partially funded by IVADO Fundamental Research Project grant PRF-2019-3583139727. The computation resource of this project is supported by Mila[4], Calcul Québec[5] and the Digital Research Alliance of Canada[6].

This work was funded in part by the National Science Foundation (NSF) awards, CCF-1918483, CAREER IIS-1943364 and CNS-2212160, Amazon Research Award, AnalytiXIN, and the Wabash Heartland Innovation Network (WHIN). Computing infrastructure was supported in part by CNS-1925001 (CloudBank). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

[4]https://mila.quebec/
[5]https://www.calculquebec.ca/
[6]https://alliancecan.ca/

References

[1] Dimitrios Alivanistos, Max Berrendorf, Michael Cochez, and Mikhail Galkin. Query embedding on hyper-relational knowledge graphs. In *International Conference on Learning Representations*, 2022.

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

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

[4] Erik Arakelyan, Pasquale Minervini, Daniel Daza, and Michael Cochez Isabelle Augenstein. Adapting neural link predictors for data-efficient complex query answering. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1G7CBp8o7L.

[5] Yushi Bai, Xin Lv, Juanzi Li, and Lei Hou. 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, 2023. URL https://proceedings.mlr.press/v202/bai23b.html.

[6] Pablo Barcelo, Mikhail Galkin, Christopher Morris, and Miguel Romero Orth. Weisfeiler and leman go relational. In *The First Learning on Graphs Conference*, 2022. URL https://openreview.net/forum?id=wY_IYhh6pqj.

[7] Pablo Barceló, Tamara Cucumides, Floris Geerts, Juan Reutter, and Miguel Romero. A neuro-symbolic framework for answering conjunctive queries. *arXiv preprint arXiv:2310.04598*, 2023.

[8] Mingyang Chen, Wen Zhang, Yuxia Geng, Zezhong Xu, Jeff Z Pan, and Huajun Chen. Generalizing to unseen elements: A survey on knowledge extrapolation for knowledge graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 6574–6582, 2023.

[9] Xuelu Chen, Ziniu Hu, and Yizhou Sun. Fuzzy logic based logical query answering on knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI*, pages 3939–3948. AAAI Press, 2022.

[10] Daniel Daza and Michael Cochez. Message passing query embedding. *Proceedings of the ICML 2020 Workshop on Graph Representation Learning and Beyond*, 2020.

[11] Caglar Demir, Michel Wiebesiek, Renzhong Lu, Axel-Cyrille Ngonga Ngomo, and Stefan Heindorf. LitCQD: Multi-hop reasoning in incomplete knowledge graphs with numeric literals. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 617–633, 2023.

[12] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[13] Mikhail Galkin, Etienne Denis, Jiapeng Wu, and William L. Hamilton. Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xMJWUKJnFSw.

[14] Mikhail Galkin, Zhaocheng Zhu, Hongyu Ren, and Jian Tang. Inductive logical query answering in knowledge graphs. In *Advances in Neural Information Processing Systems*, 2022.

[15] Mikhail Galkin, Xinyu Yuan, Hesham Mostafa, Jian Tang, and Zhaocheng Zhu. Towards foundation models for knowledge graph reasoning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jVEoydFOl9.

[16] Jianfei Gao, Yangze Zhou, Jincheng Zhou, and Bruno Ribeiro. Double equivariance for inductive link prediction for both new nodes and new relation types. In *NeurIPS 2023 Workshop: New Frontiers in Graph Learning*, 2023.

[17] Thomas Gebhart, Jakob Hansen, and Paul Schrater. Knowledge sheaves: A sheaf-theoretic framework for knowledge graph embedding. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*. PMLR, 2023. URL https://proceedings.mlr.press/v206/gebhart23a.html.

[18] Yuxia Geng, Jiaoyan Chen, Jeff Z Pan, Mingyang Chen, Song Jiang, Wen Zhang, and Huajun Chen. Relational message passing for fully inductive knowledge graph completion. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 1221–1233. IEEE, 2023.

[19] Will Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[20] Xingyue Huang, Miguel Romero Orth, Ismail Ilkan Ceylan, and Pablo Barcelo. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=7hLlZNrkt5.

[21] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[22] Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang. InGram: Inductive knowledge graph embedding via relation graphs. In *Proceedings of the 40th International Conference on Machine Learning*, pages 18796–18809. PMLR, 2023. URL https://proceedings.mlr.press/v202/lee23c.html.

[23] Xueyuan Lin, Haihong E, Chengjin Xu, Gengxian Zhou, Haoran Luo, Tianyi Hu, Fenglong Su, Ningyuan Li, and Mingzhi Sun. TFLEX: Temporal feature-logic embedding framework for complex reasoning over temporal knowledge graph. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=oaGdsgB18L.

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32, NeurIPS 2019*, pages 8024–8035, 2019.

[25] Hongyu Ren and Jure Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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

[27] Hongyu Ren, Mikhail Galkin, Michael Cochez, Zhaocheng Zhu, and Jure Leskovec. Neural graph reasoning: Complex logical query answering meets graph databases. *arXiv preprint arXiv:2303.14617*, 2023.

[28] Haitian Sun, Andrew Arnold, Tania Bedrax Weiss, Fernando Pereira, and William W Cohen. Faithful embeddings for knowledge base queries. In *Advances in Neural Information Processing Systems*, 2020.

[29] Komal Teru, Etienne Denis, and Will Hamilton. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pages 9448–9457. PMLR, 2020.

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

[31] Emile van Krieken, Erman Acar, and Frank van Harmelen. Analyzing differentiable fuzzy logic operators. *Artificial Intelligence*, 302, 2022. URL https://www.sciencedirect.com/science/article/pii/S0004370221001533.

[32] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. Composition-based multi-relational graph convolutional networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BylA_C4tPr.

[33] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[34] Zihao Wang, Yangqiu Song, Ginny Y. Wong, and Simon See. Logical message passing networks with one-hop inference on atomic formulas. In *The Eleventh International Conference on Learning Representations, ICLR*, 2023. URL https://openreview.net/forum?id=SoyOsp7i_l.

[35] Hang Yin, Zihao Wang, Weizhi Fei, and Yangqiu Song. EFO$_k$-cqa: Towards knowledge graph complex query answering beyond set operation. *arXiv preprint arXiv:2307.13701*, 2023.

[36] Hang Yin, Zihao Wang, and Yangqiu Song. Rethinking complex queries on knowledge graphs with neural link predictors. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=1BmveEMNbG.

[37] Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 9061–9073, 2021.

[38] Yongqi Zhang and Quanming Yao. Knowledge graph reasoning with relational digraph. In *Proceedings of the ACM Web Conference 2022*, pages 912–924, 2022.

[39] Yongqi Zhang, Zhanke Zhou, Quanming Yao, Xiaowen Chu, and Bo Han. Adaprop: Learning adaptive propagation for graph neural network based knowledge graph reasoning. In *KDD*, 2023.

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

[41] Jincheng Zhou, Beatrice Bevilacqua, and Bruno Ribeiro. An ood multi-task perspective for link prediction with new relation types and nodes. *arXiv preprint arXiv:2307.06046*, 2023.

[42] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems*, 34:29476–29490, 2021.

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

[44] Zhaocheng Zhu, Xinyu Yuan, Mikhail Galkin, Sophie Xhonneux, Ming Zhang, Maxime Gazeau, and Jian Tang. A*net: A scalable path-based reasoning approach for knowledge graphs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=dAJrxQz1lk.

A Datasets

A.1 Dataset statistics

First, Table 4, Table 5, and Table 6 provide the necessary details on the graphs behind the CLQA datasets. Then, Table 7, Table 8, and Table 9 list the query statistics. Transductive datasets are BetaE [25] datasets (MIT license), inductive (e) datasets are adopted from Galkin et al. [14] (CC BY 4.0 license) where validation and test inference graphs extend the training graph. The ratio denotes the size of the inference graph to the size of the training graph (in the number of nodes), that is, $\mathcal{V}_{inf}/\mathcal{V}_{train}$. In the following Appendix A.2 we provide more details in sampling 11 new inductive (e,r) datasets WikiTopics-CLQA (available under the CC BY 4.0 license).

Table 4: Graph in transductive datasets (3) from Ren and Leskovec [25]. Inverse triples and edge types are included in the splits. Train, Valid, Test denote triples in the respective set.

Dataset	Entities	Rels	Train	Valid	Test
FB15k	14,951	2,690	966,284	100,000	118,142
FB15k237	14,505	474	544,230	35,052	40,876
NELL995	63,361	400	228,426	28,648	28,534

Table 5: Graphs in inductive (e) datasets (9) from Galkin et al. [14]. Inverse triples and edge types are included in all graphs. Validation and Test splits contain an inference graph $(\mathcal{V}_{inf}, \mathcal{E}_{inf})$ which is a superset of the training graph with new nodes, and missing edges to predict (Valid and Test, respectively).

Ratio, %	Rels	Training Graph		Validation Graph			Test Graph		
		Entities	Triples	Entities	Triples	Valid	Entities	Triples	Test
106%	466	13,091	493,425	13,801	551,336	10,219	13,802	538,896	8,023
113%	468	11,601	401,677	13,022	491,518	15,849	13,021	486,068	14,893
122%	466	10,184	298,879	12,314	413,554	20,231	12,314	430,892	23,289
134%	466	8,634	228,729	11,468	373,262	25,477	11,471	367,810	24,529
150%	462	7,232	162,683	10,783	311,462	26,235	10,782	331,352	29,755
175%	436	5,560	102,521	9,801	265,412	28,691	9,781	266,494	28,891
217%	446	4,134	52,455	9,062	227,284	30,809	9,058	212,386	28,177
300%	412	2,650	24,439	8,252	178,680	27,135	8,266	187,156	28,657
550%	312	1,084	5,265	7,247	136,558	22,981	7,275	133,524	22,503

Table 6: Graphs in the newly sampled inductive entity and relation (e,r) WikiTopics-CLQA datasets (11). Triples denote the number of edges of the graph given at training, validation, or test. Valid and Test denote triples to be predicted in the validation and test sets in the respective validation and test graph.

Dataset	Training Graph			Validation Graph				Test Graph			
	Entities	Rels	Triples	Entities	Rels	Triples	Valid	Entities	Rels	Triples	Test
Art	10000	65	27262	10000	65	27262	3026	10000	65	28023	3113
Award	10000	17	23821	10000	13	23821	2646	10000	17	25056	2783
Education	10000	19	14355	10000	19	14355	1594	10000	19	14193	1575
Health	10000	31	15539	10000	31	15539	1725	10000	31	15337	1703
Infrastructure	10000	37	21990	10000	37	21990	2443	10000	37	21646	2405
Location	10000	62	85063	10000	62	85063	9451	10000	62	80269	8917
Organization	10000	34	33325	10000	34	33325	3702	10000	34	31314	3357
People	10000	40	55698	10000	40	55698	6188	10000	40	58530	6503
Science	10000	66	12576	10000	66	12576	1397	10000	66	12516	1388
Sport	10000	34	47251	10000	34	47251	5250	10000	34	46717	5190
Taxonomy	10000	59	18921	10000	59	18921	2102	10000	59	19416	2157

A.2 WikiTopics-CLQA

The WikiTopics dataset introduced by Gao et al. [16] was used to evaluate link prediction model's zero-shot performance in the inductive (e,r) setting, i.e., when the test-time inference graph contains *both* new entities and new relations unseen in training. It grouped relations into 11 different topics, or

Table 7: Statistics of 3 transductive datasets

Split	Query Type	FB15k	FB15k-237	NELL995
Train	1p/2p/3p/2i/3i	273,710	149,689	107,982
	2in/3in/inp/pin/pni	27,371	14,968	10,798
Valid	1p	59,078	20,094	16,910
	Others	8,000	5,000	4,000
Test	1p	66,990	22,804	17,021
	Others	8,000	5,000	4,000

Table 8: Statistics of 9 inductive (e) datasets.

Ratio	Graph	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
	training	135,613	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	40,000	50,000	50,000	50,000
106%	validation	6,582	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	1,000	1,000	1,000	1,000	1,000
	test	5,446	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	1,000	1,000	1,000	1,000	1,000
	training	115,523	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	40,000	50,000	50,000	50,000
113%	validation	10,256	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	1,000	1,000	1,000	1,000	1,000
	test	9,782	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	1,000	1,000	1,000	1,000	1,000
	training	91,228	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	40,000	50,000	50,000	50,000
121%	validation	12,696	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	5,000	5,000	5,000	5,000	5,000
	test	14,458	10,000	10,000	10,000	10,000	10,000	10,000	10,000	10,000	5,000	5,000	5,000	5,000	5,000
	training	75,326	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	40,000	50,000	50,000	50,000
133%	validation	15,541	50,000	50,000	50,000	50,000	50,000	50,000	20,000	20,000	5,000	5,000	5,000	5,000	5,000
	test	15,270	50,000	50,000	50,000	50,000	50,000	50,000	20,000	20,000	5,000	5,000	5,000	5,000	5,000
	training	56,114	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	40,000	50,000	50,000	50,000
150%	validation	16,229	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	5,000	5,000	5,000	5,000	5,000
	test	17,683	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	5,000	5,000	5,000	5,000	5,000
	training	38,851	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	40,000	50,000	50,000	50,000
175%	validation	17,235	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	test	17,476	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	training	22,422	30,000	30,000	50,000	50,000	50,000	50,000	50,000	50,000	30,000	30,000	50,000	50,000	50,000
217%	validation	18,168	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	test	16,902	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	training	11,699	15,000	15,000	40,000	40,000	50,000	50,000	50,000	50,000	15,000	15,000	50,000	40,000	50,000
300%	validation	16,189	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	test	17,105	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	training	3,284	15,000	15,000	40,000	40,000	50,000	50,000	50,000	50,000	10,000	10,000	30,000	30,000	30,000
550%	validation	13,616	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000
	test	13,670	50,000	50,000	50,000	50,000	50,000	50,000	50,000	50,000	10,000	10,000	10,000	10,000	10,000

Table 9: WikiTopics-CLQA statistics: the number of queries generated per query pattern for each topic knowledge graph of WikiTopics [16]. Numbers are the same for both the training and inference graph. We follow the same 14 query patterns introduced by Ren and Leskovec [25].

Topics	1p	2p	3p	2i	3i	pi	ip	2in	3in	pin	pni	inp	2u	up
Art	3113	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Award	2783	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Education	1575	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Health	1703	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Infrastructure	2405	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Location	8000	8917	4000	8000	8000	8000	8000	1000	1000	1000	1000	1000	8000	8000
Organization	3357	8000	4000	8000	8000	8000	8000	1000	1000	1000	1000	1000	8000	8000
People	6503	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Science	1388	10000	10000	10000	10000	10000	10000	1000	1000	1000	1000	1000	10000	10000
Sport	5190	8000	4000	8000	8000	8000	8000	1000	1000	1000	1000	1000	8000	8000
Taxonomy	2157	8000	8000	8000	8000	8000	8000	1000	1000	1000	1000	1000	8000	8000

domains, such as art, education, health care, and sport. Two graphs, $\mathcal{G}_{train}^{(T)}$ and $\mathcal{G}_{inf}^{(T)}$ along with the missing triples $\mathcal{E}_{valid}^{(T)}$ and $\mathcal{E}_{test}^{(T)}$, were provided for each topic T, which had the same set of relations but different (potentially overlapping) set of entities. The goal was to train models on the training graphs $\mathcal{G}_{train}^{(T)}$ of some topic T, and test on the inference graph $\mathcal{G}_{inf}^{(T')}$ of an unseen topic T'. The model's validation performance was evaluated on the missing triples $\mathcal{E}_{valid}^{(T)}$ when observing training graph $\mathcal{G}_{train}^{(T)}$ as inputs, and its test performance was evaluated on $\mathcal{E}_{test}^{(T')}$ when observing the test inference graph $\mathcal{G}_{inf}^{(T')}$ as inputs. Table 6 shows the statistics of the 11 topic-specific knowledge graphs in WikiTopics.

We follow the procedures in BetaE [25] to generate queries and answers of the 14 query patterns using the knowledge graphs in WikiTopics. We name the resulting dataset WikiTopics-CLQA. For each topic T, we generate three sets of queries and answers (training, validation, test), using the training graph $\mathcal{G}_{train}^{(T)}$ for training and validation, and inference graph $\mathcal{G}_{inf}^{(T)}$ for test queries, respectively.

Training queries on $\mathcal{G}^{(T)}_{train}$ only have easy answers, validation (test) set easy answers are attained by traversing $\mathcal{G}^{(T)}_{train}$ ($\mathcal{G}^{(T)}_{inf}$), whereas the full set of answers (easy and hard) are attained by traversing the graph $\mathcal{G}^{(T)}_{train}$ merged with $\mathcal{E}^{(T)}_{valid}$ ($\mathcal{G}^{(T)}_{inf}$ merged with $\mathcal{E}^{(T)}_{test}$). Hence, the hard answers cannot be found unless the model is capable of imputing missing links. In our experiments, we only use the inference graph $\mathcal{G}^{(T)}_{inf}$ and the test queries and answers for evaluating zero-shot inference performance. Table 9 shows the statistics of the WikiTopics-CLQA dataset.

B Hyperparameters and Baselines

Both ULTRAQUERY and ULTRAQUERY LP are implemented with PyTorch [24] (BSD-style license) and PyTorch-Geometric [12] (MIT license). Both ULTRAQUERY and ULTRAQUERY LP are initialized from the pre-trained ULTRA checkpoint published by the authors and have the same GNN architectures with parameter count (177k). ULTRAQUERY is further trained for 10,000 steps on complex queries from the FB15k237 dataset. Hyperparameters of ULTRAQUERY and its training details are listed in Table 10.

ULTRAQUERY LP employs thresholding of intermediate fuzzy set scores as one of the ways to alleviate the multi-source propagation issue (Section 4.1). Generally, the threshold is set to 0.8 with a few exceptions:

- 0.97 in NELL995

Below, we discuss the best available baselines for each dataset family.

Transductive (3 datasets): QTO [5]. QTO requires $2000d$ ComplEx [30] entity and relation embeddings pre-computed for each graph, *e.g.*, taking 30M parameters on the FB15k237 graph with 15k nodes. Further, QTO materializes the whole $(\mathcal{V} \times \mathcal{V} \times \mathcal{R})$ 3D tensor of scores of all possible triples for each graph. Pre-computing such tensors on three datasets takes considerable space and time, *e.g.*, 8 hours for FB15k with heavy sparsification settings to fit onto a 24 GB GPU.

Inductive (e) (9 datasets): GNN-QE [43]. The framework of GNN-QE is similar to ULTRAQUERY but the backbone relation projection operator is implemented with NBFNet [42] which is only inductive to entities and still learns graph-specific relation embeddings. Parameter count, therefore, depends on the number of unique relation types, *e.g.*, 2M for the FB 175% split with 436 unique relations.

Inductive (e, r) (11 datasets): for newly sampled datasets, due to the absence of fully inductive trainable baselines, we compare against a rule-based heuristic baseline similar to the baseline in Galkin et al. [14]. The baseline looks up candidate entities that have the same incoming relation type as the current relation projection (note that the identity of the starting head node in this case is ignored). The heuristic filters entities into two classes (satisfying the incoming relation type or not), hence, in order to create a ranking, we randomly shuffle entities in each class. This baseline is non-parametric, does not require any training, and represents a sanity check of CLQA models. Still, as shown in Galkin et al. [14], the baseline might outperform certain inductive reasoning approaches parameterized by neural networks.

C More Results

Table 11 corresponds to Figure 1 and Table 2 and provides MRR and Hits@10 results for ULTRAQUERY, ULTRAQUERY LP, and Best Baseline for each dataset averaged across 9 EPFO query types and 5 negation query types. Figure 7 is the full version of Figure 4 and illustrates the performance of all 3 compared models on 9 inductive (e) datasets on 14 query types together with their averaged values (EPFO avg and neg avg, respectively).

Table 10: ULTRAQUERY hyperparameters. GNN_r denotes a GNN over the graph of relations $\mathcal{G}_r$, GNN_e is a GNN over the original entity graph $\mathcal{G}$.

Hyperparameter		ULTRAQUERY training
GNN_r	# layers	6
	hidden dim	64
	message	DistMult
	aggeregation	sum
GNN_e	# layers	6
	hidden dim	64
	message	DistMult
	aggregation	sum
	$g(\cdot)$	2-layer MLP
Learning	optimizer	AdamW
	learning rate	0.0005
	training steps	10,000
	adv temperature	0.2
	traversal dropout	0.25
	batch size	64
	training queries	FB15k237

Table 11: Full results (MRR, Hits@10) of ULTRAQUERY LP and ULTRAQUERY in the zero-shot inference regime on transductive, entity-inductive (e), and fully inductive (e, r) datasets compared to the best-reported baselines averaged across 9 EPFO query types (EPFO avg) and 5 negation query types (Negation avg). ULTRAQUERY was fine-tuned only on FB15k237 queries. The numbers correspond to Table 2 and Figure 1.

	ULTRAQUERY LP				**ULTRAQUERY**				**Best Baseline**			
	EPFO avg		**Negation avg**		**EPFO avg**		**Negation avg**		**EPFO avg**		**Negation avg**	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
transductive datasets												
FB15k237	0.216	0.362	0.082	0.164	0.242	0.378	0.08	0.174	0.335	0.491	0.155	0.295
FB15k	0.501	0.672	0.291	0.465	0.764	0.834	0.567	0.725	0.740	0.837	0.492	0.664
NELL995	0.249	0.395	0.079	0.160	0.226	0.341	0.073	0.159	0.329	0.483	0.129	0.268
inductive (e) datasets												
FB 550%	0.340	0.518	0.134	0.251	0.373	0.535	0.186	0.332	0.222	0.331	0.091	0.158
FB 300%	0.320	0.496	0.117	0.227	0.359	0.526	0.168	0.312	0.291	0.426	0.125	0.224
FB 217%	0.332	0.509	0.133	0.252	0.375	0.537	0.186	0.337	0.346	0.492	0.174	0.301
FB 175%	0.297	0.469	0.110	0.214	0.338	0.499	0.159	0.297	0.351	0.507	0.188	0.336
FB 150%	0.279	0.445	0.097	0.190	0.316	0.473	0.132	0.255	0.339	0.493	0.167	0.303
FB 133%	0.266	0.426	0.087	0.167	0.298	0.451	0.122	0.238	0.353	0.514	0.197	0.341
FB 121%	0.246	0.400	0.081	0.164	0.279	0.430	0.119	0.232	0.323	0.462	0.173	0.291
FB 113%	0.217	0.362	0.067	0.136	0.240	0.387	0.097	0.192	0.352	0.494	0.214	0.339
FB 106%	0.200	0.340	0.054	0.114	0.226	0.370	0.086	0.162	0.373	0.504	0.256	0.377
inductive (e, r) datasets												
Art	0.249	0.389	0.086	0.157	0.248	0.349	0.083	0.137	0.016	0.031	0.006	0.014
Award	0.224	0.413	0.046	0.098	0.227	0.354	0.152	0.274	0.004	0.006	0.002	0.002
Edu	0.142	0.258	0.066	0.122	0.179	0.249	0.119	0.176	0.008	0.014	0.003	0.005
Health	0.317	0.466	0.159	0.231	0.317	0.394	0.419	0.525	0.019	0.040	0.010	0.020
Infrastructure	0.392	0.551	0.087	0.170	0.337	0.461	0.235	0.356	0.010	0.018	0.003	0.005
Location	0.508	0.678	0.198	0.371	0.569	0.679	0.402	0.585	0.007	0.017	0.001	0.002
Organization	0.098	0.190	0.023	0.048	0.169	0.270	0.082	0.171	0.005	0.008	0.001	0.002
People	0.373	0.530	0.182	0.281	0.332	0.443	0.194	0.285	0.005	0.010	0.002	0.003
Science	0.206	0.348	0.048	0.093	0.158	0.255	0.071	0.131	0.041	0.085	0.010	0.020
Sport	0.215	0.357	0.095	0.166	0.252	0.371	0.178	0.269	0.015	0.030	0.005	0.008
Taxonomy	0.230	0.315	0.151	0.255	0.290	0.360	0.183	0.261	0.034	0.066	0.009	0.017

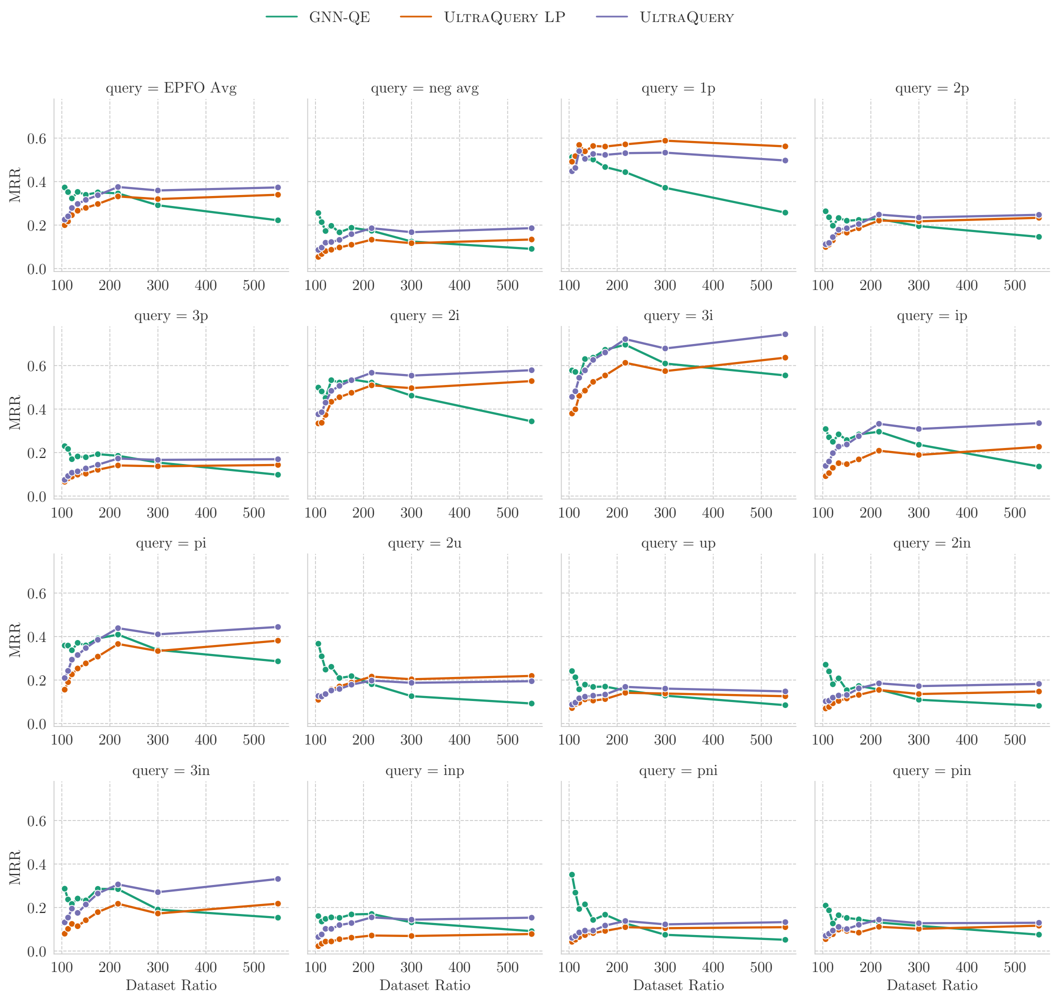


Figure 7: Full results on 9 inductive (e) datasets corresponding to Figure 4: albeit ULTRAQUERY LP outperforms the main ULTRAQUERY on simple *1p* queries, it suffers from the multi-source propagation issue on complex queries. ULTRAQUERY trades a fraction of 1p query performance for significantly better average performance on 9 EPFO and 5 negation query types with particularly noticeable gains on intersection and *2in*, *3in* queries.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper claims (1) the first inductive CLQA method where a single model is able to generalize to arbitrary KGs with any entity and relation vocabulary; (2) study of the multi-source propagation issue. Experiments in Section 5 support the claims, *i.e.*, a single pre-trained ULTRAQUERY transfers to 23 different query answering datasets with a competitive performance (setting state of the art on 15 of those datasets), the ablation study shows that ULTRAQUERY effectively alleviates the multi-source propagation issue.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: No theorems.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The experimental setup is described in Section 5, more details on implementation and hyperparameters are in Appendix B. The code is available to reviewers.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is available at: `https://github.com/DeepGraphLearning/ULTRA`, the datasets are openly available either. Appendix A.2 explains the creation process of the WikiTopics-CLQA dataset.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The details are in Section 5 and in Appendices A, B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Zero-shot inference results on ULTRAQUERY are stable and do not depend on random seeds. The variance of baselines is negligible (as in most KG reasoning works) and is therefore omitted.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The details are provided in Section 5 and Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The presented research conforms with the Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Appendices A, B.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We curated a novel WikiTpoics-CLQA dataset for inductive complex query answering, the dataset creation process is described in Appendix A.2.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing was used for this paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.