# Prediction and Calibration: Complex Reasoning over Knowledge Graph with Bi-directional Directed Acyclic Graph Neural Network

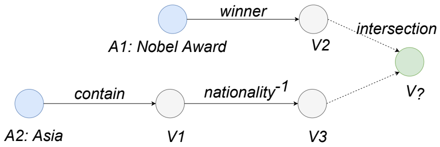
**Yao Xu[1,2], Shizhu He[1,2], Li Cai[3], Kang Liu[1,2], Jun Zhao[1,2]**
[1] The Laboratory of Cognition and Decision Intelligence for Complex Systems,
Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] Meituan, Beijing, China
{yao.xu, shizhu.he, kliu, jzhao}@nlpr.ia.ac.cn, caili03@meituan.com

## Abstract

Answering complex logical queries is a challenging task for knowledge graph (KG) reasoning. Recently, query embedding (QE) has been proposed to encode queries and entities into the same vector space, and obtain answers based on numerical computation. However, such models obtain the node representations of a query only based on its predecessor nodes, which ignore the information contained in successor nodes. In this paper, we proposed a Bi-directional Directed Acyclic Graph neural network (BiDAG) that splits the reasoning process into prediction and calibration. The joint probability of all nodes is considered by applying a graph neural network (GNN) to the query graph in the calibration process. By the prediction in the first layer and the calibration in deep layers of GNN, BiDAG can outperform previous QE based methods on FB15k, FB15k-237, and NELL995.

| Logical Query | Natural language Interpretation |
|---|---|
| $q_1 = V_?., \exists V1: Winner(NobelAward, V_?)$ $\wedge Contain(Aisa, V1) \wedge Nationalty(V_?, V1)$ | Who in an Asian country has won the Nobel Prize. |

(A) An example of CQA



(B) The corresponding computation graph

Figure 1: An Example and its corresponding computation graph of CQA.

## 1 Introduction

Knowledge Graphs (KGs) organize world knowledge as interlinked triples which describe entities and their relationships (Ji et al., 2020). Compared with link prediction (Rossi et al., 2021), answering logical queries (i.e., complex query answering, CQA (Wang et al., 2021), as shown in Figure 1 (A)) is a more challenging task because it needs to perform first-order logic (FOL) operators such as conjunction (∧), disjunction (∨), and negation (¬).

Recently, Query Embedding (QE) models (Hamilton et al., 2018; Ren et al., 2020) have been proposed to jointly encode logical queries and entities into the same vector space, and then retrieve answers (entities) based on the similarity scores.

Although QE models can obtain answers in linear time and implicitly reason over incomplete KGs by iteratively predicting the representation of intermediate and target nodes, such models obtain the representation of the current node only based on its predecessor nodes, which causes (1) The joint probability of all nodes in the query graph is ignored. Take the example in Figure 1, the probability distribution of node $V1$ will be more concentrated in *Japan* and *China* after considering node $A1$. (2) The information contained in successor nodes is ignored. As shown in Figure 1, the type of node $V1$ can only be *country* after considering the successor relation *nationality*.)

To address the above drawbacks, in this paper, we propose a novel QE based method called Bi-directional Directed Acyclic Graph neural network (BiDAG), which splits the reasoning process into the following two processes: (1) **Prediction** is used to obtain the initial representation of nodes by aggregating the information of predecessor nodes, which is similar to previous QE models. (2) **Calibration**. In this process, the original unidirectional query graph is extended to a bidirectional graph, then we apply GNN to the bidirectional graph. In this way, BiDAG can take the joint probability into account, as each node is continuously calibrated by information of its predecessor and successor nodes.

Our contributions can be summarized as follows: (1) We propose a framework that predicts first and then calibrates in CQA, which enables the model to take the joint probability of all nodes into account.

*Findings of the Association for Computational Linguistics: ACL 2023*, pages 7189–7198
July 9-14, 2023 ©2023 Association for Computational Linguistics

(2) We conducted experiments on three standard benchmarks, and show that calibration can improve model performance significantly. The source codes and data can be found on https://github.com/YaooXu/BiDAG.

## 2 Related Work

Modeling entity and query representations and logical operators are critical points of QE models. GQE (Hamilton et al., 2018) answers the conjunctive queries by representing queries and entities as points in Euclidean space. To represent queries with a large set of answer entities, Query2Box (Ren et al., 2020) utilized hyper-rectangles to encode queries. By converting union queries into Disjunctive Normal Form (DNF) (Davey and Priestley, 2002), Query2Box can handle arbitrary existential positive first-order (EPFO) queries (i.e., queries that include any set of $\wedge, \vee, \exists$). To further support the negation operator ($\neg$), BetaE (Ren and Leskovec, 2020) was proposed to support a full set of operations in FOL by encoding entities and queries into Beta distributions. MLPMix (Amayuelas et al., 2022) utilized MLP-mixer (Tolstikhin et al., 2021) to model logical operators. By encoding each query into multiple points in the vector space, Query2Particles (Bai et al., 2022) can retrieve a set of diverse answers from the embedding space. In this paper, we not only predict the intermediate and target node representations but also constantly calibrate them by modeling the joint probability of all nodes in the query graph.

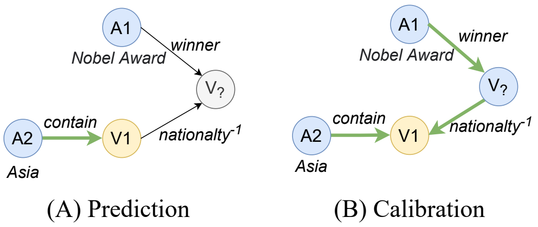
## 3 Preliminary

In this section, we formally describe the task of complex query answering over KGs. We denote a KG as $G = (\mathcal{V}, \mathcal{R})$, where $v \in \mathcal{V}$ represents an entity, and each $r \in \mathcal{R}$ represents a binary function as $r : \mathcal{V} \times \mathcal{V} \rightarrow \{0, 1\}$ which indicates whether a directed relationship $r$ exists between a pair of entities.

**First-order logic queries** The complex queries in KGs are described in logic form with first-order logic (FOL) operators such as existential quantification ($\exists$), conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$). A complex query $q$ consists of a set of anchor entities $V_a \subseteq \mathcal{V}$, some existential quantified variables $V_1, ...V_k$, and a single target variable $V_?$. The disjunctive normal form (DNF) of a FOL query $q$ is defined as follows:

$$q[V_?] = V_? : \exists V_1, ..., V_k : (e_{11} \wedge ... \wedge e_{1n_1}) \vee ... \vee (e_{m1} \wedge ... \wedge e_{mn_m}) \quad (1)$$

where each $e_{ij}$ represents a literal containing anchor node or variables, i.e., $e_{ij} = r(v_a, V') \, or \, r(V, V')$, where $v_a \in V_a, V \in \{V_1, ...V_k\}, V' \in \{V_?, V_1, ...V_k\}$. The goal of CQA is finding the answer set $S = \{v | v \in \mathcal{V}, q[v] = 1\}$.



(A) Prediction (B) Calibration

Figure 2: Illustration of Prediction and Calibration. Gray/blue nodes indicate nodes whose representations haven't/have been computed. Yellow nodes indicate nodes whose representations are computed or updated. Green lines indicate the flow of the message.

**Computation Graph** Each logical query can convert to a corresponding computation graph in the form of directed acyclic graph (DAG, as shown in Figure 1 (B)), where each node corresponds to an entity, and each edge corresponds to a logical operation. The logical operations are defined as follows.

(1) **Relation projection**: Given a set of entities $S \subseteq \mathcal{V}$ and a relation $r \in \mathcal{R}$, the relation projection will return entities $\cup_{v \in S} P_r(v)$ related to $v \in S$ via $r$, where $P_r(v) = \{v' \in \mathcal{V} : r(v, v') = 1\}$.
(2) **Intersection/union**: Given sets of entities $\{S_1, ..., S_n\}$, compute their intersection $\cap_{i=1}^{n} S_i$ or union $\cup_{i=1}^{n} S_i$.

It should be noticed that, in QE models, all these operations are executed in the embedding space. So, we can obtain the target node representation by iteratively computing the node representation following the neural logic operators in the DAG.

## 4 Bi-directional Directed Acyclic Graph Neural Network

The key idea of BiDAG is utilizing information of predecessor nodes to obtain the current node representation and then calibrating the representation with global information, as shown in Figure 2. Specifically, BiDAG includes two modules: 1)

**Representation prediction** module; 2) **Representation calibration** module. In the view of GNN, BiDAG can be regarded as the stack of one prediction module (the first layer) and multiple calibration modules (the deep layers).

### 4.1 Representation prediction

In this module, we define neural logic operations. We can obtain the representation of each node by applying logical operations based on the predecessor node representations.

**Projection** Given a node embedding $\boldsymbol{h}$ and an edge embedding $\boldsymbol{r}$, the projection operator $P$ outputs a new node embedding $\boldsymbol{h}' = P(\boldsymbol{h}, \boldsymbol{r})$. Compared with the geometric projection operator and multi-layer perceptron (MLP) used in the previous works (Hamilton et al., 2018; Ren and Leskovec, 2020), we use the gates mechanism to dynamically adjust the transformation of each node embedding under the specific relation, which is implemented by Gated Recurrent Units (GRU) (Cho et al., 2014): $\boldsymbol{h}' = GRU(\boldsymbol{r}, \boldsymbol{h})$, where $\boldsymbol{r}$, $\boldsymbol{h}$, and $\boldsymbol{h}'$ are treated as the input, past state, and updated state/output of a GRU.

**Intersection** We model the intersection of a set of query embeddings $\{\boldsymbol{q_1}, ..., \boldsymbol{q_n}\}$ as the weighted sum of them, which can be regarded as performing sets intersection in the embedding space. We implement it by adopting attention mechanisms:

$$\boldsymbol{q}_{inter} = \sum_i \alpha_i \cdot \boldsymbol{q}_i, \;\; \alpha_i = \frac{exp(MLP(\boldsymbol{q}_i))}{\sum_j exp(MLP(\boldsymbol{q}_j))} \quad (2)$$

where $\boldsymbol{q}_{inter}$ is the intersection of these query embeddings, $\alpha_i$ is the weight of query embedding $\boldsymbol{q}_i$, MLP is a multi-layer perceptron that takes $\boldsymbol{q}_i$ as input and outputs a single attention scalar.

**Union** Following Ren, Hu, and Leskovec (2020), we handle queries with union operators by transforming them into equivalent Disjunctive Normal Form (DNF). By doing so, the original query can be transformed into the union of $n$ conjunctive queries $\{q^1, ...., q^n\}$ that without union operator. Then, we can apply the existing methods to obtain the embeddings of these conjunctive queries as $\{\boldsymbol{q}^1, ...., \boldsymbol{q}^n\}$. The distance between a query $q$ and the answer entity $e$ is defined as:

$$d(q, e) = min(\{sim(\boldsymbol{q}^1, \boldsymbol{e}), ..., sim(\boldsymbol{q}^n, \boldsymbol{e})\}) \quad (3)$$

where $\{\boldsymbol{q}^1, ..., \boldsymbol{q}^n\}$ are the embeddings of these conjunctive queries, $\boldsymbol{e}$ is the embedding of entity $e$, $sim$ is a similarity function such as cosine function.

### 4.2 Representation calibration

In this module, the representation of each node is calibrated continuously by context information contained in the predecessor and successor nodes, which can address the drawback of ignoring the joint probability of all nodes.

Context information aggregating is completed by multi-head attention mechanism (Vaswani et al., 2017) in GNN, which is first introduced by GAT (Velickovic et al., 2018). Compared to the attention mechanism used in GAT which uses a shared linear transformation for all nodes. We make the following improvements: (1) We extend the graph attention mechanisms to handle directed relational graphs like KGs; (2) We introduce three weight matrices $\boldsymbol{Q} \in \mathcal{R}^{d\times d}, \boldsymbol{K}, \boldsymbol{V} \in \mathcal{R}^{d\times 2d}$ as query, key, and value matrix to enable the model to capture the higher-level information among neighbor nodes. (3) To enable the model to choose what to remain and update, we use GRU to update node representation in calibration, which is first used by (Li et al., 2017). (4) To avoid the calibrated representation being too different from the original representation, we adopt residual connection (He et al., 2015) to make adjustments to the original representation at each step. The representation for node $j$ at $(t+1)$-th calibration defined formally as follows (for simplicity, we only consider the single-head self-attention):

$$\mathbf{h}_j^{t+1} = \mathbf{h}_j^t + GRU(\sum_{i\in\mathcal{N}(j)} \alpha_{i,j}\mathbf{V}([\mathbf{h}_i^t \parallel \mathbf{e}_{i,j}]), \mathbf{h}_j^t), \quad (4)$$

$$\alpha_{i,j} = \frac{\exp\left(\mathrm{LeakyReLU}(w_{i,j})\right)}{\sum_{k\in\mathcal{N}(j)} \exp\left(\mathrm{LeakyReLU}(w_{i,k})\right)}, \quad (5)$$

$$w_{i,j} = \frac{(\mathbf{Q}\mathbf{h}_i^t)^T(\mathbf{K}[\mathbf{h}_j^t \parallel \mathbf{e}_{i,j}])}{\sqrt{d}}. \quad (6)$$

where $\parallel$ represents the concatenation operation, $\mathbf{h}_j^t$ is the representation for node $j$ after $t$-th calibration, $\mathbf{e}_{i,j}$ is the representation of edge from node $i$ to $j$, $\alpha_{i,j}$ is the attention coefficients, $\mathcal{N}(j)$ is the neighbor nodes of node $j$.

### 4.3 Model Training

Our objective is to minimize the distance between the query embedding and its answers while maximizing the distance between the query embedding and other random entities via negative sampling (Bordes et al., 2013), which we define as follows:

$$L = -log\,\sigma(\gamma - d(q, e)) - \sum_{j=1}^{k} \frac{1}{k} log\,\sigma(d(q, e_j) - \gamma) \quad (7)$$

|  | FB15k-237 |  |  |  |  |  |  |  |  |  | FB15k | NELL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **1p** | **2p** | **3p** | **2i** | **3i** | **pi** | **ip** | **2u** | **up** | **avg** | **avg** | **avg** |
| GQE | 35.0 | 7.2 | 5.3 | 23.3 | 34.6 | 16.5 | 10.7 | 8.2 | 5.7 | 16.3 | 28.0 | 18.6 |
| Q2B | 40.6 | 9.4 | 6.8 | 29.5 | 42.3 | 21.2 | 12.6 | 11.3 | 7.6 | 20.1 | 38.0 | 22.9 |
| BetaE | 39.0 | 10.9 | 10.0 | 28.8 | 42.5 | 22.4 | 12.6 | 12.4 | 9.9 | 20.9 | 41.6 | 24.6 |
| Q2P | 39.1 | 11.4 | 10.1 | 32.3 | 47.7 | 24.0 | 14.3 | 8.7 | 9.1 | 21.9 | 46.8 | 25.5 |
| MLPMix | 42.7 | 11.5 | 9.9 | 33.5 | 46.8 | **25.4** | 14.0 | 14.0 | 9.2 | 22.9 | 43.4 | 27.4 |
| **BiDAG (w/o res)** | 43.4 | **12.3** | 10.1 | 34.9 | 47.7 | 22.8 | 14.3 | 14.4 | 10.2 | 23.3 | 46.9 | 28.4 |
| **BiDAG (w/ res)** | **43.7** | 12.0 | **10.2** | **35.0** | **48.8** | 24.8 | **14.9** | **14.5** | **10.2** | **23.8** | **48.3** | **28.9** |

Table 1: The MRR results for existential positive first-order (EPFO) queries on different datasets. "w/o res" indicates “without the residual connection” while "w/ res" indicates “with the residual connection”. The full results are shown in Appendix C.

where $e_j$ represents a random negative sample, $\gamma$ represents the margin, $d(q, e)$ represents the distance between query q and entity e.

## 5 Experiment

### 5.1 Experimental Setup

**Datasets and Evaluation Protocol** We conduct experiments on three public KGs: FB15k (Bordes et al., 2013), FB15K-237 (Toutanova and Chen, 2015), and NELL995 (Xiong et al., 2017). For a fair comparison, we adopt the logical queries generated by Ren and Leskovec (2020) in model training and testing. In this paper, similar to Ren, Hu, and Leskovec (2020), we consider nine query types for evaluation. For these nine query types, we utilize the same evaluation protocol as Query2Box (Ren et al., 2020). Details about these datasets and query types can be found in Appendix A.

**Comparison with Baselines** First, we compare BiDAG with GQE, Q2B, BetaE, Q2P, and MLPMix on the EPFO queries (containing only $\wedge$, $\exists$, and $\vee$). The results are reported in Table 1. More details can be found in Appendix B.

From the table, we can find that: (1) BiDAG demonstrates an average relative improvement in Mean Reciprocal Rank (MRR) of 3.2%, 3.9%, and 5.4% over previous QE based models on the FB15k, FB15k-237, and NELL995 datasets, respectively. (2) Residual connection can improve model performance consistently on all datasets, which means residual connection is essential in the calibration process.

Even with the naive strategy that represents queries as point vectors like GQE, our BiDAG achieves a significant performance gain compared with all baselines. Furthermore, BiDAG also outperforms well on conjunctive queries (2i/3i). In our opinion, the main reason is that the target node has more processor nodes which will provide more information for calibrating. All these results demonstrate that calibration is helpful in complex query answering.

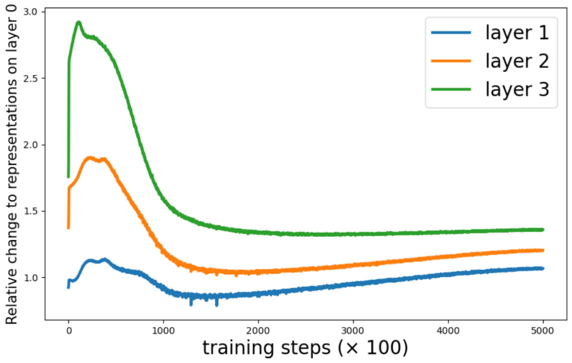


Figure 3: The curve of relative change of representations in each layer with different training steps.

**Ablation Study for BiDAG** To better demonstrate the effectiveness of bi-directional calibration (BC), we conduct further ablations studies by adopting different settings on FB15k. The experimental results are demonstrated in Table 2. From the table, we can find that compared to BiDAG-0BC (model without calibration), calibration can improve performance significantly. Besides, the significant improvement on multi-hop queries (2p/3p) demonstrates that calibration can also effectively alleviate the error cascading.

**Further study the effect of calibration** To further investigate how calibration affects the node representations in each layer, we record the relative change of the calibrated representations to the initial representations (layer-0 representations obtained by the prediction module), which is defined as follows:

$$c^t = \frac{\|\mathbf{h}^t_{tgt} - \mathbf{h}^0_{tgt}\|_2}{\|\mathbf{h}^0_{tgt}\|_2} \tag{8}$$

| Method | 1p | 2p | 3p | 2i | 3i | pi | ip | 2u | up | avg |
|---|---|---|---|---|---|---|---|---|---|---|
| BiDAG-0BC | 75.2 | 27.6 | 23.2 | 61.1 | 71.4 | 46.6 | 29.2 | 46.4 | 24.1 | 45.0 |
| BiDAG-1BC | 76.5 | 28.0 | 23.8 | 63.4 | 73.4 | 45.8 | 32.3 | 48.0 | 25.4 | 46.3 |
| BiDAG-2BC | 77.8 | 29.3 | 24.9 | 64.3 | 73.8 | 46.2 | 33.3 | 49.6 | 26.7 | 47.3 |
| BiDAG-3BC | **78.6** | **31.0** | **25.3** | **65.2** | **74.4** | **46.6** | **35.3** | **50.8** | **27.8** | **48.3** |

Table 2: Ablation studies of the BiDAG on FB15k. BC represents bi-directional calibration. (e.g. BiDAG-2BC indicates utilizing two bi-directional calibration layers).

where $\mathbf{h}_{tgt}^0$ is the initial representation for the target node, $\mathbf{h}_{tgt}^t$ is the representation for the target node after $t$-th calibration. The larger the $c^t$ value, the greater the difference between the $t$-th calibrated representation and the initial representation.

As shown in Figure 3, it can be founded that: (1) Throughout the training process, the relative change of final representations ($c^3$, the green line) increases initially and then decreases. This observation suggests that at the early stages of training, the initial representation is insufficiently accurate, so calibration mechanism changes representations a lot to get correct answers. However, as training progresses, the initial representations become increasingly precise, resulting in a relatively diminished influence of calibration later on. (2) In the middle and late stages of training, The values of $c^1$ (the blue line) and $c^2$ (the orange line) rise slowly, while $c^3$ remains stable. This observation implies that the first two calibration steps remain crucial even as the initial representations become increasingly accurate.

## 6 Conclusion

In this paper, we propose BiDAG, a query embedding method for answering complex queries over incomplete KGs. BiDAG splits the reasoning process into prediction and calibration. In the calibration process, the joint probability of all nodes is considered by applying GNN to the query graph that is extended to bidirectional message passing. The extensive experiments on multiple open datasets demonstrate that BiDAG outperforms previous QE based models and the effect of calibration in CQA.

## Limitations

There are three main limitations of our approach: (1) Our model cannot handle negation operation. Enabling BiDAG to support negation operation is a direction for future work. (2) The modeling for query representation and logical operators is too simple. Improving BiDAG by more ingenious modeling for query representation and logical operators is also a direction for future work. (3) The training process cannot be parallelized well, which is a common drawback of QE models, as QE models have to predict node representations one by one.

## Ethics Statement

This paper proposes a method for complex query answering in knowledge graph reasoning, and the experiments are conducted on public available datasets. As a result, there is no data privacy concern. Meanwhile, this paper does not involve human annotations, and there are no related ethical concerns.

## 7 Acknowledgment

This work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences (No.XDA27020100) and the National Natural Science Foundation of China (No.U1936207, No.61922085, No.61976211). This research work was supported by the Youth Innovation Promotion Association CAS, Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004).

## References

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

Jiaxin Bai, Zihao Wang, Hongming Zhang, and Yangqiu Song. 2022. Query2Particles: Knowledge Graph Reasoning with Particle Embeddings. Technical report.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on*

*Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Brian A Davey and Hilary A Priestley. 2002. *Introduction to lattices and order*. Cambridge university press.

William L. Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. 2018. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2030–2041.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. ArXiv:1512.03385 [cs].

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2020. A Survey on Knowledge Graphs: Representation, Acquisition and Applications. *ArXiv preprint*, abs/2002.00388.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2017. Gated Graph Sequence Neural Networks. ArXiv:1511.05493 [cs, stat].

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hongyu Ren and Jure Leskovec. 2020. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Andrea Rossi, Donatella Firmani, Antonio Matinata, Paolo Merialdo, and Denilson Barbosa. 2021. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Trans. Knowl. Discov. Data*, 15:14:1–14:49.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. 2021. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Zihao Wang, Hang Yin, and Yangqiu Song. 2021. Benchmarking the Combinatorial Generalizability of Complex Query Answering on Knowledge Graphs. *ArXiv preprint*, abs/2109.08925.

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

## A Data Details

The nine query types are shown in Figure 4. Specifically, there are five query types (1p/2p/3p/2i/3i) in the training set and also evaluated in a supervised manner, and the remaining four query types (2u/up/pi/ip) are evaluated in a zero-shot manner. Given the query type, a sample is generated by random walking on the KG. Datasets statistics are shown in Table 3.

## B Implement Details

To compare with baselines fairly, we set the same size of embedding vectors as 400. And we directly use the mean reciprocal rank (MRR) scores of these baselines reported by Ren and Leskovec (2020); Amayuelas, Zhang, Rao, and Zhang (2022); Bai, Wang, Zhang, and Song (2022).

In the comparison experiment with baseline, we used BiDAG-3BC for FB15k and FB15k-237, BiDAG-2BC for NELL. We tune the hyperparameters of BiDAG on the validation set for each dataset by grid search. We consider the batch size from {512, 1024, 2048}, learning rate from {2e-4, 3e-4,

|  | **Training** | **Validation** |  | **Test** |  |
|---|---|---|---|---|---|
| **Dataset** | 1p/2p/3p/2i/3i | 1p | others | 1p | others |
| FB15k | 273,710 | 59,097 | 8,000 | 67,016 | 8,000 |
| FB15k-237 | 149,689 | 20,101 | 5,000 | 22,812 | 5,000 |
| NELL | 107,982 | 16,927 | 4,000 | 17,034 | 4,000 |

Table 3: Number of training, validation, and test queries generated for different query types.

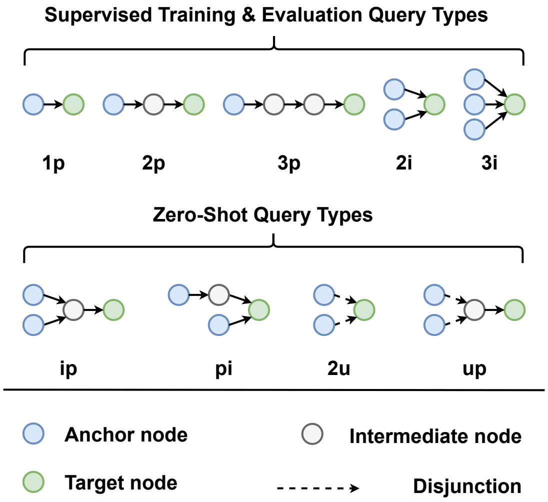


Figure 4: Query structures in this work, the operations contain projection (p), intersection (i), and union (u).

4e-4}. Our experiments are conducted on GTX 3090 with PyTorch 1.11, and the random seed is fixed for each experiment.

## C Full experimental results

The full results of Comparison with Baselines are shown in Table 4.

| **Dataset** | **Model** | **1p** | **2p** | **3p** | **2i** | **3i** | **pi** | **ip** | **2u** | **up** | **avg** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FB15k | GQE | 54.6 | 15.3 | 10.8 | 39.7 | 51.4 | 27.6 | 19.1 | 22.1 | 11.6 | 28.0 |
| | Q2B | 68.0 | 21.0 | 14.2 | 55.1 | 66.5 | 39.4 | 26.1 | 35.1 | 16.7 | 38.0 |
| | BetaE | 65.1 | 25.7 | 24.7 | 55.8 | 66.5 | 43.9 | 28.1 | 40.1 | 25.4 | 41.6 |
| | Q2P | **82.6** | 30.8 | **25.5** | 65.1 | **74.7** | **49.5** | 34.9 | 32.1 | 26.2 | 46.8 |
| | MLPMix | 69.7 | 27.7 | 23.9 | 58.7 | 69.9 | 46.7 | 30.8 | 38.2 | 24.8 | 43.4 |
| | **BiDAG (w/o res)** | 77.8 | 30.0 | 25.0 | 64.2 | 73.7 | 41.5 | 33.2 | 49.6 | 27.0 | 46.9 |
| | **BiDAG (w/ res)** | 78.6 | **31.0** | 25.3 | **65.2** | 74.4 | 46.7 | **35.3** | **50.8** | **27.8** | **48.3** |
| FB15k-237 | GQE | 35.0 | 7.2 | 5.3 | 23.3 | 34.6 | 16.5 | 10.7 | 8.2 | 5.7 | 16.3 |
| | Q2B | 40.6 | 9.4 | 6.8 | 29.5 | 42.3 | 21.2 | 12.6 | 11.3 | 7.6 | 20.1 |
| | BetaE | 39.0 | 10.9 | 10.0 | 28.8 | 42.5 | 22.4 | 12.6 | 12.4 | 9.9 | 20.9 |
| | Q2P | 39.1 | 11.4 | 10.1 | 32.3 | 47.7 | 24.0 | 14.3 | 8.7 | 9.1 | 21.9 |
| | MLPMix | 42.7 | 11.5 | 9.9 | 33.5 | 46.8 | **25.4** | 14.0 | 14.0 | 9.2 | 22.9 |
| | **BiDAG (w/o res)** | 43.3 | **12.3** | 10.1 | 34.9 | 47.7 | 22.8 | 14.3 | 14.4 | 10.2 | 23.3 |
| | **BiDAG (w/ res)** | **43.7** | 12.0 | **10.2** | **35.0** | **48.8** | 24.9 | **14.9** | **14.5** | **10.2** | **23.8** |
| NELL | GQE | 32.8 | 11.9 | 9.6 | 27.5 | 35.2 | 18.4 | 14.4 | 8.5 | 8.8 | 18.6 |
| | Q2B | 42.2 | 14.0 | 11.2 | 33.3 | 44.5 | 22.4 | 16.8 | 11.3 | 10.3 | 22.9 |
| | BetaE | 53.0 | 13.0 | 11.4 | 37.6 | 47.5 | 24.1 | 14.3 | 12.2 | 8.6 | 24.6 |
| | Q2P | 56.5 | 15.2 | 12.5 | 35.8 | 48.7 | 22.6 | 16.1 | 11.1 | 10.4 | 25.5 |
| | MLPMix | 55.4 | 16.2 | 13.9 | 39.5 | 51.0 | 25.7 | 18.3 | 14.7 | 11.2 | 27.4 |
| | **BiDAG (w/o res)** | 58.7 | 17.2 | 14.3 | 42.1 | 52.9 | 25.0 | 18.2 | 15.8 | 11.5 | 28.4 |
| | **BiDAG (w/ res)** | **59.0** | **17.5** | **14.5** | **42.3** | **53.0** | **26.7** | **18.9** | **16.1** | **11.8** | **28.9** |

Table 4: The MRR results for existential positive first-order (EPFO) queries on different datasets. "w/o res" indicates “without the residual connection” while "w/ res" indicates “with the residual connection”. Bold and underline indicate top-two results, respectively.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*In the limitation section.*

☑ A2. Did you discuss any potential risks of your work?
*In the Ethics Stateme.*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*In the abstract.*

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

### B ☑ Did you use or create scientific artifacts?

*In the Experiment section.*

☑ B1. Did you cite the creators of artifacts you used?
*In the Experiment section.*

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Not applicable. Left blank.*

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*In the Experiment section.*

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*Not applicable. Left blank.*

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*Not applicable. Left blank.*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*In the Appendix B.*

### C ☑ Did you run computational experiments?

*In the Experiment section.*

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

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

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*In the Appendix A.*

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

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

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

*Left blank.*

☒ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*Left blank.*

☒ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*Left blank.*

☒ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*Left blank.*

☒ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*Left blank.*

☒ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*Left blank.*