

# What's Wrong with Gradient-based Complex Query Answering?

Ouns El Harzli<sup>1,3</sup>, Samy Badreddine<sup>1,\*</sup> and Tarek R. Besold<sup>2</sup>

<sup>1</sup>*Sony AI Tokyo, Sony Group Corporation*

<sup>2</sup>*Sony AI Barcelona, Sony Group Corporation*

<sup>3</sup>*Department of Computer Science, University of Oxford*

## Abstract

Multi-hop query answering on knowledge graphs is known to be a challenging computational task. Neurosymbolic approaches using neural link predictors have shown promising results but are still outperformed by combinatorial optimization methods on several benchmarks, including the FB15k dataset. We analyze the task on the FB15k dataset and propose two new gradient-based methods, one learning simultaneously the representations of several candidate answers and the other learning a skolem function projecting to candidate answers instead of learning direct candidate representations. We implement both using Logic Tensor Networks. As part of this investigation we identified two important factors that limit the ability of differentiable methods to learn correct answers. The first factor is the (un)reliability of the pre-trained neural link predictors which biases the guesses of the query solver. To account for this, we suggest new evaluation metrics using satisfiability scores, that better reflect the true performance of neurosymbolic approaches on multi-hop query answering. The second factor is the regularization technique proposed in previous works, which limits the exploration of the gradient-based solver. Our results provide the foundation for future work mitigating these bottlenecks.

## Keywords

neurosymbolic AI, multi-hop querying, knowledge graphs, gradient-based solver

## 1. Introduction

Knowledge graphs (KGs) are graph-structured knowledge bases which enable representing knowledge as a set of relationships (binary predicates) between entities (variables), and constitute an important data structure with many use cases [1, 2, 3, 4]. Deep learning models have been proposed as neural link predictors to predict missing edges in incomplete KGs [5]. The predictions are typically based on high-dimensional, continuous embeddings of the entities and the relationships. A challenging computational problem in incomplete KGs is multi-hop query answering [6, 7] (see Figure 1). As any pair of nodes is a candidate link, computing all possible paths has intractable complexity in the number of hops. State-of-the-art results on several benchmarks generally require learning models that predict answers based on example sets of positive and negative answers to complex queries[8, 9].

---

*NeSy'23: 17th International Workshop on Neural-Symbolic Learning and Reasoning, July 03–05, 2023, Siena, Italy*

\*Corresponding author.

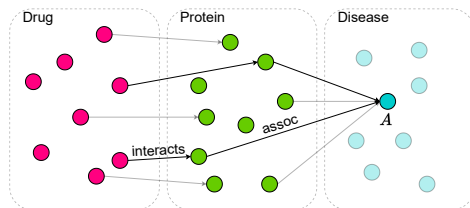
✉ [samy.baddredine@sony.com](mailto:samy.baddredine@sony.com) (S. Badreddine)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).



CEUR Workshop Proceedings (CEUR-WS.org)



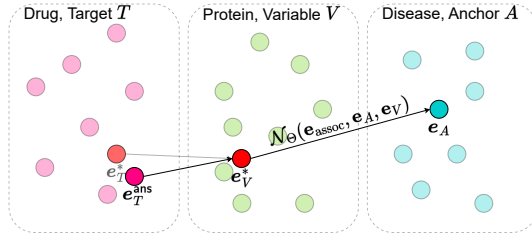
**Figure 1:** 2-hop query example "Which drugs interact with proteins associated with the disease A".

Instead, recent literature [10] has proposed to rely on pre-trained link predictors to answer queries without relying on new sets of examples. In particular, they couple link predictors with continuous relaxations of logical operators using fuzzy logic [11] to represent complex queries in fully differentiable representations and open the ground for differentiable methods to optimize for the objective. We focus our study on answering First-Order Logic queries as formulated by Arakelyan et al. [10]. In the simplest form, such queries have multiple *variable nodes*  $V_1, \dots, V_n$ , one *target node*  $T$  and one *anchor node*  $A$ . They can be written  $? T : \exists V_1, \dots, V_n R_0(A, V_1) \wedge R_1(V_1, V_2) \wedge \dots \wedge R_n(V_n, T)$ . In plain English, the objective is to decide whether there is an entity  $T$  which verifies that there exists  $n$  entities  $V_1, \dots, V_n$  such that entities  $A$  and  $V_1$  are linked by relation  $R_0$ , entities  $V_1$  and  $V_2$  are linked by relation  $R_1$ , etc and entities  $V_n$  and  $T$  are linked by relation  $R_n$ . Still, the best-performing approach in [10] is a hybrid method that is making use of neural link predictors but performs the search for candidates using a beam search heuristic. On 2-hop queries from the FB15K dataset [12], the discrepancy between the differentiable solution CQD-CO and the heuristic method CQD-Beam proposed by [10] is large. In this paper, we ask the question: what is holding fully differentiable methods back from always performing better in multi-hop querying?

As part of our investigation, we propose two new improvements of the fully differentiable approach. The first idea is to diversify the learned candidates by learning multiple embeddings simultaneously, which must all verify the query and then pick the one closest to the data. The second idea is to learn a skolem function [13], approximated by a neural network, which maps the target node to a corresponding variable node, instead of learning directly variable node embeddings. We have implemented both approaches using Logic Tensor Networks (LTNs) [14], a neurosymbolic framework based on fuzzy logic. Although the resulting improvements upon state-of-the-art turned out to be only marginal, our investigation allows us to identify two bottlenecks of which researchers and practitioners need to be acutely aware when using differentiable methods to perform complex querying.

First, even best-performing neural link predictors may hold some biases that make optimization in a continuous space difficult. We found, on the FB15k dataset, that the fully differentiable method CQD-CO is consistently finding candidate answers with high satisfiability—but since the satisfiability scores are so close to one another, this is not reflected in the Hits@3 score.<sup>1</sup> Computing satisfiability scores is therefore a better way to evaluate the performance of gradient-based querying, and reveals that the fully differentiable approach may perform better

<sup>1</sup>By satisfiability, we mean the fuzzy relaxation of the score of a complex query, which is calculated based on neural link predictors for the atomic queries.



**Figure 2:** CQD-CO [10]: (Learning) The system finds embedding candidates  $(\mathbf{e}_T^*, \mathbf{e}_V^*)$  (that do not correspond to real data) maximizing the satisfiability of the query, (Testing) Real target embeddings are ranked based on their satisfiability scores against the learned intermediary variable node  $\mathbf{e}_V^*$ .

with suitable neural link predictors and datasets. Secondly, in the fully differentiable approach, regularization has a huge impact on the distance to the data of the learned candidates, by inducing some constraints on the values taken by the candidate embeddings which counteracts the need for diversification. We also established that the regularization must be adapted to the formulation, e.g. existing regularizations are not adapted to the parameterized neural network in skolemization—thus resulting in learned candidates further from the data.

## 2. Gradient-based complex querying

2-hop queries from the FB15k dataset consist in a collection of 8000 queries. Each can be seen as a tuple  $(A, R_1, R_2)$ , where  $A$  is an anchor node and  $R_1, R_2$  are two relations to form queries  $? T : \exists V R_1(A, V) \wedge R_2(V, T)$ . An answer to that query is a target node  $T$  that satisfies the query. We can also define a full answer as a tuple  $(T, V)$  that satisfies  $R_1(A, V) \wedge R_2(V, T)$ . In the FB15k dataset, there are 2690 distinct relations and 14,951 distinct entities.

### 2.1. CQD-CO

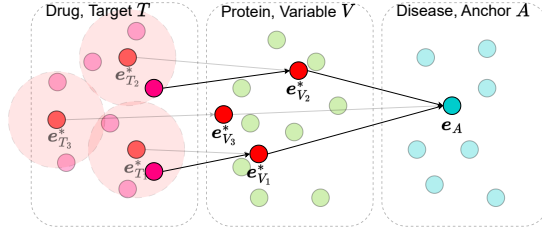
A neural link predictor  $\mathcal{N}_\Theta$  is pre-trained together with embeddings  $\mathbf{e}_V, \mathbf{e}_R \in \mathbb{R}^k$  for any entity  $V$  and any relation  $R$ .  $\mathcal{N}_\Theta(\mathbf{e}_R, \mathbf{e}_{V_1}, \mathbf{e}_{V_2}) \in [0, 1]$  provides a truth value for the atom  $R(V_1, V_2)$  for any relation  $R$  and any two entities  $V_1, V_2$ . Using these predictions, the *satisfiability score* to the full query is computed using a fuzzy t-norm [14] as a continuous generalization of the conjunction.

The CQD-CO method [10] proceeds in two phases which we denote as *learning* and *testing*.

**Learning** CQD-CO solves the following problem by gradient-based optimization:

$$(\mathbf{e}_T^*, \mathbf{e}_V^*) = \operatorname{argmax}_{\mathbf{e}_T, \mathbf{e}_V \in \mathbb{R}^k} \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_V) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_V, \mathbf{e}_T) \quad (1)$$

using the product t-norm  $\cdot$  [14] as the continuous generalization of conjunction in logic. The idea is to optimize for entity embeddings  $(\mathbf{e}_T^*, \mathbf{e}_V^*) \in \mathbb{R}^k$  such that the truth value (or satisfiability) is maximized. One can see that the expression is indeed fully differentiable. Also, the anchor node embedding, the relation embeddings and the neural link predictor are all pre-trained and fixed.



**Figure 3:** CQD-CO Multi: (Learning) Multiple answer paths are explored via  $P$  learned candidates ( $\mathbf{e}_{T_i}^*, \mathbf{e}_{V_i}^*$ ,  $i \in \{1 \dots P\}$ ) (Testing) Real targets are tested against a path decided based on their nearest neighbor. The theoretical advantage over CQD-CO is that up to  $P$  intermediate paths can be found.

**Testing** Then, CQD-CO fixes the variable node  $\mathbf{e}_V^*$  and tests all real candidate node embeddings  $\mathbf{e}_T$  to find the answer:

$$\mathbf{e}_T^{\text{ans}} = \operatorname{argmax}_{\mathbf{e}_T \in D} \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_V^*) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_V^*, \mathbf{e}_T) \quad (2)$$

where  $D$  denotes the dataset of existing node embeddings. The motivation is that, after optimization, there is no guarantee that  $\mathbf{e}_T^*$  has converged to a real data point. CQD-CO [10] proposes to query the satisfiability of the 14,951 entities with the fixed intermediate nodes and to rank their scores. The exhaustive query is acceptable because its complexity does not depend on the number of hops, and hence is tractable.

## 2.2. Proposed Improvements

In practical knowledge graphs, it is not uncommon to encounter multiple valid answers for a given query, as depicted in Figure 1. However, CQD-CO, upon completing its learning phase, only retains a single candidate path when querying in the testing phase. To address this limitation, we suggest two approaches that retain more candidate paths after the learning phase.

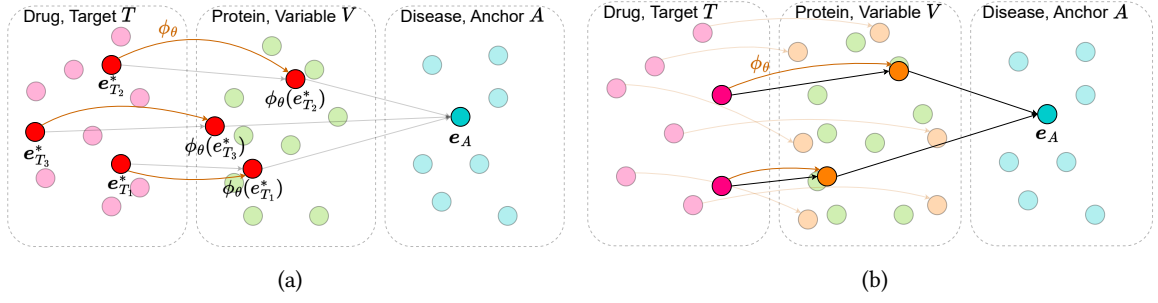
### 2.2.1. CQD-CO Multi: Learning multiple candidates

Our first approach learns multiple candidate paths after the learning phase and employs nearest neighbor strategies to select the optimal querying path during testing.

**Learning** To learn multiple candidates, we used, similarly to query parallelization, the mean-squared error MSE as a differentiable aggregator that approximates  $\forall$ . Notice however that other options such as the log-product aggregator are possible. Let  $P$  denote the number of candidates. The optimization problem yields  $(\mathbf{e}_{T_1}^*, \dots, \mathbf{e}_{T_P}^*, \mathbf{e}_{V_1}^*, \dots, \mathbf{e}_{V_P}^*)$  values:

$$\operatorname{argmax}_{\mathbf{e}_{T_1}^*, \dots, \mathbf{e}_{T_P}^*, \mathbf{e}_{V_1}^*, \dots, \mathbf{e}_{V_P}^* \in \mathbb{R}^k} \operatorname{MSE}_{i \in \{1, \dots, P\}} \left( \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_{V_j}^*) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_{V_j}^*, \mathbf{e}_{T_j}^*) \right) \quad (3)$$

**Testing** Given the embedding of an entity  $\mathbf{e}_T$ , let  $j$  be the index of its nearest neighbor within  $\mathbf{e}_{T_1}^*, \dots, \mathbf{e}_{T_P}^*$  (in terms of euclidean norm). Then, let the corresponding  $\mathbf{e}_{V_j}^*$  variable node be



**Figure 4:** CQD-CO Skolem: (a, Learning) A projection from target node to intermediate nodes in the context of the query is trained via  $P$  learned candidates  $(\mathbf{e}_{T_i}^*, \phi_\theta(\mathbf{e}_{T_i}^*))$ ,  $i \in \{1 \dots P\}$  (b, Testing) Real targets are tested using their own projections. The theoretical advantage over CQD-CO is that intermediate paths are not fixed.

denoted as  $\text{proj}_V^*(\mathbf{e}_T)$ . For each of the 14,951 entities, use these candidates to compute and rank satisfiability scores according to:

$$\mathbf{e}_T^{\text{ans}} = \underset{\mathbf{e}_T \in D}{\text{argmax}} \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \text{proj}_V^*(\mathbf{e}_T)) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \text{proj}_V^*(\mathbf{e}_T), \mathbf{e}_T) \quad (4)$$

### 2.2.2. CQD-CO Skolem: Non-fixed Intermediate Paths via Skolemization

Our second approach, which we name CQD-CO Skolem, aims to not fix any variable node during the testing phase of the candidates. We implement a dependence between the target node  $\mathbf{e}_T$  and the variable node  $\mathbf{e}_V$  in the form of a skolem function, materialized by a neural network  $\phi_\theta$ . Doing so, we leave potentially every answer path open (instead of limiting it to at most 1 path in CQD-CO or  $P$  paths in CDQ-CO Multi).

**Learning** We also have to learn multiple candidates at the same time, so that the neural network captures a data-dependent mapping between target nodes and variable nodes  $\phi_\theta(\mathbf{e}_T) = \mathbf{e}_V$  in the context of the given query. The optimization problem thus becomes:

$$\underset{\mathbf{e}_{T_1}, \dots, \mathbf{e}_{T_P}, \in \mathbb{R}^k, \theta \in \mathbb{R}^N}{\text{argmax}} \text{MSE}_{i \in \{1, \dots, P\}} \left( \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \phi_\theta(\mathbf{e}_{T_i})) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \phi_\theta(\mathbf{e}_{T_i}), \mathbf{e}_{T_i}) \right) \quad (5)$$

**Testing** After training, we fix  $\phi_\theta$  and use it to generate a variable node embedding for each of the 14,951 entities, which is then used to compute and rank satisfiability scores according to:

$$\mathbf{e}_T^{\text{ans}} = \underset{\mathbf{e}_T \in D}{\text{argmax}} \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \phi_\theta(\mathbf{e}_T)) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \phi_\theta(\mathbf{e}_T), \mathbf{e}_T) \quad (6)$$

## 3. Results

We reproduce CQD-CO [10] and our proposed improvements using the LTN framework [14]. For each method, the optimization phase can be parallelized on all 8000 queries using a mean-squared error aggregator. The number of queries optimized in parallel, which can be viewed as

**Table 1**

Hits@3 score of different training instances on FB15k. Notice that the number of learned candidates  $P$  and the query batch size must be balanced to obtain good results.

Method	$P$	Batch size	Hits@3
CQD-CO	1	100	30%
CQD-CO Multi	10	100	24%
CQD-CO Multi	10	10	32%
CQD-CO Multi	50	2	31.5%
CQD-CO Multi	100	1	32%
CQD-CO Skolem	10	100	1%
CQD-CO Skolem	10	10	2%
CQD-CO Skolem	50	2	2%
CQD-CO Skolem	100	1	2%

a *batch size*, has an influence on the performance. Indeed, too big batch sizes can result in a deterioration of the satisfiability for some of the queries, whereas a too small batch will favor the maximization of the satisfiability for each query, which can lead to overfitting.

CQD-CO proposes to include a differentiable regularization term  $\text{Reg}$  which is computed through canonical tensor decomposition [15]. The full optimization problem is:

$$(\mathbf{e}_T^*, \mathbf{e}_V^*) = \operatorname{argmax}_{\mathbf{e}_T, \mathbf{e}_V \in \mathbb{R}^k} \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_V) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_V, \mathbf{e}_T) - \text{Reg}(\mathbf{e}_T, \mathbf{e}_V) \quad (7)$$

For our proposed methods, we adapt it by summing over the candidates  $\sum_{i=1}^P \text{Reg}(\mathbf{e}_T, \mathbf{e}_{V_i})$ .

The results in terms of Hits@3 for varying numbers of learned candidates and batch sizes are reported in Table 1. The Hits@3 is the percentage of queries such that a correct answer is in the top 3 results of the testing phase in terms of satisfiability score. On FB15k, CQD-CO reaches a Hits@3 score of 30%. CQD-CO Multi, which diversifies the learned candidates by learning  $P$  paths at the same time, allows us to push the performance to a 32% Hits@3 score only. Upon further investigation, we find that the regularization method is largely limiting the diversity of the  $P$  paths. We discuss this in the next section. CQD-CO Skolem is rather unsuccessful, plateauing at a 2% Hits@3 score. While we believe that the idea is promising, since having a dependency between target and variable node is undoubtedly desirable, the trained neural network  $\phi_\theta$  projects real entity embeddings onto intermediary points that were further from the data. That is likely due to our decision to train with no more than 100 candidates per query, as training with too many examples ultimately defeats the purpose of reducing the complexity of answering multi-hop queries. We leave further exploration of this idea to future work.

## 4. Challenges and Discussions

### 4.1. Prominent effects of regularization

We performed an ablation study on the regularization term to understand its impact on the performance. Also, we performed a qualitative analysis of the L2 norms and euclidean distances to data of the learned candidates. Our findings were as follows: 1) with CQD-CO and regularization, learned candidates have L2 norms one order of magnitude smaller than real

**Table 2**

Hits@3 scores and satisfiability metric with and without canonical tensor decomposition (CTD) or L2 regularization, for different number of learned candidates  $P$  and batch sizes, on FB15k.

Method	Reg.	$P$	Batch size	Hits@3	Sat metric
CQD-CO	none	1	100	13%	18%
CQD-CO	CTD	1	100	30%	40%
CQD-CO Multi	none	10	10	15.5%	20%
CQD-CO Multi	none	100	1	16.5%	
CQD-CO Multi	L2	10	10	32%	
CQD-CO Multi	CTD	10	10	32%	45%
CQD-CO Skolem	none	10	10	2%	2%
CQD-CO Skolem	CTD	10	10	2%	2%

entity embeddings. Without regularization, the L2 norms of learned candidates are of the same order of that of the real entities; 2) with CQD-CO Multi and regularization, the distance of candidates to one another is several orders of magnitudes smaller than the distance to real entity embeddings. Without regularization, this effect is less important: learned candidates are further apart from one another, but distances to data are still much higher; and 3) with CQD-CO Skolem and regularization, learned candidates are one order of magnitude further away in euclidean distance from real data points than learned candidates by the previous method. Without regularization, the distances to data remain similar.

The ablation study reveals important differences in performance when removing regularization for both CQD-CO and our extension to learning multiple candidates. The effect is not observable with skolemization. Results are reported in Table 2. Noticing that the regularizer tends to push learned candidates towards points with small norms and close to one another, we implement a more explicit norm regularization, namely L2-regularization, and interestingly, it is possible to reproduce state-of-the-art results with the right choice of hyperparameter.

Our interpretation of these empirical observations is that the regularizer constrains the exploration of the search space onto embeddings with smaller norms, which also limits the ability to learn correct patterns. This could be one of the glass ceilings of performance by these methods. Yet, when no constraint is applied, nothing prevents overfitting from happening, which explains why performance deteriorates although the learned embeddings are closer to real embeddings. Skolemization is a different story: the search space is different and regularization fails to mitigate overfitting. The trained neural network  $\phi_\theta$  projects real embeddings onto points far away from the data, and the regularization term is not adapted to its parameterization which is why it has little to no effect. In our opinion, one would therefore need to come up with an entirely different regularization technique, adapted to this formulation, in order to harness the power of skolemization.

## 4.2. Satisfiability scores reveal biases in neural link predictors

In order to gain a finer understanding of what gradient-based complex querying is actually learning, we studied directly the satisfiability scores instead of computing only Hits@3. The idea was to evaluate the satisfiability scores of the candidate answers to 2-hop queries to see by how much they were off, in comparison with ground truth answers.

For each query  $(A, R_1, R_2)$ , we consider the target node  $T$  that was selected by each of the gradient-based complex querying methods through ranking of satisfiability scores. For each entity  $V$  amongst the 14,951 possible entities, we evaluate the satisfiability  $\mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_V) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_V, \mathbf{e}_T)$ , and we pick the entity  $V$  with the highest satisfiability to form the full answer  $(T, V)$ . Similarly, considering a ground truth answer  $T_g$ , we use the same procedure to form a full answer  $(T_g, V_g)$ . If the satisfiability of a full answer obtained by our gradient-based methods is higher or equal to the satisfiability of the full answer from ground truth, we classify the answer as correct; otherwise, we classify it as incorrect. This allows us to compute a percentage of correct answers that we call *satisfiability metric*. The results are reported in Table 2 and show that the performance of gradient-based methods is underestimated by the Hits@3 metric. In other words, in some cases, it is possible to find an answer  $(T, V)$  with a higher satisfiability than that of ground truth answers  $(T_g, V_g)$ , i.e.  $\mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_V) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_V, \mathbf{e}_T) > \mathcal{N}_\Theta(\mathbf{e}_{R_1}, \mathbf{e}_A, \mathbf{e}_{V_g}) \cdot \mathcal{N}_\Theta(\mathbf{e}_{R_2}, \mathbf{e}_{V_g}, \mathbf{e}_{T_g})$ , which reveals that the neural link predictor  $\mathcal{N}_\Theta$  may be biased.

We computed the typical difference between the highest satisfiability of a learned candidate answer and the highest satisfiability of a ground truth answer, when the learned answer is incorrect. For comparison, we computed the same typical difference between a random answer and a ground truth answer. Surprisingly, the difference with the random answer is already quite low ( $10^{-3}$ ). But the difference with our learned candidates is even lower ( $10^{-4}$ ) for our best-performing gradient-based method. There are two take-aways from these evaluations: 1) the neural link predictor creates a bias towards high values of satisfiability. For a given 2-hop query, for any entity chosen as the target node  $T$ , it is possible to find a corresponding variable node  $V$  which would yield a satisfiability close to the highest possible satisfiability, as much as a difference of  $10^{-3}$ ; 2) the answers returned by our best-performing gradient-based complex querying methods, although they may be incorrect, are not just any answers: they are answers with a satisfiability very close to that of ground truth answers, as much as a difference of  $10^{-4}$ .

Our analysis sheds light on inherent difficulties that gradient-based complex querying methods may encounter on some particular datasets. Here, the task is to find, amongst entities with satisfiability scores close to one another as much as  $10^{-3}$ , the entities with best satisfiability to a  $10^{-4}$  precision. In our opinion, this provides an explanation on why it is difficult to solve it with continuous optimization methods, which are inherently approximative. It comes as no surprise that a combinatorial optimization method which treats entities as discrete, no matter the difference between them, would perform better on this instance. Yet, our observations suggest that the gradient-based methods are learning what they should be learning and that some signal is definitely captured. We can infer that the performance of gradient-based methods could be dramatically better in configurations where the neural link predictors and the dataset are suitable, e.g. if the correct answers stick out in terms of satisfiability.

## 5. Conclusion

We have investigated the task of complex query answering on incomplete knowledge graphs, with one question in mind: why don't gradient-based methods always produce the best performance? Although our attempts to improve on the existing methods have achieved only marginal improvements on the FB15k dataset, our in-depth analysis of the behavior of these



methods in terms of distance to the data and satisfiability has given us ways to understand these results. We highlight the role of regularization in driving the learned candidates away or closer to the data and advocate that this should be carefully taken into account when using and designing these methods. Furthermore, we see that the Hits@3 metric may underestimate the performance of gradient-based complex querying in its inherent task, which is to generalize in terms of satisfiability. We also pinpoint a bottleneck that would perhaps need to be mitigated independently from the complex querying task: proximities in terms of satisfiability induced by the neural link predictor and the data. By demystifying in part the behavior of gradient-based complex querying, we hope that our work will serve as foundation for developing new solutions.

## References

- [1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, Dbpedia: A nucleus for a web of open data, in: K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, P. Cudré-Mauroux (Eds.), *The Semantic Web*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 722–735.
- [2] F. M. Suchanek, G. Kasneci, G. Weikum, Yago: A core of semantic knowledge, in: *Proceedings of the 16th International Conference on World Wide Web, WWW '07*, Association for Computing Machinery, New York, NY, USA, 2007, p. 697–706. URL: <https://doi.org/10.1145/1242572.1242667>. doi:10.1145/1242572.1242667.
- [3] M. Dumontier, A. Callahan, A. Cruz-Toledo, P. Ansell, V. Emonet, F. Belleau, A. Droit, Bio2rdf release 3: A larger, more connected network of linked data for the life sciences, in: *International Semantic Web Conference, CEUR Workshop Proceedings*, volume 1272, 2014, p. 401–404.
- [4] N. F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, J. Taylor, Industry-scale knowledge graphs: lessons and challenges, in: *Communications of the ACM*, volume 62, 2019, p. 36–43.
- [5] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, *Proceedings of the IEEE* 104 (2016) 11–33. doi:10.1109/JPROC.2015.2483592.
- [6] H. Ren, J. Leskovec, Beta embeddings for multi-hop logical reasoning in knowledge graphs, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), *Advances in Neural Information Processing Systems*, volume 33, Curran Associates, Inc., 2020, pp. 19716–19726. URL: <https://proceedings.neurips.cc/paper/2020/file/e43739bba7cdb577e9e3e4e42447f5a5-Paper.pdf>.
- [7] H. Ren\*, W. Hu\*, J. Leskovec, Query2box: Reasoning over knowledge graphs in vector space using box embeddings, in: *International Conference on Learning Representations*, 2020. URL: <https://openreview.net/forum?id=BJgr4kSFDS>.
- [8] X. Chen, Z. Hu, Y. Sun, Fuzzy logic based logical query answering on knowledge graphs, *Proceedings of the AAAI Conference on Artificial Intelligence* 36 (2022) 3939–3948. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/20310>. doi:10.1609/aaai.v36i4.20310.
- [9] Z. Zhu, M. Galkin, Z. Zhang, J. Tang, Neural-symbolic models for logical queries on knowledge graphs, in: K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, S. Sabato

- (Eds.), Proceedings of the 39th International Conference on Machine Learning, volume 162 of *Proceedings of Machine Learning Research*, PMLR, 2022, pp. 27454–27478. URL: <https://proceedings.mlr.press/v162/zhu22c.html>.
- [10] E. Arakelyan, D. Daza, P. Minervini, M. Cochez, Complex query answering with neural link predictors, in: International Conference on Learning Representations, 2021. URL: <https://openreview.net/forum?id=Mos9F9kDwkz>.
- [11] E. van Krieken, E. Acar, F. van Harmelen, Analyzing differentiable fuzzy logic operators, *Artificial Intelligence* 302 (2022) 103602. URL: <https://www.sciencedirect.com/science/article/pii/S0004370221001533>. doi:<https://doi.org/10.1016/j.artint.2021.103602>.
- [12] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K. Weinberger (Eds.), *Advances in Neural Information Processing Systems*, volume 26, Curran Associates, Inc., 2013. URL: <https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf>.
- [13] *A Shorter Model Theory*, Cambridge University Press, 1997.
- [14] S. Badreddine, A. d’Avila Garcez, L. Serafini, M. Spranger, Logic tensor networks, *Artificial Intelligence* 303 (2022) 103649. URL: <https://www.sciencedirect.com/science/article/pii/S0004370221002009>. doi:<https://doi.org/10.1016/j.artint.2021.103649>.
- [15] T. Lacroix, N. Usunier, G. Obozinski, Canonical tensor decomposition for knowledge base completion, in: J. Dy, A. Krause (Eds.), Proceedings of the 35th International Conference on Machine Learning, volume 80 of *Proceedings of Machine Learning Research*, PMLR, 2018, pp. 2863–2872. URL: <https://proceedings.mlr.press/v80/lacroix18a.html>.