ReliK: A Reliability Measure for Knowledge Graph Embeddings [Extended Abstract]

Maximilian K. Egger[1], Wenyue Ma[1], Davide Mottin[1,*], Panagiotis Karras[1,2], Ilaria Bordino[3], Francesco Gullo[3] and Aris Anagnostopoulos[4]

[1]Aarhus University, Department of Computer Science, Aabogade 34, 8200 Aarhus, Denmark
[2] Copenhaghen University, Department of Computer Science, Universitetsparken 1, 2100 København, Denmark
[3] Unicredit, Via Molfetta 101, 00171 Rome, Italy
[4] Dept of Computer, Control, and Management Eng., Sapienza University of Rome, Via Ariosto 25, 00185 Rome, Italy

Abstract

Can we assess a priori how well a knowledge graph embedding will perform on a specific downstream task and in a specific part of the knowledge graph? Knowledge graph embeddings (KGEs) represent entities and relationships of a knowledge graph (KG) as vectors. KGEs are generated by optimizing an *embedding score*, which assesses whether a triple exists in the graph. KGEs have been proven effective in a variety of downstream tasks, including, for instance, predicting relationships among entities. However, the problem of anticipating the performance of a given KGE in a certain downstream task and locally to a specific individual triple, has not been tackled so far. In this paper, we fill this gap with *ReliK*, a **Reli**ability measure for **K**GEs. *ReliK* relies solely on KGE embedding scores, is task- and KGE-agnostic, and requires no KGE retraining. Through extensive experiments, we attest that *ReliK* correlates well with both common downstream tasks, such as tail or relation prediction and triple classification, and advanced downstream tasks, such as rule mining and question answering, while preserving locality.

Keywords

Knowledge Graph Embeddings, Reliability, Knowledge Graphs

1. Introduction

Knowledge graphs (KGs) are sets of *facts* (i.e., *triples* such as "da Vinci," "painted," "Mona Lisa") that interconnect *entities* ("da Vinci," "Mona Lisa") via *relationships* ("painted") [1, 2]. Entities and relationships correspond to nodes and (labeled) edges of the KG, respectively. *Knowledge graph embeddings* (KGEs) [3] are popular techniques to generate a vector representation for entities and relationships of a KG. A KGE is computed by optimizing a *scoring function* that provides an *embedding score* as an indication of whether a triple actually exists in the KG. KGEs have been extensively used as a crucial building block of state-of-the-art methods for a variety of *downstream tasks* commonly carried out on the Web, such as knowledge completion [4], whereby a classifier is trained on the embeddings to predict the existence of a triple; or head/tail prediction [5], which aims to predict entities of a triple, as well as more advanced ones, including rule mining [6], query answering [7], and entity alignment [8, 9, 10, 11].

SEBD 2024: 32nd Symposium on Advanced Database System, June 23-26, 2024 - Villasimius, Sardinia, Italy
maximilian.egger@cs.au.dk (M. K. Egger); wenyuema@cs.au.dk (W. Ma); davide@cs.au.dk (D. Mottin); piekarras@gmail.com (P. Karras); ibordino@acm.org (I. Bordino); gullof@acm.org (F. Gullo); aris@diag.uniroma1.it (A. Anagnostopoulos)

Motivation. So far, the choice of an appropriate KGE method has depended on the downstream task, the characteristics of the input KG, and the computational resources. The existence of many different scoring functions, including linear embeddings [12], bilinear [6], based on complex numbers [13], or projections [14] further complicates this choice. Alas, the literature lacks a unified measure to quantify how *reliable* the performance of a KGE method can be for a certain task *beforehand*, without performing such a potentially slow task. Furthermore, KGE performance on a specific downstream task is typically assessed in a *global* way, that is, in terms of how accurate a KGE method is for that task on *the entire KG*. However, the performance of KGEs for several practical applications (e.g., knowledge completion [4]) typically varies across the parts of the KG. This requires carrying out a performance assessment of KGE *locally* to specific parts of the KG, rather than globally.

Contributions. We address all the above shortages of the state of the art in KGE and introduce *ReliK* (**Reli**ability for **K**GEs), a simple, yet principled measure that quantifies the *reliability* of how a KGE will perform on a certain downstream task in a specific part of the KG, without executing that task or (re)training that KGE. To the best of our knowledge, no measure like *ReliK* exists in the literature. *ReliK* relies exclusively on embedding scores as a black box, particularly on the ranking determined by those scores (rather than the scores themselves). *ReliK* is simple, intuitive, and easy-to-implement. Despite that, its exact computation requires processing all the possible combinations of entities and relationships, for every single fact of interest. We address this major challenge by devising an approximation to *ReliK*.

Apart from experimenting with *ReliK* in basic downstream tasks, such as entity/relation prediction, we also showcase *ReliK* on two advanced downstream tasks, *query answering*, which finds answers to complex logical queries over KGs, and *rule mining*, deduces logic rules, with the purpose of cleaning the KG from spurious facts or expanding the information therein.

2. Preliminaries

A *knowledge graph* (KG) $\mathcal{K} : \langle \mathcal{E}, \mathcal{R}, \mathcal{F} \rangle$ is a triple consisting of a set $\mathcal{E}$ of n entities, a set $\mathcal{R}$ of relationships, and a set $\mathcal{F} \subset \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ of m facts. A *fact* is a *triple* $x_{hrt} = (h, r, t)$, where $h \in \mathcal{E}$ is the *head*, $t \in \mathcal{E}$ is the *tail*, and $r \in \mathcal{R}$ is the *relationship*. For instance, entities "Leonardo da Vinci" and "Mona Lisa," and relationship "painted" form the triple ("Leonardo da Vinci," "painted," "Mona Lisa"). The set $\mathcal{F}$ of facts form an edge-labeled graph whose nodes and labeled edges correspond to entities and relationships, respectively. We say a triple x_{hrt} is *positive* if it actually exists in the KG (i.e., $x_{hrt} \in \mathcal{F}$), *negative* otherwise (i.e., $x_{hrt} \notin \mathcal{F}$).

Knowledge graph embedding. A *KG embedding* (KGE) [3, 5, 15] is a representation of entities and relationships in a d-dimensional ($d \ll |\mathcal{E}|$) space, typically, the real $\mathbb{R}^d$ space or the complex $\mathbb{C}^d$ space. For instance, TransE [12] represents a triple x_{hrt} as entity vectors $\mathbf{e}_h, \mathbf{e}_t \in \mathbb{R}^d$ and relation vector $\mathbf{e}_r \in \mathbb{R}^d$, and DistMult [6] represents the relationship as a matrix $\mathbf{W}_r \in \mathbb{R}^{d \times d}$. Although KGEs can differ (significantly) from one another in their definition, a common key aspect of all KGEs is that they are typically defined based on a so-called *embedding scoring function* or simply *embedding score*. This is a function $s \colon \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to \mathbb{R}$, which quantifies how likely a triple $x_{hrt} \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ exists in $\mathcal{K}$ based on the embeddings of its head (h),

relationship (r), and tail (t). Specifically, the higher $s(x_{hrt})$, the more likely the existence of x_{hrt}. For instance, TransE's score $s(x_{hrt}) = -\|\mathbf{e}_h + \mathbf{e}_r - \mathbf{e}_t\|$ is the (ℓ_1 or ℓ_2) distance between the "translation" from h's embedding to t's embedding through r's embedding [12].

KGEs are typically learned via optimization (e.g., gradient descent) of a loss function defined based on the embedding score. This training process can be computationally expensive, especially if it has to be repeated for multiple KGEs. KGEs learned this way are shown to be effective for a number of downstream tasks [5], such as predicting the existence of a triple, but do not offer any prior indication on their performance [16]. Moreover, existing benchmarks [15] show global performance on the entire graph rather than *local* on subgraphs. To this end, in this work, we provide an answer to the following key question: *Is there a measure that provides a prior indication on the performance of an embedding on a specific subgraph?*

3. KGE Reliability

We would like a measure of *reliability* that properly assesses how the embedding of a triple would perform on certain tasks and data, without knowing them in advance. To this end, we aim to fulfill the following desiderata for a KGE reliability measure.

(R1) Embedding-agnostic. Is independent of the specific KGE method. This is to have a measure fully general. **(R2) Learning-free.** Requires no further KGE training. This is primarily motivated by efficiency, but also for other reasons, such as privacy or unavailability of the data used for KGE training. **(R3) Task-agnostic.** Is independent of the specific downstream task. by anticipating the performance of a KGE in general, for *any* downstream task. **(R4) Locality.** Is a good predictor of KGE performance *locally* to a given triple, that is, in a close surrounding neighborhood of that triple. This is important, as a KGE model may be more or less effective based on the different parts of the KG it is applied to.

3.1. The Proposed *ReliK* Measure

Design principles. To fulfill **(R1)** and **(R2)**, the KGE reliability measure should not engage with the internals of the computation of KGEs. Thus, we need to treat the embeddings as vectors and the embedding score as a black-box function that provides only an indication of the actual existence of a triple. Although the absolute embedding scores are incomparable to one another, we observe that the distribution of positive and negative triples is significantly different. Specifically, we assume the *relative ranking* of a positive triple to be higher than that of a negative. Otherwise, we multiply the score by -1. This leads to the following main observation.

Observation 1. *A KGE reliability measure that uses the position of a triple relative to other triples via a ranking defined based on the embedding score fulfills **(R1)** and **(R2)**.*

Furthermore, comparing a triple to all other (positive or negative) triples might be ineffective. This is because the absolute score does not provide an indication of performance. We thus advocate that a local approach that considers triples relative to a neighborhood is more appropriate, and propose a measure that fulfills **(R4)**. The soundness of **(R4)** is better attested in our

experiments in Section 4. To meet **(R3)**, the KGE reliability measure should not exploit any peculiarity of a downstream task in its definition. Indeed, this is accomplished by our measure, as we show next.

Definition. For a triple $x_{hrt} = (h, r, t)$ we compute the neighbor set $\mathcal{N}^-(h)$ of all possible negative triples, that is, triples with head h that do not exist in $\mathcal{K}$. Similarly, we compute $\mathcal{N}^-(t)$ for tail t. We define the *head-rank* h of a triple x_{hrt} as the position of the triple in the rank obtained using score s for a specific KGE relative to all the negative triples having head h.

$$rank_H(x_{hrt}) = \left|\{x \in \mathcal{N}^-(h) : s(x) > s(x_{hrt})\}\right| + 1$$

The *tail-rank* $rank_T(x_{hrt})$ for tail t is defined similarly.

Our reliability measure, *ReliK*, for a triple x_{hrt} is ultimately defined as the average of the reciprocal of the head- and tail-rank

$$\textit{ReliK}(x_{hrt}) = \frac{1}{2}\left(\frac{1}{rank_H(x_{hrt})} + \frac{1}{rank_T(x_{hrt})}\right). \tag{1}$$

We define the *ReliK* score of a set $S \subseteq \mathcal{F}$ of triples as the average *ReliK* of the triples in the set.

Rationale. *ReliK* ranges from $(0, 1]$, with higher values corresponding to better reliability. In fact, the lower the head-rank $rank_H(x_{hrt})$ or tail-rank $rank_T(x_{hrt})$, the better the ranking of x_{hrt} induced by the underlying embedding scores, relatively to the nonexisting triples in x_{hrt}'s neighborhood, complies with the actual existence of x_{hrt} in the KG.

It is easy to see that *ReliK* achieves **(R1)** and **(R2)** by relying on the relative ranking rather than the absolute scores. It also fulfills **(R3)** as it involves no downstream tasks at all, and **(R4)** as it is based on the local (i.e., 1-hop) neighborhood of a target triple.

3.2. Efficiently Computing *ReliK*

Computing *ReliK* (Eq. (1)) requires $\Omega\left(|\mathcal{E}| \cdot |\mathcal{R}|\right)$ time, as it needs to scan the entire negative neighborhood of the target triple. For large KGs, repeating this for a (relatively) high number of triples may be computationally too heavy. For this purpose, here we focus on approximate versions of *ReliK*, which properly trade off between accuracy and efficiency.

The main intuition behind the *ReliK* approximation is that the precise ranking of the various potential triples is not actually needed. Rather, what it matters is *just the number* of those triples that exhibit a higher embedding score than the target triple. As such, we estimate *ReliK* by evaluating the fraction of triples in the sample that have a higher score than the triple under consideration and then scaling this fraction to the total number of negative triples.

Let S_H be a random subset of k elements selected without replacement independently and uniformly at random from the negative neighborhood $\mathcal{N}^-(h)$ of the head h of a triple x_{hrt}. The size $|S_H|$ trades off between efficiency and accuracy of the estimator, and it may be defined based on the size of $\mathcal{N}^-(h)$. Define also $rank^S_H(x_{hrt}) = |\{x \in S_H : s(x) > s(x_{hrt})\}| + 1$, to be the rank of the score $s(x_{hrt})$ that the KGE assigns to x_{hrt}, among all the triples *in the sample*. We similarly compute S_T and $rank^S_T$ for tail's neighborhood $\mathcal{N}^-(t)$.

We define our estimator as

$$\mathit{ReliK}_{\mathit{Apx}} = \frac{1}{2}\left[\left(rank_H^S(x_{hrt})\frac{|\mathcal{N}^-(h)|}{|S_H|}\right)^{-1} + \left(rank_T^S(x_{hrt})\frac{|\mathcal{N}^-(t)|}{|S_T|}\right)^{-1}\right]. \quad (2)$$

In words, we simply scale up the rank induced by the sample to the entire set of negative triples.

Theorem 1. *Equation 2 is an upper bound for ReliK; that is* $\mathbb{E}[\mathit{ReliK}_{\mathit{Apx}}(x_{hrt})] \geq \mathit{ReliK}(x_{hrt})$.

Theorem 1 follows immediately from the Jensen's inequality [17] and the fact that $\mathbb{E}[rank_H^S(x_{hrt})] = |S_H| \cdot \frac{rank_H(x_{hrt})}{|\mathcal{N}^-(h)|}$.

Algorithm. To compute $\mathit{ReliK}_{\mathit{Apx}}$, we first sample, uniformly at random, k negative triples from the head neighborhood and the tail neighborhood. We can save computation time by first filtering the triples in $S_H \cup S_T$ by score, that is, by considering only those with score higher than the input triple x_{hrt}, and then checking whether a triple in $S_H \cup S_T$ has either the head or the tail in common with x_{hrt} to update the corresponding rank.

Time complexity. $\mathit{ReliK}_{\mathit{Apx}}$ runs in $\mathcal{O}(k)$ time to sample the negative triples for each positive triple.The sample size k trades off between accuracy and efficiency of the estimation.

4. Experimental evaluation

We evaluate *ReliK* on four downstream tasks, six embeddings, and four datasets.

Embeddings. We include six established KGE methods, TransE [12], DistMult [6], RotatE [13], PairRE [14], ComplEx [18], ConvE [19] representative of the major embedding families (Sec. 5).

Datasets. We perform experiments on KGs with different characteristics: **Countries** [20] is a small graph from geographical locations; **FB15k237** [21] is a sample of Freebase KG [22] with 14k nodes, 310k facts, and 237 relationships; **Codex** [23] is a collection of three datasets of incremental size, Codex-S ($2k$ entities, $36k$ triples), Codex-M ($17k$ entities, $200k$ facts), and Codex-L ($78k$ entities, $610k$ facts) extracted from Wikidata and Wikipedia.; **YAGO2** [24] is a KG automatically extracted from Wikidata, which comprises 834k entities and 948k facts. .

Experimental setup. We implement our approximate and exact *ReliK* in Python v3.9.13 (Code at: https://github.com/AU-DIS/ReliK). We train the embedding with Pykeen v1.10.1 [15], with default parameters besides the embedding dimension $dim = 50$ and training loop sLCWA. We report an average of 5 experiments using 5-fold cross validation with 80-10-10 split.

4.1. Common Downstream Tasks

We test *ReliK* on the ability to anticipate the results of common tasks for KGEs [5, 3]. We measure the statistical significance of Pearson correlation among two ranking tasks, tail and relation prediction, and the triple classification task. To evaluate *ReliK* on different areas of the graph and different graph topologies, we sample random subgraphs of Codex-S with 60 nodes by initially selecting a starting node uniformly at random and then including nodes and edges by a random walk with restart [25] with restart probability $1 - \alpha = 0.2$, until the subgraph comprises 60 nodes. For Codex-M and Codex-L we use size 100 and for FB15k237 we use 200 nodes. We report the average *ReliK* on 100 random subgraphs.

Ranking tasks (T1). In the first experiments, we measure the Pearson correlation between *ReliK* and the performance on ranking tasks with mean reciprocal rank (MRR) [26]. The first

task, *tail prediction* [12, 14, 13], assesses the ability of the embedding to predict the tail given the head and the relation, thus answering the query $(h, r, ?)$ where the tail is unknown. The second task, *relation prediction*, assesses the ability of the embedding to predict the undisclosed relation of a triple $(h, ?, t)$. The common measure used for tail and relation prediction is MRR, which provides an indication of how close to the top the score ranks the correct tail (or relation). Consistently with previous approaches [12, 14, 13], we employ the filtered approach in which we consider for evaluation only negative triples that do not appear in either the train, test, or validation set. Table 1 reports the correlations alongside the statistical significance in terms of the p-value. We marked in red, high p-values (> 0.05), which suggest no correlation, and Pearson score values that indicate inverse correlation. Generally, *ReliK* exhibits significant correlation across embeddings and tasks. Noteworthy, even though *ReliK* (see Eq. (1)) does not explicitly target tail or head rankings by including both, we observe significant correlation on tail prediction in most embeddings and datasets. Because of the considerable training time, we only report results for RotatE on Codex-S. Comparing the actual results of the various tasks, it is also clear in most cases in which we do not have correlation that the results are too close to distinguish; for example, ComplEx having only results close to 0. In such cases, the results indicate that the particular embedding method needs additional training.

		Tail (MRR)		**Relation (MRR)**		**Classific. (Acc.)**	
	KGE	Pearson	p-value	Pearson	p-value	Pearson	p-value
Codex-L	TransE	0.83	$1.13e^{-26}$	0.97	$3.812e^{-64}$	0.63	$2.54e^{-12}$
	DistMult	0.49	$2.10e^{-07}$	0.78	$4.68e^{-22}$	0.60	$3.74e^{-11}$
	RotatE	–	–	–	–	–	–
	PairRE	-0.04	0.68	0.95	$3.33e^{-52}$	$-4.47e^{-4}$	0.99
	ComplEx	0.82	$1.03e^{-25}$	0.91	$3.96e^{-39}$	0.06	0.57
	ConvE	0.59	$4.26e^{-11}$	-0.07	0.48	0.31	$1.57e^{-3}$
FB15k237	TransE	0.24	0.02	0.86	$2.83e^{-30}$	0.34	$5.79e^{-4}$
	DistMult	-0.05	0.65	0.64	$5.57e^{-13}$	0.39	$5.58e^{-05}$
	RotatE	–	–	–	–	–	–
	PairRE	0.80	$1.51e^{-23}$	0.65	$1.74e^{-13}$	0.08	0.44
	ComplEx	0.20	0.05	0.88	$3.53e^{-34}$	0.14	0.18
	ConvE	0.09	0.37	0.85	$4.47e^{-30}$	0.01	0.93

Table 1
Pearson correlation and statistical significance of *ReliK* for tail prediction, relation prediction, and triple classification; Results for Codex-M and Codex-S are similar to those of Codex-L.

Classification task (T2). In this experiment, we test the correlation between *ReliK* and the accuracy of a threshold-based classifier on the embeddings. The classifier predicts the presence of a triple in the KG if the embedding score is larger than a threshold, a common scenario for link prediction [5]. Table 1 (right column) reports the correlations and their significance for all datasets. At close inspection, we observe that in cases of unclear correlation, such as with PairRE, the respective classification results are too close to observe a difference. Those cases notwithstanding, *ReliK* is significantly correlated with accuracy. This result confirms that *ReliK* can serve as a proxy for the quality of complex models trained on embeddings.

Tuning Subgraph Size. Next, we analyze how *ReliK* correlates with the tasks presented in Section 4.1 on subgraphs of varying size with the TransE embedding. Figure 1 reports the correlation values for all three tasks, only including those values where the p-value is below 0.05. We observe that *ReliK*'s correlation generally increases with subgraphs of up to 100 nodes on Codex-S. After that point, we note an unstable behavior in all tasks. This is consistent with the assumption that *ReliK* is a measure capturing local reliability. To strike a balance between quality and time we test on subgraphs with 60 nodes for Codex-S in all experiments. Yet, as tasks are of different nature, the subgraph size can be tuned in accordance with the task to provide more accurate results.

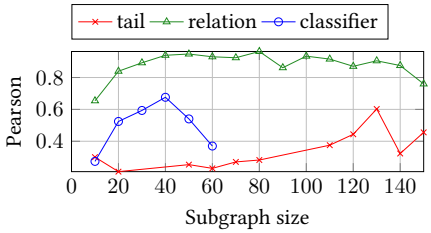

Figure 1: Correlation vs subgraph size on Codex-S.

4.2. Complex Downstream Tasks

Query answering (T3). We show how *ReliK* can improve query-answering tasks. Complex logical queries on KGs are working with different query structures. We focus on queries of chaining multiple predictions or having an intersection of predictions, from different query structures that have been described in recent work [27, 28]. We keep the naming convention introduced by Ren and Leskovec [27]. We evaluate a selection of 1000 queries per type ($1p$,$2p$,$3p$,$2i$,$3i$) from their data on the FB15k237 graph.The queries of type p are 1 to 3 hops from a given entity with fixed relation labels that point to a solution, whereas queries of type i are the intersection of 2 or 3 predictions pointing towards the same entity. We evaluate *ReliK* on the ability to detect whether an instance of an answer is true or false. We compute *ReliK* on TransE embeddings trained on the entire FB15k237. Figure 2 shows the average *ReliK* scores for positive and negative answers. *ReliK* clearly discriminates between positive and negative instances, often by a large margin.

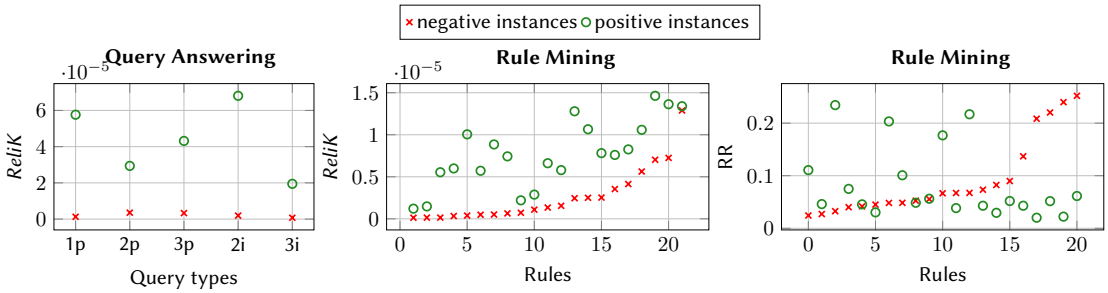

Figure 2: Comparison between positive and negative instances for query answering on FB15k237 (left) and rule mining on Yago2 with *ReliK* (middle) and RR (right).

Rule mining (T4). *ReliK* effectively improves on the rule mining task as well. Rule mining methods [29, 30, 31] automatically retrieve logic rules over KGs having a predefined minimum confidence. A logic rule is an assertion such as $A \Rightarrow B$, which states that B follows from A. For instance, a rule could imply that all presidents of a country are citizens of the country. An instance of a rule is triples matching B, given that A is true. Logic rules are typically harvested with slow exhaustive algorithms similar to the apriori algorithm for association rules [32]. We show that *ReliK* can discriminate between true and false instances.

Detecting true instances. To showcase performance on the downstream task (**T4**), we compare *ReliK* with the reciprocal rank (RR) of a combination of the tail and the relation embeddings on the ability to detect whether an instance of a rule is true or false. This task is particularly important to quantify the real confidence of a rule [33]. To this end, we use a dataset [34] comprising 23 324 manually annotated instances over 26 rules extracted from YAGO2 using the AMIE [29] and RudiK [31] methods. We compute *ReliK* on TransE embeddings trained on the entire YAGO2. Figure 2 shows the average *ReliK* scores for positive and negative instances. *ReliK* discriminates between positive and negative instances, often by a large margin, whereas RR often confounds positive and negative instances.

5. Related Work

Knowledge graph embeddings are commonly used to detect missing triples, correcting errors, or question answering [3, 5]. *Translational embeddings* in the TransE [12] family and the recent PairRE [14] assume that the relationship performs a translation from the head to the tail. *Semantic embeddings*, such as DistMult [6] or HolE [35], interpret the relationship as a multiplicative operator. *Complex embeddings*, such as RotatE [13] and ComplEx [18], use complex-valued vectors and operations in the complex plane. *Neural-network embeddings*, such as ConvE [19], perform sequences of nonlinear operations.

Embedding calibration. An orthogonal direction to ours is embedding calibration [36, 37]. Calibration methods provide effective ways to improve the existing embeddings on various tasks, by altering the embedding vectors in subspaces with low accuracy [36], by reweighing the output probabilities in the respective tasks [37], or by matrix factorization [38].

Evaluation of embeddings. *ReliK* bears an interesting connection with ranking-based quality measures, in particular with the mean reciprocal rank (MRR) and HITS@k for head, tail, and relation prediction [3, 36, 39, 12, 14, 13]. Even though MRR and HITS@k provide a global indication of performance, *ReliK* is suitable for local analysis. Yet, current global measures have been recently shown to be biased towards high-degree nodes [40].

6. Conclusion

Aiming to develop a measure that prognosticates the performance of a knowledge graph embedding on a specific subgraph, we introduced *ReliK*, a KGE reliability measure agnostic to the choice of the embeddings, the dataset, and the task. To allow for efficient computation, we proposed a sampling-based approximation, which we show to achieve similar results to the exact *ReliK* in a fraction of the time. Our experiments confirm that *ReliK* anticipates the performance on a number of common and complex downstream tasks for KGEs. These results suggest that *ReliK* may be used in other domains, as well as a debugging tool for KGEs.

Acknowledgments

M. Egger is supported by the Danish Council for Independent Research grant DFF-1051-00062B. W. Ma is supported by the China Scholarship Council grant 202110320012.

References

[1] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab, A. Zimmermann, Knowledge graphs, ACM CSUR 54 (2022) 71:1–71:37.

[2] G. Weikum, Knowledge graphs 2021: A data odyssey, PVLDB 14 (2021) 3233–3238.

[3] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: A survey of approaches and applications, TKDE 29 (2017) 2724–2743.

[4] X. Wang, L. Chen, T. Ban, M. Usman, Y. Guan, S. Liu, T. Wu, H. Chen, Knowledge graph quality control: A survey, Fundamental Research 1 (2021) 607–626.

[5] S. Ji, S. Pan, E. Cambria, P. Marttinen, S. Y. Philip, A survey on knowledge graphs: Representation, acquisition, and applications, Trans. Neural Netw. Learn. Syst. 33 (2021) 494–514.

[6] B. Yang, S. W.-t. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, in: ICLR, 2015.

[7] Y. Wu, Y. Xu, X. Lin, W. Zhang, A holistic approach for answering logical queries on knowledge graphs, in: ICDE, 2023, pp. 2345–2357.

[8] S. S. Bhowmick, E. C. Dragut, W. Meng, Globally aware contextual embeddings for named entity recognition in social media streams, in: ICDE, 2023, pp. 1544–1557.

[9] J. Huang, Z. Sun, Q. Chen, X. Xu, W. Ren, W. Hu, Deep active alignment of knowledge graph entities and schemata, PACMMOD 1 (2023) 159:1–159:26.

[10] A. Zeakis, G. Papadakis, D. Skoutas, M. Koubarakis, Pre-trained embeddings for entity resolution: An experimental analysis, PVLDB 16 (2023) 2225–2238.

[11] Z. Zhong, M. Zhang, J. Fan, C. Dou, Semantics driven embedding learning for effective entity alignment, in: ICDE, 2022, pp. 2127–2140.

[12] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, NeurIPS 26 (2013).

[13] Z. Sun, Z. Deng, J. Nie, J. Tang, Rotate: Knowledge graph embedding by relational rotation in complex space, in: ICLR, 2019.

[14] L. Chao, J. He, T. Wang, W. Chu, Pairre: Knowledge graph embeddings via paired relation vectors, in: ACL, 2021, pp. 4360–4369.

[15] M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, M. Galkin, S. Sharifzadeh, A. Fischer, V. Tresp, J. Lehmann, Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework, TPAMI 44 (2021) 8825–8845.

[16] N. Jain, J.-C. Kalo, W.-T. Balke, R. Krestel, Do embeddings actually capture knowledge graph semantics?, in: ESWC, 2021, pp. 143–159.

[17] J. L. W. V. Jensen, Sur les fonctions convexes et les inégalités entre les valeurs moyennes, Acta mathematica 30 (1906) 175–193.

[18] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: ICML, PMLR, 2016, pp. 2071–2080.

[19] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: AAAI, volume 32, 2018.

[20] G. Bouchard, S. Singh, T. Trouillon, On approximate reasoning capabilities of low-rank vector spaces, in: AAAI, 2015.
[21] K. Toutanova, D. Chen, Observed versus latent features for knowledge base and text inference, in: Proceedings of the 3rd workshop on continuous vector space models and their compositionality, 2015, pp. 57–66.
[22] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: SIGMOD, 2008, pp. 1247–1250.
[23] T. Safavi, D. Koutra, Codex: A comprehensive knowledge graph completion benchmark, in: EMNLP, 2020, pp. 8328–8350.
[24] J. Hoffart, F. M. Suchanek, K. Berberich, G. Weikum, Yago2: A spatially and temporally enhanced knowledge base from wikipedia, Artificial intelligence 194 (2013) 28–61.
[25] H. Tong, C. Faloutsos, J.-Y. Pan, Fast random walk with restart and its applications, in: ICDM, IEEE, 2006, pp. 613–622.
[26] N. Craswell, Mean reciprocal rank., Encyclopedia of database systems 1703 (2009).
[27] H. Ren, J. Leskovec, Beta embeddings for multi-hop logical reasoning in knowledge graphs, Advances in Neural Information Processing Systems 33 (2020) 19716–19726.
[28] Y. Bai, X. Lv, J. Li, L. Hou, Answering complex logical queries on knowledge graphs via query computation tree optimization (2023).
[29] L. A. Galárraga, C. Teflioudi, K. Hose, F. M. Suchanek, AMIE: association rule mining under incomplete evidence in ontological knowledge bases, in: TheWebConf, 2013, pp. 413–422.
[30] L. Galárraga, C. Teflioudi, K. Hose, F. M. Suchanek, Fast rule mining in ontological knowledge bases with amie ++, VLDBJ 24 (2015) 707–730.
[31] S. Ortona, V. V. Meduri, P. Papotti, Robust discovery of positive and negative rules in knowledge bases, in: ICDE, 2018, pp. 1168–1179.
[32] R. Agrawal, R. Srikant, et al., Fast algorithms for mining association rules, in: VLDB, volume 1215, Santiago, Chile, 1994, pp. 487–499.
[33] M. Loster, D. Mottin, P. Papotti, J. Ehmüller, B. Feldmann, F. Naumann, Few-shot knowledge validation using rules, in: TheWebConf, 2021, pp. 3314–3324.
[34] M. Loster, D. Mottin, P. Papotti, J. Ehmüller, B. Feldmann, F. Naumann, Colt dataset, 2021. URL: https://hpi.de/naumann/projects/repeatability/datasets/colt-dataset.html, accessed on November 1, 2023.
[35] M. Nickel, V. Tresp, H.-P. Kriegel, et al., A three-way model for collective learning on multi-relational data., in: ICML, volume 11, 2011, pp. 3104482–3104584.
[36] T. Safavi, D. Koutra, E. Meij, Evaluating the calibration of knowledge graph embeddings for trustworthy link prediction, in: EMNLP, 2020.
[37] P. Tabacof, L. Costabello, Probability calibration for knowledge graph embedding models, in: ICLR, 2020.
[38] C. Demir, J. Lienen, A.-C. Ngonga Ngomo, Kronecker decomposition for knowledge graph embeddings, in: HT, 2022, pp. 1–10.
[39] F. Bianchi, G. Rossiello, L. Costabello, M. Palmonari, P. Minervini, Knowledge graph embeddings and explainable ai, in: Knowledge Graphs for eXplainable Artificial Intelligence: Foundations, Applications and Challenges, IOS Press, 2020, pp. 49–72.

[40] S. Tiwari, I. Bansal, C. R. Rivero, Revisiting the evaluation protocol of knowledge graph completion methods for link prediction, in: TheWebConf, 2021, pp. 809–820.