

Knowledge Graph Embeddings or Bias Graph Embeddings? A Study of Bias in Link Prediction Models

Andrea Rossi^a, Donatella Firmani^b and Paolo Merialdo^a

^aRoma Tre University, Roma, Italy

^bSapienza University, Roma, Italy

Abstract

Link Prediction aims at tackling Knowledge Graph incompleteness by inferring new facts based on the existing, already known ones. Nowadays most Link Prediction systems rely on Machine Learning and Deep Learning approaches; this results in inherent opaque models in which assessing the robustness to data biases is not trivial. We define 3 specific types of Sample Selection Bias and estimate their presence in the 5 best-established Link Prediction datasets. We then verify how these biases affect the behaviour of 9 systems representative for every major family of Link Prediction models. We find that these models do indeed learn and incorporate each of the presented biases, with a heavily negative effect on their behaviour. We thus advocate for the creation of novel more robust datasets and of more effective evaluation practices.

Keywords

Link Prediction, Knowledge Graph Embeddings

1. Introduction

Knowledge Graphs (KGs) are structured repositories of information where nodes modeling real-world *entities* are linked by labeled directed edges; each label represents a semantic *relation*, therefore each edge linking a pair nodes represents a *fact* conveying that the corresponding two entities are connected via that relation.

KGs have recently achieved widespread popularity in a variety of contexts. Large open KGs, such as DBpedia [1], YAGO [2] and Wikidata [3], are used on a daily basis for Semantic Web projects [4] and question answering [5]. Meanwhile, a growing number of companies rely on private KGs to support their services. Google and Microsoft use respectively the Google KG [6] and Satori [7] to enhance their search engines; Amazon [8] and Ebay [9] use product graphs to improve their recommendations; social networks like Facebook [10] and LinkedIn [11] use KGs for user profiling and advertisement.

It is therefore unsurprising that many KGs have achieved web-scale dimensions, featuring millions of entities and billions of facts. Nonetheless, it is well-known that even even the largest


DL4KG 2021: Workshop on Deep Learning for Knowledge Graphs, held as part of ISWC 2021: the 20th International Semantic Web Conference, October 24 - 28, 2021 (Online Event)

✉ andrea.rossi3@uniroma3.it (A. Rossi); donatella.firmani@uniroma1.it (D. Firmani);

paolo.merialdo@uniroma3.it (P. Merialdo)



© 2021 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)

and richest KGs suffer from *incompleteness*, as they only hold a small portion of the real-world knowledge they should encompass [12].

Link Prediction (LP) tackles this issue by leveraging the already known facts in the KG to infer new ones. Nowadays the vast majority of LP models learn vectorized representations of entities and relations called *embeddings* using Machine Learning (ML) techniques; many of them rely on Deep Learning architectures, featuring sequences of neural layers interspersed by activation functions. These models have been shown to achieve state-of-the-art performances [13, 14]. In the last decade embedding-based LP has become a sparkling research area, with dozens of novel models being proposed every year (see the work by Wang *et al.* [15] for a comprehensive survey). The pioneering model TransE [16] has established the practice of evaluating these systems computing global metrics of their predictive performances over datasets obtained from real-world KGs.

LP datasets are usually obtained by extracting the most mentioned entities from real-world KGs. We find that this policy leads to various forms of imbalances and *biases*. For instance, the datasets sampled from Freebase [17] tend to only feature people with nationality *USA*. Furthermore, the same biases observed in training are also present in validation and testing: this indirectly *incentivizes* models to incorporate the biases, as they can be instrumental to produce the correct predictions and boost the evaluation results. In short, our models are yielding the correct answers for the wrong reasons. For instance, in FB15k-237, which is considered the most reliable dataset [13], among the 1210 training facts and 151 test facts with relation *film_budget_currency*, 1164 and 146 facts respectively feature the same tail *USA_dollar*. We find that in those test facts, models always predict the correct tail on the first try when the answer is *USA_dollar*, whereas they never manage to guess if it is a different currency.

A few studies in the past have highlighted criticalities in LP benchmarks, mainly with regard to test leakage [18, 19] or of unnatural distributions and structures [20, 13]. Nonetheless, to the best of our knowledge the presence of straight-up biases had gone almost unnoticed so far. This motivates a systematic analysis of how they affect datasets and models, especially considering that KG embeddings have been shown to be just as vulnerable to biases as word embeddings [21, 22]. We report an accurate comparison with these other researches in Section 5.

We provide a formal definition of 3 types of data bias that can affect LP models. We focus on the 5 best-established LP datasets and estimate for each of them the number of test samples affected by each type of bias. We then conduct an extensive re-evaluation of 9 models representative for all the major families of LP systems, showing how removing the biased test facts affects their overall predictive performance. All the code and resources generated in our work are available at our public repository.¹

The paper is organized as follows. In Section 2 we overview how embedding-based models perform the LP task; in Section 3 we discuss the main types of data bias we analyze in our work; in Section 4 we report our experimental findings on how they affect LP models; in Section 5 we present works related to ours and in Section 6 we provide concluding remarks.

¹<https://github.com/merialdo/research.lpbias>

2. Link Prediction on Knowledge Graphs

We define any Knowledge Graph as $KG = (\mathcal{E}, \mathcal{R}, \mathcal{G})$, where \mathcal{E} is a set of entities, \mathcal{R} is a set of relations, and $\mathcal{G} \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is a set of facts connecting the entities via relations. Each fact can be formulated as a triple $\langle h, r, t \rangle$, where h is the *head*, r is the *relation*, and t is the *tail*.

Most LP models nowadays map entities and relations to vectorized representations called *KG embeddings*. These models usually define a *scoring function* ϕ to estimate the plausibility of facts based on the embeddings of their elements. Embeddings are initialized randomly; then, they are trained with ML methods to optimize the scores of the known facts. When the training is over, the learned embeddings should be able to generalize and yield good ϕ values for unseen true facts as well. Models may also feature deep architectures of neural layers, which can be used in ϕ to process the embeddings of elements of the facts to score. The weights of neural layers are trained jointly with the KG embeddings.

Given a trained model, a *tail prediction* $\langle h, r, t \rangle$ is the process that finds t to be the best-scoring entity to complete the triple $\langle h, r, ? \rangle$: i.e., t is the answer to the question «What is the most likely tail for head h and relation r ?»²:

$$t = \operatorname{argmax}_{e \in \mathcal{E}} \phi(h, r, e). \quad (1)$$

A formulation for *head predictions* can be defined analogously.

LP research typically relies on datasets sampled from real-world KGs. Any dataset has its own sets \mathcal{E} , \mathcal{R} , and \mathcal{G} , and \mathcal{G} is usually split into a training set \mathcal{G}_{train} , a validation set \mathcal{G}_{valid} and a test set \mathcal{G}_{test} . To evaluate the predictive performance of a model on a dataset, head and tail predictions are performed on each test fact in its \mathcal{G}_{test} . For each prediction, the *target entity* featured in the test fact (i.e., the expected prediction) is ranked against all other entities in \mathcal{E} . Given any test fact $\langle h, r, t \rangle$, the *tail rank* can be thus computed as:

$$\text{tailRank}(h, r, t) = |\{e \in \mathcal{E} | \phi(h, r, e) \geq \phi(h, r, t)\}| \quad (2)$$

Head ranks can be computed analogously. An ideal model, or an *oracle*, would obtain rank 1 in the head and tail predictions of all test facts. The set T of all the head and tail ranks obtained in testing are then gathered into global metrics:

- *Hits@K (H@K)*: the fraction of ranks in T lesser or equal than a value k .
- *Mean Reciprocal Rank (MRR)*: the average of the inverse values of all the ranks in T .

Both $H@K$ and MRR are always between 0 and 1; the higher their value, the better the result they convey. These metrics can be computed in two separate settings: in the *raw setting* correct answers that outrank the target one are still deemed wrong; in the *filtered setting* they are not considered mistakes and do not contribute to rank computation. *Raw* metrics can sometimes be misleading, so *filtered* metrics are generally preferred in literature [16]; therefore, in this work we always use filtered metrics.

²In this formulation we assume higher ϕ scores convey better plausibility; analogous formulations are defined for models where higher scores convey worse plausibility.

3. Forms of Data Bias

In this section, we define 3 main types of bias commonly found in LP datasets. All of them are forms of *Sample Selection Bias* [23], i.e., unwanted, unrealistic patterns in a dataset caused by imbalances in the processes and sources used to construct it. We provide definitions from the perspective of a tail prediction $\langle h, r, t \rangle$; analogous definitions can be used for head predictions.

Type 1 Bias. A tail prediction $\langle h, r, t \rangle$ is prone to *Type 1 Bias* if the training facts mentioning r tend to always feature t as tail. For example, the tail prediction $\langle Barack_Obama, gender, male \rangle$ is prone to this type of bias if the vast majority of gendered entities in the training set are males: this artificially favours the prediction of male genders. In practice, we verify if the fraction between the number of training facts featuring both r and t and the number of training facts featuring r exceeds a threshold τ_1 . In our experiments we set $\tau_1 = 0.75$.

Type 2 Bias. A tail prediction $\langle h, r, t \rangle$ in which r is a one-to-many or a many-to-many relation is prone to *Type 2 Bias* if, whenever an entity e is seen as head for relation r , fact $\langle e, r, t \rangle$ also exists in \mathcal{G}_{train} . Type 2 Bias affects relations that have a "default" correct answer. Differently from Type 1, facts mentioning r may feature a variety of tails different from t ; however, for each entity e seen as head these facts, t tends to always be among the correct tails too. This makes $\langle e, r, t \rangle$ artificially easier to predict. For instance, the tail prediction $\langle Cristiano_Ronaldo, language, English \rangle$ is prone to Type 2 Bias if most people, in addition to other languages, also speak *English*. In practice, we verify if the fraction of entities e seen as heads for relation r and that also display a fact $\langle e, r, t \rangle$ exceeds a threshold τ_2 . In our experiments we use $\tau_2 = 0.5$.

Type 3 Bias. A tail prediction $\langle h, r, t \rangle$ is prone to *Type 3 Bias* if a relation s exists such that: (i) whenever s links two entities, r links them as well; and (ii) the fact $\langle h, s, t \rangle$ is present in the training set. For example, in the FB15k dataset the producer of a TV program is almost always its creator too; this may lead to assume that creating a program implies being its producer. In practice, to verify if s and r share this correlation we check if the fraction of s mentions in which s also co-occurs with r is greater than a threshold τ_3 . In our experiments we set $\tau_3 = 0.5$.

4. Data Bias in Link Prediction Benchmarks

We discuss in this section our experimental findings on the three forms of data bias defined in Section 3. We first describe how these biases affect LP datasets; we then analyze their consequences on the behaviour of LP models.

4.1. Data Biases in LP Datasets

We take into account the 5 most popular LP datasets in literature: FB15k, WN18, FB15k-237, WN18RR and YAGO3-10. **FB15k** and **WN18** have been built by Bordes *et al.* [16] by selecting the facts featuring the richest entities in Freebase [17] and WordNet [24]. Toutanova and Chen [18] and Dettmers *et al.* [19] have observed that such datasets suffer from test leakage due to the pervasive presence of inverse and equivalent relations; they have filtered away such relations to create the more challenging subsamples **FB15k-237** and **WN18RR**. Finally, Dettmers *et al.* [19]

	Entities	Relations	Facts			Test Predictions				
			Train	Valid	Test	All	w/o B1	w/o B2	w/o B3	w/o B*
FB15k	14.951	1.345	483.142	50.000	59.071	118.142	115.165 (-2.5%)	108.705 (-8.0%)	101.406 (-14.2%)	93.005 (-21.3)
FB15k-237	14.541	237	272.115	17.535	20.466	40.932	39.145 (-4.4%)	36.482 (-5.5%)	40.932 (-0.0%)	36.912 (-9.8%)
WN18	40.943	18	141.442	5.000	5.000	10.000	10.000 (-0.0%)	10.000 (-0.0%)	10.000 (-0.0%)	10.000 (-0.0%)
WN18RR	40.943	11	86.835	3.034	3.134	6.268	6.268 (-0.0%)	6.268 (-0.0%)	6.268 (-0.0%)	6.268 (-0.0%)
YAGO3-10	123.155	37	1.078.868	5.000	5.000	10.000	9.725 (-2.8%)	9.992 (-0.1%)	4.876 (-51.2%)	4.593 (-54.1%)

Table 1
Dataset statistics and numbers of test predictions prone to bias.

have also created the **YAGO3-10** dataset selecting the facts featuring entities linked by at least 10 different relations in the YAGO3 KG [25].

In these datasets the training, validation and test sets are sampled from the same distributions, so any bias seen in training will also be featured in validation and testing. For each test prediction in each dataset we verify if it is prone to any type of bias applying the definitions in Section 3. Table 1 reports the main statistics and the number of test predictions not affected by Type 1 Bias (w/o B1), Type 2 Bias (w/o B2), Type 3 Bias (w/o B3), and by any type of bias at all (w/o B*). Note that these numbers are not cumulative because the same prediction may be affected by multiple types of bias.

The most affected dataset is YAGO3-10, with 54.1% of its test facts being prone to bias, especially of Type 3. This confirms the finding of Akrami *et al* [13] that the two most common relations in the dataset (*affiliated_to* and *plays_for*) are almost interchangeable. FB15k and FB15k-237 are noticeably affected too, with 26.7% and 9.8% of their test facts prone to bias respectively. The Type 3 Bias is not present at all in FB15k-237; this is not surprising, because this dataset was built by design with no inverse or equivalent relations. WN18 and WN18RR, finally, seem completely immune to any type of bias; this is possibly due the nature of WordNet, which is a lexical ontology rather than a KG.

4.2. Data Biases and LP models

In this section we study how data bias affects the behaviour of LP models. As mentioned in Section 4.1, LP benchmarks display the same biases across their training and test sets. Hence, in addition to being exposed to bias, unbeknownst to developers, LP models are actually encouraged to incorporate it, because such bias can help to identify the correct answers in testing. In the long run this may favour the architectures most vulnerable to bias over the most robust ones.

To assess the severity of this condition we remove from each dataset the test predictions prone to each type of bias, and measure how this affects the evaluation results of LP models relying on diverse architectures. We focus on H@1 and MRR, usually considered the most characterizing metric in LP. We stress that we do not modify the training sets of the datasets: we just filter away the bias-prone test predictions, in the quantities reported in Table 1.

We use as a starting point the evaluation results of 19 LP models publicly shared by Rossi *et al.* [14]. Due to space constraints, we report our findings on 9 models represent-

		FB15k					FB15k-237					YAGO3-10					
		Orig.	w/o B1	w/o B2	w/o B3	w/o B*	Orig.	w/o B1	w/o B2	w/o B3	w/o B*	Orig.	w/o B1	w/o B2	w/o B3	w/o B*	
		Matrix Decomposition		Geometric		Deep Learning		AnyBURL									
Matrix Decomposition	Complex	H@1	0.823	0.819 (-0.6%)	0.814 (-1.1%)	0.799 (-2.9%)	0.788 (-4.3%)	0.272	0.239 (-12.0%)	0.241 (-11.3%)	0.272 (0.0%)	0.205 (-24.6%)	0.501	0.487 (-2.7%)	0.501 (-0.1%)	0.232 (-53.7%)	0.186 (-62.9%)
		MRR	0.855	0.851 (-0.4%)	0.846 (-1.0%)	0.835 (-2.4%)	0.824 (-3.6%)	0.367	0.338 (-7.8%)	0.337 (-8.1%)	0.367 (0.0%)	0.306 (-16.7%)	0.577	0.566 (-2.0%)	0.577 (-0.1%)	0.307 (-46.8%)	0.265 (-54.1%)
	TuckER	H@1	0.729	0.722 (-0.9%)	0.716 (-1.8%)	0.705 (-3.3%)	0.689 (-5.5%)	0.259	0.225 (-13.0%)	0.228 (-12.0%)	0.259 (0.0%)	0.191 (-26.3%)	0.466	0.460 (-1.1%)	0.466 (0.0%)	0.186 (-60.1%)	0.158 (-66.1%)
		MRR	0.788	0.783 (-0.7%)	0.777 (-1.4%)	0.767 (-2.7%)	0.754 (-4.4%)	0.352	0.323 (-8.4%)	0.322 (-8.5%)	0.352 (0.0%)	0.290 (-17.8%)	0.544	0.537 (-1.3%)	0.544 (0.0%)	0.257 (-52.7%)	0.225 (-58.6%)
Geometric	TransE	H@1	0.494	0.492 (-0.3%)	0.467 (-5.3%)	0.463 (-6.2%)	0.438 (-11.2%)	0.217	0.184 (-15.3%)	0.189 (-12.8%)	0.217 (0.0%)	0.153 (-29.6%)	0.406	0.390 (-4.0%)	0.406 (0.0%)	0.135 (-66.8%)	0.083 (-79.4%)
		MRR	0.628	0.624 (-0.6%)	0.608 (-3.1%)	0.597 (-4.9%)	0.577 (-8.1%)	0.310	0.280 (-9.7%)	0.282 (-9.1%)	0.310 (0.0%)	0.249 (-19.8%)	0.501	0.487 (-2.8%)	0.501 (0.0%)	0.219 (-56.2%)	0.172 (-65.6%)
	CrossE	H@1	0.601	0.592 (-1.5%)	0.590 (-1.8%)	0.573 (-4.6%)	0.561 (-6.6%)	0.212	0.178 (-15.9%)	0.182 (-14.2%)	0.212 (0.0%)	0.145 (-31.7%)	0.331	0.321 (-3.0%)	0.33 (-0.1%)	0.128 (-61.4%)	0.093 (-71.8%)
		MRR	0.702	0.695 (-1.0%)	0.693 (-1.3%)	0.677 (-3.6%)	0.666 (-5.1%)	0.298	0.267 (-10.3%)	0.267 (-10.2%)	0.298 (0.0%)	0.233 (-21.6%)	0.446	0.435 (-2.5%)	0.445 (-0.1%)	0.207 (-53.7%)	0.167 (-62.4%)
	HAKE	H@1	0.745	0.739 (-0.8%)	0.734 (-1.5%)	0.721 (-3.2%)	0.708 (-4.9%)	0.249	0.218 (-12.5%)	0.223 (-10.6%)	0.249 (0.0%)	0.189 (-24.2%)	0.463	0.453 (-2.2%)	0.462 (0.0%)	0.199 (-57.1%)	0.161 (-65.2%)
		MRR	0.796	0.791 (-0.6%)	0.786 (-1.2%)	0.775 (-2.6%)	0.763 (-4.1%)	0.347	0.319 (-8.1%)	0.320 (-7.6%)	0.347 (0.0%)	0.289 (-16.5%)	0.546	0.536 (-1.9%)	0.546 (0.0%)	0.273 (-50.0%)	0.234 (-57.2%)
Deep Learning	InteractE	H@1	0.726	0.719 (-1.0%)	0.711 (-2.1%)	0.695 (-4.2%)	0.677 (-6.7%)	0.263	0.231 (-12.5%)	0.233 (-11.6%)	0.263 (0.0%)	0.197 (-25.3%)	0.466	0.451 (-3.2%)	0.466 (-0.1%)	0.213 (-54.4%)	0.165 (-64.7%)
		MRR	0.786	0.780 (-0.7%)	0.774 (-1.5%)	0.760 (-3.3%)	0.745 (-5.2%)	0.355	0.326 (-8.2%)	0.325 (-8.4%)	0.355 (0.0%)	0.293 (-17.4%)	0.543	0.531 (-2.4%)	0.543 (-0.1%)	0.277 (-49.1%)	0.232 (-57.2%)
	CapsE	H@1	0.019	0.015 (-23.7%)	0.014 (-29.4%)	0.011 (-40.9%)	0.007 (-66.0%)	0.073	0.040 (-45.5%)	0.064 (-34.4%)	0.073 (0.0%)	0.028 (-62.1%)	0.000	0.000 (0.0%)	0.000 (0.0%)	0.000 (0.0%)	0.000 (0.0%)
		MRR	0.087	0.075 (-13.0%)	0.073 (-15.5%)	0.072 (-16.5%)	0.056 (-35.9%)	0.160	0.126 (-21.1%)	0.145 (-9.6%)	0.160 (0.0%)	0.108 (-32.4%)	0.000	0.000 (0.0%)	0.000 (0.0%)	0.000 (0.0%)	0.000 (0.0%)
	RSN	H@1	0.723	0.717 (-0.9%)	0.71 (-1.9%)	0.690 (-4.6%)	0.674 (-6.8%)	0.198	0.163 (-17.8%)	0.169 (-15.0%)	0.198 (0.0%)	0.130 (-34.6%)	0.492	0.477 (-2.9%)	0.426 (-1.1%)	0.164 (-61.6%)	0.118 (-72.3%)
		MRR	0.777	0.771 (-0.7%)	0.765 (-1.6%)	0.748 (-3.7%)	0.734 (-5.6%)	0.280	0.247 (-11.5%)	0.249 (-10.9%)	0.280 (0.0%)	0.214 (-23.6%)	0.560	0.547 (-2.2%)	0.51 (-0.1%)	0.231 (-54.8%)	0.187 (-63.5%)
AnyBURL	H@1	0.815	0.810 (-0.5%)	0.804 (-1.3%)	0.795 (-2.4%)	0.782 (-4.0%)	0.269	0.236 (-12.2%)	0.237 (-12.0%)	0.269 (0.0%)	0.201 (-25.5%)	0.492	0.477 (-2.9%)	0.491 (-0.1%)	0.218 (-55.7%)	0.169 (-65.6%)	
	MRR	0.837	0.833 (-0.4%)	0.827 (-1.2%)	0.818 (-2.2%)	0.806 (-3.7%)	0.353	0.324 (-8.2%)	0.322 (-8.9%)	0.353 (0.0%)	0.290 (-18.0%)	0.560	0.547 (-2.2%)	0.559 (-0.1%)	0.282 (-49.5%)	0.238 (-57.4%)	

Table 2

Evaluation results on FB15k, FB15k-237 and YAGO3-10 before and after removing the test predictions prone to bias.

ing all the families in their work: Complex [26] and TuckER [27] for the Matrix Decomposition models; TransE [16], CrossE [28] and HAKE [29] for the Geometric models; InteractE [30], RSN [31] and CapsE [32] for the Deep Learning models; and the rule-based AnyBURL [33] as a baseline. Complete results for all the 19 models are available in our online repository.

Table 2 shows the results for datasets FB15k, FB15k-237 and YAGO3-10. If models were immune to biases, removing bias-prone test predictions should not heavily affect their metrics. On the contrary, we observe impressive performance drops across all models. YAGO3-10 results display the largest worsening, with models losing 50% to 70% of their H@1 and MRR. In FB15k-

		WN18					WN18RR				
		Orig.	w/o B*	w/o inverse	w/o symmetric	w/o inverse or symmetric	Orig.	w/o B*	w/o inverse	w/o symmetric	w/o inverse or symmetric
		Matrix Decomposition									
TuckER	H@1	0.944	0.944 (0.0%)	0.807 (-14.5%)	0.929 (-1.6%)	0.215 (-77.2%)	0.443	0.443 (0.0%)	0.443 (0.0%)	0.161 (-63.6%)	0.161 (-63.6%)
	MRR	0.951	0.951 (0.0%)	0.827 (-13.0%)	0.937 (-1.4%)	0.295 (-69.0%)	0.489	0.489 (0.0%)	0.489 (0.0%)	0.230 (-52.9%)	0.230 (-52.9%)
TuckER	H@1	0.946	0.947 (0.0%)	0.812 (-14.2%)	0.932 (-1.5%)	0.232 (-75.4%)	0.429	0.429 (0.0%)	0.429 (0.0%)	0.141 (-67.1%)	0.141 (-67.1%)
	MRR	0.951	0.951 (0.0%)	0.826 (-13.1%)	0.938 (-1.4%)	0.290 (-69.5%)	0.459	0.459 (0.0%)	0.459 (0.0%)	0.185 (-59.6%)	0.185 (-59.6%)
Geometric											
TransE	H@1	0.406	0.406 (0.0%)	0.009 (-97.8%)	0.514 (+26.6%)	0.037 (-91.0%)	0.028	0.028 (0.0%)	0.028 (0.0%)	0.042 (+50.5%)	0.042 (+50.5%)
	MRR	0.646	0.646 (0.0%)	0.334 (-48.4%)	0.713 (+10.2%)	0.135 (-79.1%)	0.206	0.206 (0.0%)	0.206 (0.0%)	0.110 (-46.5%)	0.110 (-46.5%)
CrossE	H@1	0.733	0.732 (0.0%)	0.774 (+5.6%)	0.668 (-8.9%)	0.148 (-79.8%)	0.381	0.381 (0.0%)	0.381 (0.0%)	0.073 (-80.8%)	0.073 (-80.8%)
	MRR	0.834	0.834 (0.0%)	0.797 (-4.5%)	0.793 (-4.9%)	0.208 (-75.1%)	0.405	0.405 (0.0%)	0.405 (0.0%)	0.106 (-73.7%)	0.106 (-73.7%)
HAKE	H@1	0.943	0.944 (0.0%)	0.803 (-14.9%)	0.928 (-1.6%)	0.196 (-79.2%)	0.453	0.453 (0.0%)	0.453 (0.0%)	0.177 (-61.0%)	0.177 (-61.0%)
	MRR	0.950	0.950 (0.0%)	0.823 (-13.4%)	0.936 (-1.4%)	0.278 (-70.7%)	0.497	0.497 (0.0%)	0.497 (0.0%)	0.243 (-51.1%)	0.243 (-51.1%)
Deep Learning											
InteractE	H@1	0.946	0.946 (0.0%)	0.811 (-14.3%)	0.932 (-1.5%)	0.232 (-75.4%)	0.427	0.427 (0.0%)	0.427 (0.0%)	0.138 (-67.7%)	0.138 (-67.7%)
	MRR	0.950	0.949 (0.0%)	0.821 (-13.5%)	0.936 (-1.4%)	0.273 (-71.3%)	0.458	0.458 (0.0%)	0.458 (0.0%)	0.184 (-59.8%)	0.184 (-59.8%)
CapsE	H@1	0.846	0.846 (0.0%)	0.699 (-17.4%)	0.828 (-2.1%)	0.042 (-95.0%)	0.337	0.337 (0.0%)	0.337 (0.0%)	0.071 (-79.0%)	0.071 (-79.0%)
	MRR	0.890	0.890 (0.0%)	0.748 (-15.9%)	0.873 (-1.8%)	0.123 (-86.2%)	0.415	0.415 (0.0%)	0.415 (0.0%)	0.158 (-61.9%)	0.158 (-61.9%)
RSN	H@1	0.912	0.912 (0.0%)	0.773 (-15.2%)	0.895 (-1.9%)	0.142 (-84.5%)	0.346	0.346 (0.0%)	0.346 (0.0%)	0.074 (-78.5%)	0.074 (-78.5%)
	MRR	0.928	0.928 (0.0%)	0.793 (-14.6%)	0.912 (-1.7%)	0.190 (-79.5%)	0.395	0.395 (0.0%)	0.395 (0.0%)	0.124 (-68.7%)	0.124 (-68.7%)
AnyBURL											
AnyBURL	H@1	0.947	0.947 (0.0%)	0.815 (-13.9%)	0.933 (-1.5%)	0.247 (-73.9%)	0.458	0.458 (0.0%)	0.458 (0.0%)	0.184 (-59.9%)	0.184 (-59.9%)
	MRR	0.953	0.953 (0.0%)	0.833 (-12.6%)	0.94 (-1.3%)	0.318 (-66.6%)	0.497	0.497 (0.0%)	0.497 (0.0%)	0.243 (-51.1%)	0.243 (-51.1%)

Table 3

Variations in the evaluation results on WN18 and WN18RR after removing the test predictions prone to bias or influenced by inverse and symmetric relations.

237, often considered the most reliable dataset, their decrease is still between 20% and 35%. In FB15k they lose around 5% of the original metrics, but this apparent robustness is likely due to the presence of inverse relations facilitating predictions.

Table 3 reports the results for WN18 and WN18RR. As already described in Section 4.1, in these datasets test predictions do not appear prone to bias. Nonetheless this does not make them more reliable; on the contrary, their test predictions seem only enabled by the presence of symmetric and (in WN18) inverse relations. Table 3 also displays that removing the test predictions featuring symmetric or inverse relations results in plummeting H@1 and MRR values, with a decrease usually between 50% and 75%.

4.3. Key Takeaways

We find that the policies used to generate LP datasets have led to severe forms of selection bias; this, in turn, has made significant fractions of their test sets artificially easier to predict than the others. Our results prove that LP models are indeed sensitive to these forms of bias, as filtering away the affected test predictions heavily worsens their evaluation metrics. Both neural and rule-based LP models appear equally vulnerable to this phenomenon: this proves that the issue is not rooted in the technique used to learn the facts, but rather in the data sources themselves. Not all datasets are interested by this condition to the same extent. YAGO3-10 and FB15k-237 are the most affected, and show the heaviest drop in performance when removing the biased test facts. Wordnet-based datasets, on the other hand, do not display bias at all.

The bias problem, in itself, can probably be avoided by just skipping the test facts prone to bias during evaluation. However, suggesting to just adopt this practice would be naive from our part. Quite worryingly, even in bias-free datasets we observe that most correct predictions are just enabled by the presence of inverse and symmetric relations. In other words, in *all* datasets, when removing the test predictions affected by either bias or inverse/symmetric relations, the predictive performance of models plummets.

This makes us wonder how many of the remaining test facts are actually predictable. If we just removed the bias-prone test predictions, we may mostly end up with test sets that not even humans, with the information available in training, can infer; if this was the case, the whole task would become pointless. We intend to conduct further studies in this regard, asking this question directly to human workers. If the outcome will prove that most non-biased test facts are indeed unpredictable, then it will be painfully necessary to replace the current datasets with novel ones extracted in more sensible ways, possibly keeping humans in the loop for the selection of training and test facts.

5. Related Works

The works most related to ours consist in analyses that highlight criticalities of the current LP benchmarking techniques. So far, compared to the wide body of literature proposing new LP models, a relatively small effort has been devoted to analyzing their evaluation practices.

Toutanova and Chen [18] have been the first to notice the presence of test leakage in FB15k due to inverse relations. They have assessed the severity of the issue by proving that a simple model based on observable features achieves competitive performance on the dataset; they have proceeded to remove these relations from FB15k generating the more challenging FB15k-237.

A similar study has been carried out by Dettmers *et al.* [19] on FB15k and WN18, showing that a trivial system based on inverse relations can achieve state-of-the-art results on both datasets; the authors have then generated WN18RR as a more challenging subsample of WN18.

Akrami *et al.* [13] have carried out an extensive analysis quantifying the effect of various artificial patterns in LP datasets, such as inverse relations and Cartesian product relations. In addition to confirming the above mentioned observations on FB15k and WN18 they have found that LP performances are boosted in FB15k and FB15k-237 by the redundant structures of Cartesian product relations, and in YAGO3-10 by the presence of equivalent relations.

Nayyeri *et al.* [34] refer to KGs in which facts also feature additional numerical weights to convey various meanings, e.g., the confidence of each fact. They acknowledge that the presence of bias in these values hinders the effectiveness of the learned embeddings, and propose a Weighted Triple Loss that, while taking advantage of these weights, is also robust to their biases.

Rossi and Matinata [20] have shown that the distributions of entities in all LP datasets are wildly skewed: a few entities are featured in thousands of training facts, making them easier to learn and predict, whereas the others may only occur a handful of times. The rich and easily predictable entities are over-represented in testing, thus affecting fairness of such benchmarks.

The same concerns have been shared by Mohamed *et al.* [35]; to overcome this issue, they have proposed novel stratified versions of the Hits@K and MRR metrics, called Strat-Hits@K and Strat-MRR respectively. These metrics should estimate of the predictive performance of models in a fairer way, unbiased by the popularity over-represented entities.

All these works share the same spirit of ours in their goal to identify the shortcomings of current LP evaluation approaches, in order to drive research towards more realistic and healthy practices. Our main difference lies in our definition and identification of *bias structures* that had so far gone unnoticed, as well as our systematic methodology of re-computing the predictive performances of a wide variety of models after removing the biased test facts.

6. Conclusion

We have reported an analysis on the presence of bias across the 5 best-established datasets for Link Prediction on KGs. We have defined 3 main types of Selection Sample Bias, and we have observed that they affect significant portions of the test predictions in 3 datasets out of 5. We have then analyzed how removing such bias-prone predictions alters the evaluation results of 9 models representing the main families of Link Prediction systems. The result is generally a significant drop in predictive performance. This proves that a large part of the correct predictions output by models on those datasets is indeed facilitated by the presence of bias. The very low values obtained on this de-biased test scenario suggests that many of the remaining test facts may not be predictable at all.

We thus call for the production of more effective and robust datasets for Link Prediction, and for the definition of more thorough evaluation methods that take into account their properties.

References

- [1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, Dbpedia: A nucleus for a web of open data, in: *The semantic web*, Springer, 2007.
- [2] T. P. Tanon, G. Weikum, F. M. Suchanek, YAGO 4: A reason-able knowledge base, in: *ESWC*, 2020.
- [3] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledge base, *CACM* (2014).
- [4] E. Hovy, R. Navigli, S. P. Ponzetto, Collaboratively Built Semi-structured Content and Artificial Intelligence: The Story So Far, *Artif. Intell.* (2013).
- [5] W. Yih, M. Chang, X. He, J. Gao, Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base, in: *ACL*, 2015.
- [6] A. Singhal, Introducing the knowledge graph: things, not strings, 2012. URL: <https://www.blog.google/products/search/introducing-knowledge-graph-things-not/>, blogpost in the Official Google Blog.

- [7] R. Qian, Understand your world with bing, 2013. URL: <https://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/>, blogpost in Bing Blogs.
- [8] X. L. Dong, Building a broad knowledge graph for products, in: ICDE, 2019.
- [9] R. Pittman, Cracking the code on conversational commerce, 2017. URL: <https://www.ebayinc.com/stories/news/cracking-the-code-on-conversational-commerce/>, blogpost in Ebay Inc. Stories.
- [10] T. Stocky, L. Rasmussen, Introducing graph search beta, 2014. URL: <https://newsroom.fb.com/news/2013/01/introducing-graph-search-beta/>, blogpost in Facebook Newsroom.
- [11] Q. He, B.-C. Chen, D. Agarwal, Building the linkedin knowledge graph, 2016. URL: <https://engineering.linkedin.com/blog/2016/10/building-the-linkedin-knowledge-graph/>, blogpost in LinkedIn Blog.
- [12] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, D. Lin, Knowledge base completion via search-based question answering, in: WWW, 2014.
- [13] F. Akrami, M. S. Saeef, Q. Zhang, W. Hu, C. Li, Realistic Re-evaluation of Knowledge Graph Completion Methods: An Experimental Study, in: SIGMOD, 2020.
- [14] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, P. Merialdo, Knowledge graph embedding for link prediction: A comparative analysis, ACM TKDD (2021).
- [15] M. Wang, L. Qiu, X. Wang, A survey on knowledge graph embeddings for link prediction, Symmetry (2021).
- [16] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: NIPS, 2013.
- [17] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: a collaboratively created graph database for structuring human knowledge, in: SIGMOD, 2008.
- [18] K. Toutanova, D. Chen, Observed versus latent features for knowledge base and text inference, in: CVSC, 2015.
- [19] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: AAAI, 2018.
- [20] A. Rossi, A. Matinata, Knowledge Graph Embeddings: Are Relation-Learning Models Learning Relations?, in: PIE, 2020.
- [21] J. Fisher, D. Palfrey, C. Christodoulopoulos, A. Mittal, Measuring social bias in knowledge graph embeddings, arXiv preprint arXiv:1912.02761 (2019).
- [22] S. Bourli, E. Pitoura, Bias in knowledge graph embeddings, in: ASONAM, IEEE, 2020.
- [23] J. J. Heckman, Sample selection bias as a specification error, *Econometrica* (1979).
- [24] G. A. Miller, Wordnet: a lexical database for english, CACM (1995).
- [25] F. Mahdisoltani, J. Biega, F. M. Suchanek, YAGO3: A knowledge base from multilingual wikipedias, in: CIDR, 2015.
- [26] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex Embeddings for Simple Link Prediction, in: ICML, 2016.
- [27] I. Balazevic, C. Allen, T. M. Hospedales, TuckER: Tensor Factorization for Knowledge Graph Completion, in: EMNLP - IJCNLP, 2019.
- [28] W. Zhang, B. Paudel, W. Zhang, A. Bernstein, H. Chen, Interaction embeddings for prediction and explanation in knowledge graphs, in: WSDM, 2019.
- [29] Z. Zhang, J. Cai, Y. Zhang, J. Wang, Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction, in: AAAI, 2020.
- [30] S. Vashishth, S. Sanyal, V. Nitin, N. Agrawal, P. P. Talukdar, InteractE: Improving Convolution-based Knowledge Graph Embeddings by Increasing Feature Interactions, in: AAAI, 2020.
- [31] L. Guo, Z. Sun, W. Hu, Learning to Exploit Long-term Relational Dependencies in Knowledge Graphs, in: ICML, 2019.
- [32] D. Q. Nguyen, T. Vu, T. D. Nguyen, D. Q. Nguyen, D. Q. Phung, A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization, in: NAACL-HLT, 2019.
- [33] C. Meilicke, M. W. Chekol, M. Fink, H. Stuckenschmidt, Reinforced Anytime Bottom Up Rule Learning for Knowledge Graph Completion, arXiv preprint arXiv:2004.04412 (2020).
- [34] M. Nayyeri, G. M. Cil, S. Vahdati, F. Osborne, A. Kravchenko, S. Angioni, A. A. Salatino, D. R. Recupero, E. Motta, J. Lehmann, Link prediction of weighted triples for knowledge graph completion within the scholarly domain, *IEEE Access* (2021).
- [35] A. Mohamed, S. Parambath, Z. Kaoudi, A. Aboulnaga, Popularity agnostic evaluation of knowledge graph embeddings, in: UAI, PMLR, 2021.