Friendly Neighbors: Contextualized Sequence-to-Sequence Link Prediction

Adrian Kochsiek
University of Mannheim
Germany
akochsiek
@uni-mannheim.de

Apoorv Saxena
Adobe Research
India
apoorvs
@adobe.com

Inderjeet Nair
Adobe Research
India
inderjeetnair1
@gmail.com

Rainer Gemulla
University of Mannheim
Germany
rgemulla
@uni-mannheim.de

Abstract

We propose *KGT5-context*, a simple sequence-to-sequence model for link prediction (LP) in knowledge graphs (KG). Our work expands on KGT5, a recent LP model that exploits textual features of the KG, has small model size, and is scalable. To reach good predictive performance, however, KGT5 relies on an ensemble with a knowledge graph embedding model, which itself is excessively large and costly to use. In this short paper, we show empirically that adding contextual information—i.e., information about the direct neighborhood of the query entity—alleviates the need for a separate KGE model to obtain good performance. The resulting KGT5-context model is simple, reduces model size significantly, and obtains state-of-the-art performance in our experimental study.

1 Introduction

A knowledge graph (KG) is a collection of facts describing relations between real-world entities. Facts are represented in the form of subject-relation-object ((s, r, o)) triples such as (*Brendan Fraser, hasWonPrize, Oscar*). In this paper, we study the link prediction (LP) problem, which is to infer missing links in the KG. We focus on KGs in which the entities and relations have textual features, such as mention names or descriptions.

Saxena et al. (2022) made a case for large language models (LM) for this task. They proposed the KGT5 model, which posed the link prediction problem as a sequence-to-sequence (seq2seq) task. The main advantages of this approach are that

(i) it allows for small model sizes, and

(ii) it decouples inference cost from the graph size (and, in particular, the number of entities).

They found that KGT5's performance was particularly strong when predicting the object of new relations for a query entity (e.g., the birthplace of a person), but fell short of alternative approaches when predicting additional objects for a known relation (e.g., additional awards won by someone).

To avoid this problem, Saxena et al. (2022) used an ensemble of KGT5 with a large knowledge graph embedding (KGE) model (ComplEx (Trouillon et al., 2016)). This ensemble did reach good performance but destroyed both advantages (i) and (ii) of using a LM. In fact, KGE models learn a low-dimensional representation of each entity and each relation in the graph (Bordes et al., 2013; Sun et al., 2019; Trouillon et al., 2016). Consequently, model size and LP cost are linear in the number of entities in the graph, which can be expensive to use for large-scale KGs. For example, the currently best-performing model (Cattaneo et al., 2022) for the large-scale WikiKG90Mv2 benchmark (Hu et al., 2021) consists of an ensemble of 85 KGE models; each taking up more than 86 GB of space for parameters. Though KGE model sizes can be reduced by using compositional embeddings based on text mentions (Wang et al., 2021; Clouatre et al., 2021; Wang et al., 2022; Jiang et al., 2023), inference cost remains high for large graphs.

We propose and study KGT5-context, which expands on KGT5 by providing contextual information about the query entity—i.e., information about the direct neighborhood of the query entity—to facilitate link prediction. Our work is motivated by the KGE model HittER (Chen et al., 2021), which follows a similar approach; we use the seq2seq model KGT5 instead of a Transformer-based KGE model. KGT5-context is very simple: The only change to KGT5 is that we add a verbalization of the neighborhood of the query entity to the description of a given LP task; see Fig. 1 for an example. KGT5-context retains advantages (i) and (ii) of KGT5.

We performed an experimental study using the Wikidata5M (Wang et al., 2021) and

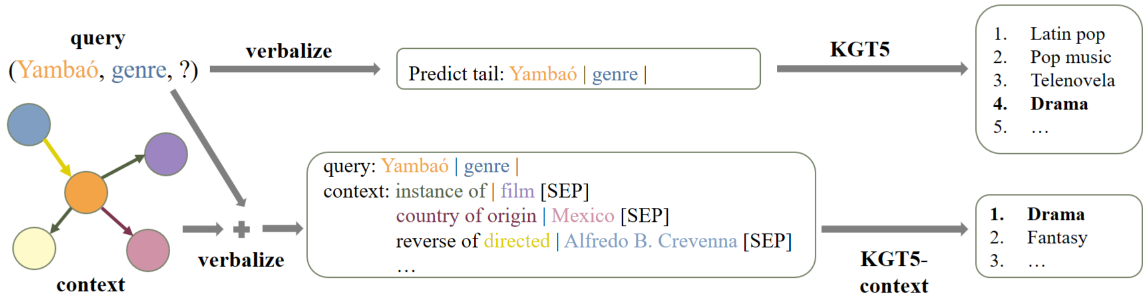

Figure 1: Overview of KGT5-context (at bottom) and comparison to KGT5 (on top); real example from Wikidata5M, best viewed in color. KGT5-context differs from KGT5 in that it appends the neighboring relations and entities of *Yambaó* (a drama movie) to the verbalized query. Both models then apply T5, sample predictions from the decoder, map the samples to entities, and rank by sample logit scores.

WikiKG90Mv2 (Hu et al., 2021) benchmarks. We found that—without further hyperparameter tuning—KGT5-context reached or exceeded state-of-the-art performance on both benchmarks using a significantly smaller model size than alternative approaches. The simple KGT5-context model thus provides a suitable baseline for further research.

2 Expanding KGT5 with Context

Given a *query* $(s, r, ?)$ and a KG, LP is the task to predict new answer entities, i.e., the ? slot of the query. An example is given in Fig. 1.

KGT5 (Saxena et al., 2022) treats link prediction as a seq2seq task. It exploits available textual information for entities and relations, such as mention names (for both entities and relations) or descriptions. KGT5's architecture is based on the encoder-decoder Transformer model T5 (Raffel et al., 2020). It uses canonical mentions to verbalize the LP query to a text sequence of form “predict tail: <subject mention> | <relation mention> | ”. To predict answers, KGT5 samples (exact) candidate mentions from the decoder; the cost of sampling is independent of the number of entities in the KG. To train KGT5, Saxena et al. (2022) use standard training techniques for LLMs: KGT5 is trained on facts in the KG and asked to generate the true answer using teacher forcing and a cross-entropy loss.

KGT5-context (ours) proceeds in the same way as KGT5 but extends the verbalization of the query. In particular, we append a textual sequence of the one-hop neighborhood of the query entity s to the verbalized query of KGT5. As a result, the query entity is contextualized, an approach that has been applied successfully to KGE models before (Chen et al., 2021). KGT5-context simplifies the prediction problem because additional information that is readily available in the KG is provided along with the query. In the example of Fig. 1, the contextual information states that *Yambaó* is a Mexican movie. This information is helpful; e.g., it already rules out the top two predictions of KGT5, which incorrectly suggest that *Yambaó* is a piece of music. For a more detailed analysis, see Sec. 3.3.

Verbalization details. To summarize, we obtain mentions of the entities and relations in the query as well as in the one-hop neighborhood of the query entity. We use these mentions to verbalize the query together with the neighborhood as “query: <query entity mention> | <query relation mention> | context: <context relation 1 mention> | <context entity 1 mention> <SEP> …”.[1] To keep direction of relations, we prepend the relation mention with “reverse of” if the query entity acts as an object, i.e., the relation “points towards” the query entity. A real-world example is given in Fig. 1. Inspired by neighborhood sampling in GNNs (Hamilton et al., 2017), we sample up to k (default: $k = 100$) relation-neighbor pairs uniformly, at random, and without replacement.

3 Experimental Study

We conducted an experimental study to investigate (i) to what extent integrating context in terms of the entity neighborhood into KGT5 improves link prediction performance, (ii) whether the use of context can mitigate the necessity for an ensemble of the text-based KGT5 model with a KGE model, and

[1] When entity descriptions are available, we include “description: <description of query entity>” right before the query context.

Dataset	Entities	Relations	Edges
Wikidata5M	4.8M	828	21M
WikiKG90Mv2	91M	1,387	601M

Table 1: Dataset statistics.

(iii) for what kind of queries context is helpful. We found that:

1. KGT5-context improved the state-of-the-art performance on Wikidata5M using a smaller model (Tab. 2).
2. KGT5-context was orders of magnitudes smaller than the leading models on WikiKG90Mv2 and reached competitive performance (Tab. 3).
3. KGT5-context did not benefit further from ensembling with a KGE model (Tab. 4).

3.1 Experimental Setup

Source code and configuration are available at `https://github.com/uma-pi1/kgt5-context`.

Datasets. We evaluate KGT5-context on two commonly used large-scale link prediction benchmarks. Wikidata5M (Wang et al., 2021) is the induced graph of the 5M most-frequent entities of the Wikidata KG. WikiKG90Mv2 (Hu et al., 2021) contains more than 90M entities and over 600M facts. In contrast to Wikidata5M, it is only evaluated on tail prediction, i.e., $(s, r, ?)$ queries. Dataset statistics are summarized in Tab. 1. For Wikidata5M and WikiKG90Mv2[2], we used the entity mentions provided on the KGT5 webpage. For Wikidata5M, we also consider the usefulness of entity descriptions, which are provided with the dataset and have been used in some prior studies (Wang et al., 2022; Jiang et al., 2023). Note that we do not use these descriptions by default, and clearly mark throughout when they have been used.

Metrics. We follow the standard procedure to evaluate model quality for the link prediction task. In particular, for each test triple (s, r, o), we rank all triples of the form $(s, r, ?)$ (and $(?, r, o)$ on Wikidata5M) by their predicted scores. For KGT5 and KGT5-context, we instead sample from the decoder and ignore outputs that do not correspond to an existing entity mention. For all models, we filter out all true answers other than the test triple that occur either in the train, valid or test data. Finally, we determine the mean reciprocal rank (MRR) and Hits@K over all test triples. In case of ties, we use the mean rank to avoid misleading results (Sun et al., 2020).

Settings. We mainly follow the setting of KGT5. For all experiments, we used the same T5 architecture (T5-small for Wikidata5M, T5-base for WikiKG90Mv2) without any pretrained weights. Training from scratch ensures test data is unseen during (pre-)training and avoids leakage. We used the SentencePiece tokenizer pretrained by (Raffel et al., 2020). We trained on 8 A100-GPUs with a batch size of 32 (effective batch size of 256) using the AdaFactor optimizer. No dataset-specific hyperparameter optimization was performed. For KGT5-context, we sampled up to 100 neighbors per query entity or up to an input sequence length of 512 tokens. For inference, we obtained 500 samples from the decoder.

Models. On Wikidata5M, we compare KGT5-context to the KGE models ComplEx (Trouillon et al., 2016) and SimplE (Kazemi and Poole, 2018) (only graph structure used), the compositional KGE model SimKGC (Wang et al., 2022), its extension utilizing hard negatives (Jiang et al., 2023), and the seq2seq model KGT5 (Saxena et al., 2022). The model of Jiang et al. (2023) is an ensemble of multiple SimKGC models, each trained with a different strategy for selecting negatives. Note that in contrast to KGT5 and KGT5-context, SimKGC is based on pretrained models (BERT transformers). During prediction, all text-based models require access to entity and relation mentions and, when used, the description of the query entity. SimKGC additionally requires access to precomputed entity embeddings, KGT5-context to the 1-hop neighborhood of the query entity in the KG.

On WikiKG90M, we compare to the models presented on the official leaderboard.[2] Here, the best-performing approaches are large ensembles of multiple KGE models.

3.2 Link Prediction Performance

Link prediction performance on Wikidata5M is shown in Tab. 2; additional baselines are given in Tab. 5 (appendix). Generally, we found that textual

[2] We directly used mentions of entities and relations for WikiKG90Mv2, instead of the textual embeddings used by other models. For this reason, the benchmark authors (Hu et al., 2021) did not provide us with scores on the hidden test set. The mentions used to be provided with the dataset but have been removed by now; we obtained them from `https://github.com/apoorvumang/kgt5`.

Model	MRR	Hits@1	Hits@3	Hits@10	Params	Add. requirements for inference	Pre-trained
SimplE †	0.296	0.252	0.317	0.377	2,400M	-	no
ComplEx ††	0.308	0.255	-	0.398	614M	-	no
SimKGC	0.212	0.182	0.223	0.266	220M	entity embeddings	yes
KGT5 ‡	0.300	0.267	0.318	0.365	60M	-	no
+ ComplEx ‡	0.336	0.286	0.362	0.426	674M	-	no
KGT5-context (ours)	0.378	0.350	0.396	0.427	60M	1-hop neighborhood	no
SimKGC + Desc. ‡‡	0.358	0.313	0.376	0.441	220M	entity embeddings	yes
+ Hard Negative Ensemble $	0.420	0.381	0.435	**0.490**	1,100M	entity embeddings	yes
KGT5 + Desc.	0.381	0.357	0.397	0.422	60M	-	no
KGT5-context + Desc. (ours)	**0.426**	**0.406**	**0.440**	0.460	60M	1-hop neighborhood	no

Table 2: Link prediction results on Wikidata5M, test split. The first group does not make use of textual information, the second group uses mention names, the third group additionally entity descriptions. Best per group underlined, best overall bold. Marked results are from † Zhu et al. (2019), †† Kochsiek and Gemulla (2021), ‡ Saxena et al. (2022), ‡‡ Wang et al. (2022), $ Jiang et al. (2023). Additional results in Tab. 5 (appendix).

information was highly beneficial. KGT5-context was the only model that improved upon KGE models (which do not use textual information) when only mention information was available. Moreover, KGT5-context obtained better predictive performance than the ensemble of KGT5 with the ComplEx KGE model. Entity descriptions provided further improvements; they hold valuable information for this benchmark. With these descriptions, KGT5-context outperformed traditional KGE models by up to 12pp in terms of MRR, with a model size reduction of 90-98%. Likewise, KGT5-context improved on KGT5 by 12pp, on the KGT5+Complex ensemble by almost 9pp, and performed roughly on-par with the current state-of-the-art SimKGC ensemble model, which is significantly larger.

The results on the much larger WikiKG90Mv2 are shown in Tab. 3.[2] Here, KGT5-context is multiple orders of magnitude smaller than the currently best-performing models,[3] and improves validation MRR by almost 1pp.

Model	Test MRR	Valid MRR	Params
ComplEx	0.141	0.182	18.2B
TransE	0.082	0.110	18.2B
ComplEx-Concat	0.176	0.205	18.2B
TransE-Concat	0.176	0.206	18.2B
PIE-RM	0.212	0.254	18.2B[3]
DGLKE + Rule Mining	0.249	0.292	18.2B[3]
BESS	**0.254**	0.292	23.3B[3]
KGT5, T5 small[2]	-	0.221	60M
KGT5-context, T5 base (ours)[2]	-	**0.301**	220M

Table 3: Link prediction results on WikiKG90Mv2. Baseline numbers are from the official leaderboard of OGB-LSC (Hu et al., 2021).

Model	0	1-10	>10	All
ComplEx	0.534	0.351	**0.045**	0.296
KGT5	0.624	0.215	0.015	0.300
KGT5-context (ours)	**0.738**	**0.415**	0.014	0.378
KGT5 + ComplEx	0.624	0.351	0.045	0.336
KGT5-context + ComplEx	**0.738**	0.351	**0.045**	**0.379**

Table 4: Test MRR on Wikidata5M grouped by query frequency during training.

3.3 Analysis

To investigate in which cases context information was beneficial, we empirically analyzed LP performance w.r.t. (i) query frequency and (ii) the degree of the query entity. We also sampled predictions and summarize our general observations.

Query frequency. The *frequency* of a test query $(s, r, ?)$ is the number of answers to the query already available in the training data. For example, queries for N:1 relations have frequency 0, whereas queries for 1:N relations can have large frequency for high-degree query entities. We bucketized the test queries of Wikidata5M into low, medium, and high frequency queries and report average MRR for various models in Tab. 4. Generally, high-frequency queries appear harder to answer. These queries have many known true answers already (tying up model capacity); there may be many additional, potentially unrelated answers and incom-

[3]The parameter count in Tab. 3 corresponds to the size of the largest model in an ensemble, not the overall model size. For example, BESS (Cattaneo et al., 2022) consists of 85 models and the complete ensemble has 2.6T parameters; the KGT5-context model is 5 orders of magnitude smaller.

pleteness of the KG may be a concern during evaluation. In contrast, a low-frequency query such as *(Brendan Fraser, instance Of, ?)* has few or no known answer and might be easier to infer, even when the combination of this particular subject and relation was not yet seen during training.

Ensemble with KGE models. In general, the prior KGT5 model performed reasonably well on queries that did not occur in the training data, but was outperformed by a large amount by ComplEx on queries seen multiple times. Hence, both models complemented each other in an ensemble. KGT5-context strongly improved performance over ComplEx, KGT5, and the KGT5+Complex ensemble for low- and medium-frequency queries. For this reason, an ensemble between KGT5-context and ComplEx only brought negligible benefits, but has substantial drawbacks. Consequently, an ensemble of KGT5-context with a KGE model is not needed and should not be used.

Entity degree. We also investigated the benefit of contextual information w.r.t. to the degree of the query entity (see Fig. 2 in the appendix). We found that KGT5-context was beneficial and performed well on query entities with a degree of up to 100. For entities with very large degrees (i.e., nodes with more than 100 or even 1000s of neighbors), ComplEx showed benefits. As before, we feel that these performance benefits are negligible considering the increase in model size and decrease in scalability.

Anecdotal results. We manually probed some predictions of KGT5-context and found the context is especially beneficial when (i) the entity mention only provides limited information about the entity, and/or when (ii) the answer to the query is contained in the one-hop neighborhood.

A case of (i) is shown in Fig. 1, a real example. Here, KGT5 was able to capture the geographic region of the real-world entity only based on its mention. Based on this geographic notion, it proposed the music genre *Latin pop* but was unaware that the entity is a movie. This useful information can be obtained directly from the one-hop neighborhood and, indeed, was exploited by KGT5-context.

For Wikidata5M, the correct answer entity appears in the one-hop neighborhood of the query entity for about 7% of the validation triples. But even when the answer does not directly appear in the context, it may contain entities strongly hinting at the correct answer. For example, it is easier to predict that an entity has occupation *biochemist*, when the context already contains the information that the entity is a *chemist*.

4 Conclusion

We proposed and studied KGT5-context, a sequence-to-sequence model for link prediction in knowledge graphs. KGT5-context extends the KGT5 model of Saxena et al. (2022) by using contextual information of the query entity for prediction. KGT5-context is simple, small, and scalable, and it obtained or exceeded state-of-the-art performance in our experimental study. It thus provides a suitable baseline for further research in this area. A natural direction, for example, is to explore approaches that integrate contextual information in a less naive way than KGT5-context does.

Limitations

KGT5-context relies on the textual mentions of entities and relations (and, optionally, entity descriptions). Therefore, it is only applicable to KGs that provide such information. KGT5-context may be able to handle some entities without textual features when well-described by their neighborhood; we did not investigate this though.

To use KGT5-context for prediction, the KG has to be queried to obtain context information, i.e., the one-hop neighborhood of the query entity. KGT5-context thus cannot be used without the underlying KG.

The verbalized neighborhood of the query entity leads to long input sequences, which in turn may induce higher memory consumption and higher computational cost during training. Overall, training KGT5-context is typically more expensive than training traditional KGE models, which can be tuned (Kochsiek et al., 2022) and trained efficiently (Lerer et al., 2019; Kochsiek and Gemulla, 2021; Zheng et al., 2020).

For inference, KGT5-context first samples relation-neighbor pairs for contextualization, and then samples possible answers from the decoder. These sampling steps can lead to variance in predictive performance. We found this effect to be negligible on Wikidata5M, but it may be larger on other datasets.

Ethics Statement

This research uses publicly available data and benchmarks for evaluation. We believe that this research was conducted in an ethical manner and in compliance with all relevant laws and regulations.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*, pages 1–9.

Alberto Cattaneo, Daniel Justus, Harry Mellor, Douglas Orr, Jerome Maloberti, Zhenying Liu, Thorin Farnsworth, Andrew Fitzgibbon, Blazej Banaszewski, and Carlo Luschi. 2022. Bess: Balanced entity sampling and sharing for large-scale knowledge graph completion. *arXiv preprint arXiv:2211.12281*.

Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2021. Hitter: Hierarchical transformers for knowledge graph embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10395–10407.

Louis Clouatre, Philippe Trempe, Amal Zouaq, and Sarath Chandar. 2021. Mlmlm: Link prediction with mean likelihood masked language model. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4321–4331.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 1025–1035, Red Hook, NY, USA. Curran Associates Inc.

Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. 2021. Ogb-lsc: A large-scale challenge for machine learning on graphs. *Advances in Neural Information Processing Systems*, 35.

Fan Jiang, Tom Drummond, and Trevor Cohn. 2023. Don't mess with mister-in-between: Improved negative search for knowledge graph completion. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1810–1824.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems*.

Adrian Kochsiek and Rainer Gemulla. 2021. Parallel training of knowledge graph embedding models: a comparison of techniques. *Proceedings of the VLDB Endowment*, 15(3):633–645.

Adrian Kochsiek, Fritz Niesel, and Rainer Gemulla. 2022. Start small, think big: On hyperparameter optimization for large-scale knowledge graph embeddings. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part II*, pages 138–154. Springer.

Adam Lerer, Ledell Wu, Jiajun Shen, Timothee Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. 2019. Pytorch-biggraph: A large scale graph embedding system. *Proceedings of Machine Learning and Systems*, 1:120–131.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2814–2828.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Zhiqing Sun, Shikhar Vashishth, Soumya Sanyal, Partha Talukdar, and Yiming Yang. 2020. A re-evaluation of knowledge graph completion methods. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5516–5522.

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

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4281–4294.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

Da Zheng, Xiang Song, Chao Ma, Zeyuan Tan, Zihao Ye, Jin Dong, Hao Xiong, Zheng Zhang, and George Karypis. 2020. Dgl-ke: Training knowledge graph embeddings at scale. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 739–748.

Zhaocheng Zhu, Shizhen Xu, Meng Qu, and Jian Tang. 2019. Graphvite: A high-performance cpu-gpu hybrid system for node embedding. In *The World Wide Web Conference*, pages 2494–2504. ACM.

Appendix

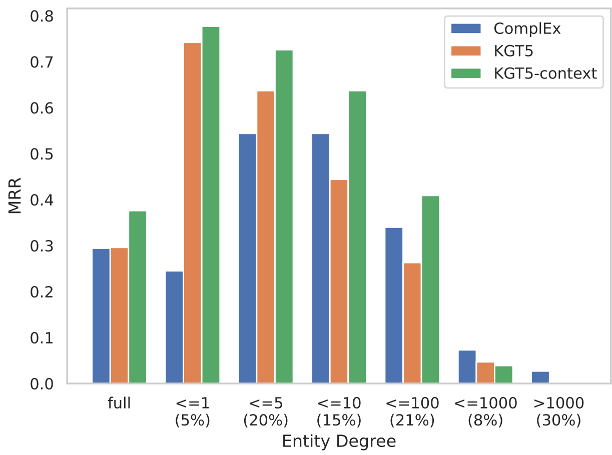

Figure 2: MRR grouped by entity degree on Wikidata5M. Group weight is given in brackets.

Model	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**	**Params**
TransE (Bordes et al., 2013) †	0.253	0.170	0.311	0.392	2,400M
DistMult (Yang et al., 2015) †	0.253	0.209	0.278	0.334	2,400M
RotatE (Sun et al., 2019) †	0.290	0.234	0.322	0.390	2,400M
DKRL (Xie et al., 2016) $	0.160	0.120	0.181	0.229	20M
KEPLER (Wang et al., 2021) $	0.210	0.173	0.224	0.277	125M
MLMLM (Clouatre et al., 2021) ††	0.223	0.201	0.232	0.264	355M

Table 5: Additional link prediction results on Wikidata5M from prior work. Results are from † Zhu et al. (2019). $ Wang et al. (2021). †† Clouatre et al. (2021).