Finding Good Neighbors: Examining the Importance of Neighborhood Selection for Link Prediction

Anonymous ACL submission

Abstract

Link Prediction (LP) approaches based on Language Models (LMs) operate over the labels and descriptions of entities and relations in a KG, achieving LP performance competitive with state-of-the-art. Recent approaches have shown that incorporating a local graph neighborhood can improve the LP capabilities of LMs. These approaches usually sample a context from the neighborhood around a query triple randomly, thereby incorporating noise that might hinder the model in making correct predictions.

In this work, we derive an approximately optimal context for a given query under the assumption that we know the correct answer. This allows us to investigate the characteristics of such contexts and the impact of a good context on LP, thereby providing an approximate upper bound on the achievable performance when using optimal contexts. We provide evidence that the neighborhoods created through random sampling are often suboptimal and unnecessarily large. Furthermore, we show that the potential improvements of using an optimal context can be significant. We conclude that research on context selection is an important step towards developing better LP models.

1 Introduction

Knowledge Graphs (KGs) are used as background knowledge in various NLP applications, e. g., in Question Answering (Schneider et al., 2022), Dialogue Systems (Park et al., 2024), and Text Generation (Wang et al., 2024). A KG is a multi-relational graph, defined by a set of $(subject, relation, object)$ triples. The KG can be called text-attributed KG when textual labels or descriptions are available for entities and relations.

KGs are inherently incomplete as the processes of creating them will always miss relevant facts, either because i) they are manually created and curators miss relevant knowledge due to time and effort constraints, ii) they are automatically created and automatic methods are error-prone and iii) because they might not have been up-dated regularly and thus miss new facts (Paulheim, 2017). To address this, approaches were developed to infer missing triples based on the triples already available in the KG. This task is called Link Prediction (LP). Given a $(subject, relation, ?)$ query, an LP model is trained to predict the missing $?$ to derive a triple that is likely in the graph. A small example KG and an LP query are shown in Fig. 1.

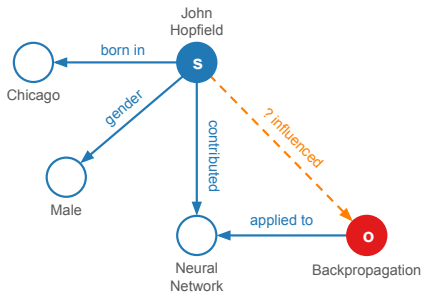


Figure 1: Example KG with LP query.

Research on LP with GNNs (Schlichtkrull et al., 2018; Hamilton et al., 2017) has shown that the local neighborhood can hold valuable information for LP. Whereas GNN-based approaches can incorporate large graph neighborhoods and reduce the neighborhood size to optimize training and inference time (Hamilton et al., 2017; Ying et al., 2018), a Language Model (LM)-based approach to LP is limited to the context size of the LM. Consequently, most LM-based LP approaches cannot include full neighborhoods and, thus, often perform random sampling to select small subgraphs around a given query. For example, given the KG depicted in Fig. 1, random sampling might include the facts about *John Hopfield*'s gender and birthplace but miss the information about his research contributions. As the latter are more relevant for the given query *(John Hopfield, influenced, ?)*, this might

limit the model in making correct predictions if those facts are not available in the sampled context.

We hypothesize that randomly sampled subgraphs introduce noise, distracting the model and occupying context space without adding value. To test this, we analyze which facts in a query's neighborhood support correct predictions at inference time. Optimizing the LP context at inference time is similar to prompt optimization for LLMs, which creates a prompt that contains the most relevant information for a given task and reduces noise.

Our key contributions are as follows. First, we introduce a method to determine the optimal context for a given query, target, and model from the local graph neighborhood. Second, we demonstrate that the derived context generalizes across models and architectures. This shows that the same information is relevant to different models and verifies the effectiveness of our approach. Finally, we show that optimal contexts are remarkably small and that common sampling strategies are suboptimal.

2 Background

A Knowledge Graph (KG) $\mathcal{G}$ is defined by a set of triples, where each triple (s, r, o) represents a relation r between a subject entity s and an object entity o, where $r \in \mathcal{R}$ and $s, o \in \mathcal{E}$. We assume that the KG is text-attributed such that every entity and relation has a label or description. Then, Link Prediction (LP) is the task to predict the $? \in \mathcal{E}$ in a given query q of the form $(s_q, r_q, ?)$ or $(?, r_q, o_q)$.

Traditional LP approaches learn patterns of relations but do not take into account the local graph neighborhood (Yang et al., 2015; Bordes et al., 2013a). Language Models (LMs) were proposed (Yao et al., 2019; Daza et al., 2021; Qiu et al., 2024) specifically for LP on text-attributed KGs. These models use a (pre-trained) LM to obtain a feature representation of entities and relations from their textual attributes. The entity descriptions enable the model to generalize well to entities with few or no relations (inductive LP) (Kochsiek and Gemulla, 2023), in situations where traditional LP models relying on the graph structure struggle. Two prominent examples are SimKGC (Wang et al., 2022) and KGT5 (Saxena et al., 2022). SimKGC uses an LM solely for encoding entity and relation descriptions, and computes triple likelihoods by combining their representations through vector multiplication. In contrast, KGT5 encodes a query and generates a target entity label instead of scoring target candidates. Therefore, the model is much more memory efficient and faster compared to approaches that are based on traditional architectures like, e. g., SimKGC.

Research on LP based on Graph Neural Networks (GNNs) (Schlichtkrull et al., 2018; Busbridge et al., 2018; Hamilton et al., 2017) has shown that the neighborhood can hold valuable information for LP. These approaches iteratively aggregate information from the direct graph neighborhood, allowing for the theoretical incorporation of an arbitrary number of neighboring triples per entity. In order to improve LMs used for LP, recent approaches have proposed to incorporate neighborhood information. One example is NNKGC by Li and Yang (2023) that first obtains entity embeddings through a BERT (Devlin et al., 2019) encoder and then uses a GNN to obtain contextualized node representations by aggregating the graph neighborhood. In contrast, KGT5-context (Kochsiek et al., 2023) is an extension of KGT5 that enriches the KGT5 input sequence with entities and relations in the direct neighborhood of the query head.

3 Related Work

Graph learning models that incorporate local graph neighborhoods face challenges with most existing KGs due to the exponential growth of the computation graph with increasing hop depth and the rapid expansion when encountering high-degree nodes. Consequently, neighborhood selection is an important research area aiming to reduce the training and inference time of models applied to large graphs (Chen et al., 2018; Ying et al., 2018) and to improve the prediction through selecting valuable neighbors (Peng et al., 2021).

We are particularly interested in graph sampling techniques that are applied to KGs. Therefore, in the following, we outline the related work for sampling KG subgraphs for LP and NLP tasks.

Neighborhood Selection for LP Most LP approaches sample contexts uniformly. For example, Kochsiek et al. (2023) sample n triples that include the query head. Other works repeat this sampling to construct multihop subgraphs (Hamilton et al., 2017; Li et al., 2024). This sampling leads to a stochastic training procedure that relies on the assumption that sampling-induced noise averages out over successive training iterations, particularly when entities appear in multiple contexts and when contexts are dynamically resampled.

There are a number of previous approaches that rely on heuristics to determine the context. Luo et al. (2025), for instance, propose to sample triples with the same relation as the query relation from the neighborhood and the entire graph, in addition to randomly selected triples. Bi et al. (2023) have proposed to sample neighbors according to the node degree. Random Walk with Restart (RWR) scores, introduced by Pan et al. (2004), are expected to indicate structural or feature-based relevance between nodes. Consequently, RWR scores are frequently utilized in neighborhood sampling strategies for GNNs (Xiong et al., 2024).

We conclude that most subgraph extraction methods for LP are based on random sampling, and, despite its importance, the selection of an optimal subgraph remains insufficiently explored. In LP, no textual information beyond the query is given that could indicate any relevance of neighboring triples. Until now, the triples that are actually relevant to a prediction and to what extent noise hinders the models have not been evaluated.

Subgraph selection also plays a crucial role in explainable AI (XAI) for GNNs. For example, GNNExplainer (Ying et al., 2019) aim to identify an input subgraph that has the highest importance to the model's prediction. In contrast, we aim to identify the subgraph with the highest impact towards the *correct* model prediction. Here, the goal is not to understand the model behavior – as in explainable AI – but to understand what context is most beneficial for the task.

Subgraph Selection for NLP Tasks Subgraph selection is an important topic in knowledge-intensive NLP applications, such as QA or graph-RAG, too. QA approaches often link entities mentioned in the question to a KG. Instead, Graph-RAG (Peng et al., 2024) computes textual similarities of the question with the text passages of the nodes. The size of the subgraph induced through this set of identified entities often increases exponentially with the number of entities.

In the literature, we found the following non-trainable methods to sample a subgraph given a set of entities: (i) k-hop paths starting at the given entities (Yasunaga et al., 2021, 2022; Taunk et al., 2023), (ii) shortest paths between the given entities (Plenz et al., 2023), or (iii) Steiner Tree between the given entities (He et al., 2024).

More advanced methods use a trainable retrieval component. E. g., Zhang et al. (2022) finetune RoBERTa (Liu et al., 2019), and Mavromatis and Karypis (2024) train a GNN for subgraph retrieval.

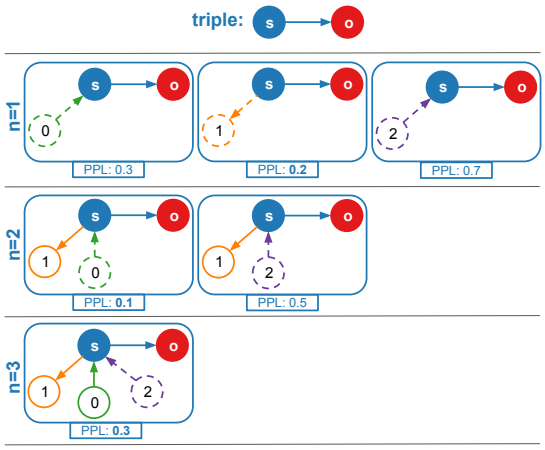


Figure 2: Schematic illustration of the greedy search algorithm for the post hoc selection of the most effective context for a given triple. With $|C| = 3$, the algorithm builds an effective context by iteratively selecting the neighbor that minimizes the perplexity of o.

4 Methodology

Given an LP query q, a trained model M, and the neighborhood $C \subseteq \mathcal{G}$, we investigate the potential benefits of providing an optimal context $C_{opt} \subseteq C$ instead of a randomly sampled context $C_{random} \subseteq C$ to M at inference time for answering q.

Post Hoc Optimal Context Selection In contrast to determining the context a priori for a query, as we are interested in understanding how an optimal context would look like, we search for a subset $C_{opt} \subseteq C$ that maximizes the model's likelihood of predicting the correct target t for a query with a known answer. Our approach is thus post hoc and assumes ground truth knowledge and can thus not be applied to (prospective) inference. Given that our research goal is to identify the characteristics of ideal context and not to propose a new state-of-the-art method, this approach is in line with our goals and research question.

An optimal context can be found by maximizing the confidence p_M of M in predicting the correct target entity. This is equivalent to minimizing the negative log-likelihood

$$C_{opt} = \underset{C' \subseteq C}{\arg\min} -p_M(t|s_q, r_q, C'). \quad (1)$$

Greedy Search for Optimal Context Selection An evaluation of Eq. 1 with all possible C' is not

tractable as C can have exponentially many subgraphs.[1] Therefore, we approximate C_{opt} with a local optimal $C_{\sim opt}$ that we derive through a greedy optimization that requires a manageable number of forward passes and is easily and generally applicable. Our greedy optimization works as follows: Given a triple (s_q, r_q, t), we start with $C' = \emptyset$ and iteratively extend the set by adding the context triple $c \in C$ that locally minimizes the negative log-likelihood. We thus iteratively compute the opticaml context as follows:

$$C'_i = \underset{c \in C \setminus C'_{i-1}}{\arg\min} -p_M(t|s_q, r_q, C'_{i-1} \cup c). \quad (2)$$

This yields a series of contexts $C' = [C'_1, ..., C'_{|C|}]$ from which we select

$$C_{\sim opt} = \underset{C'_i \in C'}{\arg\min} -p_M(t|s_q, r_q, C'_i). \quad (3)$$

For neighborhoods beyond the 1-hop, we ensure that $C_{\sim opt}$ forms a connected graph by selecting only triples connected to entities already in C'_i.

Optimal Context Selection with Generative LMs For an LP model based on a generative LM, the negative log-likelihood is equal to the perplexity of generating t's label. We denote the tokenized sequence as $t = t_1, ..., t_l$.

The perplexity (PPL) of a model M given $(s_q, r_q, ?)$ generating t is defined as

$$PPL(t|s_q, r_q) = \exp\left(-\frac{1}{l}\sum_{i=1}^{l} log M_{s_q,r_q}(t_i|t_{<i})\right), \quad (4)$$

where $M_{s,r}(t_i|t_{<i})$ denotes probability that the M generates t_i. Following Eq. 1, an optimal context for a generative LM is

$$C_{opt} = \underset{C' \subseteq C}{\arg\min}\, PPL(t|s_q, r_q). \quad (5)$$

5 Experimental Setup

In our experiments,[2] we aim to demonstrate that the greedy optimization derives approximately optimal neighborhoods and show the potential benefits of leveraging such neighborhoods for LP. Specifically, we first investigate whether greedy optimization can effectively reduce the model perplexity and the sizes of the neighborhoods for a given sample. Second, we assess the quality of the optimized neighborhoods by investigating: i) whether the optimized neighborhoods generalize to new models – i. e., whether the optimized neighborhoods improve the model if re-trained, ii) their effectiveness across different architectures – i. e., whether using the optimized neighborhood yields improved results across different model architectures, and iii) how the optimal neighborhoods compare to heuristic approaches from related work.

Models We evaluate two models: i) KGT5 (Kochsiek et al., 2023), which shows good scalability and performance comparable to state-of-the-art, and ii) Graph Language Model (GLM) from Plenz and Frank (2024), a general graph transformer for KGs that we adapt to LP.

The KGT5 model is based on the seq2seq T5 (Raffel et al., 2020) architecture, which requires linearizing the graph-structured input. However, the linearized input does not adequately reflect the graph structure, as the distance between entities in the linearized sequence does not correspond to their distance in the original graph. Consequently, the KGT5 model only incorporates the 1-hop neighborhood around the query subject entity, as complex graph structures can not be modeled.

The GLM does not linearize the input. Instead, the GLM modifies the relative positional encoding of tokens in the T5 encoder to reflect the distances between entities in the graph, thereby capturing structural information. As a result, the GLM can incorporate neighborhoods beyond the 1-hop level. Further details can be found in App. B.

We use pre-trained T5 weights and the same hyperparameters as reported by Saxena et al. (2022) and Kochsiek et al. (2023) for both models.

Datasets We conduct our experiments on two publicly available and well-established LP datasets: FB15k-237 and Wikidata5m. Statistics about both datasets can be found in App. C.

FB15k-237 is a refined subset of FB15k (Bordes et al., 2013b), which itself is derived from Freebase. Toutanova and Chen (2015) constructed FB15k-237 by filtering out inverse relations. The dataset consists of 310,116 triples, with 14,541 entities and 237 distinct relation types.

Wikidata5m is a large-scale subset of Wikidata, including only entities with descriptions of more than five words (Wang et al., 2021). The dataset

[1] E. g., the number of subgraphs for a 1-hop neighborhood C is $2^{|C|}$, where $|C|$ is the number of triples in C.

[2] The source code used for our experiments is publicly available on GitHub `https://anonymous.4open.science/r/kgt5-glm-336A/README.md`. The computational resources used for our experiments are reported in App. D.

contains 21 million triples, 5 million entities, and 822 relation types.

Evaluation We report the models' perplexity PPL as a measure of their confidence in making correct predictions, using it as an indicator of the progress of the context optimization process.

Ranking-based metrics are commonly used to evaluate LP models because LP models aim to rank the correct answer $o \in \mathcal{E}$ as highly as possible among all candidate entities $\mathcal{E}$. These metrics directly reflect how well the model orders these candidates. Following Bordes et al. (2013b), we evaluate the models' LP capabilities using filtered ranking-based metrics (MRR and Hits@k).

Baseline Context Given an LP model, we evaluate its performance using contexts constructed through different heuristic methods as baselines. More details can be found in App. E.

Random sampling (C_{random}): 100 triples are uniformly selected from the neighborhood around the query subject s_q.

Node degree (C_{degree}): We select the 10 neighboring entities with the highest node degree. We pass all triples connecting s_q with these neighbors.

Random walk with restart scores (C_{rwr}): We compute the random walk with restart (RWR) scores and use these to sort the neighbors. We then select the 10 triples associated with the neighboring entities with the highest RWR scores. We conduct our experiments with C_{rwr} only on FB15k-237 due to the high computational demands.

Entity linking (C_{link}): We run entity linking on the entity descriptions and restrict the context to triples involving those entities.

C_{degree}, C_{rwr}, and C_{link} are expected to reflect the importance of entities in the graph and, thus, contain less noise than C_{random}. We limit the context size to at most 10 neighboring entities in order to ensure comparability to $C_{\sim opt}$.

Context Optimization For each dataset, we train a KGT5 model with C_{random}, following Kochsiek et al. (2023). Then, we optimize a neighborhood $C_{\sim opt}$ up to a size of 30 for the KGT5 model on each dataset and each query.[3] We stop the optimization process at 30 triples, as most approximately optimal contexts contain only around 3-5 triples.

Dataset	C_{random}	$C^*_{\sim opt}$	rel diff. ↑
FB15k-237	3.361	1.732	**48.5%**
FB15k-237 w/ desc.	3.364	2.581	23.2%
Wikidata5m	4.466	3.485	21.9%
Wikidata5m w/ desc.	3.423	2.737	20.0%

Table 1: Comparision: Median perplexity of KGT5 on FB15k-237 and Wikidata5m for random sampling and optimized context.

We use these $C_{\sim opt}$ throughout all experiments and do not optimize any further contexts. To indicate that a context was derived specifically for a given model, we mark it with "*" – e. g., KGT5-$C^*_{\sim opt}$ denotes that the KGT5 model uses a context explicitly optimized for itself.

Optimization of 2-hop Contexts GLMs allow contexts beyond a 1-hop. Thus, we investigate the impact of a 2-hop $C_{\sim opt}$ for GLMs. First, we train a GLM model given a 2-hop C_{random}. For the 2-hop random sampling, we sample 10 direct neighboring triples in the first hop, and 10 for each of those in the second hop, leading to up to 110 sampled neighborhood triples. Then, we optimize $C_{\sim opt}$ up to a size of 30 per sample for each dataset and each model.

6 Results

Our baseline is the standard KGT5 model trained with random sampling.

6.1 Context Optimization

First, we validate our greedy context optimization by comparing the perplexities between the sampled and optimized contexts. As shown in Tab. 1, our approach decreases the median perplexity[4] of KGT5 models by 20.0% to 48.5%, depending on the dataset. As expected, the perplexity decreases more for datasets without descriptions, as models have to rely solely on the context here, which, thus, has a larger impact.

Tab. 2 shows the average size of different contexts. On average, $|C_{\sim opt}|$ is considerably smaller than the original $|C|$ and $|C_{random}|$. We note that $|C_{\sim opt}|$ is, on average, larger for FB15k-237 than for Wikidata5m, and larger for datasets with descriptions than for datasets without descriptions. Further investigating the effect of different datasets on optimal context sizes is left for future work.

[3] We optimize the context based on 100 randomly sampled triples around the query subject instead of the entire graph neighborhood to improve the runtime. For 0.66% of test instances on FB15k-237 and 0.62% on Wikidata5m, the 100 sampled triples fully capture the corresponding entity's neighborhood.

[4] We use median instead of mean, as the mean was dominated by a few outliers for some models.

Dataset	$\lvert C \rvert$	$\lvert C_{\text{random}} \rvert$	$\lvert C_{\text{degree}} \rvert$	$\lvert C_{\text{rwr}} \rvert$	$\lvert C_{\text{link}} \rvert$	$\lvert C_{\sim\text{opt}} \rvert$
FB15k-237	386.7	60.0	11.8	66.2	77.4	8.9
FB15k-237 w/ desc.	386.7	60.0	11.8	66.2	77.4	13.0
Wikidata5m	2443.8	45.5	8.2	-	262.3	4.6
Wikidata5m w/ desc.	2443.8	45.5	8.1	-	262.3	6.1

Table 2: Comparison: average context sizes per test query.

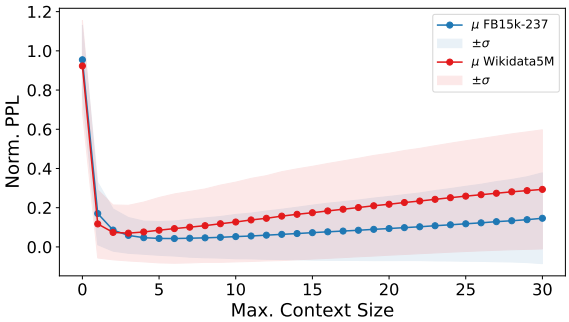


Figure 3: Perplexity across different context sizes during the $C^*_{\sim opt}$ creation. KGT5 w/o desc. on FB15k-237 and Wikidata5M.

The size of $C_{\sim opt}$ follows a long-tailed distribution, peaking around the average $C_{\sim opt}$ size, with only a few samples reaching the optimization limit of 30 neighbors (c. f. Fig. 7 in App. F).

Fig. 3 shows the model perplexity throughout the iterative context optimization.[5] We observe that the first few neighbors added to the context contribute the most to reducing the model perplexity, while adding additional neighbors results in either a minor decrease or even an increase. Furthermore, the PPL of the Wikidata5m models decreases earlier than that of the FB15k-237 model and, after reaching its minimum, rises more sharply.

6.2 Evaluating the Quality of Optimized Contexts

We investigate whether optimized contexts can improve a model's LP capabilities.

The results are shown in Tab. 3. Optimizing a context for a specific model and then evaluating the same model with this context leads to great performance increases – e. g., the MRR for KGT5 w/o desc. improves from 0.278 (for C_{random}) to 0.616. However, the extent of these improvements varies depending on the model and dataset, with notably smaller improvements observed for Wikidata5m.

[5]These results are aggregated over the entire test set, where the perplexity optimization curve for each sample was normalized to a range between 0 and 1 before the mean and variance were computed. This normalization ensures a reasonable variance in the perplexity values. We provide the perplexity curve for multiple individual samples as case studies in App. G.

Generalization Across Models So far, we investigated $C^*_{\sim opt}$, i. e., the optimized context applied to the model for which it was optimized. Thus, the context may be overfitted to that specific model. To assess the generalizability of the optimized context, we apply $C_{\sim opt}$ to separately finetuned KGT5 models. From Tab. 3, we observe that although the MRR score decreases from 0.616 to 0.362 (FB15k-237 w/o desc.) when applied to a new model, this score remains higher than that of C_{random}, which is around 0.278. This trend holds across all investigated model configurations and datasets. We conclude that $C_{\sim opt}$ generalizes to new models, though a large gap remains to the model it was originally optimized for.

Generalization Across Model Architectures In the next step, we investigated whether $C_{\sim opt}$ also generalizes to models of different architectures. As shown in Tab. 3, models incorporating $C_{\sim opt}$ outperform those using C_{random}, although the gain is smaller than for KGT5.

Comparison to Heuristic Approaches If a model is trained using C_{random} but evaluated with C_{degree}, C_{rwr}, or C_{link}, its performance decreases compared to when it is evaluated with C_{random}. It is important to note that we constructed C_{degree}, C_{rwr}, and C_{link} to contain a reasonable number of triples (see Tab. 2) to obtain a similar size as the optimized contexts and, thereby, ensure comparability.

6.3 Ablation Study: Model Sensitivity to the Sequential Order in Optimized Contexts

In an ablation study, we examine whether KGT5 models are sensitive to the order induced by the iterative context optimization strategy. We evaluate one model using the context in its original order, as determined during optimization (KGT5-$C_{\sim opt}$), and another with a shuffled context (KGT5-$C_{\sim opt}$ shuf.). Interestingly, the LP scores remain similar, suggesting that KGT5 models learn to be order-invariant during training. As GLMs are order-invariant by design, we omit input shuffling.

6.4 Ablation Study: Small Context Models

So far, all our models were trained with a randomly sampled context of up to 100 triples (60 on average). However, our optimized contexts have only 4.6 to 13.0 triples on average, depending on the dataset. Thus, the optimized contexts may be even more beneficial for models trained with smaller

	Model	FB15k-237				Wikidata5m			
		MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
w/ desc.	KGT5-$\emptyset$	0.184	0.135	0.199	0.287	0.144	0.116	0.157	0.199
	KGT5-C_{degree}	0.278	0.212	0.307	0.419	0.403	0.384	0.417	0.435
	KGT5-C_{link}	0.273	0.211	0.300	0.402	0.378	0.357	0.393	0.416
	KGT5-C_{rwr}	0.278	0.212	0.305	0.416	-	-	-	-
	KGT5-C_{random}	0.285	0.216	0.313	0.430	0.430	0.409	0.444	0.463
	GLM-C_{random}	0.284	0.215	0.316	0.429	0.424	0.405	0.438	0.457
	KGT5-$C^*_{\sim opt}$	0.402	0.322	0.443	0.561	0.455	0.436	0.467	0.487
	KGT5-$C_{\sim opt}$	0.328	0.252	0.364	0.483	0.441	0.421	0.453	0.475
	KGT5-$C_{\sim opt}$ shuf.	0.323	0.247	0.358	0.478	0.441	0.421	0.453	0.475
	GLM-$C_{\sim opt}$	0.294	0.224	0.327	0.437	0.434	0.416	0.448	0.466
w/o desc.	KGT5-$\emptyset$	0.135	0.098	0.148	0.211	0.197	0.162	0.217	0.263
	KGT5-C_{degree}	0.244	0.185	0.268	0.368	0.354	0.328	0.370	0.402
	KGT5-C_{link}	0.219	0.167	0.240	0.329	0.306	0.277	0.324	0.361
	KGT5-C_{rwr}	0.240	0.181	0.266	0.364	-	-	-	-
	KGT5-C_{random}	0.278	0.208	0.306	0.423	0.381	0.355	0.398	0.429
	GLM-C_{random}	0.284	0.211	0.315	0.436	0.377	0.353	0.391	0.423
	KGT5-$C^*_{\sim opt}$	0.616	0.534	0.664	0.776	0.416	0.392	0.430	0.459
	KGT5-$C_{\sim opt}$	0.362	0.286	0.400	0.514	0.396	0.370	0.411	0.443
	KGT5-$C_{\sim opt}$ shuf.	0.361	0.286	0.398	0.512	0.396	0.371	0.410	0.443
	GLM-$C_{\sim opt}$	0.324	0.253	0.359	0.467	0.394	0.371	0.407	0.439

Table 3: Comparison of ranking-based LP scores on FB15k-237 and Wikidata5m.

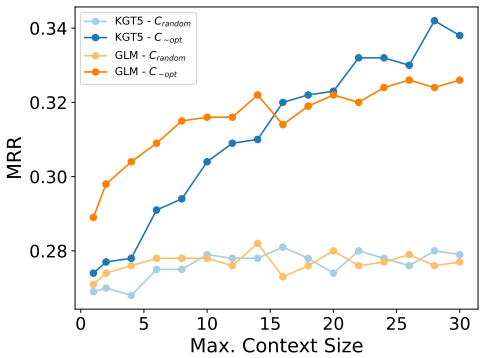


Figure 4: MRR scores of models, trained with different context sizes, when provided with an optimized context ($C_{\sim opt}$) vs. with random context (C_{random}).

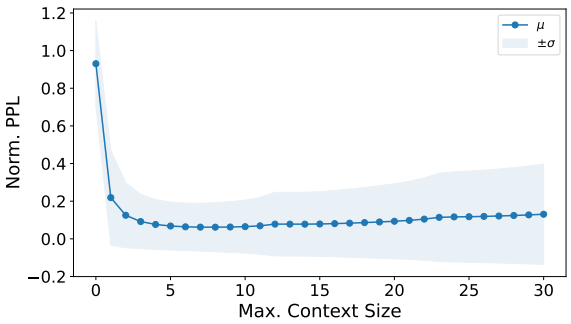


Figure 5: Perplexity across different context sizes during the $C^*_{\sim opt}$ creation for 2-hops. GLM w/o desc. on FB15k-237.

contexts. Therefore, we train multiple models with C_{random} of varying sizes, ranging from 0 to 30. During the evaluation, we provide the optimized contexts (optimized for our baseline model) up to the context size with which the model was trained.

Fig. 4 shows the results. We observe that the MRR increases only gradually as the sampled context size grows. However, when evaluated with $C_{\sim opt}$, the MRR shows a fast improvement with increased context sizes. Interestingly, a small optimized context (around 5-10 triples) already outperforms larger random contexts of 30 triples. Additionally, we find that the GLM model performs better with smaller contexts, whereas the KGT5 model shows superior results with larger contexts. Around 15 neighbors, both models achieve similar performance with optimized contexts.

6.5 Optimization for 2-hop Contexts

So far, we optimized the 1-hop contexts based on KGT5. Using GLMs, we can extend our experiments to larger contexts. Due to computational costs, we restrict ourselves to 2-hop contexts and FB15k-237. The optimization reduces the median PPL from 3.308 to 2.770, as visualized in Fig. 5. Thus, the PPL follows a similar trend to the one observed in previous experiments with the 1-hops.

We observe that triples from the second hop are incorporated into the optimized context quite early, indicating that 2-hop triples provide valuable information for LP. As shown in Fig. 6, more than 80% of the triples at the fifth position in the optimized context are from the second hop.

Tab. 4 shows that context optimization can improve the model's LP scores. However, we observe that the contexts generalize worse from one model

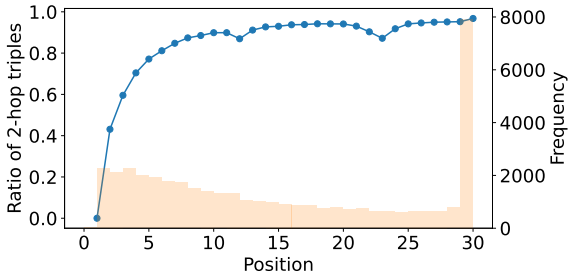


Figure 6: Ratio of 2-hop triples at different positions of $C_{\sim opt}$. The histogram indicates the frequency of the respective $C_{\sim opt}$ sizes.

Model	MRR	Hits@1	Hits@3	Hits@10
GLM-C_{random}	0.278	0.203	0.309	0.435
GLM-$C^*_{\sim opt}$	0.364	0.286	0.402	0.523
GLM-$C_{\sim opt}$	0.306	0.231	0.338	0.461

Table 4: Comparision of ranking-based LP scores on FB15k-237 with 2-hop contexts: C_{random} vs. $C_{\sim opt}$. Models w/o desc.

to another compared to the 1-hop contexts. In general, the 2-hop results are slightly worse than the 1-hop results. For the 2-hop model, we did not evaluate the context selection heuristics, as we have already shown that they do not provide a valuable context for the 1-hop models.

7 Discussion

We optimize contexts for LP inference to show the potential improvements of using a good context. Our results show that the greedy post hoc context optimization reduces the models' median perplexity and improves ranking-based LP metrics while yielding an optimized context $C_{\sim \text{opt}}$ that is smaller than a quarter of the randomly sampled C_{random}.

The characteristics of a good context may vary depending on whether it is used during training or inference. E. g., during training, a context should contain some noise to make the model robust, whereas, during inference, the context should contain as little noise as possible. Nonetheless, a context optimized for inference may serve as a valuable starting point for training contexts.

Due to the post hoc optimization, the context is optimized towards a specific target, i. e., we cannot optimize the context for a given query and target, and then reuse it for predicting new targets. We confirm this empirically in App. H. Thus, our approach can not be directly applied to predict new targets. However, it offers insights to enhance context selection and thereby improve LP.

Although models with entity descriptions typically achieve better LP scores, our optimized context experiments on FB15-237 show the opposite, that is, the model without descriptions outperforms the model provided with descriptions. We conclude that i) descriptions can be misleading or distracting when high-quality neighbors are available – or lead to truncations of good contexts; ii) descriptions, on average, offer more valuable information than randomly sampled contexts.

KGT5's graph linearization only applies to 1-hop contexts, so that we evaluated 2-hop contexts only for the GLMs. In general, the 2-hop results are slightly worse than the 1-hop results for randomly sampled contexts. We hypothesize that this is because, on average, 1-hop neighbors contain more relevant information than more distant neighbors. When we randomly select neighbors that are up to 2 hops away, then fewer 1-hop neighbors are selected with the result that LP performance decreases.

Although we consider 1-hop triples to be more informative than 2-hop triples, we observe a surprising, yet recurring pattern in the context optimization: often, one direct neighbor (1-hop) is added first, followed by all of its neighbors (2-hop) before additional direct neighbors of the query subject are included. This pattern is reflected in Fig. 6, where the ratio of 2-hop triples drops at context positions 12 and 23. This behavior is also visible in the perplexity optimization, as shown in Fig. 5. We believe that this pattern arises because the greedy algorithm gets stuck in a local optimum. However, despite this behavior, our results show that greedy context optimization can be effective for LP.

8 Conclusion & Future Work

In this work, we introduce a method to determine an approximately optimal context for a given query, target, and model that effectively improves a model's likelihood of predicting the correct target by reducing the perplexity by 20.0% to 48.5%. Second, we demonstrate that the same information is relevant to different models. Finally, we show that optimal contexts are smaller than a quarter of the randomly sampled contexts and that common sampling strategies are suboptimal.

Based on our results, future work can optimize the context sampling for LP training, e. g., by mining frequent patterns in the contexts or training a model to predict contexts.

Limitations

We are aware of three limitations of our approach:

i) Our context optimization uses a greedy strategy, which can get stuck in local optima. For example, suppose we have three possible triples x, y, and z. Adding x might reduce the negative log-likelihood more than adding y in the current step. However, if z is added next, the combination $\{y, z\}$ might lead to a better overall result than $\{x, z\}$. This shows that greedy optimization can, in theory, miss better combinations of triples. However, if getting stuck in local optima were a serious problem, we would expect a significant drop in negative log-likelihood when triples are added later in the optimization process. However, our case studies in Fig. 8 show a stable and consistent decrease in negative log-likelihood over time, without sudden jumps that would suggest escaping a local optimum. Therefore, we believe that our greedy optimization is not strongly affected by local optima.

ii) The context optimization is performed post hoc, i. e., it requires the correct query answer to determine the optimal context for a given query. As a result, our algorithm cannot be directly used to generate optimal context to enhance existing link prediction methods. Nonetheless, we believe that our insights into the potential improvements and characteristics of optimal contexts offer a valuable contribution to the community and can support the development of more advanced context selection strategies.

iii) We also use random sampling as a starting point due to computational constraints. However, we consider large subgraphs for random sampling, so the impact should not be too large for our experiments.

Ethics

Our work builds on established datasets and methods and does not introduce any new risks related to bias or harmful content. Instead, our findings can support the development of more effective and computationally efficient link prediction approaches.

References

Zhen Bi, Siyuan Cheng, Jing Chen, Xiaozhuan Liang, Feiyu Xiong, and Ningyu Zhang. 2023. Relphormer: Relational graph transformer for knowledge graph representations. *Neurocomputing*, page 127044.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013a. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013b. Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, 26.

Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y. Hammerla. 2018. Relational graph attention networks. *ArXiv*, abs/1904.05811.

Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*.

Daniel Daza, Michael Cochez, and Paul Groth. 2021. Inductive entity representations from text via link prediction. In *Proceedings of the Web Conference 2021*, WWW '21, page 798–808, New York, NY, USA. Association for Computing Machinery.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Adrian Kochsiek and Rainer Gemulla. 2023. A benchmark for semi-inductive link prediction in knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10634–10643, Singapore. Association for Computational Linguistics.

Adrian Kochsiek, Apoorv Saxena, Inderjeet Nair, and Rainer Gemulla. 2023. Friendly neighbors: Contextualized sequence-to-sequence link prediction. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 131–138, Toronto, Canada. Association for Computational Linguistics.

Irene Li and Boming Yang. 2023. NNKGC: Improving Knowledge Graph Completion with Node Neighborhoods. In *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2023)*.

Qingyang Li, Yanru Zhong, and Yuchu Qin. 2024. MoCoKGC: Momentum contrast entity encoding for knowledge graph completion. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14940–14952, Miami, Florida, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Kangyang Luo, Yuzhuo Bai, Cheng Gao, Shuzheng Si, Yingli Shen, Zhu Liu, Zhitong Wang, Cunliang Kong, Wenhao Li, Yufei Huang, Ye Tian, Xuantang Xiong, Lei Han, and Maosong Sun. 2025. Gltw: Joint improved graph transformer and llm via three-word language for knowledge graph completion. *Preprint*, arXiv:2502.11471.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning. *Preprint*, arXiv:2405.20139.

Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, and Pinar Duygulu. 2004. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 653–658, New York, NY, USA. Association for Computing Machinery.

Jinyoung Park, Minseok Joo, Joo-Kyung Kim, and Hyunwoo J. Kim. 2024. Generative subgraph retrieval for knowledge graph–grounded dialog generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21167–21182, Miami, Florida, USA. Association for Computational Linguistics.

Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024. Graph retrieval-augmented generation: A survey. *Preprint*, arXiv:2408.08921.

Hao Peng, Ruitong Zhang, Yingtong Dou, Renyu Yang, Jingyi Zhang, and Philip S. Yu. 2021. Reinforced neighborhood selection guided multi-relational graph neural networks. *ACM Transactions on Information Systems (TOIS)*.

Moritz Plenz and Anette Frank. 2024. Graph language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4477–4494, Bangkok, Thailand. Association for Computational Linguistics.

Moritz Plenz, Juri Opitz, Philipp Heinisch, Philipp Cimiano, and Anette Frank. 2023. Similarity-weighted construction of contextualized commonsense knowledge graphs for knowledge-intense argumentation tasks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6130–6158, Toronto, Canada. Association for Computational Linguistics.

Chenyu Qiu, Pengjiang Qian, Chuang Wang, Jian Yao, Li Liu, Fang Wei, and Eddie Y.k. Eddie. 2024. Joint pre-encoding representation and structure embedding for efficient and low-resource knowledge graph completion. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15257–15269, Miami, Florida, USA. Association for Computational Linguistics.

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, Dublin, Ireland. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

Phillip Schneider, Tim Schopf, Juraj Vladika, Mikhail Galkin, Elena Simperl, and Florian Matthes. 2022. A decade of knowledge graphs in natural language processing: A survey. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 601–614, Online only. Association for Computational Linguistics.

Dhaval Taunk, Lakshya Khanna, Siri Venkata Pavan Kumar Kandru, Vasudeva Varma, Charu Sharma, and Makarand Tapaswi. 2023. Grapeqa: Graph augmentation and pruning to enhance question-answering. In *Companion Proceedings of the ACM Web Conference 2023*, WWW '23 Companion, page 1138–1144, New York, NY, USA. Association for Computing Machinery.

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

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, Dublin, Ireland. Association for Computational Linguistics.

Ruoyao Wang, Graham Todd, Ziang Xiao, Xingdi Yuan, Marc-Alexandre Côté, Peter Clark, and Peter Jansen. 2024. Can language models serve as text-based world simulators? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–17, Bangkok, Thailand. Association for Computational Linguistics.

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.

Fei Xiong, Haoran Sun, Guixun Luo, Shirui Pan, Meikang Qiu, and Liang Wang. 2024. Graph attention network with high-order neighbor information propagation for social recommendation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 2478–2486. International Joint Conferences on Artificial Intelligence Organization. Main Track.

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*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *Preprint*, arXiv:1909.03193.

Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D. Manning, Percy Liang, and Jure Leskovec. 2022. Deep bidirectional language-knowledge graph pretraining. In *Neural Information Processing Systems (NeurIPS)*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 974–983, New York, NY, USA. Association for Computing Machinery.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5773–5784, Dublin, Ireland. Association for Computational Linguistics.

Dataset	FB15k-237	Wikidata5m
Number of triples (training set)	272115	20614279
Number of triples (validation set)	17535	5163
Number of triples (test set)	20466	5133
Number of entities	14541	4594485
Number of distinct subject entities	13781	4573330
Number of distinct object entities	13379	1068854
Number of distinct relations	237	822
Relation occurrences (least frequent)	37	1
Relation occurrences (most frequent)	15989	3839805
Mean node degree	37.520	8.974
Maximum node degree	7614	1519673
Graph density (directed)	1.29e-03	9.77e-07

Table 5: LP dataset statistics.

A Details about Models

B Graph Language Model

Normal Language Models (LMs) operate on texts, where one token comes after the other. So-called positional encodings inform the language model about the sequential ordering of input texts. Graph Language Models (GLMs; Plenz and Frank, 2024) extend the positional encoding of langauge models in order to encode graphs instead of sequences. This allows GLMs to encode graphs efficiently, as they are a type of graph transformer. As only the positional encoding has to be adjusted, the pre-trained LM paramters from LMs can be used for GLMs. This enables GLMs to process text as a LM would, while also being able to process graphs.

We use the global GLM (gGLM) from Plenz and Frank (2024), which is based on the T5 encoder. For text generation, we pass the graph encoding to the T5 decoder without any modifications.

C Link Prediction Datasets

The characteristics of the two investigated LP benchmark datasets are shown in Tab. 5.

D Computational Resources

All experiments were conducted on our GPU cluster equipped with A40 GPUs. The runtime for a single run was as follows: i) training required approximately 24 hours on FB15k-237 and 100 hours on Wikidata5m, ii) context optimization took 24 hours (FB15k-237) and 2 hours (Wikidata5m), and iii) evaluation lasted 7 hours (FB15k-237) and 1 hour (Wikidata5m). Since we repeated these experiments multiple times for multiple configurations, the total compute amounted to approximately 1500 GPU hours.

Dataset	Node Degree
FB15k-237	36.2
Wikidata5m	29.8

Table 6: Average node degree of entities in the test set.

E Context Selection Baselines

E.1 Node Degree

We compute the undirected node degree. The average node degree is shown in Tab. 6.

E.2 Random Walk with Restarts

We compute the random walk with restart scores with a start probability of 0.15.

E.3 Entity Linking

We run entity linking on the entity descriptions and restrict the context to triples involving those entities. We use a Wikidata entity linking tool based on SpaCy.[6] If no entities can be linked, we randomly sample a neighborhood of a maximum size of 100 around the query source without any restrictions.

For the FB15k-237 dataset, we mapped the Wikidata IDs back to Freebase. 82% of the entity descriptions in the FB15k-237 test set have at least one link. In the Wikidata5m test set, 66% of the entities have at least one link.

F Neighborhood Optimization Details

C_{opt} is not necessarily unique, as different contexts can result in the same prediction confidence of M. If we find multiple equally valuable cotexts, we are interested in the smallest C_{opt} possible.

Fig. 7 shows histograms of the sizes of the optimized contexts.

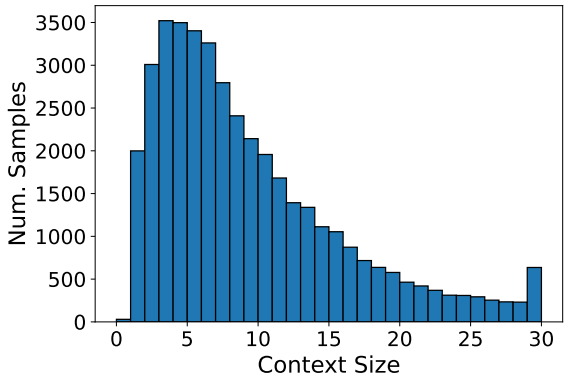
G Case Study: Context Optimization

Fig. 8 shows how the greedy algorithm optimizes the perplexity for 6 random test samples from the FB15k-237 dataset.

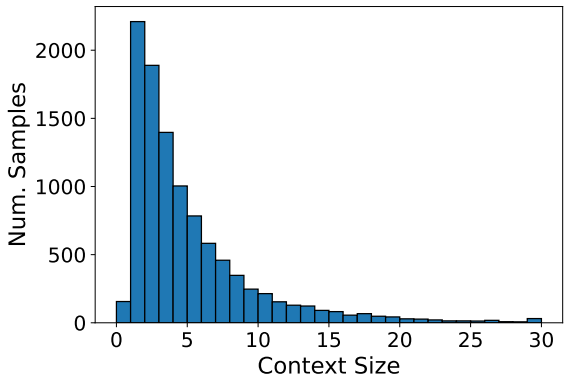
H Generalization of Neighborhoods from Train to Test

The greedy optimization derives an effective post hoc context; therefore, the context depends on the triple, i. e., the query and the correct target entity.

[6] See https://github.com/egerber/spaCy-entity-linker.



(a) FB15k-237



(b) Wikidata5m

Figure 7: Overview of the frequency of optimal context sizes. KG-T5 w/o context.

However, some relations can connect one entity with multiple entities; these are called 1-n relations. Consequently, one query can have multiple correct answers.

This raises the question: Are the effective contexts triple-specific, or do they generalize and are query-specific?

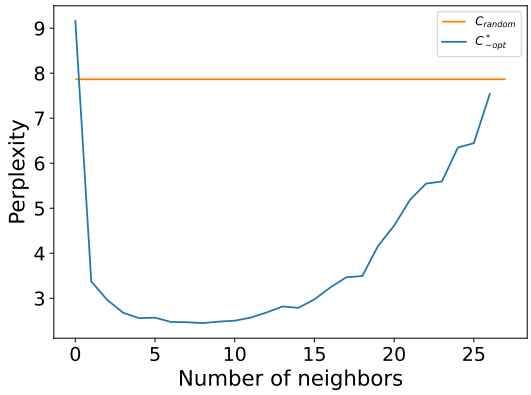
We investigate whether the context of a given query derived for one target generalizes to other targets. The queries of the train and test set of LP datasets usually have a significant overlap, i. e., many test queries are already contained in the training set. This is caused by the random splitting of triples with 1-n relations into train and test sets. E. g., 68.5% of the FB15k-237 test queries are present in the train set, see Tab. 7 for details.

In order to investigate whether the optimized neighborhoods generalize from train to test, we optimize test samples associated with queries in the test set. During inference, we either use these neighborhoods optimized on the training queries if available, or we stay with random sampling.

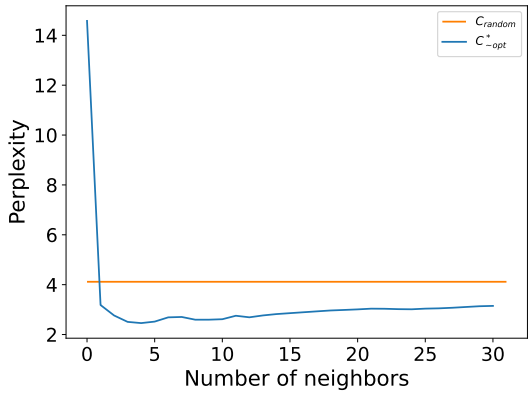
We run this experiment with a KGT5 model on FB15k-237 and observe that the MRR score is significantly worse (0.226) compared to when random sampling is conducted (0.278). We conclude that the neighborhood is highly target-dependent and does not generalize to new triples with the same query aiming for a different target.

Dataset	Train Queries	Test Queries	Intersection
FB15k-237	243,061	36,587	25,058 (68.5%)
Wikidata5m	36,518,652	13,235	4,897 (37.0%)

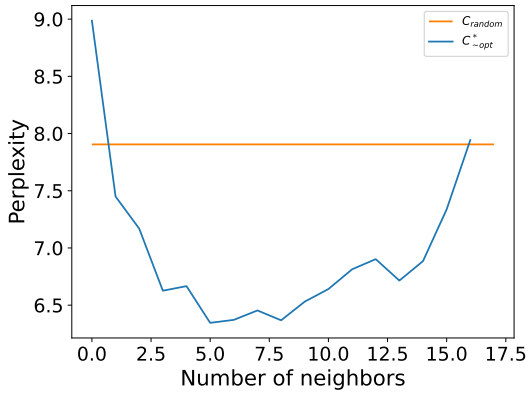
Table 7: Overlap of queries contained in the train and test set.



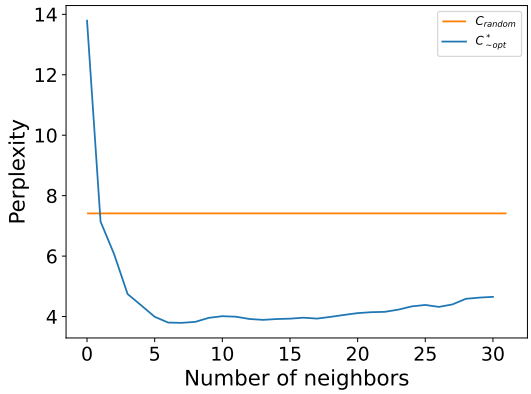
(a) Sample 1



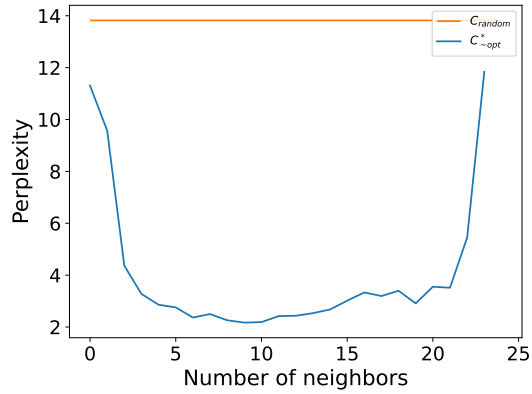
(b) Sample 2



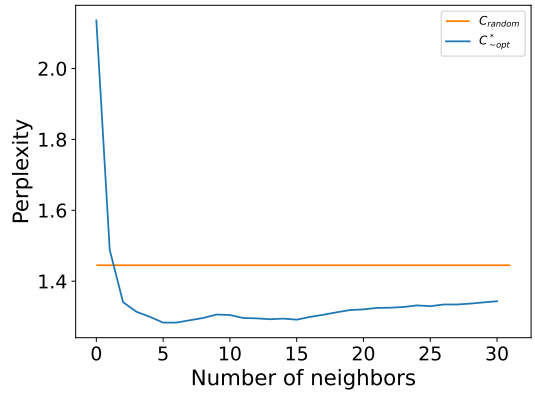
(c) Sample 3



(d) Sample 4



(e) Sample 5



(f) Sample 6

Figure 8: PPL throughout the greedy neighborhood optimization for KGT5 (w/o desc.) for six random samples in the FB15k-237 dataset.