Are Missing Links Predictable? An Inferential Benchmark for Knowledge Graph Completion

Yixin Cao[1][2] Xiang Ji[1] Xin Lv[3]
Juanzi Li[3] Yonggang Wen[1] Hanwang Zhang[1]
[1]School of CSE, Nanyang Technological University, Singapore
[2]S-Lab, Nanyang Technological University, Singapore
[3]Department of CST, Tsinghua University, Beijing, China
caoyixin2011@gmail.com, lv-x18@mails.tsinghua.edu.cn
lijuanzi@tsinghua.edu.cn, {ygwen,hanwangzhang}@ntu.edu.sg

Abstract

We present InferWiki, a Knowledge Graph Completion (KGC) dataset that improves upon existing benchmarks in inferential ability, assumptions, and patterns. First, each testing sample is predictable with supportive data in the training set. To ensure it, we propose to utilize rule-guided train/test generation, instead of conventional random split. Second, InferWiki initiates the evaluation following the open-world assumption and improves the inferential difficulty of the closed-world assumption, by providing manually annotated negative and unknown triples. Third, we include various inference patterns (e.g., reasoning path length and types) for comprehensive evaluation. In experiments, we curate two settings of InferWiki varying in sizes and structures, and apply the construction process on CoDEx as comparative datasets. The results and empirical analyses demonstrate the necessity and high-quality of InferWiki. Nevertheless, the performance gap among various inferential assumptions and patterns presents the difficulty and inspires future research direction. Our datasets can be found in https://github.com/TaoMiner/inferwiki.

1 Introduction

Knowledge Graph Completion (KGC) aims to predict missing links in KG by inferring new knowledge from existing ones. Attributed to its reasoning ability, KGC models are crucial in alleviating the KG's incompleteness issue and benefiting many downstream applications, such as recommendation (Cao et al., 2019b) and information extraction (Hu et al., 2021; Cao et al., 2020a). However, the KGC performance on existing benchmarks are still unsatisfactory — 0.51 Hit Ratio@1 and 187 Mean Rank of the top-ranked model (Wang et al., 2019) on the widely used FB15k237 (Toutanova and Chen, 2015). Do we have a slow progress of

	Head	Predicate	Tail
Test	David	location	? (Ans: Florida)
Train	David David	placeOfBirth nationality	Atlanta U.S.A.
Test	Zurich	travelMonth	? (Ans: October)
Train	Zurich	travelMonth	Jan., Feb., Mar., Apr., May., Jun., Jul., Aug., Sep., Nov., Dec.

Table 1: Low-quality examples in FB15k237. We only present related triples. Ans denotes the missing entity.

models (Akrami et al., 2020)? Or should we blame for the low-quality of benchmarks?

In this paper, we re-think the task of KGC and construct a new benchmark dubbed InferWiki that highlights three fundamental objectives:

Test triples should be inferential: this is the essential requirement of KGC. Each test triple should have supportive samples in the train set. However, we observe two major issues of current KGC datasets: unpredictable and meaningless test triples, which may hinder evaluating and advancing state-of-the-arts. As shown in Table 1, the first example of inferring the location for David (i.e., Florida) is even impossible for humans — not to mention machines — merely based on his birthplace and nationality (i.e., Atlanta and USA). In contrast, the second one is predictable but meaningless to find the missing month from a list of months within a year. The above cases are very common in existing datasets, e.g., YAGO3-10 (Dettmers et al., 2018) and CoDEx (Safavi and Koutra, 2020), mainly due to their construction process: first collecting a high-frequency subset of entities and then randomly splitting their triples into train/test. In this setting, KGC models may be over- or under-estimated, as we are even unsure if a human can perform better.

Test triples may be inferred positive, negative, or unknown. Following open-world assumption: what is not observed in KG is not necessar-

	FB15k237	**WN18RR**	**YAGO3-10**	**CoDEx-m**	**Kinship**	**Country**	**InferWiki16k/64k**	
Source	FreeBase	WordNet	YAGO	Wikidata	Artificial		Wikidata	
Inferential	✗	✗	✗	✗	✓	✓	✓	✓
#Entity	14,541	40,943	123,182	17,050	104	272	16,288	64,718
#Relation	237	11	37	51	26	2	197	239
#train	272,115	86,835	1,079,040	185,584	8,548	1,111	162,424	782,243
#valid (+)	17,535	3,034	5,000	10,310	1,069	24	3,398	7,747
(-\UNK)	-\-	-\-	-\-	10,310\-	-\-	-\-	1,910\1,456	6,125\1,605
#test (+)	20,466	3,134	5,000	10,311	1,069	24	3,398	7,747
(-\UNK)	-\-	-\-	-\-	10,311\-	-\-	-\-	1,868\1,501	6,062\1,685

Table 2: Statistics of KGC datasets. A more detailed survey table can be found in Appendix A.

ily false, but unknown (Shi and Weninger, 2018). However, existing benchmarks generate unseen triples as negatives (i.e., the closed-world assumption), because KG contains only positive triples. They usually randomly corrupt the head or tail entity in a triple, sometimes with type constraints (Li et al., 2019a). This leads to trivial evaluation (almost 100% accuracy in triple classification (Safavi and Koutra, 2020)). Besides, the lack of unknown test ignores a critical inference capacity and may cause false negative errors in knowledge-driven tasks (Kotnis and Nastase, 2017).

Inference has various patterns. Concentrating on limited patterns in evaluation may bring in severe bias. Domain-specific datasets Kinship (Kemp et al., 2006) and Country (Bouchard et al., 2015) only focus on a few relations and are nearly solved (Das et al., 2017). General-domain WN18RR (Dettmers et al., 2018) contains prevalent symmetry relation types, which incorrectly boosts the performance of RotatE (Abboud et al., 2020). Clearly, limited patterns leads to unfair comparisons among KGC models.

To this end, we curated an **Infer**ential KGC dataset extracted from **Wiki**data and establish the benchmark with two settings of varying in sizes and structures: **InferWiki64k** and **InferWiki16k**. Instead of random split, we mine rules via AnyBURL (Meilicke et al., 2019) to guide train/test generation. All test triples are thus guaranteed inferential from training data. To avoid the rule leakage, we utilize two sets of triples: a large set for high-quality rule extraction and a small set for train/test split. Moreover, we infer unseen triples and manually annotate them with positive, negative and unknown labels to improve the difficulty of evaluation following both closed-world and open-world assumptions. For inference patterns, we include and balance triples with different reasoning path length, relation types and patterns (e.g., symmetry and composition).

Our contributions can be summarized as follows:

- We summarize three principles of KGC: inferential ability, assumptions and patterns, and construct a rule-guided dataset.
- We highlight the importance of negatives and unknowns, and initiate open-world evaluation.
- We conduct extensive experiments to establish the benchmark. The results and deep analyses verify the necessity and challenge of InferWiki, providing insights for future research.

2 Related Work

We can roughly classify current KGC datasets into two groups: inferential and non-inferential datasets. The first group is usually manually curated to ensure each testing sample can be inferred from training data through reasoning paths, while they only focus on specific relations, such as Families (Garcia-Duran et al., 2015), Kinship (Kemp et al., 2006), and Country (Bouchard et al., 2015). The limited scale and inference patterns make them not challenging. HOLE (Nickel et al., 2016) achieves 99.7% ACU-PR on the dataset of Country.

The second group of datasets are automatically derived from public KGs and randomly split positive triples into train/test, leading to a risk of testing samples non-inferential from training data. Popular datasets include FB15k-237 (Toutanova and Chen, 2015), WN18RR (Dettmers et al., 2018), and YAGO3-10 (Dettmers et al., 2018). CoDEx (Safavi and Koutra, 2020) argues the scope and difficulty of the above datasets, thus propose a comprehensive dataset with manually verified hard negatives.

In fact, inference is an important ability for intelligence. Various fields study how inference is done in practice, ranging from logic to cognitive psychology. Inference helps people make reliable predictions, which is also an expected ability for AI models. Indeed, once deployed, a model may have to make a prediction when there is no evidence in the training set. But, instead of an unreliable guess, we highlight the ability to know unknown, a.k.a. open-world assumption. Therefore, we aim to curate an large-scale inferential benchmark InferWiki including various inference patterns and testing samples (i.e., positive, negative, and unknown), for better evaluation. We list the statistics in Table 2.

3 Dataset Design

We describe our dataset construction that comprises four steps: data preprocessing, rule mining, rule-guided train/test generation, and inferred test labeling. We then give a detailed analysis.

3.1 Data Preprocessing

More and more studies utilize Wikidata[1] as a knowledge resource due to its high quality and large quantity. We utilize the September 2019 English dump in experiments. Data preprocessing aims to define relation vocabulary and extract two sets of triples from Wikidata: a large one for rule mining $\mathcal{T}^r$ and a relatively small one for dataset generation $\mathcal{T}^d$. The reason for using two sets is to avoid the leakage of rules. In other words, some frequent rules on the large set may be very few on the small set. The different distributions shall avoid that rule mining methods will easily achieve high performance. Besides, more triples can improve the quality of mined rules. In contrast, the relatively small set is enough for efficient KGC training and evaluation.

In specific, we first extract all triples that consist of two entity items and one relation with English labels. We then remove the repeated triples and obtain 40,199,175 triples with 7,734,841 entities and 1,170 different relation types. Considering rule mining efficiency, we reduce the relation vocabulary by (1) manually filtering out meaningless relations, such as movie ID or film rating, (2) removing relations of *InstanceOf* and *subClassOf* following existing benchmarks (Toutanova and Chen, 2015), (3) select the most frequent 500 relation types. We focus on the most frequent 800,000 entities, which result in 8,632,777 triples as the large set for rule mining. To obtain the small set for dataset construction, we further select the most frequent 120,000 entities and 300 relations, which result in 1,283,246 triples. Note that we also infer new triples and label them as positive, negative, or unknown later.

[1]https://www.wikidata.org/

3.2 Rule Mining

Since developing advanced rule mining models is not the focus of this paper and several mature tools are available online, such as AMIE+ (Galárraga et al., 2015) and AnyBURL (Meilicke et al., 2019). We utilize AnyBURL[2] in experiments due to its efficiency and effectiveness.

Given a set of triples (i.e., the large set $\mathcal{T}^r$), this step aims to automatically learn rules $\mathcal{F} = \{(f_p, \lambda_p)\}_{p=1}^{P}$, where f_p denotes a horn rule, e.g., $\mathit{spouse}(x, y) \wedge \mathit{father}(x, z) \Rightarrow \mathit{mother}(y, z)$, and $\lambda_p \in [0, 1]$ denotes the confidence of f_p. For each rule f_p, the left side of $\Rightarrow$ is called the premise, and the right side is called the conclusion, where the conclusion contains a single atom and the premise is a conjunction of several atoms in the Horn rule scheme. We can ground specific entities to replace x, y, z in f_p, which shall denote an inferential relationship between premise and conclusion triples. For example, given *spouse*(LeBron James, Savannah Brinson) and *father*(LeBron James, Bronny James), we may infer a new triple *mother*(Savannah Brinson, Bronny James).

Of course, not all of the mined rules are reasonable. To alleviate the negative impacts of unreasonable rules, we rely on more data (a large set of triples) and keep high-confidence rules only. Particularly, we follow the suggested configuration of AnyBURL. We run it for 500 seconds to ensure that all triples can be traversed at least once and obtain 251,317 rules, where 168,996 out of them whose confidence meets $\lambda_p > 0.1$ have been selected as the rule set to guide dataset construction.

3.3 Rule-guided Dataset Construction

Different from existing benchmarks, InferWiki provides inferential testing triples with supportive data in the training set. Moreover, it aims to include as many inference patterns as possible and these patterns are better evenly distributed to avoid biased evaluation. Thus, this step has four objectives: rule-guided split, path extension, negative supplement, and inference pattern balance.

Rule-guided Split grounds the mined rules $\mathcal{F}$ on triples $\mathcal{T}^d$ to obtain premise triples and corresponding conclusion triples. All premise triples form a training set, and all conclusion triples form a test set. Thus, they are naturally guaranteed to be inferential. For correctness, all of premise triples

[2]http://web.informatik.uni-mannheim.de/AnyBURL/

must exist in the given triple set $\mathcal{T}^d$, while conclusion triples are not necessarily in $\mathcal{T}^d$ and may be generated for further annotation (i.e., Section 3.4).

For example, given a rule *spouse*(x, y) $\wedge$ *father*(x, z) $\Rightarrow$ *mother*(y, z), we traverse all of the given triples and find entities *LeBron James*, *Savannah Brinson*, and *Bronny James* that meet the premise. We then add the premise triples *spouse*(LeBron James, Savannah Brinson) and *father*(LeBron James, Bronny James) into the training set, and generate the conclusion triple *mother*(Savannah Brinson, Bronny James) for testing, no matter it is given or not.

Path Extension aims to increase the inference path patterns by (1) adding more reasoning paths for the same testing triple, and (2) elongating paths by replacing those premise triples that have reasoning paths. For example, we replace *father*(LeBron James, Bronny James) with two triples that can infer it: *father*(LeBron James, Bryce James) and *brother*(Bronny James, Bryce James). The original path is then extended by one hop. Correspondingly, we define the confidence of extended paths as the multiplication of all involved rules. Longer paths will challenge long-distance reasoning ability.

Negative Supplement is to generate negative triples if we cannot annotate the same number of negatives with positive triples. Otherwise, we will face an imbalance issue. Following conventions, we randomly corrupt the head or tail entities in a positive triple with the following constraints: (1) the relation of the positive triple is exclusive, e.g., *placeOfBirth*, if the ratio from head to tail entities is smaller than a threshold (we choose 1.2 heuristically in experiments); otherwise, the corrupted negative triple may be actually positive, leading to false negative errors. (2) We choose positive triples from the test set for corruption to improve the difficulty — the model has to correctly infer the corresponding positive triple from training data, then classify the corrupted triple as negative through the confliction. Particularly, for non-exclusive relation types, most of their corrupted results should be unknown following open-world assumption. The inferred test set covers such cases, which will be discussed in Section 3.4.

Inference Pattern Balance aims to balance various inference patterns, including path length, relation types, and relation patterns (i.e., symmetry, inversion, hierarchy, composition, and others). This is because concentrating on some patterns may lead to severe bias and unfair comparison between KGC models (Zhang et al., 2020). We first count the frequency of testing triples according to the path lengths, relation types and patterns, respectively. For each of them, we rank their counting and choose highest ranked groups of triples as frequent ones, instead of setting a threshold. We then carefully remove some frequent triples randomly, until the new distributions reach an accepted range (checked by humans).

3.4 Inferred Test Triple Labeling

Different from existing datasets, InferWiki aims to include positive, negative, and unknown testing triples, to evaluate the model under two types of assumptions: open-world assumption and closed-world assumption. The main difference between them is whether unknown triples are regarded as negatives. That is, the open-world evaluation is a three-class classification problem (i.e., positive, negative, and unknown). The closed-world evaluation targets only positive and negative triples, and we can simply relabel unknown triples as negatives without changing the test set.

So far, we have two test sets: one is generated via rule guidance, and the other contains the supplemented negatives. This section aims to label the generated triples. First, we automatically label the triples with positive if they exist in Wikidata. Then, we manually annotate the remaining 4,053 triples. The annotation guideline can be found in Appendix B. Note that all of the unknowns are factually incorrect but not inferential. To assess the quality of annotations, we verify a random selection of 300 test triples (100 for each label). The annotators agree with our labels 84.3% of the time. We further investigate the disagreements by relabeling 100 samples. 85% of the time, humans prefer an unknown, while automatic labeling tends to assign them with positive or negative labels. This suggests the inferential difference between humans and machines — the capacity of knowing unknown.

Finally, we remove the entities that are not in any of the grounded paths and their triples. We randomly select half of the test set as valid. This forms **InferWiki64k**. We further extract a dense subset **InferWiki16k** by filtering out the positive triples whose confidence is smaller than 0.6. Correspondingly, negative/unknown triples are reduced to keep balance. The statistics is listed in Table 2.

Test (+)	Japan National Route $1_{Q1191191}$, *connectsWith*, Japan National Route $4_{Q1055023}$
Train	Japan National Route 4, *terminus*, Japan National Route 1
Test (+)	$\text{Agnese}_{Q2726556}$, *sibling*, $\text{Lucia}_{Q3838490}$
Train	$\text{Maddalena}_{Q329555}$, *sibling*, Agnese $\wedge$ Maddalena, *mother*, Beatrice_{Q51089} $\wedge$ Valentina, *mother*, Beatrice$\wedge$ $\text{Viridis}_{Q271827}$, *sibling*, $\text{Valentina}_{Q943180}$ $\wedge$ Viridis, *sibling*, $\text{Estorre}_{Q3733572}$ $\wedge$ $\text{Elisabetta}_{Q1941886}$, *sibling*, Estorre $\wedge$ Lucia, *sibling*, Elisabetta
Test (-)	$\text{Quimper}_{Q702161}$, *capital*, Versailles_{Q621} ($\perp$ Yvelines_{Q12820}, *capital*, Versailles)
Train	Yvelines, *replaces*, $\text{Seine-et-Oise}_{Q979470}$ $\wedge$ Seine-et-Oise, *capital*, Versailles
Test (-)	Roberto_{Q53003}, *placeOfBirth*, Baton_{Q28218} ($\perp$ Roberto, *placeOfBirth*, Rome_{Q220})
Train	$\text{Renzo}_{Q1397252}$, *sibling*, Roberto $\wedge$ Renzo, *placeOfBirth*, Rome
Test (UNK)	Midōsuji $\text{Line}_{Q1192413}$, *connectsWith*, Keiyō $\text{Line}_{Q741145}$
Train	Shin-Ōsaka $\text{Station}_{Q801438}$, *connectingLine*, Midōsuji Line $\wedge$ Tōkaidō $\text{Shinkansen}_{Q660895}$, *terminus*, Shin-Ōsaka Station $\wedge$ Tōkaidō Shinkansen, *connectsWith*, Keihin-Tōhoku $\text{Line}_{Q1197028}$ $\wedge$ Keiyō Line, *connectsWith*, Keihin-Tōhoku Line
Test (UNK)	$\text{Mary}_{Q104109}$, *workLocation*, London_{Q84}
Train	Mary, *memberOfPoliticalParty*, Republican Party_{Q29468} $\wedge$ $\text{Carl}_{Q127437}$, *memberOfPoliticalParty*, Republican Party $\wedge$ $\text{Carl}_{Q127437}$ *workLocation*, London $\wedge$ Mary, *occupation*, actor_{Q33999} $\wedge$ Mary, *spouse* $\text{Owen}_{Q966972}$

Table 3: Positive, negative, and unknown examples of InferWiki, where the triples with brackets are not in train set (inferred from related training triples), $\perp$ denotes contradicted triples and subscripts denote Wikidata ID.

3.5 Dataset Analysis

Table 3 shows positive, negative, and unknown examples of InferWiki and their (possible) supportive training data. For positives, their paths seem reasonable and vary in length, relation types, and patterns. The 7-hop path of the sibling example is even difficult for a human. For negatives and unknowns, they are indeed incorrect and more challenging. There are no direct contradicted triples in the train set — the model is encouraged to reason related triples and justify if there is a confliction (i.e., negative) or not (i.e., unknown). Nevertheless, there are two minor issues. First, some unreasonable paths may corrupt the predictability. We thus increase the rule confidence threshold $\lambda > 0.6$ for InferWiki16k and manually annotate uncertain test triples for the correctness of labels. More advanced rule mining models can improve the construction pipeline. We leave it in the future. Second, does unknown triples have a bias on certain relation types? The answer is yes but not exactly. As shown in Table 3, the relation $connectsWith$ is involved in both positive and unknown triples, which is also determined by the paths.

Next, we analyze the relation patterns and path length distribution through comparisons with existing KGC datasets. Due to the different construction pipelines, existing datasets are difficult to offer quantitative statistics. We thus apply our pipeline on CoDEx (Safavi and Koutra, 2020). Only inferential test triples remain, and the training set keeps unchanged, namely CoDEx-m-infer, which reduces the test and valid positives from 20,622 to 7,050. This agree with the original paper that reports 20.56% triples are symmetry or compositional through AMIE+ analysis. We find more paths due to more extensive rules extracted from a large set of triples. This also demonstrates the necessity of rule-guided train/test generation — most test triples are not guaranteed inferential when using random split.

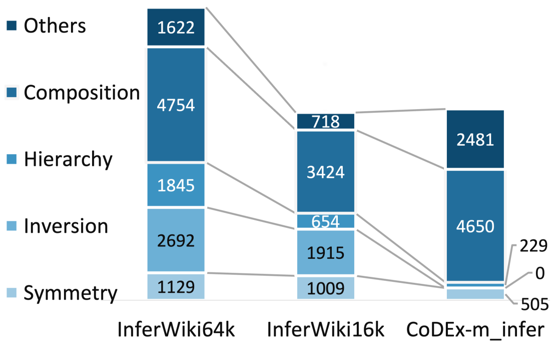


Figure 1: Distribution of paths in relation patterns.

Relation Pattern Following convention, we count reasoning paths for various patterns: symmetry, inversion, hierarchy, composition, and others, whose detailed explanations and examples can be found in Appendix C. If a triple has multiple paths, we count all of them. As Figure 1 shows, we can see that (1) there are no inversion and only a few symmetry and hierarchy patterns in CoDEx-m, as most current datasets remove them to avoid train/test leakage. But, we argue that learning and remembering such patterns are also an essential capacity of inference. It just needs to control their numbers for a fair comparison. (2) The patterns of InferWiki is more evenly distributed. Note that the patterns

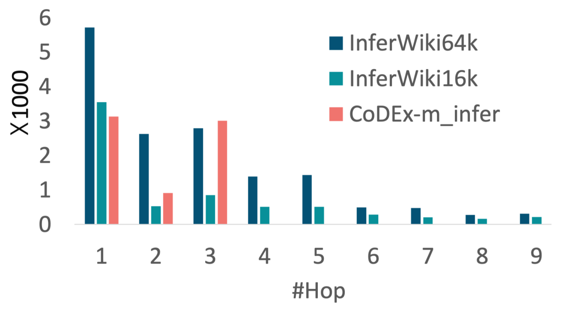


Figure 2: Comparison of paths in different lengths.

of symmetry, inversion, and hierarchy refer to 1-hop paths, while composition and others refer to multi-hop paths. So, the total number of the former three is almost the same as that of the latter two, to balance paths with varying lengths, which will be discussed next.

Path Length Distribution The reasoning paths can ensure test triples' predictability but may not be the shortest ones, as there may be undiscovered paths connecting two entities. Thus, our statistics concerning path length offer a conservative analysis and give an upper bound. For a test triple with multiple paths, we count the shortest one. As shown in Figure 2, we can see that InferWiki has more long-distance paths, while CoDEx-m-infer normally concentrates on maximum 3-hop reasoning paths. In specific, the maximum path length of InferWiki is 9 (4 before path extension) and the average length is 2.9 (1.5 before path extension).

Further analysis of relation, entity and neighbor distributions can be found in Appendix D&E.

3.6 Limitation

Although we carefully design the construction of inferWiki, there are still two types of limitations: rule biases and dataset errors, that can to be addressed along with the development of KG techniques in the future. In terms of rule biases, AnyBURL may be over-estimated due to its role in the construction. Although we utilize two triple sets to avoid rule leakage, their overlap may still bring unfair performance gain to AnyBURL. We consider synthesize several rule mining results to improve InferWiki in the next version. In terms of dataset errors, first, to balance positive and negative triples in the larger InferWiki64k, we follow conventions to randomly sample a portion of negatives. These negatives may be unknown if following open-world assumption. We manually assess the randomly sampled negatives and find a 15.7% error rate. Therefore, we conduct open-world experiments on the smaller InferWiki16k, all of whose testing negatives are verified by humans. The second type of errors is due to unreasonable rules for dataset split, which is caused by prediction errors of existing rule mining models. However, there is no suitable evaluation in this field to provide quantitative analysis. Our ongoing work aims to develop an automatic evaluation for path rationality to improve the mining quality, and thus facilitate our inferential pipeline.

4 Benchmarking

4.1 Tasks

We benchmark performance on InferWiki for the tasks: (1) **Link Prediction**, the task of predicting the missing head/tail entity for a given query triple (?, r, t) or (h, r, ?). Models are encouraged to rank correct entities higher than others in the vocabulary. We adopt the filtering setting (Bordes et al., 2013) that excludes those entities, if the predicted triples have been seen in the train set. Mean reciprocal rank (MRR) and hits@k are standard metrics for evaluation. (2) **Triple Classification** aims to predict a label for each given triple (h, r, t). The label following open-world assumption is trinary $y \in \{-1, 0, 1\}$ and becomes binary $y \in \{-1, 1\}$ when adopting closed-world assumption — all 0-label triples are re-labeled with -1, since our unknown triples are factually negative yet non-inferential from training data. Since KGC models output real-value scores for triples, we classify scores into labels by choosing one or two thresholds per relation type on valid. Accuracy, precision, recall, and F1 are measurements.

4.2 Models

For comprehensive comparison, we choose three types of representative models as baselines: (1) Knowledge Graph Embedding models, including **TransE** (Bordes et al., 2013), **ComplEx** (Trouillon et al., 2016), **RotatE** (Sun et al., 2019), **ConvE** (Dettmers et al., 2018), and **TuckER** (Balazevic et al., 2019), (2) multihop reasoning model **Multihop** (Lin et al., 2018), and (3) rule-based **AnyBURL** (Meilicke et al., 2019). Note that the latter two are specially designed for link prediction. The detailed implementation including parameters and thresholds can be found in Appendix F.

4.3 Triple Classification Results

Table 4 shows micro scores for triple classification. We can see that all of the baselines perform

	InferWiki64k				InferWiki16k				CoDEx-m-infer			
	Acc	Prec	Recall	F1	Acc	Prec	Recall	F1	Acc	Prec	Recall	F1
TransE	.823	.782	.895	.835	.796	.736	**.926**	.820	.763	.792	.891	.839
ComplEx	.812	.779	.872	.823	.811	.835	.778	.805	.798	.805	.936	.866
RotatE	.852	.808	**.924**	.862	.811	.769	.891	.825	.788	.790	.945	.861
ConvE	**.881**	.864	.906	**.884**	**.897**	**.887**	.911	**.899**	**.851**	.853	**.948**	**.898**
TuckER	.862	**.897**	.817	.855	.861	.836	899	.866	.803	**.919**	.784	.846

Table 4: Overall Performance of Triple Classification (Closed-world Assumption), where acc and prec stand for accuracy and precision, respectively.

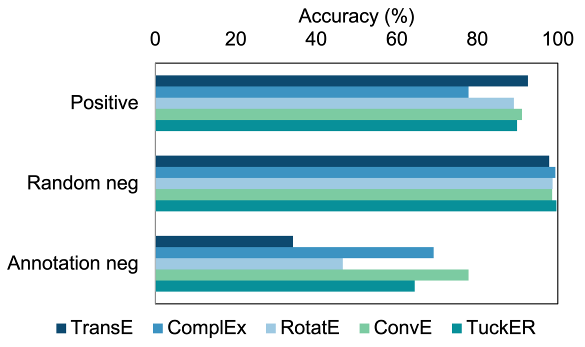


Figure 3: Accuracy regarding various negative types, where random neg denotes supplemented negatives, and annotation neg denotes annotated negatives (including unknowns).

	Acc	Prec	Recall	F1
TransE	.711	.687	.668	.676
ComplEx	.723	.703	.709	.701
RotatE	.745	.746	.750	.736
ConvE	**.803**	**.763**	**.777**	**.768**
TuckER	.709	.639	.657	.618

Table 5: Performance of Triple Classification on InferWiki16k (Open-world Assumption).

well — around 90% F1 scores. This is consistent with recent findings that triple classification is a nearly solved task (around 98% F1 scores) (Safavi and Koutra, 2020). Nevertheless, the lower performance demonstrates the difficulty of our curated datasets, mainly due to the manually annotated hard negatives of InferWiki (and CoDEx).

Impacts of Hard Negatives

Figure 3 presents the accuracy on InferWiki16k regarding various types of triples: positive, random supplemented negatives, and annotated negatives (including relabeled unknowns). We can see that (1) random negative triples are indeed trivial for all of baseline models, which motivates the necessity of harder negative triples to push this research direction forward, (2) positive triples are slightly difficult to judge than random negatives, and (3) the accuracy significantly drops on annotation negatives. This is mainly because most annotated triples are actually unknown — they are factually incorrect, but there are no obvious abnormal patterns. Such non-inferential cases may underestimate KGC models.

Open-world Assumption

Since most baselines fail in judging unknown as negative, we now investigate them following open-world assumption to see their ability in recognizing unknown triples. Table 5 shows the macro performance[3] on InferWiki16k. We can see that all of the baseline models perform worse than those under the closed-world assumption. On one hand, the trinary classification is intuitively more difficult than binary classification. On the other hand, it is a rather straightforward method to search two decision thresholds — one between positive and unknown and the other between unknown and negative. This motivates us future works on advanced models to represent KG, which should also be able to detect the limitation and boundaries of given KG. It is a fundamental capacity of inference to respond “I do not know”, to avoid false negatives in downstream applications.

Figure 4 presents a detailed analysis of each model regarding their search thresholds. We can see that although their best performance seems not bad, the worst scores are only around 10%. That is, they are very sensitive to thresholds. Besides, most of the time, the average F1 scores of ComplEx, RotatE, and TuckER are around 20%, while transE achieves higher scores. Maybe that is the reason why it is still the most widely used KGC method. ConvE stably outperforms other baselines, no matter in terms of best, worst, or average performance.

4.4 Link Prediction Results

Table 6 shows the average scores for head and tail prediction. We can see that (1) AnyBURL performs the best most of the time, but the performance gap is not significant. This is mainly due to its role in

[3]Micro performance is only applicable to binary classification, while open-world evaluation is trinary.

	InferWiki64k			InferWiki16k			CoDEx-m-infer		
	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
TransE	.357	.129	.709	.474	.214	.842	.366	.363	.567
ComplEx	.350	.218	.595	.537	.377	.789	.252	.160	.430
RotatE	.465	.297	.735	.629	.450	.883	.352	**.476**	.561
ConvE	**.575**	.475	.747	.748	.678	.868	.450	.369	.585
TuckER	.573	.466	.755	**.754**	.677	.886	**.451**	.365	.603
AnyBURL	-	**.559**	**.783**	-	**.714**	**.892**	-	.394	**.620**

Table 6: Results of Link Prediction. Bold fonts denote the best scores and underlines highlight the second best.

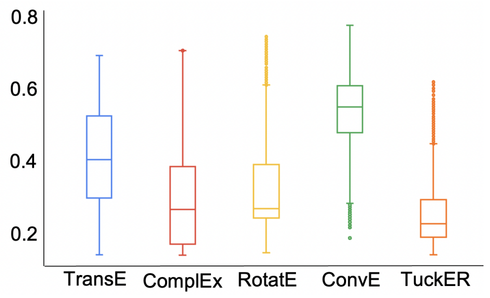


Figure 4: Macro F1 variance when we search the best thresholds for open-world triple classification.

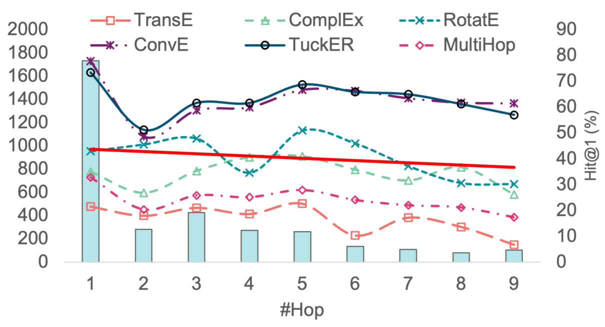


Figure 5: Hit@1 curves of baseline models for tail prediction. The x-axis denotes the number of hops, and the bars denote the number of examples that have corresponding hops. Red solid line is a performance trend line of six models.

dataset construction, although we utilize two sets of triples to minimize rule leakage. Actually, inference of rules may be more important than we thought to improve the reliability and interpretability of knowledge-driven models. This also motivates us to incorporate rule knowledge into KGC training for advanced reasoning ability (Guo et al., 2018; Li et al., 2019b). (2) KGC models perform better on InferWiki16k than InferWiki64k, due to the higher structure density and rule confidence. (3) Models have higher hit@10 and lower hit@1 on InferWiki than other datasets (e.g., CoDEx). This agrees with an intuition that most entities are irrelevant, making it trivial to judge these corrupted triples as in triple classification. And, only a small portion of entities is difficult to predict, which requires strong inference ability. Besides, hit@1 varies a lot, so that we can better compare among models.

Impacts of Inferential Path Length

Figure 5 presents Hit@1 curves for tail prediction regarding varying path length on InferWiki64k[4]. We can see an overall downwards trend along with the increasing path length. Meanwhile, the large fluctuation may be due to two possible reasons: (1) as discussed in Section 3.5, the inferential paths ensure the predictability, but may not be the shortest ones. This thus offers a conservative analysis and give an upper bound of the performance concerning k-hop paths. Our paths are of high coverage and quality compared with existing datasets, which either conduct case study or post-process datasets via rule mining. (2) Relation types and patterns also have significant impacts. Shorter paths contain more long-tail relations, and longer paths tend to cover many common relations. This improves the difficulty of shorter paths and makes longer paths easier.

Impacts of Relation Patterns

We present the Hit@1 tail prediction on InferWiki64k regarding relation patterns in Table 7. We can see that symmetry and inversion are not well-solved, which should be considered into evaluation but limited in scale. TransE performs worse on symmetry and inversion relations, consistent with the analysis in Abboud et al. (2020). Even if ComplEx and RotatE can capture such patterns, they fail to rank corresponding entities at the top. Embedding-based models perform well on hierarchy relations, even outperforms AnyBURL. For compositional relations, it is still quite challenging and worthwhile further investigation.

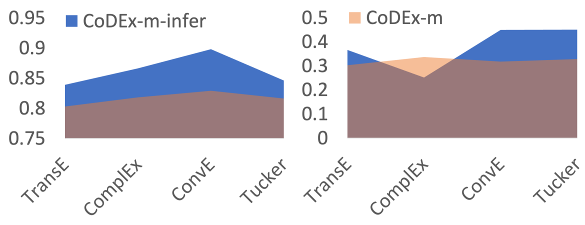
4.5 Comparison of CoDEx-infer and CoDEx

We investigate the impacts of rule-based train/test generatation by comparing CoDEx-m-infer with

[4]Multihop is designed for tail prediction, and Hit@1 on InferWiki64k is more distinct for following ablation study.

	Sym	Inv	Hier	Comp	Others
TransE	.000	.049	.479	.211	.296
ComplEx	.130	.279	.502	.368	.414
RotatE	.191	.246	.694	.477	.610
ConvE	.558	.668	.855	.602	.784
TuckER	.527	.612	.850	.625	.753
Multihop	.231	.309	.345	.240	.296
AnyBURL	.782	.793	.782	.686	.809

Table 7: Hit@1 tail prediction on Relation Patterns.



(a) Triple classification (F1). (b) Link Prediction (MRR).

Figure 6: Comparison of CoDEx-infer and CoDEx.

CoDEx-m. The two datasets share the same training set. The only difference lies in how we obtain the test triples, either using our proposed pipeline (CoDEx-m-infer) or randomly (CoDEx-m). Thus, the results reflect the impacts of inferential guarantee for dataset construction and demonstrate the necessity to avoid over-estimation or under-estimation of the inferential ability of KGC models. We report the performance on CoDEx-m from the original paper (Safavi and Koutra, 2020).

We can see that all of models perform better with inferential path guarantee on CoDEx-m-infer than CoDEx-m, except ComplEx for link prediction. This is because rule guidance elimites those non-inferential testing triples, making the task easier. Nevertheless, the scores on hard cases are actually decreased (as discussed in Figure 3 and Table 7). Models are excepted a stronger reasoning ability among several related entities, instead of trivially filtering out massive irrelevant entities. This also demonstrates the necessity of InferWiki to avoid over- or under- estimation of the inferential ability of KGC models — learning new knowledge from existing ones.

5 Case Study of Relation Types

We illustrate the most frequent relation types and their distribution of InferWiki64k and InferWiki16k in Figure 8. We can see that InferWiki has a diverse relation types that are not limited to specific domains. Besides, the triples of each relation type are well balanced.

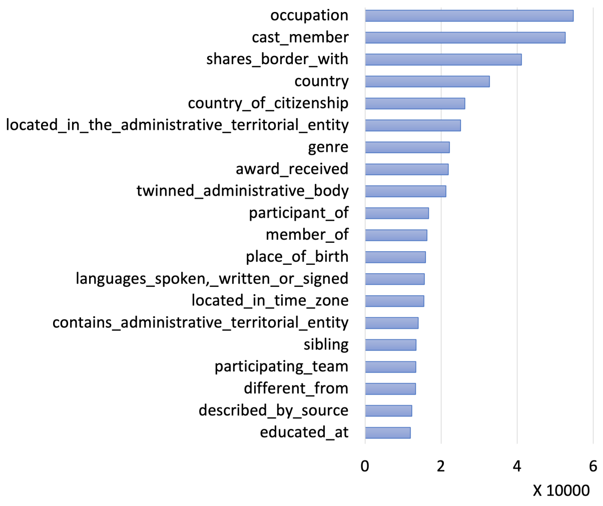


Figure 7: Distribution of most frequent relation types in InferWiki64k. A comparison with InferWiki16k is in Appendix D.

6 Conclusion

We highlighted three principles for KGC datasets: inferential ability, assumptions, and patterns, and contribute a large-scale dataset InferWiki. We established a benchmark with three types of seven KGC models on two tasks of triple classification and link prediction. The results present a detailed analysis regarding various inference patterns, which demonstrates the necessity of an inferential guarantee for better evaluation and the difficulty of new open-world triple classification.

In the future, we are interested in cross-KGs inference and transfer (Cao et al., 2019a), and investigating how to inject knowledge into deep learning architectures, such as for information extraction (Tong et al., 2020) or text generation (Cao et al., 2020b).

Acknowledgments

This research was conducted in collaboration with SenseTime. This work is partially supported by A*STAR through the Industry Alignment Fund - Industry Collaboration Projects Grant, by NTU (NTU–ACE2020-01) and Ministry of Education (RG96/20), and by the National Research Foundation, Prime Minister's Office, Singapore under its Energy Programme (EP Award No. NRF2017EWT-EP003-023) administrated by the Energy Market Authority of Singapore. This work is partially supported by Singapore MOE AcRF T1.

References

Ralph Abboud, Ismail Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. Boxe: A box embedding model for knowledge base completion. *NeurIPS*.

Farahnaz Akrami, Mohammed Samiul Saeef, Qingheng Zhang, Wei Hu, and Chengkai Li. 2020. Realistic re-evaluation of knowledge graph completion methods: An experimental study. In *SIGMOD*.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. In *EMNLP-IJCNLP*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *NeurIPS*.

Guillaume Bouchard, Sameer Singh, and Theo Trouillon. 2015. On approximate reasoning capabilities of low-rank vector spaces. In *AAAI spring symposia*.

Yixin Cao, Jun Kuang, Ming Gao, Aoying Zhou, Yonggang Wen, and Tat-Seng Chua. 2020a. Learning relation prototype from unlabeled texts for long-tail relation extraction. *arXiv preprint arXiv:2011.13574*.

Yixin Cao, Zhiyuan Liu, Chengjiang Li, Juanzi Li, and Tat-Seng Chua. 2019a. Multi-channel graph neural network for entity alignment. In *ACL*.

Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. 2020b. Expertise style transfer: A new task towards better communication between experts and laymen. In *ACL*.

Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019b. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The world wide web conference*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2017. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv preprint arXiv:1711.05851*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. 2015. Fast rule mining in ontological knowledge bases with amie+. *VLDB*.

Alberto Garcia-Duran, Antoine Bordes, and Nicolas Usunier. 2015. Composing relationships with translations. In *EMNLP*.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge graph embedding with iterative guidance from soft rules. In *AAAI*.

Zikun Hu, Yixin Cao, Lifu Huang, and Tat-Seng Chua. 2021. How does knowledge graph and attention help? a qualitative analysis into bag-level relation extraction. In *ACL*.

Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. 2006. Learning systems of concepts with an infinite relational model. In *AAAI*.

Stanley Kok and Pedro Domingos. 2007. Statistical predicate invention. In *ICML*.

Bhushan Kotnis and Vivi Nastase. 2017. Analysis of the impact of negative sampling on link prediction in knowledge graphs. *arXiv preprint arXiv:1708.06816*.

Chengjiang Li, Yixin Cao, Lei Hou, Jiaxin Shi, Juanzi Li, and Tat-Seng Chua. 2019a. Semi-supervised entity alignment via joint knowledge embedding model and cross-graph model. In *EMNLP-IJCNLP*.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019b. A logic-driven framework for consistency of neural models. In *EMNLP-IJCNLP*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. In *EMNLP*.

Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2016. Knowledge representation learning with entities, attributes and relations. In *IJCAI*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*.

Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. 2019. Anytime bottom-up rule learning for knowledge graph completion. In *IJCAI*.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *AAAI*.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *ECML PKDD*.

Tara Safavi and Danai Koutra. 2020. Codex: A comprehensive knowledge graph completion benchmark. *arXiv preprint arXiv:2009.07810*.

Baoxu Shi and Tim Weninger. 2018. Open-world knowledge graph completion. In *AAAI*.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. *NeurIPS*.

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

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *ACL*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. *ACL CVSC Workshop*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080.

Quan Wang, Bin Wang, and Li Guo. 2015. Knowledge base completion using embeddings and rules. In *IJCAI*.

Rui Wang, Bicheng Li, Shengwei Hu, Wenqian Du, and Min Zhang. 2019. Knowledge graph embedding via graph attenuated attention networks. *IEEE Access*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V Chawla. 2020. Few-shot knowledge graph completion. In *AAAI*.

A Literature Review

Table 8 lists existing KGC datasets. We can roughly classify them into two groups: inferential and non-inferential datasets. The first group are usually manually curated to ensure each testing sample can be inferred from training data through reasoning paths. Families (Garcia-Duran et al., 2015) test family relationships including cousin, ancestor, marriage, parent, sibling, and uncle, among the members of 5 families along 6 generations. Such that there are obvious compositional relationships like uncle $\approx$ sibling + parent or parent $\approx$ married + parent. Kinship (Kemp et al., 2006) contains kinship relationships among members of the Alyawarra tribe from Central Australia, while Country (Bouchard et al., 2015) contains countries, regions, and subregions as entities and is carefully designed to explicitly test the location relationship (i.e., locatedIn and neighbor) among them. The above datasets are clearly limited in scale and inference patterns, thus become not challenging. HOLE (Nickel et al., 2016) even achieves 99.7% ACU-PR on dataset Country (Bouchard et al., 2015).

The second group of datasets are automatically derived from public KGs and randomly split positive triples into train/valid/test, leading to a risk of testing samples non-inferential from training data. FB13 (Socher et al., 2013) and FB15K (Bordes et al., 2013) are commonly used benchmark from FreeBase. FB15k401 (Yang et al., 2014) is a subset of FB15k containing only frequent relations (relations with at least 100 training examples). To remove test leakage, FB15k-237 (Toutanova and Chen, 2015) removes all equivalent or inverse relations. Similarly, FB5M (Wang et al., 2014) removes all the entity pairs that appear in the testing set. WN18RR (Dettmers et al., 2018) is the challenging version of WN18 (Bordes et al., 2013) extracted from WordNet. Textual information is also included for specific task, such as FB40K (Lin et al., 2015) targeting relation extraction dataset New York Times (Riedel et al., 2010). FB24K (Lin et al., 2016) introduce Attributes. FB15K+ (Xie et al., 2016) introduce types and make fb15k more sparse by only filterring out relation with a frequency lower than one. Another popular knowledge source is YAGO, and the corresponding datasets include YAGO3-10 (Dettmers et al., 2018) and YAGO37 (Guo et al., 2018). Except for open-domain KG, NELL (Wang et al., 2015) concentrates on location and sports, and UMLS (Kok and Domingos, 2007) targets medical knowledge. CoDEx (Safavi and Koutra, 2020) argues the quality of the above benchmarks, such as NELL995 (Xiong et al., 2017) are nonsensical or overly generic. Thus they propose a comprehensive dataset consisting of three knowledge graphs varying in size and structure, entity types, multilingual labels and descriptions, and hard negatives.

B Annotation Guideline

We provide the following annotation guidelines for annotators to label inferred triples in Section 3.4.

Task This is a two-step annotations. First, you must annotate each triple with the label $y \in \{1, -1\}$, where 1 denotes that the triple is correct and -1 denotes that the triple is incorrect. You can find the answer from anywhere you want, such as commonsense, Wikipedia, and professional websites. If you cannot find any evidence to support the statement, you shall choose label -1. Second, you must annotate each incorrect triple with the label $\hat{y} \in \{0, -1\}$, where 0 denotes that you do not know the answer. Now, you can find the answer from our provided triples. If you cannot find any evidence to support the statement, you shall choose label 0.

Examples Here are some examples judged using three types of knowledge sources.

- **Commonsense**: (Cypriot Fourth Division, hasPart, 2018–19 Cypriot Third Division) is clearly incorrect, since the fourth division cannot has a part of third division.

- **Professional websites**: To annotate the triple (Bahrain-Merida 2019, hasPart, Carlos Betancur), you may search the person in professional websites, such as `https://www.procyclingstats.com/team/bahrain-merida-2019`. Since there is no Carlos Betancur listed in that website, please choose false.

- **Wikipedia**: Given the triples (Tōkaidō Shinkansen, connectsWith, Osaka Higashi Line) and (Tōkaidō Shinkansen, connectsWith, San'yō Main Line), you can find related station information from the page of Tōkaidō Shinkansen. You can find that Osaka Higashi Line shares a transfer station with Tōkaidō Shinkansen, thus label it with 1.

Datasets	source	#Entity	#Relation	#Triples (train/valid/test)
FB13 (Socher et al., 2013)	FreeBase	75,043	13	316,232/5,908/23,733
FB15k (Bordes et al., 2013)	FreeBase	14,951	1,345	483,142/50,000/59,071
FB15k237 (Toutanova and Chen, 2015)	FreeBase	14,541	237	272,115/17,535/20,466
FB15k+ (Xie et al., 2016)	FreeBase	14,951	1,855	486,446/50,000/62,374
FB15k401 (Yang et al., 2014)	FreeBase	14,541	401	560,209/-/-
FB24k (Lin et al., 2016)	FreeBase	23,634	987	402,493/-/21,067
FB40k (Lin et al., 2015)	FreeBase	39,528	1,336	370,648/67,946/96,678
FB5M (Wang et al., 2014)	FreeBase	5,385,322	1,192	19,193,556/50,000/59,071
WN11 (Socher et al., 2013)	WordNet	38,696	11	112,581/2,609/10,544
WN18 (Bordes et al., 2013)	WordNet	40,943	18	141,442/5,000/5,000
WN18RR (Dettmers et al., 2018)	WordNet	40,943	11	86,835/3,034/3,134
YAGO3-10 (Dettmers et al., 2018)	YAGO	123,182	37	1,079,040/5,000/5,000
YAGO37 (Guo et al., 2018)	YAGO	123,189	37	989,132/50,000/50,000
CoDEx (Safavi and Koutra, 2020)	Wikidata	77,951	69	551,193/30,622/30,622
NELL995 (Xiong et al., 2017)	NELL	75,492	200	154,213/-/-
NELL$_{loc}$ (Wang et al., 2015)	NELL	672	10	941/-/-
Family (Garcia-Duran et al., 2015)	Artificial	721	7	8,461/2,820/2,821
Kinship (Kemp et al., 2006)	Artificial	104	26	8,548/2,820/2,821
Countries (Bouchard et al., 2015)	Artificial	272	2	1,111/24/24
UMLS (Kok and Domingos, 2007)	UMLS	135	49	5,216/-/-

Table 8: An overview of Knowledge Graph Completion Datasets.

And, San'yō Main Line doesn't show up in the page, you may label it with -1.

C Relation Patterns

InferWiki is able to analyze relation patterns for each path, including symmetry, inversion, hierarchy, and composition, where detailed explanations and examples are listed in Table 9.

D Relation Types

We illustrate the most frequent relation types and their distribution of InferWiki64k and InferWiki16k in Figure 8.

E Comparison with Existing Datasets

Figure 9 shows the distribution of entities and their neighbors as compared to widely used datasets: FB15k237 and CoDEx-m.

F Experiment Setup

Our experiments are run on the server with the following configurations: OS of Ubuntu 16.04.6 LTS, CPU of Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz, and GPU of GeForce RTX 2080 Ti. We use OpenKE[5] for re-implementing TransE, ComplEx, and RotatE. For the rest models, we use the original codes for ConvE[6], TuckER [7], Multihop[8], and AnyBURL[9]. Because we utilize various types of KGC models including embedding-based, multi-hop reasoning (reinforcement learning), and rule-based models, these models largely have their own hyperparameters. To avoid exhaustive parameter search in a large range, we conduct a series of preliminary experiments and find that the suggested parameters work well on Wikidata-based data. We then search the embedding size in the range of $\{256, 512\}$, number of negative samples in the range of $\{15, 25\}$ and margin in the range of $\{4, 8\}$. The optimal parameters of each model on all of three datasets are listed in Table 10. The thresholds in triples classification are listed in Table 11

[5]https://github.com/thunlp/OpenKE

[6]https://github.com/TimDettmers/ConvE

[7]https://github.com/ibalazevic/TuckER

[8]https://github.com/salesforce/MultiHopKG

[9]http://web.informatik.uni-mannheim.de/AnyBURL/

Pattern	Notation	Example
symmetry	$r_1(x,y) \Rightarrow r_1(y,x)$	(Prince Christopher$_{Q44775}$, partner, Friederike$_{Q93614}$) $\Rightarrow$ (Friederike, partner, Prince Christopher)
inversion	$r_1(x,y) \Leftrightarrow r_2(y,x)$	(Amravati district$_{Q1771774}$, capital, Amravati$_{Q269899}$) $\Rightarrow$ (Amravati, capitalOf, Amravati district)
hierarchy	$r_1(x,y) \Rightarrow r_2(y,x)$	(Superman$_{Q79015}$, derivativeWork, Superman Returns$_{Q328695}$) $\Rightarrow$ (Superman, presentInWork, Superman Returns)
composition	$r_1(x,y) \wedge \cdots \wedge r_p(y,z) \Rightarrow r_{p+1}(x,z)$	(Eleanor$_{Q156045}$, mother, Joanna$_{Q171136}$) $\wedge$ (Ferdinand I$_{Q150611}$, mother, Joanna) $\wedge$ (Isabella$_{Q157884}$, sibling, Ferdinand I) $\Rightarrow$ (Eleanor, sibling, Isabella)

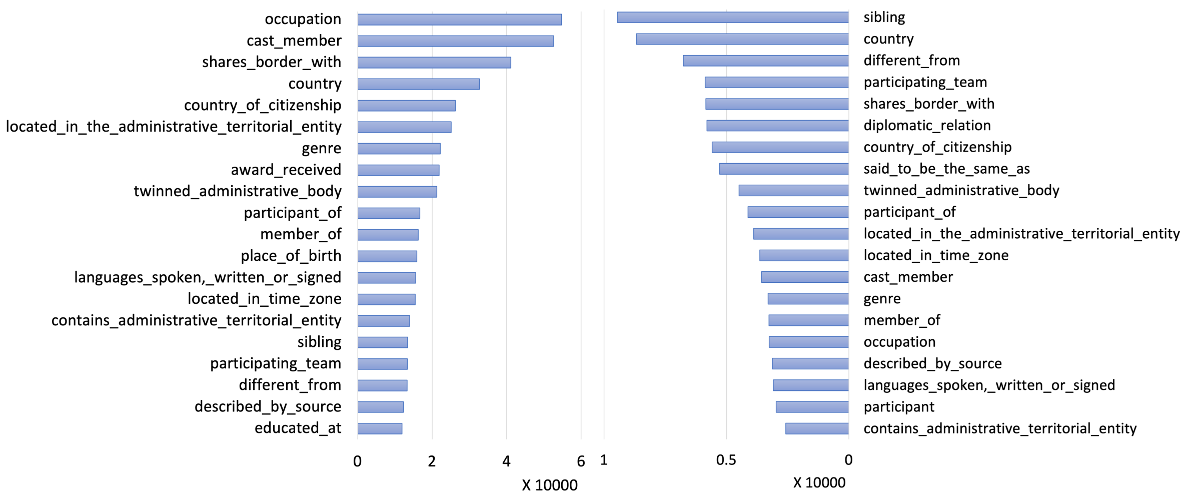
Table 9: Explanations and examples for various relation patterns.

Hyperparameter	TransE	ComplEx	RotatE	ConvE	TuckER	Multihop
InferWiki16k						
Embedding Size	256	512	512	512	512	256
# Negatives	15	25	25	-	-	-
Margin	4	4	8	-	-	-
Learning Rate	1.0	0.5	2e-5	1e-4	1e-4	1e-3
Optimizer	SGD	adagrad	adam	adam	adam	-
Batch Size	1,625	1,625	2,000	256	256	128
InferWiki64k						
Embedding Size	256	512	512	256	512	256
# Negatives	15	15	25	-	-	-
Margin	4	4	8	-	-	-
Learning Rate	1.0	0.5	2e-5	1e-4	1e-4	1e-3
Optimizer	SGD	adagrad	adam	adam	adam	-
Batch Size	7,823	7,823	2,000	256	256	128
CoDEx-m-infer						
Embedding Size	512	256	512	256	512	256
# Negatives	25	25	25	-	-	-
Margin	8	4	4	-	-	-
Learning Rate	1.0	0.5	2e-5	1e-4	1e-4	1e-3
Optimizer	SGD	adagrad	adam	adam	adam	-
Batch Size	1,856	1,856	2000	256	256	128

Table 10: Best hyperparameter configurations.

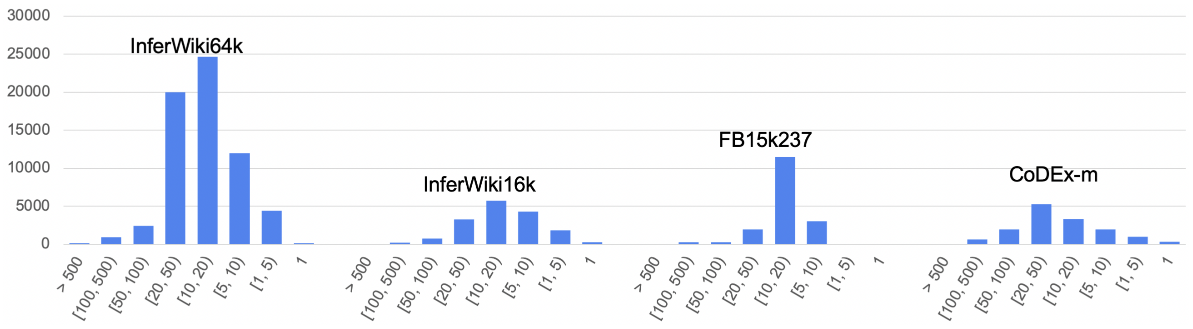
	InferWiki	TransE	ComplEx	RotatE	ConvE	TuckER
Closed World	64k	[-24.4663, -9.0235] -16.7449	[-43.0342, 30.6942] -0.2717	[-15.7235, 7.8291] -0.6498	[0.0, 0.9999] 0.1	[0.0, 0.9982] 0.01
	16k	[-24.0588, -4.333] -13.4069	[-21.5906, 24.7742] 2.5191	[-21.2362, 7.8282] -0.6005	[0.0, 1.0] 0.19	[0.0, 0.9734] 0.0097
Open World	16k	[-24.0588, -4.333] -16.1685, -11.8288	[-21.5906, 24.7742] -3.5084, 3.4464	[-21.2362, 7.8282] -2.3444, 0.8527	[0.0, 1.0] 0.01, 0.37	[0.0, 0.9734] 0.0097, 0.0389

Table 11: Best thresholds in triple classification, where the upper side is the search range and the lower side is the best values. They are searched on validation.

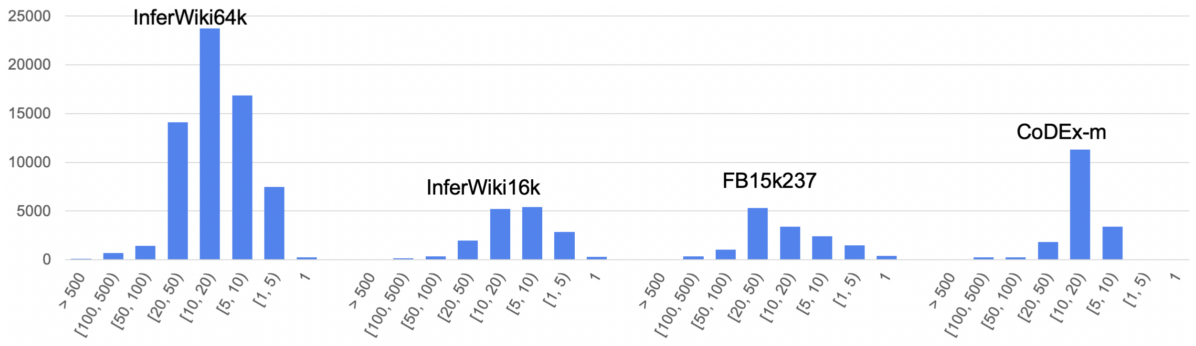


(a) InferWiki64k. (b) InferWiki16k.

Figure 8: Distribution of most frequent relation types.



(a) Distribution of entities, where x-axis denotes different ranges regarding entity frequency in the train set.



(b) Distribution of entity neighbors, where x-axis denotes ranges regarding average number of neighbors in the train set.

Figure 9: Distribution of entities and their neighbors.