Synthesizing Knowledge Graphs for Link and Type Prediction Benchmarking

André Melo and Heiko Paulheim

University of Mannheim
B6 26, 68159 Mannheim, Germany
{andre,heiko}@informatik.uni-mannheim.de
http://dws.informatik.uni-mannheim.de

Abstract. Despite the growing amount of research in link and type prediction in knowledge graphs, systematic benchmark datasets are still scarce. In this paper, we propose a synthesis model for the generation of benchmark datasets for those tasks. Synthesizing data is a way of having control over important characteristics of the data, and allows the study of the impact of such characteristics on the performance of different methods. The proposed model uses existing knowledge graphs to create synthetic graphs with similar characteristics, such as distributions of classes, relations, and instances. As a first step, we replicate already existing knowledge graphs in order to validate the synthesis model. To do so, we perform extensive experiments with different link and type prediction methods. We show that we can systematically create knowledge graph benchmarks which allow for quantitative measurements of the result quality and scalability of link and type prediction methods.

Keywords: Knowledge Graphs, Link Prediction, Type Prediction, Benchmarking

1 Introduction

Benchmarking is an important way of evaluating and comparing different methods for a given task. Having datasets with various characteristics is a crucial part of designing good benchmarking tests, allowing to thoroughly analyze the performance of a method under various conditions.

With the growing adoption and usage of Web-scale knowledge graphs, the data quality of those graphs has drawn some attention, and methods for improving the data quality, e.g., by predicting missing types and links, have been proposed. While there are a few benchmarking datasets for other tasks in the Semantic Web community, like SPARQL query performance [18, 30], ontology matching [6], entity linking [8], machine learning [28], and question answering [15], benchmarks for the task of type and link prediction are still missing. In contrast, the majority of approaches is only tested on one or few datasets, most prominently different versions of DBpedia, which makes it difficult to compare the approaches [24]. Thus, it would be desirable to have benchmarking datasets

with different characteristics, such as the number of entities, relation assertions, number of types, the taxonomy of types, the density of the knowledge graph, etc. Furthermore, it would be interesting to be able to have some control over these characteristics, vary them if necessary, and generate a knowledge graph following defined settings.

Generating data artificially for evaluation purposes is not something new. Data synthesizers have been widely used in some other research areas. IBM Quest Synthetics Data Generator[1] is probably the most famous of them. It generates transaction tables for frequent pattern mining. There are also generators, e.g., for spatial-temporal data [31], clustering and outlier detection [5], data for information discovery and analysis systems [29], and high-dimensional datasets [1].

The overall goal is to synthesize a multitude of knowledge graphs to design benchmarkings for the tasks of link and type prediction. A first step to achieve this goal is to be able to replicate already existing datasets. In this paper, we propose knowledge graph models, and a synthesis process that is able to generate data based on the models. To show the validity of the synthesis approach, our main goal is to replicate the performance measures obtained for evaluation measures when performing link and type prediction with various state-of-the-art methods. We want to minimize the distance between the original dataset and the synthesized replicas for these measures, and also preserve method rankings. In our case, we select five methods for each task.

In order to be able to run systematic scalability tests with different approaches, we also explore the possibility to generate replicas of different sizes (number of entities and facts). The results should be preserved when varying the size of the synthesized data.

The rest of this paper is structured as follows. Section 2 discusses related work. We introduce our model for knowledge graphs in section 3, and discuss the synthesis approach in section 4. In a set of experiments, we discuss the validity of our approach in section 5, and conclude with a summary and an outlook on future work.

2 Related Work

There have been works which address the synthesis of knowledge graphs for benchmarking purposes. However, most efforts were focused on synthesizing A-box assertions for a specific T-box. Moreover, these works generate benchmarking datasets for different tasks in the Semantic Web, but none of them focus on link and type prediction.

Guo et al. [12] propose a method for benchmarking Semantic Web knowledge base systems on large OWL applications. They present the Lehigh University Benchmark (LUBM), which has an ontology for the university domain and includes the Univ-Bench artificial data generator (UBA), as well as a set of

[1] `http://www.philippe-fournier-viger.com/spmf/datasets/IBM_Quest_data_generator.zip`

queries and performance measures for evaluation. The data generator synthesizes A-boxes of arbitrary size to evaluate scalability. The data contains information about universities, which are artificially created based on some predefined restrictions, e.g. minimum and maximum number of departments, student/faculty ratio, which are based on arbitrary defined ranges.

SP2Bench [30] is a SPARQL performance benchmark based on DBLP data. It features a data generator, which can create arbitrarily large datasets. Similarly to UBA, the authors synthesize the A-box based on an existing T-box, in this case the DBLP ontology, and a dataset specific model used to generate the synthetic data. The model uses logistic curves and simple intervals to describe characteristics of the DBLP data, such as the number and types of publications, distribution of citations, and level of incompleteness over years.

Morsey et al. [18] created a SPARQL query benchmark based on DBpedia to evaluate knowledge base storage systems. They gather a set of real world queries extracted with query log mining, and run them on datasets of different sizes generated from DBpedia. Their "data generation" process consists of sampling the original DBpedia dataset and changing the entities namespace. Two sampling methods are considered: *rand*, which basically randomly selects a fraction of the triples, and *seed*, which first sample a subset of the classes, then instances of these classes are also sampled and added to a queue. This process is iterated until the target dataset size is reached.

Linked Data Benchmark Council (LDBC) [2] developed the social network benchmark (SNB) and the semantic publishing benchmark (SPB). The SNB which includes a data generator that enables the creation of synthetic social network data representative of a real social network. The data generated includes properties occurring in real data, e.g. irregular structure, structure/value correlations and power-law distributions. The benchmark covers main aspects of social network data management, including interactive, business intelligence and graph analytics workload. The SPB is similar to the SNB, but it concerns the scenario of a media organization that maintains RDF descriptions of its catalogue of creative works.

3 Knowledge Graph Model

We define a knowledge graph $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is the T-box and $\mathcal{A}$ is the A-box containing relations assertions $\mathcal{A}_R$ and type assertions $\mathcal{A}_C$. We define N_C as the set of concepts (types), N_R as the set of roles (object properties) and N_I as the set of individuals (entities which occur as subject or object in relations). The set of relation assertions is defined as $\mathcal{A}_R = \{p(s,o) | p \in N_R \wedge s, o \in N_I\}$ and the set of type assertion as $\mathcal{A}_C = \{C(s) | C \in N_C \wedge s \in N_I\}$.

In our proposed model, we learn the joint distribution of types over instances. To that end, we compute $P(T)$, which is the probability of an individual having a set of types T. We define the set of types $\tau(s)$ of a given individual s as $\tau(s) = \{C | C(s) \in \mathcal{A}_C\}$ and the set of individuals of given set of types T as $E_T = \{s | \tau(s) = T\}$. This is important because most knowledge graphs allow

instances to have multiple types, and by modeling the distribution of instances over sets of types we can capture the dependencies between types, which is relevant for the problem described in this paper. It is important to notice that, e.g., `Arnold_Schwarzenegger` with set of types $T = \{\texttt{Actor}, \texttt{Politician}, \texttt{BodyBuilder}\}$ is not considered to belong to $\{\texttt{Actor}, \texttt{Politician}\}$ when computing the distributions. With that, we make sure that $\sum_{T \in \mathcal{P}(N_C)} P(T) = 1$, where $\mathcal{P}(N_C)$ is the powerset of types containing all possible combinations of types

$$P(T) = \frac{|\{s|\tau(s) = T\}|}{|N_I|} \tag{1}$$

We also model the joint distribution of relations and the type set of their subject (T_s) and object (T_o), which we call $P(r, T_s, T_o)$. This distribution allows us to model how different types are related, and capture domain and range restrictions of relations in a fine grained way. For example, we can model not only that the relation `playsFor` has domain `Athlete` and range `SportsTeam`, but also how athletes are distributed over more specific types (e.g., `FootballPlayer`, `BasketballPlayer`, etc.) and how teams are distributed over subclasses of `SportsTeam` (e.g., `FootballTeam`, `BasketballTeam`, etc.), and most importantly, we can model that `FootballPlayer playsFor FootballTeam` and `BasketballPlayer playsFor BasketballTeam`.

We model the joint distribution $P(r, T_s, T_o)$ with the chain rule (3). We decompose it into the distribution of relations over facts $P(r)$, conditional distributions of subject type set given relation $P(T_s|r)$ and a conditional distributions of object type set given subject type set and relation $P(T_o|r, T_s)$.

$$P(r, T_s, T_o) = P(r)P(T_s|r)P(T_o|r, T_s) \tag{2}$$

$$P(r) = \frac{|\{p(s,o) \in \mathcal{A}_R | p = r\}|}{|\mathcal{A}_R|} \tag{3}$$

$$P(T_s|r) = \frac{|\{p(s,o) \in \mathcal{A}_R | p = r \wedge \tau(s) = T_s\}|}{|\{p(s,o) \in \mathcal{A}_R | p = r\}|} \tag{4}$$

$$P(T_o|r, T_s) = \frac{|\{p(s,o) \in \mathcal{A}_R | p = r \wedge \tau(s) = T_s \wedge \tau(o) = T_o\}|}{|\{p(s,o) \in \mathcal{A}_R | p = r \wedge \tau(s) = T_s\}|} \tag{5}$$

It is important to note that in case there are inconsistencies in the knowledge graph, such as domain/range violations or the assignment of inconsistent types, they are also captured in the distribution $P(r, T_s, T_o)$, and can be later replicated with their respective probabilities.

Besides the probability distributions of types and relations, individuals also follow a certain probability distribution, and not all relations have a uniform distribution w.r.t. their subjects and objects. In many cases, when selecting the individuals from E_{T_s} and E_{T_o}, there might be some bias which we should take into account. For instance, if we select $r = \texttt{livesIn}$, $T_s = \{\texttt{Person}\}$ and $T_o = \{\texttt{Country}\}$, we should not select the individual for `Country` based on an uniform distribution. The distribution should be biased towards more populous countries, e.g., the probability of selecting `China` should be much higher that of `Vatican`. At the same time, for the $r = \texttt{capitalOf}$ with $T_o = \{\texttt{Country}\}$, the

distribution of countries should be uniform since all the countries are equally likely to have a capital.

After selecting the relation r and type set of subject T_s and object T_o, we then need to select the subject and object individuals. Since in our synthesis process we first generate the individuals and their type assertions and then generate the relations assertions, there exist a limited number of individuals belonging to a given type set T which we define as $n_T = |E_T|$.

Following those considerations, we compute the conditional distributions of subject and object individuals given a relation and type set of subject and object, which we call $P(e|r, T_s)$ and $P(e|r, T_o)$, respectively. To that end, we count the occurrences of subject individuals for all relations r and subject type set T_s, and occurrences of object individuals for all r and T_o. We then sort the individuals by frequency in descending order and fit a distribution model.

We need to select an instance from a finite set E_T, and we should be able to vary the size n_T in order to be able to scale the knowledge base up and down. Therefore we consider the use of uniform and exponential truncated distributions (c.f. Equations 6 and 7).

$$f(x,b) = \begin{cases} \frac{1}{b} & \text{, if } 0 \leq x < b \\ 0 & \text{, otherwise} \end{cases} \quad (6) \qquad f(x,b) = \begin{cases} \frac{e^{-x}}{1-e^{-b}} & \text{, if } 0 \leq x < b \\ 0 & \text{, otherwise} \end{cases} \quad (7)$$

In truncated distributions, occurrences are limited to values which lie inside a given range. In the case of Equations 6 and 7, that interval is $0 \leq x < b$. It is important to use truncated functions, because when synthesizing relation assertions and selecting the individual for a given type, we can set $b = n_T$, and select an individual amongst the limited number of individuals that have the required type.

All distributions presented earlier in this section can effectively replicate some characteristics of a knowledge graph, such as in and out degree and density of the graph, however, they are not able to replicate more complex patterns involving paths in the graph. An example of such pattern in a knowledge graph containing data about families is that people who are married to the parent of a given child are also the parent of that child with some confidence. This pattern can be represented with the horn rule below.

$$\texttt{marriedTo(x,y)} \wedge \texttt{childOf(x,z)} \Rightarrow \texttt{childOf(y,z)} \quad [conf = 0.93]$$

Horn rules are basis of inductive logic programming (ILP) systems, such as ALEPH [19], WARMR [11], DL-Learner [14], and AMIE [9]. There are also ILP extensions with probabilistic methods [27] and that can efficiently handle numerical attributes [17]. We choose to use AMIE especially because of its better scalability in comparison to ALEPH and WARMR.

As most ILP systems, AMIE uses techniques to restrict the search space. AMIE mines only *closed* and *connected* rules. A rule is connected if all of its atoms are connected transitively to every other atom of the rule, and two atoms are connected if they share a variable or a constant. A rule is closed if every variable in the rule appears at least twice. Such rules do not predict merely the

existence of a fact (e.g. `diedIn(x,y)` ⇒ `wasBornIn(x,z)`, which is connected rule, but not closed), but the concrete arguments for it (e.g. `diedIn(x,y)` ⇒ `wasBornIn(x,y)`).

We use the horn rules learned by AMIE in our KB model in order to represent more complex patterns and use their associated PCA (partial close-world assumption) confidence value in the synthesis. In our model, we are able to ensure various relation characteristics. The RDF Schema domain and range restrictions can be ensured by the joint distribution $P(r, T_s, T_o)$. The horn rules can model symmetric, transitive, equivalent, and inverse properties.

To cover even more complex schemas, we additionally learn functionality, inverse functionality and non-reflexiveness from the data. All relations which do not have any same individual as both subject and object of a triple are considered non-reflexive, all relations with object cardinality of 1 are considered functional, and with subject cardinality of 1 are considered inverse functional. Learning these characteristics from data allows us to detect relations which might not have been conceived as, or not defined as such in the schema, but which in the available data present the characteristics. For instance, a dataset with the `childOf` relation, which is not functional, might contain data about people which have exclusively one child, and with our approach we ensure this characteristic is replicated.

4 Synthesis Process

Algorithm 1 summarizes the process of synthesizing a knowledge graph. As input, it uses the probability distributions $P(T)$, $P(r, T_s, T_o)$, $P(e|r, T_s)$, and $P(e|r, T_o)$, a set of horn rules $\mathcal{H}$, as well as the desired number of individuals n_e and relation assertions n_f to be synthesized.

The function VERIFY_TRIPLE first verifies if the exact same triple is already present in the synthesized KG. Then it checks whether functionality, inverse functionality, and non-reflexiveness are satisfied. That is, it verifies if there is no assertion with the given subject already present in the KG for functional relations, no assertion with the given object for inverse functional relations, and the given subject and object are different individuals for non-reflexive relations.

The function CHECK_HORN_RULES ensures that the patterns learned with the horn rules are replicated in the synthesized data. It checks if a newly synthesized fact triggers any of the learned horn rules. If a rule is triggered, the rule will produce a new fact with a probability equal to that of its confidence. The new facts produced by rules also need to be checked against the horn rules again, which means that the CHECK_HORN_RULES function is called recursively until it does not produce any new facts.

The function UPDATE_DISTRIBUTION makes sure that the original distribution $P(r, T_s, T_o)$ is not distorted by the production of new facts from horn rules, which may not follow $P(r, T_s, T_o)$. Therefore, it is necessary to adjust the joint distribution in order to compensate this effects. We do that by simply keeping counts for the relations, subject and object type sets, and based on the number

Algorithm 1 Knowledge base synthesis process

function GEN_KB($n_e, n_f, P(T), P(r), P(r, T_s, T_o), \mathcal{H}$)
  $\mathcal{A} \leftarrow \emptyset$ ▷ Create empty A-Box
  $E \leftarrow \{\}$ ▷ Map of type sets and their entities
  for $i \leftarrow 1$ to n_e **do** ▷ synthesize entities
    $T_i \leftarrow$ randomly choose from $P(T)$
    $E[T_i] \leftarrow E[T_i] \cup \{e_i\}$
    for $C \in T_i$ **do**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{C(e_i)\}$
    end for
  end for
  $i \leftarrow 0$
  while $i < n_f$ **do** ▷ synthesize relation assertions
    $r_i, T_{s_i}, T_{o_i} \leftarrow$ randomly choose from $P(r, T_s, T_o)$ ▷ use chain rule
    $s_i \leftarrow$ SELECT_ENTITY($E[T_{s_i}], P(e|r_i, T_{s_i})$)
    $o_i \leftarrow$ SELECT_ENTITY($E[T_{o_i}], P(e|r_i, T_{o_i})$)
    if VERIFY_TRIPLE(s_i, r_i, o_i) **then**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{r_i(s_i, o_i)\}$
      CHECK_HORN_RULES($\mathcal{A}, (s_i, r_i, o_i), \mathcal{H}$)
      UPDATE_DISTRIBUTION($P(r, T_s, T_o)$)
      $i \leftarrow i + 1$
    end if
  end while
  return $\mathcal{A}$
end function

of facts to be synthesized and the distribution of already synthesized facts we can adjust $P(r, T_s, T_o)$.

Another detail not shown in algorithm 1 is the use of a pool of subjects for functional and pool of objects for inverse functional relations. We do that in order to avoid generating facts which violate the functionality and inverse functionality restrictions. If no pools are considered, the probability of generating violating facts for a given relation increases linearly with the number of already existent facts. With the pools, all individuals of a given type are initially in the pool, and whenever an individual is picked to generate a new fact, this individual is removed from the pool and cannot be picked again, therefore preventing the violations.

In the synthesis process some characteristics can be easily changed. Noise can be introduced by smoothing the distribution $P(r, T_s, T_o)$, making the probability for invalid combinations of relations, subject and object types non-zero. The density of the knowledge graph can be altered by modifying the ratio n_f/n_e between number of facts and individuals. It is possible to change the scale of the synthetic knowledge graphs by simply multiplying the original number of individuals n_e and facts n_f by a constant. That is, assuming that the number of relations in the knowledge graph are linear, i.e., the number of relation assertions grows linearly with the number of individuals.

However, some knowledge graphs might have relations which are quadratic, e.g. `owl:differentFrom` that indicates individuals that are not the same. Therefore, for the quadratic relations of a knowledge base, we need to scale the number of relation assertions quadratically with the number of individuals. This kind of relations are rather rare, and they can be difficult to automatically detect. We use a simple heuristic based on thresholds for the average number of different objects per subject and different subjects per object. If both thresholds are reached, we assume the relation to be quadratic.

One important characteristic is that the synthesis process is based on pseudo random number generators (PRNG), therefore, the process is deterministic and identical datasets can be generated if the same seed is used. By using different seeds, it is also possible to generate different datasets from the same model and with similar characteristics, allowing us to test the stability of methods.

5 Experiments

The link prediction task consists of predicting the existence (or probability of correctness) of edges in the graph (i.e., triples). This is important since existing knowledge graphs are often missing many facts, and some of the edges they contain are incorrect. Nickel et al. [20] present a review of multirelational models, many of which have been used for the link prediction task. In this paper, we select five popular methods to be used in our experiments: Path Ranking algorithm [13], SDValidate [26], Holographic embeddings (HolE) [21], Translation embeddings (TransE) [4] and RESCAL [22]. In our experiments, we evaluate the prediction of relation assertions only. All the measurements reported were obtained using 5-fold cross-validation. The test set consists of the 20% of positive positive triples selected in the cross-validation, plus negative examples. There are the same number of positive and negative examples in the test set, and the negative examples are generated by corrupting each of the positive triples following the method described by Bordes et al. [4].

Type prediction can be considered a subtask of link prediction where we are interested on prediction links for the relation `rdf:type`. There are several type prediction approaches which rely on external features [32, 3, 10, 23], however, in this paper, we concentrate on methods which rely on features extracted from the knowledge graph. The methods used in the experiments are SDType [25] and SLCN [16], as well as the state-of-art multilabel classifiers MLC4.5 [7], MLP [33] and MLkNN [34] – multilabel versions of decision tree, multilayer perceptron and k-nearest neighbors – with ingoing and outgoing links used as features as described in [16].

As input knowledge graphs, we use Wikidata, DBpedia (2015-10), and NELL. We use the following smaller domain specific datasets: Thesoz[2], Semantic Bible[3]

[2] `http://www.gesis.org/fileadmin/upload/dienstleistung/tools_standards/thesoz_skos_turtle.zip`

[3] `http://www.semanticbible.com/`

Dataset	Entities	Types	Rels	Type ass.	Relation ass.	Density
Wikidata	19060716	474	482	40198183	18955236	$1.082 \cdot 10^{-10}$
DBpedia	4940352	1027	646	31521734	14747048	$9.353 \cdot 10^{-10}$
NELL	1475674	276	248	5565472	174621	$3.233 \cdot 10^{-10}$
AIFB	27100	63	82	59613	59349	$9.855 \cdot 10^{-7}$
Mutagenesis	14157	91	4	48111	26533	$3.310 \cdot 10^{-5}$
SemanticBible	789	71	31	2563	2482	$1.286 \cdot 10^{-4}$
Thesoz	48540	10	16	109960	275430	$7.306 \cdot 10^{-6}$
NobelPrize	10013	23	18	19506	30148	$1.671 \cdot 10^{-5}$
ESWC2015	1285	16	25	1285	4062	$9.840 \cdot 10^{-5}$
ISWC2013	2548	20	39	2545	9992	$3.946 \cdot 10^{-5}$
WWW2012	3836	22	43	3907	15406	$2.435 \cdot 10^{-5}$
LREC2008	3502	7	24	3502	16514	$5.611 \cdot 10^{-5}$

Table 1: Statistics about the datasets used in the experiments

AIFB portal[4], Nobel Prize[5] and Mutagenesis. We also select four of the largest conference datasets from the Semantic Web dog food corpus[6], i.e., LREC2008, WWW2012, ISWC2013, and ESWC2015. Some relevant statistics about the datasets used in the experiments are shown in Table 1.

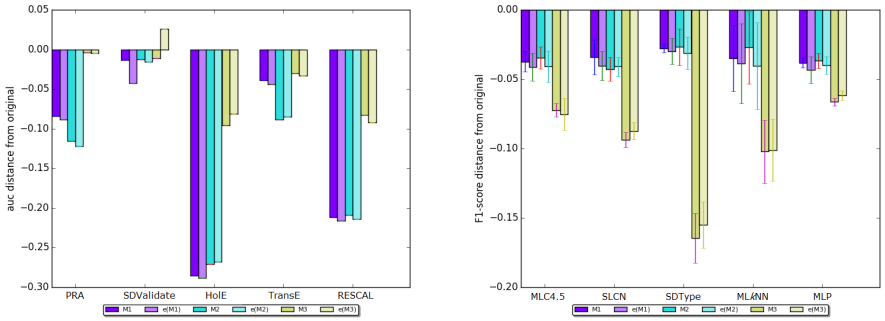
For every input KG, we synthesize replicas of three different sizes increased by factors of 10. For smaller datasets we also scale the replicas up. On the Semantic Web dog food datasets we synthesize replicas of sizes 10%, 100% and 1000%. For large datasets we scale the replicas down (DBpedia and Wikidata replicas are of sizes 0.01%, 0.1% and 1%, and the remaining datasets 1%, 10% and 100%).

We use the scikit-kge[7] implementation of HolE, TransE and RESCAL, and the scikit-learn implementation of MLkNN, MLC4.5 and MLP. We implemented the remaining methods ourselves. The proposed synthesis process code is available to download.[8]

The evaluation measures used in the link experiments are the area under the precision-recall curve (PR AUC) and area under the ROC curve (ROC AUC). For the type prediction experiments we use micro-averaged F_1-score and accuracy. We compute the distance of these evaluation measures between the results on the original datasets, and their synthetic replicas. In order to compare the ranking of methods, we use the Spearman-ρ rank correlation coefficient. All the results reported in this paper were obtained with 5-fold cross-validation.

In order to evaluate how the different parts of the proposed knowledge base model affect the results on link and type prediction tasks, we use 6 different models in our evaluation: M1, M2, M3, e(M1), e(M2) and e(M3):

[4] `http://www.aifb.kit.edu/web/Web_Science_und_Wissensmanagement/Portal`
[5] `http://www.nobelprize.org/nobel_organizations/nobelmedia/nobelprize_org/developer/manual-linkeddata/terms.html`
[6] `http://data.semanticweb.org/dumps/conferences/`
[7] `https://github.com/mnick/scikit-kge`
[8] `https://github.com/aolimelo/kbgen`



(a) Link prediction PR AUC distance on Nobel Prize dataset

(b) Type prediction F_1-score distance on Wikidata

Fig. 1: Distances of performance measures to original datasets

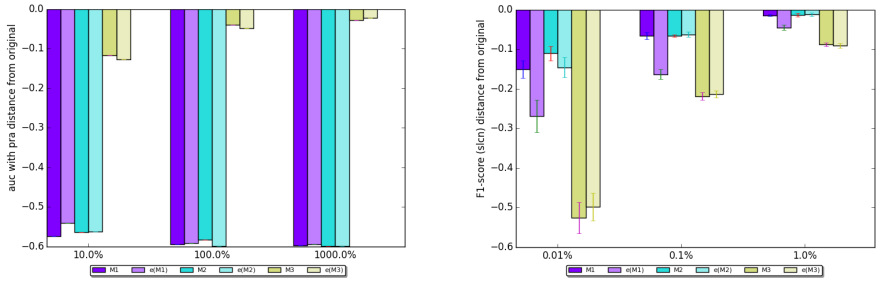
- M1 is the simplest version, which considers only the distributions $P(T)$ and $P(r, T_s, T_o)$. The bias to selection of individuals is not considered, and individuals are always selected from an uniform distribution. No relation characteristics (apart from domain and range restrictions covered by $P(r, T_s, T_o)$) are considered.
- M2 is M1 plus functionality, inverse functionality and non-reflexiveness of relations.
- M3 is M2 plus the horn rules learned with AMIE.
- The models e(Mi) are the model Mi plus the biases to selection of individuals $P(e|r, T_s)$ and $P(e|r, T_o)$.

We use AMIE with its default parameter settings (i.e., no rules with constants, maximum rule length = 3, confidence computed with PCA, minimum support = 100 examples, minimum head coverage = 0.01).

We use PRA with maximum path length of 3 for all datasets. For HolE, TransE and RESCAL we learn embeddings with 20 dimensions and maximum of 100 epochs. While this may not be the optimal settings for most datasets, we consistently use the same settings throughout all of our experiments, since our aim is not to achieve optimal results, but to show that the benchmark synthesis works as desired.

Figure 1a shows an example of PR AUC distance on link prediction from for the Nobel Prize datasets between original and replica (100% size) with the 5 selected methods. It is clear that the use of horn rules significantly improves the results, as M3 and e(M3) performs better than the other methods, except from SDValidate, which relies on exclusively on distributions of relations and object types and does not exploit more complex path patterns.

Figure 1b shows an example of F_1-score distance on type prediction for Wikidata between original and replica (0.1% size). It is noticeable that horn rules do not improve the results, as M1, M2, e(M1) and e(M2) perform better than M3 and e(M3). This is explained by the fact that most of the evaluated type prediction methods rely solely on ingoing and outgoing links of entities. More-



(a) Link prediction on LREC2008 (PRA) (b) Type prediction on DBpedia (SLCN)

Fig. 2: Effect of scaling the replica sizes up and down

	PR AUC						ROC AUC					
	M1	e(M1)	M2	e(M2)	M3	e(M3)	M1	e(M1)	M2	e(M2)	M3	e(M3)
ρ_{all}	0.527	0.643	0.567	0.607	0.643	**0.653**	0.647	0.613	**0.657**	0.613	0.610	0.577
ρ_{large}	0.640	0.740	0.590	0.580	0.730	**0.800**	0.650	0.610	0.640	0.630	**0.670**	0.620
d_{all}	0.243	0.247	0.247	0.245	**0.112**	0.115	0.231	0.230	0.231	0.231	**0.109**	0.111
d_{large}	0.215	0.228	0.216	0.219	**0.082**	0.089	0.211	0.215	0.208	0.211	**0.087**	0.095

Table 2: Summary of the link prediction results

over, as explained in Section 3, horn rules can disturb the original distribution $P(r, T_s, T_o)$, which is crucial for the replication of ingoing and outgoing links.

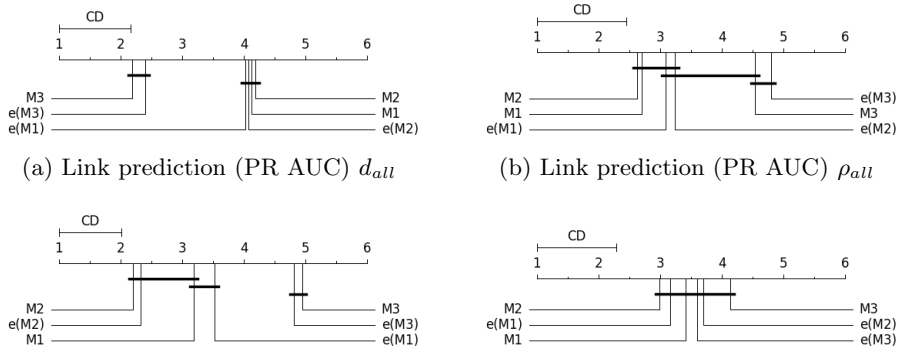
Tables 2 and 3 show a summary of the results obtained over all datasets for type prediction and link prediction, respectively. The values with subscript *all* report the average of the results over all different sizes of replicas, while those with subscript *large* report the averages over the largest size of replicas only. We do that because different models, especially M3 and e(M3), perform worse than others for smaller replica sizes, and we also want to know how the models perform when ruling out this effect.

The results of Table 2 indicate that in terms of distance, M3 is the best method overall, however, when it comes to preserving the rankings, the results become more mixed. It is clear that introducing the horn rules does have a positive effect on the model, especially for the distances which are reduced to less than half of that of other models. In Table 3 we can see that M2 is the best overall in terms of distance for both PR and ROC AUC, while for the rankings, the use of horn rules again have a positive impact with M3 being the best method overall. The link prediction results were reported for all datasets apart from DBpedia and Wikidata. Because of the large size of these two datasets and the complexity of the approaches, the experiments did not finish in less than a week.

We also perform the Nemenyi test in order to find how significant the differences of the evaluated models is, both in terms of distance and ranking. Figure 3 shows the critical distance diagrams. For the distances d the models on the left side are the best performers, since lower distances are desired, while for Spearman's rank correlations ρ the models on the right side are the best performers, since higher correlations are desired.

	F_1-score						*Accuracy*					
	M1	e(M1)	M2	e(M2)	M3	e(M3)	M1	e(M1)	M2	e(M2)	M3	e(M3)
ρ_{all}	0.208	0.221	0.195	0.259	**0.362**	0.265	0.290	0.334	0.315	0.343	**0.406**	0.307
ρ_{large}	0.343	0.273	0.251	0.400	**0.456**	0.410	0.470	0.420	0.357	0.498	**0.502**	0.433
d_{all}	0.086	0.098	**0.082**	0.083	0.131	0.130	0.061	0.064	**0.057**	0.061	0.065	0.066
d_{large}	0.059	0.065	**0.055**	0.057	0.083	0.084	0.056	0.060	**0.054**	0.057	0.061	0.062

Table 3: Summary of the type prediction results



(a) Link prediction (PR AUC) d_{all}

(b) Link prediction (PR AUC) ρ_{all}

(c) Type prediction (F_1) d_{all}

(d) Type prediction (F_1) ρ_{all}

Fig. 3: Nemenyi Critical distance diagrams for link and type prediction

In Figure 3a we can see that PR AUC distances on link prediction between the models with horn rules (M3 and e(M3)) and the others is very significant, while the differences in terms of Spearman-ρ from Figure 3b are closer to the critical distance (CD). We can also observe that the difference between M3 and e(M3) is not significant, indicating that the use of bias to selection of instances does not have a great impact. One possible explanation for that is the fact that, in order to simplify our model and abstract from specific instances, we assume that, for a given type set, the most frequent instances are always the same. That is, if we consider the type set {`Country`} as object of `livesIn` and `beatifiedPlace`, we assume that the most frequent country in both cases is the same individual, while in reality the most frequent country for `livesIn` would be China and for `beatifiedPlace` Italy. Since the computation of the bias can be very expensive, especially for larger datasets with high number of types and individuals, M3 would be a more reasonable choice than e(M3).

When analyzing Figure 3c, we notice that, in terms of F_1-score distance, the M2, e(M2) and M1 are not significantly different from each other, and the use of horn rules has a significant negative effect. The Spearman-ρ from Figure 3d values are very close to each other, without any significant difference between the evaluated models.

We illustrate the difference in runtime for the synthesis processes with different methods with Figure 4. The plot shows the number of facts generated over

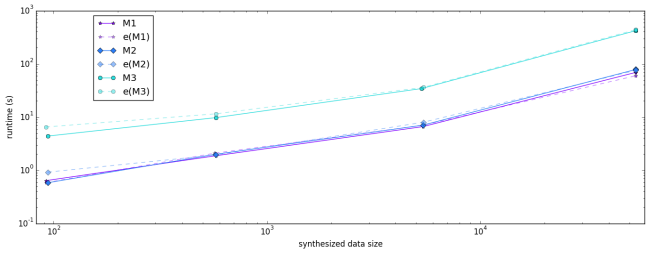


Fig. 4: Synthesis process runtime over dataset size for the ESWC2015 model

time for the ESWC2015 dataset. It is clear that M3 and e(M3) are significantly slower than the others. It is also worth noting that these two models require horn rules, which need to be learned with AMIE increasing the model learning time as well.

6 Conclusion and Outlook

In this paper, we have proposed a knowledge graph model and synthesis process which is able to capture essential characteristics of existing knowledge graphs, which allows us to create replicas of those graphs at different scales.

Extensive experiments comparing the replicas and original datasets in the link and type prediction tasks were conducted. We have performed evaluations with five different methods for each tasks and comparisons of distances and methods rankings between replicas and original datasets. Overall, the model M3 was the best performer, and the use of horn rules significantly improved the results. The use of a bias to selection of subject and object individuals did not show any significant improvement. In general, we recommend the use of M3, unless the objective is to replicate the results of type prediction on a single methods, without performing any comparisons. In that case M2, which does not include horn rules, would be the best option.

In the future, we intend to start synthesizing knowledge graphs from scratch, which would involve the synthesis of whole schemas. We plan to create a system which enables users to synthesize data based on a set of parameters that gives control on important characteristics of a knowledge base, such as number of entities, types, relations, assertions of types and relations, density, connectivity. Finally, we want to synthesize a set of knowledge bases of different characteristics to create a larger collection of benchmarks for link prediction and type prediction.

Acknowledgements

The work presented in this paper has been partly supported by the Ministry of Science, Research and the Arts Baden-Württemberg in the project SyKo^{2}W^2 (Synthesis of Completion and Correction of Knowledge Graphs on the Web).

References

1. Albuquerque, G., Löwe, T., Magnor, M.: Synthetic generation of high-dimensional datasets. IEEE Transactions on Visualization and Computer Graphics (TVCG, Proc. Visualization / InfoVis) 17(12), 2317–2324 (Dec 2011)
2. Angles, R., Boncz, P., Larriba-Pey, J., Fundulaki, I., Neumann, T., Erling, O., Neubauer, P., Martinez-Bazan, N., Kotsev, V., Toma, I.: The linked data benchmark council: A graph and rdf industry benchmarking effort. SIGMOD Rec. 43(1), 27–31 (May 2014)
3. Aprosio, A.P., Giuliano, C., Lavelli, A.: Automatic expansion of DBpedia exploiting Wikipedia cross-language information. In: ESWC 2013 (2013)
4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 2787–2795. Curran Associates, Inc. (2013)
5. Chawla, S., Gionis, A.: k-means-: A unified approach to clustering and outlier detection. In: Proceedings of the 13th SIAM International Conference on Data Mining, Austin, Texas, USA. pp. 189–197. SIAM (2013)
6. Cheatham, M., Dragisic, Z., Euzenat, J., Faria, D., Ferrara, A., Flouris, G., Fundulaki, I., Granada, R., Ivanova, V., Jiménez-Ruiz, E., et al.: Results of the ontology alignment evaluation initiative 2015. In: 10th ISWC workshop on ontology matching (OM). pp. 60–115 (2015)
7. Clare, A., King, R.D.: Knowledge discovery in multi-label phenotype data. In: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery. pp. 42–53. PKDD'01, Springer-Verlag, London, UK (2001)
8. van Erp, M., Mendes, P., Paulheim, H., Ilievski, F., Plu, J., Rizzo, G., Waitelonis, J.: Evaluating entity linking: An analysis of current benchmark datasets and a roadmap for doing a better job. In: of the Language Resources and Evaluation Conference. ELRA (2016)
9. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.M.: AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In: WWW 2013, Rio de Janeiro, Brazil, 2013. pp. 413–422. ACM (2013)
10. Gangemi, A., Nuzzolese, A.G., Presutti, V., Draicchio, F., Musetti, A., Ciancarini, P.: Automatic typing of dbpedia entities. In: ISWC 2012. pp. 65–81 (2012)
11. Goethals, B., Van den Bussche, J.: Relational Association Rules: Getting Warmer, pp. 125–139. Springer Berlin Heidelberg, Berlin, Heidelberg (2002)
12. Guo, Y., Pan, Z., Heflin, J.: Lubm: A benchmark for owl knowledge base systems. Web Semant. 3(2-3), 158–182 (Oct 2005)
13. Lao, N., Cohen, W.W.: Relational retrieval using a combination of path-constrained random walks. Mach. Learn. 81(1), 53–67 (Oct 2010)
14. Lehmann, J.: Dl-learner: Learning concepts in description logics. Journal of Machine Learning Research 10, 2639–2642 (2009)
15. Lopez, V., Unger, C., Cimiano, P., Motta, E.: Evaluating question answering over linked data. Web Semantics: Science, Services and Agents on the World Wide Web 21, 3–13 (2013)
16. Melo, A., Paulheim, H., Völker, J.: Type prediction in RDF knowledge bases using hierarchical multilabel classification. In: Proceedings of the International Conference on Web Intelligence, Mining and Semantics, WIMS 2016, Nîmes, France, 2016. pp. 14:1–14:10 (2016)

17. Melo, A., Theobald, M., Völker, J.: Correlation-based refinement of rules with numerical attributes. In: Proceedings of the International Florida Artificial Intelligence Research Society Conference, FLAIRS, Pensacola, Florida, 2014. (2014)
18. Morsey, M., Lehmann, J., Auer, S., Ngonga Ngomo, A.C.: DBpedia SPARQL Benchmark—Performance Assessment with Real Queries on Real Data. In: ISWC 2011 (2011)
19. Muggleton, S.: Learning from positive data. In: 6th International Workshop on Inductive Logic Programming. pp. 358–376. Springer-Verlag (1997)
20. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. Proceedings of the IEEE 104(1), 11–33 (2016)
21. Nickel, M., Rosasco, L., Poggio, T.A.: Holographic embeddings of knowledge graphs. CoRR abs/1510.04935 (2015)
22. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). pp. 809–816. ACM (2011)
23. Nuzzolese, A.G., Gangemi, A., Presutti, V., Ciancarini, P.: Type inference through the analysis of wikipedia links. In: WWW 2012 Workshop on Linked Data on the Web, Lyon, France, 2012. CEUR Workshop Proceedings, vol. 937 (2012)
24. Paulheim, H.: Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web 8(3), 489–508 (2017)
25. Paulheim, H., Bizer, C.: Type inference on noisy RDF data. In: ISWC 2013, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I. pp. 510–525 (2013)
26. Paulheim, H., Bizer, C.: Improving the quality of linked data using statistical distributions. Int. J. Semant. Web Inf. Syst. 10(2), 63–86 (Apr 2014)
27. Raedt, L.D., Frasconi, P., Kersting, K., Muggleton, S. (eds.): Probabilistic Inductive Logic Programming - Theory and Applications, Lecture Notes in Computer Science, vol. 4911. Springer (2008)
28. Ristoski, P., de Vries, G.K.D., Paulheim, H.: A collection of benchmark datasets for systematic evaluations of machine learning on the semantic web. In: ISWC. Springer (2016)
29. Samadi, B., Cipolone, A., Lin, P.J., Xiao, R., Jeske, D.R., Holt, D., Rend, C., Cox, S.: Development of a synthetic data set generator for building and testing information discovery systems. Third International Conference on Information Technology
30. Schmidt, M., Hornung, T., Lausen, G., Pinkel, C.: Sp2bench: A SPARQL performance benchmark. CoRR abs/0806.4627 (2008)
31. Theodoridis, Y., Nascimento, M.A.: Generating spatiotemporal datasets on the www. SIGMOD Rec. 29(3), 39–43 (Sep 2000)
32. Yosef, M.A., Bauer, S., Hoffart, J., Spaniol, M., Weikum, G.: HYENA: hierarchical type classification for entity names. In: COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Posters, Mumbai, India. pp. 1361–1370 (2012)
33. Zhang, M.L., Zhou, Z.H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. on Knowl. and Data Eng. 18(10), 1338–1351 (Oct 2006)
34. Zhang, M.L., Zhou, Z.H.: ML-KNN: A lazy learning approach to multi-label learning. Pattern Recogn. 40(7), 2038–2048 (2007)