LiterallyWikidata - A Benchmark for Knowledge Graph Completion Using Literals

Genet Asefa Gesese[1,2(✉)], Mehwish Alam[1,2(✉)], and Harald Sack[1,2(✉)]

[1] FIZ Karlsruhe – Leibniz Institute for Information Infrastructure,
Eggenstein-Leopoldshafen, Germany
{genetasefa.gesese,mehwish.alam,harald.sack}@fiz-karlsruhe.de
[2] Karlsruhe Institute of Technology, Institute AIFB, Karlsruhe, Germany

Abstract. In order to transform a Knowledge Graph (KG) into a low dimensional vector space, it is beneficial to preserve as much semantics as possible from the different components of the KG. Hence, some link prediction approaches have been proposed so far which leverage literals in addition to the commonly used links between entities. However, the procedures followed to create the existing datasets do not pay attention to literals. Therefore, this study presents a set of KG completion benchmark datasets extracted from Wikidata and Wikipedia, named LiterallyWikidata. It has been prepared with the main focus on providing benchmark datasets for multimodal KG Embedding (KGE) models, specifically for models using numeric and/or text literals. Hence, the benchmark is novel as compared to the existing datasets in terms of properly handling literals for those multimodal KGE models. LiterallyWikidata contains three datasets which vary both in size and structure. Benchmarking experiments on the task of link prediction have been conducted on LiterallyWikidata with extensively tuned unimodal/multimodal KGE models. The datasets are available at https://doi.org/10.5281/zenodo.4701190.

Keywords: Knowledge graph completion · Knowledge graph embedding · Link prediction · Literals · Benchmark dataset

1 Introduction

Knowledge Graphs (KGs) are composed of structured information describing facts about a particular domain through entities and interrelations between them. Recently, KGs have become crucial to improve diverse real-world applications mainly in the areas of Natural Language Processing (NLP) such as question answering, named entity disambiguation, information extraction, and etc. [9,38]. Due to the Open World Assumption, KGs are never complete, i.e., there are always some facts missing. In order to solve this problem, different KG embedding models have been proposed for automated KG Completion (KGC). Most of these models are based on the tasks such as link prediction, triple classification, and entity classification/typing. Some of these embedding models make

A. Hotho et al. (Eds.): ISWC 2021, LNCS 12922, pp. 511–527, 2021.
https://doi.org/10.1007/978-3-030-88361-4_30

use of only relational triples (triples with object properties), such as TransE [6], DistMult [43], ConVE [10], RotatE [34], and etc. On the other hand, some models such as LiteralE [21], KBLRN [13], MTKGNN [35], and MKBE [27] use relational triples together with attributive triples (i.e., triples with datatype properties which take literals as values) and images of entities (refer to [15] for more details).

The performance of various KGE approaches, mainly link prediction models, has been evaluated using some commonly known KGC datasets. Most of these datasets except CoDEx [29], are outdated and easy for link prediction tasks such as FB15K [6] and FB15K-237 [36] which are subsets of the no longer maintained KG Freebase [5]. Moreover, attributive triples have not been handled properly in any of the current datasets. For instance, in CoDEx-M [29], it is not possible to find a single datatype property in Wikidata with numerical literal values for some of the entities. Apart from numerical properties, the major existing datasets also contain a significant number of entities for which there is no textual description available. For instance, in CoDEx among the total number of 77,951 entities, 17,276 of them do not have textual descriptions in English, i.e., they are not represented in English Wikipedia. Hence, in those studies which combine KG and textual entity descriptions for representation learning (such as DKRL [41]) it is common to filter out these entities in order to train the embedding models. This indicates that a high-quality benchmark that covers both relational and attributive triples is required to evaluate the performance of the state-of-the-art KGC models.

Therefore, in this work a KGC benchmark **LiterallyWikidata** which properly combines attributive triples with relational triples by taking into account the aforementioned concerns is presented. **LiterallyWikidata** consists of a set of KGC datasets extracted from Wikidata and Wikipedia. In addition to Github, all of the datasets are made available also on Zenodo under Creative Commons Attribution 4.0 International license to ensure long-term findability through a persistent identifier[1].

The contributions of this work are summarized as follows:

- **Datasets**: **LiterallyWikidata** which is a benchmark containing three subsets of Wikidata varying in size and structure is introduced. Each of these subsets contains both relational and attributive triples along with entity types.
- **Automatic dataset creation pipeline**: As compared to the way the current benchmarks are created, for instance, CoDEx, the pipeline used in this work requires very little human intervention. In CoDEx, the first step taken was defining a set of initial classes in some specific domains whereas in our pipeline it is not required for the domains and initial classes to be predefined. Moreover, it is possible to adapt the pipeline to create new datasets with newer Wikidata dumps.
- **Benchmarking**: Extensive KGC experiments have been conducted on **LiterallyWikidata** for selected embedding models with and without attributive triples on the task of link prediction.

[1] The details including the DOI are given under the reference [14].

- **Review of existing link prediction datasets**: A review of the existing KGC datasets in terms of their sources, domain, and support for literals has been conducted and presented in Table 1.

The rest of the paper is organized as follows: Sect. 2 discusses the existing KGC frameworks/datasets followed by Sect. 3, where a detailed description of the procedure followed to generate the LiterallyWikidata datasets is presented. Section 4 demonstrates the comparison between existing datasets and Literally-Wikidata whereas Sect. 5 presents benchmarking experiments on the generated datasets with uni/multimodal KGE models. Finally, concluding remarks along with directions for future work are stated in Sect. 6.

2 Related Work

A summary of the recent and the most common existing KGC benchmarks, specifically link prediction datasets, is given in Table 1. The sources of the majority of these datasets are Freebase [5], WordNet [25], YAGO [33], Wikidata [37], and NELL [8].

Freebase Extracts. FB15K and FB15K-237 are among the most popular datasets to evaluate KGC models. Even though the original releases of both datasets do not include any attributive triples, they have been extended with textual and numerical attributes [21,40,41]. However, different studies [10,15,29] have claimed that FB15K does not possess the required qualities to be actually used as a benchmark, i.e., it contains multiple inverse relations. On the other hand, in FB15K-237 which is a subset of FB15K without inverse relations, all validation and test triples containing entity pairs directly linked in the training set have been removed. Moreover, FB15K-237 contains a significant amount of triples with skewed relations towards either some head or tail entities [29] (see Sect. 4 for more details).

WordNet Extracts. Among the WordNet datasets, WN18 [6] and WN18RR [10] are the most popular ones. Both datasets are smaller in size and domain-specific as compared to the other datasets such as FB15K-237. Besides, the original releases do not contain any numerical attributive triples.

YAGO Extracts. YAGO3-10 [10] is the widely used dataset among those extracted from YAGO. It is a dataset that contains only relational triples from YAGO3 [23] mostly about locations and people. The dataset has been extended with numerical attributes, textual entity descriptions, and entity images in [27] and only with numerical attributes in [21]. Most of all, as discussed in [1], YAGO3-10 has a significant number of triples with two duplicate relations *isAffiliatedTo* and *playsFor* which makes the dataset easy for a link prediction task.

Wikidata (and Wikipedia) Extracts. Wikidata-authors [30] is a domain-specific dataset containing relational triples from Wikidata where the head entities are persons who are authors or writers. Apart from having a narrow scope and a small set of triples (i.e., 86,376), this dataset doesn't have any attributive triples. CoDEx [29] is a recent KGC benchmark extracted from Wikidata and Wikipedia. The relational triples in this dataset are from Wikidata and the attributive triples have been provided as auxiliary information taken from both Wikidata and Wikipedia. The auxiliary information contains Wikidata labels, descriptions of entities and relations along with Wikipedia page extracts for entities. This dataset does not include any numeric attribute and if we try to extract them from Wikidata, there are only limited number of entities in the dataset which have numeric attributes. Moreover, in CoDEx the set of triples already contain classes and this may decrease the level of difficulty of the dataset for tasks other than link prediction and triple classification that involve classes, i.e., entity typing/classification.

Others. There are other datasets such as NELL-995 [42] and MovieLens [27] (see Table 1 for more details). NELL-995 is a dataset extracted from the 995th iteration of NELL [8]. Due to the fact that the triples in NELL-995 are nonsensical or overly generic, the dataset is not suitable to be used as a KGC benchmark [29]. Moreover, the dataset does not have any attributive triples. MovieLens [27] is a dataset about movies where relational triples, numerical attributes, and textual attributes are from ML100K [17] and images are movie posters from TMDB[2]. This dataset contains few entities, relations, and triples as compared to the widely used KGC datasets, such as FB15K-237. Moreover, not all of the entities have textual attributes. Another very recently released benchmark is Kgbench [4] which could be used for both node classification and link prediction. However, baseline results are only provided for node classification task because the datasets are generated primarily for that particular task. Kgbench provides a set of different domain-specific datasets and in each dataset the source for the multimodality are mainly images and hence, numeric literals are available only for a limited number of entities whereas LiterallyWikidata is a collection of domain-generic datasets with every entity having some numeric literals.

In general, the existing KGC benchmarks do not give proper emphasis to attributive triples, i.e., attributes are treated as auxiliary information. Consequently, the attributive triples are either way unbalanced, less in number, or have few unique attributes. Therefore, in this work, a new KGC benchmark called **LiterallyWikidata** is presented which properly handles literals, specifically, numerical attributes and textual descriptions.

3 Dataset Creation

In this section, the procedure followed to create the LiterallyWikidata benchmark is discussed in detail. First, attributive triples with numerical literals are extracted from the Wikidata full dump from 07 September, 2020[3]. Then, relational triples are retrieved from the dump for the entities with the attributive triples. Once the triples are extracted, duplicate triples are filtered out and different datasets varying in size and structure are generated, namely, **LitWD1K**,

[2] https://www.themoviedb.org/.

[3] https://dumps.wikimedia.org/wikidatawiki/.

Table 1. Existing KGC datasets for the task of link prediction.

Dataset	Sources	Domain: Specific (●) Generic (⋆)	Attributive **triples**: Text (●), Numerical (⋆), Image (✓)	
			Original	Extended
CoDEx [29]	Wikidata [37], Wikipedia	⋆	●	
Wikidata-authors [30]	Wikidata	●		
FB15K [6]	Freebase	⋆		● [41] ⋆[40] ⋆[21]
FB15K-237 [36]				⋆[21] ● [21]
FB15k-237-OWE [31]			●	
FB20K [41]			●	
FB13 [32]				
FB5M [39]				
FB24K [22]				
FB15K-401 [43]				
WN18 [6]	WordNet [25]	●		
WN18RR [10]				
WN11 [32]				
YAGO3-10 [10]	YAGO	⋆		⋆[21] ● ⋆ ✓ [27]
YAGO37 [16]				
YG58K [40]				⋆[40]
NELL-995 [42] and other Nell varieties [26]	NELL [8]			
MovieLens [27]	ML 100K [17], TMDB[a]	●	● ⋆ ✓	
UMLS [20]	UMLS [24]			
kinship [20]	Alyawarra kinship [19]			
Nations [20]	Nations Project [28]			
Countries [7]	Countries data[b]			
Family [11,12]	Families [18]			

[a] https://www.themoviedb.org/.

[b] https://github.com/mledoze/countries.

LitWD19K, and **LitWD48K**. Finally, each of the datasets is divided into training, validation, and testing triples. Note that classes explicitly have not been considered as entities in this framework in order to enable the adaptability of the datasets for tasks other than link prediction such as entity type prediction. Classes in Wikidata are those items which occur either as the value/object in an instance-of (P31) statement/triple or they are subject or value/object in a subclass-of (P279) statement. In the subsequent sections, the steps taken to generate the datasets are discussed in detail, i.e., i) extracting attributive triples, ii) extracting relational triples, and iii) filtering the triples.

3.1 Extracting Attributive Triples

Note that in this phase the main focus is on extracting attributive triples with datatype properties taking numerical values. Therefore, the first step is identifying those data type properties in Wikidata. The Wikidata properties which are typed with any of the three Wikimedia datatypes *Wikimedia:Time*, *Wikimedia:GlobeCoordinate*, and *Wikimedia:Quantity* are considered, in this work, as properties taking numeric values.

Wikimedia:Time. Those properties which take *point in time* values, such as P569 (date of birth) are categorized as *Wikimedia:Time* properties.

Wikimedia:GlobeCoordinate. The values of *Wikimedia:GlobeCoordinate* typed properties such as P625 (coordinate location), are geographic coordinates given as latitude-longitude pairs. We have separated these pairs by attaching the postfix “longtiude” and “latitude” to the ID of the properties. For instance, the triple

<Q100000 P62‘‘Point(5.7678 50.8283)"^^geo[4]:wktLiteral .>

is transformed into the following two triples:

<Q100000 P625_Longtiude ‘‘5.7678"^^xsd[5]:double .> and
<Q100000 P625_Latitude ‘‘50.8283"^^xsd:double .>

Note that some entities have multiple values per property. For such entities, splitting their corresponding triples might create a logical problem, i.e., it would be difficult to associate longitude and latitude values once the triples are split. Therefore, only one triple per <*entity, property*> pair has been randomly selected before splitting.

Wikimedia:Quantity. Properties of wikimedia type *Wikimedia:Quantity* take quantities representing decimal numbers, such as P2049 (width). In the case of these properties, for every <*entity, property*> pair statements ranked as “preferred” are retrieved if there are any. Otherwise, all statements which are ranked as “Normal” are extracted. In Wikidata, such statements have units associated with their values. These units might be either SI units or non-SI units. Those values with non-SI units are normalized to their corresponding SI unit whenever possible. There are still properties with more than one unit after normalization. These units are either not normalizable or are outliers. For each statement with a non-normalizable unit, the unit is attached to the ID of the property as a postfix. For example, the property P3362 (Operating Income) takes currencies such as Q4916 (Euro), Q4917 (United States Dollar), and Q25224 (Pound sterling), as units that could not be converted to one base unit and thus, they will be combined with the property ID as in P3362_Q4916, P3362_Q4917, and P3362_Q25224 respectively. For each property, units that occur less than 1% of the time are considered outliers and are removed.

[4] http://www.opengis.net/ont/geosparql#.

[5] http://www.w3.org/2001/XMLSchema#.

Note that the extracted triples with the aforementioned data type properties do not include those entities which satisfy at least one of the following conditions:

- The entities do not have site-links at least to the English Wikipedia. This step is required in order to support those link prediction models which leverage textual descriptions of entities.
- The entities have types only from the set of subclasses of the class `Q17379835` (Wikimedia page outside the main knowledge tree). This is imposed in order to keep only those entities which describe real-world concepts.

3.2 Extracting Relational Triples

As mentioned in Sect. 1, those triples with properties of Wikibase type *wikibase:Item* are referred to as relational triples in this paper. Once the entities with numerical literals are obtained as discussed above in Sect. 3.1, the next step is to extract relational triples for these entities. At this phase, we address both inverse properties and symmetric properties as follows:

- **Inverse properties:** Given two inverse properties p_1 and p_2 connected with the property `P1696` (inverse property) where the frequency of p_1 is greater than or equal to that of p_2, the subject and object entities of those triples with p_2 have been swapped and p_2 is replaced with p_1.
- **Symmetric Properties:** In these relational triples, every relation, except `P1889` (different from) whose head-tail pairs overlap with its tail-head pairs at least 50% of the time is considered as symmetric and hence, for each pair of redundant triples belonging to this relation, only one of them is kept. The property `P1889` (different from) has been removed due to the fact that it occurs in a significantly high number of triples but the semantic information captured in this property is not that much beneficial for KG embedding approaches to learn better KG representation.

3.3 Filtering the Triples

Taking as inputs the extracted attributive and relational triples, the goal in this phase is to create three datasets that vary in structure and size to be used for different purposes. The smallest dataset could be used for debugging and testing KGE models with and without literals whereas the medium size dataset would suit for evaluating KGE approaches on multiple tasks in general. On the other hand, the largest dataset could be used for few-shot evaluations in addition to general evaluations for KGEs. In this section, these datasets are referred to as small, medium, and large. The following three steps are applied to create these datasets:

Seeding Entities. The top N entities with the highest number of datatype properties are considered as seed entities. The value of N is $200,000$ for the small and large datasets and $50,000$ for the medium datasets. Different values have been tried out for N and those particular values are chosen because they suit well to generate appropriate-sized datasets.

Extending the Seed Entities. At this phase, fractions of the relational triples are taken by extending the seed entities with their **one-hop** entities for the small and large datasets and with their **two-hop** neighbors for the medium dataset.

Creating k-cores. The size of the triples extracted using the steps discussed so far is huge as it is from the entire Wikidata dump. Hence, the relational triples have been further filtered into $k - cores$, i.e., maximal-subgraphs G' of a given graph G where each node in the sub-graphs has at least a degree of k [3]. The value of k is 15 for the small and medium datasets and 6 for the large datasets. Note that the values for k are determined by taking into consideration both the size and structure of the datasets to be generated. The value of k is less for the largest dataset as compared to the others because this dataset is intended to be used for few-shot evaluations. In case of few-shot evaluations, it would be possible to see the advantages of literals in learning representations for entities occurring in few structured triples. Once the k-cores are created, some triples have been removed from each of the k-cores due to the following factors:

- Either the head or the tail entity doesn't have a summary section on the corresponding English Wikipedia page or the section contains less than 3 non-stop words.
- All entities having exactly the same Wikipedia pages for various reasons have been excluded in order to avoid having meaningless descriptions.
- Relations (object properties) with more than 50% subject-object overlap have been considered as duplicates and only one of them is kept.
- Relations occurring less than 3 times have been removed to ensure that every relation has a chance to appear in the training, validation, and test sets.
- Attributes (data properties) skewed 100% of the time towards a single (head) entity have been excluded.

In the subsequent sections, the created small, medium, and large datasets are referred to as LitWD1K, LitWD19K, and LitWD48K respectively. The statistics and analysis of these datasets are presented in Table 2. Each of these datasets has been split into 90/5/5 train/valid/test sets. While splitting the datasets, we have ensured that the entities which occur in validation and test sets also occur in the respective training sets. Moreover, the test sets do not contain any relation which is 100% skewed towards a single head or tail entity. LitWD48K contains more than double the number of entities in LitWD19K. However, both datasets have almost the same number of structured triples. This is due to the way the datasets are created, i.e., LitWD19K is based on two-hop whereas LitWD48K is based on one-hop as discussed above. Table 2 also presents a summary of the analysis of the datasets in terms of graph connectivity, diameter, and density.

3.4 Textual Information

In addition to the relational and attributive (numerical) triples discussed in Sect. 3.2 and Sect. 3.1, textual information about the entities and relations has also been extracted. The textual information includes **Wikidata labels**,

Table 2. Dataset Statistics and Analysis

		LitWD1K	LitWD19K	LitWD48K
Statistics	#Entities	1,533	18,986	47,998
	#Relations	47	182	257
	#Attributes	81	151	291
	#StruTriples	29,017	288,933	336,745
	#AttrTriples	10,988	63,951	324,418
	#Train	26,115	260,039	303,117
	#Test	1,451	14,447	16,838
	#Valid	1,451	14,447	16,838
Analysis	Connectivity	Yes	Yes	No[a]
	Diameter	5	7	8[b]
	Density	0.01235	0.0008	0.00014

[a] LitWD48K contains 3 connected components and the largest component contains 47,994 entities.

[b] The diameter of the largest component of LitWD48K is 8.

aliases, and **descriptions of entities, relations, and attributes**. Moreover, for each entity, the **summary** sections of the corresponding English, German, Russian, and Chinese Wikipedia pages have been extracted. The statistics of the text literals for each dataset are given in Table 3.

Table 3. Short and long text literals extracted from Wikidta and Wikipedia for entities, relations and attributes. The values are presented in percentage.

	Wikipedia summary				Wikidata (entity/relation/attrb) (en)		
	en	de	ru	zh	Labels	Aliases	Descriptions
LitWD1K	100	78	72	66	100/100/100	38/83/81	95/98/100
LitWD19K	100	80	65	39	100/100/100	44/87/81	99/99/100
LitWD48K	100	88	75	29	100/100/100	47/87/79	99/99/100

3.5 Domain of the Datasets

Since the pipeline developed in this study to create LiterallyWikidata framework does not require pre-defining the domains or classes of entities or relations, the created datasets are generic and their domains could be identified only after they are created. Based on the types/classes of entities, People, Geography, Entertainment, Transportation, Sport, Travel, Business, and Research are among the domains covered in LiterallyWikidata. The classes/types of the entities are also released along with the datasets.

4 Comparison with Existing Datasets

Link prediction benchmark datasets are usually characterized based on the nature of the relations such as symmetricity, inversion, skeweness, and cartesian product (fixed-set). Link prediction with symmetric/inverse/cartesian product relations is easy and does not require a complex embedding model [1,29]. It could also be done with simple rule based approaches. Here, the comparison will be with two existing datasets, FB15K-237 as the most popular extension of FB15K and CoDEx-M as the most recent dataset extracted from Wikidata. In order to make a fair comparison, the LitWD19K dataset is chosen to be compared against these datasets as it is comparable to both in terms of size.

Skeweness. As reported in CoDEx [29], 15.98% and 1.26% of test triples in FB15K-237 and CoDEx-M contain relations which are skewed 50% or more toward a single head or tail entity. In our case, as it has already been mentioned above, while splitting the LiterallyWikidata datasets we have made sure to exclude any relation which is 100% skewed towards a single head or tail entity in each of the datasets. However, for a fair comparison with the numbers reported in CoDEx [29], we also consider skewed relations as relations which are skewed 50% or more (instead of 100%) towards a single head or tail entity and find 6.48% of the test sets of LitWD19K to contain such skewed relations. This number does not have much of an impact as its coverage of the test set is low and also as already mentioned, none of the relations are 100% skewed.

Symmetricity. 4.01% of the triples in CoDEx-M contain symmetric relations [29]. In case of FB15K-237, every validation and test triple containing entity pairs that are directly linked in the training set were removed, which leads to deleting any symmetric relations from its test/validation sets. LitWD19K does not contain any symmetric relation in the entire dataset not only test/valid sets.

Inversion. Similar to the existing datasets FB15K-237 and CoDEx-M, LitWD19K also do not contain any inverse relations (see Sect. 3.2 for more details).

Cartesian Product or Fixed-set Relations. As reported in [29], about 12.7% of test triples in FB15K-237 contain fixed-set relations, i.e., relations which connect entities to fixed sets of values. On the other hand, both CoDEx-M and our dataset (LitWD19K) do not contain any such kind of relation.

5 Benchmarking Experiments on Link Prediction

In this section, the benchmarking experiments conducted on the link prediction task are discussed. The chosen KGE approaches, the model selection strategy, and the obtained results are presented. Note that there are properties in the LiterallyWikidata datasets which take date values. In order to treat those date values as numeric literals, for the experiments, the values are converted to decimals. This allows leveraging the semantics present in all parts of the date values, i.e., the year, the month, the days, and so on.

5.1 KGE Models

In this study, the models DistMult-LiteralE, DistMult, and ComplEx have been chosen to conduct the benchmarking experiments. The model DistMult-LiteralE was selected because the main focus of this study lies in providing benchmark datasets for KGE with literals whereas the other models DistMult and ComplEx are included to show the comparisons with and without literals. For more details on KGEs with literals please refer to [15]. **DistMult** scores a given triple using a diagonal bilinear interaction function between the head and tail entity embeddings and the relation embeddings - $f(h, t, r) = h^T diag(r)t$. This model can only deal with symmetric relations due to the fact that $f(h, t, r) = f(r, t, h)$. **ComplEx** is an extension of DistMult, which uses complex-valued embeddings in order to better handle asymmetric relations. Its scoring function is defined as - $f(h, t, r) = Re(h^T diag(r)\bar{t})$ where $Re(.)$ is the real part and $\bar{t}$ is the conjugate of t. **DistMultLiteral** extends DistMult by modifying the scoring function f such that the entity embeddings of h and t are replaced with their respective literal enriched representations h^{lit} and t^{lit}.

5.2 Model Selection

As it has been demonstrated in [2], in addition to a model's architecture, the combination of the training approach and the loss function used also plays an important role to determine a model's performance. Hence, we used a pytorch-based configurable KGE framework **Pykeen**[6] to search from a large range of hyperparameters listed in Table 4. First, around 70 different combinations of datasets, models, training approaches, losses, regularizers and optimizers (for example, **LitWD1K + DistMult + LCWA + CEL + LP + Adam**) were defined as configurations. Then, for each of these configurations, **random search** has been used to perform the hyper-parameter optimizations over all other hyper-parameters in order to select the best models. The details on the training approaches, losses, and search strategies are given as follows:

Training Approaches and Loss Functions. The models have been trained based on the sLCWA (Stochastic Local Closed World Assumption) and LCWA (Local Closed World Assumption) approaches. The sLCWA training approach has been used with UNS (Uniform Negative Sampler) to generate negative samples. The loss functions Cross Entropy Loss (CEL) and Binary Cross Entropy Loss (BCEL) are used together with LCWA whereas BCEL and Margin Ranking Loss (MRL) are used with sLCWA. In order to learn more about these training approaches and losses refer to [2].

Search Strategies. For each configuration with LitWD1K, a maximum of 100 trials are generated within a bound of 24 h for DistMult and DistMultLiteral, and 36 h for ComplEx. During each trial, the model is trained for 1000 epochs. On the other hand, for LitWD19K and LitWD48K a maximum of 100 trials are

[6] https://pykeen.readthedocs.io/en/latest/.

Table 4. Hyper-parameter search space

Hyper-parameter	Range
Embedding dimension	{64,128,256}
Initialization	{Xavier}
Optimizers[a]	{Adam, Adadgrad}
Regulaizer	{None, L1, L2}
Weight for L1 and L2	[0.01, 1.0)
Learning Rate (log scale)	[0.001, 0.1)
Batch size	{128, 256, 512, 1024}
Input dropout[b]	{0,0.1,0.2,0.3,0.4,0.5}
Training Approach[c]	
sLCWA	
Loss	{BCEL, MRL}
Number of Negatives	{1, 2, ... , 100}
Margin for MRL	{0.5, 1.5, ... , 9.5}
LCWA	
Loss	{BCEL, CEL}
Label Smoothing (log scale)	[0.001, 1.0)

[a] We evaluated both Adam & Adagrad using DistMult & DistMultLiteral on LitWD1K and using DistMult on LitWD19K & LitWD48K(sLCWA). The result indicates that Adagrad performs better than Adam on the smallest dataset whereas Adam is better on the larger ones. Hence, for that reason and also due to the fact that Adam is known for addressing the problem of decreasing learning rate in Adagrad, for the two larger datasets, we sticked to Adam for the rest of the experiments in order to reduce computational cost.

[b] The input dropout range is applied to DistMultLiteral

[c] We have evaluated both sLCWA & LCWA using DistMult & DistMultLiteral on all the three datasets and learned that LCWA performs better at all times. Hence, we used only LCWA for the rest of the experiments.

generated within 48 h for DistMult and DistMultLiteral, and 60 h for ComplEx. Every trial is run for a maximum of 500 epochs where early stopping is performed by evaluating the model every 25 epochs with a patience of 50 epochs on the validation set using MRR. Finally, for each dataset and embedding model pair (e.g., LitWD1K+DistMult), the best configuration is chosen based on the evaluation result on the validation set. Then, evaluation is carried out using the test set by retraining the selected models on each dataset for 1000 epochs. In order to make sure that the results reported are consistent, the retraining is done three times for all models on LitWD1K and for DistMult on LitWD19K and since we find the results to be very close, we run the retraining only once for the rest of the experiments.

The experiments with LitWD1K and LitWD19K are run on TITAN X (Pascal) 12 GB whereas those on LitWD48K are run on NVIDIA Tesla V100S-PCIE-32GB. The optimal hyperparameter values for each of the models on all the datasets are provided along with the datasets on Github[7].

5.3 Results

The results of the experiments on link prediction are presented in Table 5. Three different comparisons can be made from the results, i.e., i) unimodal vs. multimodal, ii) between uni-modals, and ii) proposed datasets vs. existing datasets.

- **Unimodal vs. Multi-modal:** Here, we compare DistMult with DistMutLiteral because DistMutLiteral is a multimodal KGE that extends DistMult. As it is seen in the results, for all of the three datasets DistMultLiteral outperforms DistMult w.r.t. almost all metrics. This indicates that making use of literals (numeric literals) improves entity representations.
- **Unimodal vs. Unimodal:** When comparing the unimodals, ComplEx, and DistMult, we see that ComplEx performs better than DistMult on the largest dataset LitWD48K. On the other two datasets, the results of the two models are comparable.
- **Proposed datasets vs. Existing datasets:** In order to show the level of difficulty of the proposed datasets, here we compare the results of the two unimodals on LitWD19K and the existing datasets FB15K-237 and CoDEx-M. For both ComplEx and DistMult, w.r.t. all metrics, the results on LitWD19K are worse than those on FB15K-237 and CoDEx-M.

Table 5. Results of link prediction

Dataset		Model	MRR	Hits@1	Hits@10
Ours	LitWD1K	DistMult	0.419	0.283	0.697
		ComplEx	0.413	0.28	0.673
		DistMultLiteral	**0.431**	**0.297**	**0.703**
	LitWD19K	DistMult	0.195	0.138	0.308
		ComplEx	0.181	0.122	0.296
		DistMultLiteral	**0.245**	**0.168**	**0.399**
	LitWD48K	DistMult	0.261	0.195	0.4
		ComplEx	0.277	**0.207**	0.428
		DistMultLiteral	**0.279**	0.204	**0.434**
Existing*	FB15K-237	DistMult	0.343	0.250	0.531
		ComplEx	0.348	0.253	0.536
	CoDEx-M	ComplEx	0.337	0.262	0.476

* The results are copied from LibKGE (https://github.com/uma-pi1/kge.)

[7] https://github.com/GenetAsefa/LiterallyWikidata.

6 Conclusion and Future Work

This study presents LiterallyWikidata which is a set of KGC datasets extracted from Wikidata and Wikipedia with a special focus on literals. The existing datasets FB15K-237 (popular) and CoDEx (recent) are both valuable datasets for link prediction with unimodal KGC models. However,we have shown that LiterallyWikidata is appropriate for both unimodal and multimodal link prediction tasks. Besides, directions for future work on LiterallyWikidata are indicated as follows:

- **More tasks**: Using the datasets with other tasks such as triple classification.
- **More Experiments**: Conducting experiments with text literals and also by fusing relational triples, numeric literals, short text literals (aliases and labels), and long text literals all together. Moreover, experiments with more varieties of KGE models will be performed.
- **Detailed analysis**: Conducting further analysis on the datasets in terms of compositionality will be undertaken, so as to explore its use for models which leverage paths.
- **Studying data bias**: Bias in training data is one of the crucial aspects of Machine Learning that needs to be carefully addressed. Since Wikidata is one of the crowd-sourced KGs, it is susceptible to biases. These biases in Wikidata reflect the real-world and hence, LiterallyWikidata may as well be biased. However, the current version of the dataset is not yet de-biased. We are currently investigating whether de-biasing should be done and what methods exist for such purpose.

We hope that the release of LiterallyWikidata fosters research on more sophisticated KGE models that exploit the additional semantics provided with literals.

References

1. Akrami, F., Saeef, M.S., Zhang, Q., Hu, W., Li, C.: Realistic re-evaluation of knowledge graph completion methods: An experimental study. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (2020)
2. Ali, M., et al.: Bringing light into the dark: a large-scale evaluation of knowledge graph embedding models under a unified framework. arXiv preprint arXiv:2006.13365 (2020)
3. Batagelj, V., Zaveršnik, M.: Fast algorithms for determining (generalized) core groups in social networks. Adv. Data Anal. Classif. **5**(2), 129–145 (2011)
4. van Berkel, L., de Boer, V.: kgbench: A collection of knowledge graph datasets for evaluating relational and multimodal machine learning. In: ESWC (2021)
5. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the ACM SIGMOD international conference on Management of data (2008)

6. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS (2013)
7. Bouchard, G., Singh, S., Trouillon, T.: On approximate reasoning capabilities of low-rank vector spaces. In: AAAI Spring Symposia (2015)
8. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence. AAAI Press (2010)
9. Daza, D., Cochez, M., Groth, P.: Inductive entity representations from text via link prediction. In: Proceedings of the Web Conference 2021, pp. 798–808 (2021)
10. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
11. García-Durán, A., Bordes, A., Usunier, N.: Effective blending of two and three-way interactions for modeling multi-relational data. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) ECML PKDD 2014. LNCS (LNAI), vol. 8724, pp. 434–449. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44848-9_28
12. García-Durán, A., Bordes, A., Usunier, N.: Composing relationships with translations. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 286–290. Association for Computational Linguistics (2015)
13. García-Durán, A., Niepert, M.: KBLRN: End-to-end learning of knowledge base representations with latent, relational, and numerical features. In: Globerson, A., Silva, R. (eds.) Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, pp. 372–381. AUAI Press (2018)
14. Gesese, G.A., Alam, M., Sack, H.: LiterallyWikidata - A Benchmark for Knowledge Graph Completion using Literals April 2021. https://doi.org/10.5281/zenodo.4701190
15. Gesese, G.A., Biswas, R., Alam, M., Sack, H.: A survey on knowledge graph embeddings with literals: Which model links better literal-ly?. arXiv preprint arXiv:1910.12507 (2019)
16. Guo, S., Wang, Q., Wang, L., Wang, B., Guo, L.: Knowledge graph embedding with iterative guidance from soft rules. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (2018)
17. Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. ACM Trans. Interact. Intell. Syst. **5**(4), 1–19 (2015)
18. Hinton, G.E., et al.: Learning distributed representations of concepts. In: Proceedings of the Eighth Annual Conference of the Cognitive Science Society, vol. 1, p. 12. Amherst (1986)
19. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Proceedings of the 21st National Conference on Artificial Intelligence, vol. 1, pp. 381–388. AAAI 2006, AAAI Press (2006)
20. Kok, S., Domingos, P.: Statistical predicate invention. In: Proceedings of the 24th International Conference on Machine Learning. Association for Computing Machinery (2007)
21. Kristiadi, A., Khan, M.A., Lukovnikov, D., Lehmann, J., Fischer, A.: Incorporating literals into knowledge graph embeddings. In: Ghidini, C., et al. (eds.) ISWC 2019. LNCS, vol. 11778, pp. 347–363. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30793-6_20

22. Lin, Y., Liu, Z., Sun, M.: Knowledge representation learning with entities, attributes and relations. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence IJCAI 2016, pp. 2866–2872. AAAI Press (2016)
23. Mahdisoltani, F., Biega, J., Suchanek, F.M.: Yago3: A knowledge base from multilingual wikipedias. In: CIDR (2015)
24. McCray, A.: An upper-level ontology for the biomedical domain. Comp. Funct. Genomics **4**, 80–84 (2003)
25. Miller, G.A.: Wordnet: a lexical database for english. Commun. ACM **38**, 39–41 (1995)
26. Mitchell, T., et al.: Never-ending learning. Commun. ACM **61**(5), 103–115 (2018)
27. Pezeshkpour, P., Chen, L., Singh, S.: Embedding multimodal relational data for knowledge base completion. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3208–3218. Association for Computational Linguistics October-November 2018
28. Rummel, R.J.: Dimensionality of nations project: Attributes of nations and behavior of nation dyads, pp. 1950–1965, 16 February 1992
29. Safavi, T., Koutra, D.: CoDEx: A comprehensive knowledge graph completion benchmark. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), November 2020
30. Safavi, T., Koutra, D., Meij, E.: Improving the utility of knowledge graph embeddings with calibration. arXiv preprint arXiv:2004.01168 (2020)
31. Shah, H., Villmow, J., Ulges, A., Schwanecke, U., Shafait, F.: An open-world extension to knowledge graph completion models. In: AAAI (2019)
32. Socher, R., Chen, D., Manning, C.D., Ng, A.Y.: Reasoning with neural tensor networks for knowledge base completion. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, vol. 1 (2013)
33. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A core of semantic knowledge. In: 16th International Conference on the World Wide Web, pp. 697–706 (2007)
34. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. In: International Conference on Learning Representations (2019)
35. Tay, Y., Tuan, L.A., Phan, M.C., Hui, S.C.: Multi-task neural network for non-discrete attribute prediction in knowledge graphs. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1029–1038. Association for Computing Machinery (2017)
36. Toutanova, K., Chen, D.: Observed versus latent features for knowledge base and text inference. In: Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality (2015)
37. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
38. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: a survey of approaches and applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017)
39. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence AAAI 2014, pp. 1112–1119. AAAI Press (2014)
40. Wu, Y., Wang, Z.: Knowledge graph embedding with numeric attributes of entities. In: Proceedings of The Third Workshop on Representation Learning for NLP, pp. 132–136. Association for Computational Linguistics (2018)

41. Xie, R., Liu, Z., Jia, J., Luan, H., Sun, M.: Representation learning of knowledge graphs with entity descriptions. In: AAAI (2016)
42. Xiong, W., Hoang, T., Wang, W.Y.: DeepPath: A reinforcement learning method for knowledge graph reasoning. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (2017)
43. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: International Conference on Learning Representations (ICLR) (2015)