Creating Variants of Freebase for Robust Development of Intelligent Tasks on Knowledge Graphs

Farahnaz Akrami*, **Mohammed Samiul Saeef***, **Nasim Shirvani-Mahdavi***, **Xiao Shi, Chengkai Li**
Department of Computer Science and Engineering, University of Texas at Arlington
{farahnaz.akrami, mohammedsamiul.saeef,
nasim.shirvanimahdavi2, xiao.shi}@mavs.uta.edu
cli@uta.edu

Abstract

Knowledge graphs (KGs) are an essential asset to a wide variety of tasks and applications in the fields of artificial intelligence and machine learning. They encode rich semantic, factual information, and different datasets could be potentially linked together for purposes greater than what they support separately. Despite the growing importance of KGs, many KGs are kept proprietary and many of the publicly available datasets are relatively small. Freebase is amongst the largest public cross-domain KGs that store common facts. It possesses several data modeling idiosyncrasies rarely found in comparable datasets such as Wikidata, YAGO, and so on. It has a strong type system; its properties are purposefully represented in reverse pairs; and it uses mediator objects to facilitate representation of multiary relationships. These design choices serve important practical purposes in realistically modeling the real-world. But they also pose nontrivial challenges that could hinder the advancement of KG-oriented technologies. More specifically, when algorithms and models for intelligent tasks are developed and evaluated agnostically of these data modeling idiosyncrasies, one could either miss the opportunity to leverage such features or fall into pitfalls without knowing. This paper lays out a comprehensive analysis of the challenges associated with the aforementioned idiosyncrasies of Freebase, measures their impact on tasks such as link prediction, and provides several variants of the Freebase dataset by inclusion/exclusion of various data modeling idiosyncrasies. Furthermore, the datasets underwent thorough cleaning in order to improve their utility. The datasets and data preprocessing scripts are made publicly available. They can be a valuable resource to researchers and practitioners in developing technologies by and for knowledge graphs.

1 Introduction

The ability to exploit big data on the Web enables intelligent systems [44]. Such data include encyclopedic knowledge of real-world factual information. Knowledge graphs (KGs) encode such semantic, factual information as triples of the form (subject, predicate, object). They can potentially link together heterogeneous data sources across different domains for purposes greater than what they support separately. This makes KGs an essential asset to a wide variety of tasks and applications in the fields of artificial intelligence and machine learning [13, 29], including natural language

*Equal Contribution

Submitted to the 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks. Do not distribute.

processing [53], information retrieval and web search [52], knowledge-base question answering [25], and recommender systems [55]. Consequently, KGs are of great importance to many major technology companies [37, 20] and governments [3].

To develop and robustly evaluate models and algorithms for intelligent tasks on knowledge graphs, access to large-scale KGs is crucial. But publicly available KG datasets are often much smaller than what real-world scenarios render and require [26]. For example, FB15K and FB15k-237 [11, 45], two staple datasets for knowledge graph completion, only have less than 15,000 entities in each. As of now, only a few cross-domain common fact knowledge graphs are both large and publicly available, e.g., DBpedia [8], Freebase [9], Wikidata [47], YAGO [41], and NELL [12].

With more than 80 million nodes, Freebase is amongst the largest public KGs. It comprises factual information in a broad range of domains, making it relevant to many applications. The dataset possesses several data modeling idiosyncrasies rarely found in the aforementioned comparable datasets. These design choices serve important practical purposes in realistically modeling the real-world. *Firstly*, Freebase properties are purposefully represented in reverse pairs, making it convenient to traverse and query the graph in both directions [38]. *Secondly*, Freebase uses mediator objects to facilitate representation of n-ary relationships [38]. *Lastly*, Freebase's strong type system categorizes each entity into one or more types, and the type of an entity determines the properties it may possess [10]. Furthermore, in practice the label of a property *almost* functionally determines the types of the entities in its two ends. As a simple example of the type system's merits, when querying the graph, a filtering condition on entities can be specified using an entity type.

Albeit highly useful, the aforementioned idiosyncrasies also pose nontrivial challenges that could hinder the advancement of KG-oriented technologies. More specifically, when algorithms and models for intelligent tasks are developed and evaluated agnostically of these data modeling idiosyncrasies, one could either miss the opportunity to leverage such features or fall into pitfalls without knowing. One example is that many knowledge graph link prediction models [49, 40] proposed in the past decade were evaluated using a subset of Freebase full of reverse triple pairs. The reverse triples lead to data leakage in evaluating the models. The consequence is substantial over-estimation of the models' accuracy and unrealistic comparison of their relative strengths [6].

This paper lays out a comprehensive analysis of the challenges associated with the aforementioned idiosyncrasies of Freebase. It measures their impact on tasks such as link prediction, and provides several variants of the Freebase dataset by inclusion/exclusion of mediator objects and reverse triples. In these variants, the Freebase type system is included. Furthermore, the datasets underwent thorough cleaning in order to improve their utility and to remove irrelevant triples from domains such as */freebase/* and */dataworld/*, which are for describing internal operations of Freebase, e.g., bulk loading and transformations of data. The datasets and data preprocessing scripts are made available at `https://github.com/idirlab/freebases`. They can be a valuable resource to researchers and practitioners in developing technologies by and for knowledge graphs.

2 Freebase Basic Concepts

In this section, we provide a brief summary of some basic terminology and concepts related to Freebase. We aim to be consistent with [10, 31, 22, 38] in our description.

RDF: Freebase is stored in N-Triples RDF (Resource Description Format) [31]. RDF is a standard graph-based model for representing and interchanging highly connected data. An RDF graph is a collection of triples (`s`, *p*, `o`), each comprising a subject `s`, an object `o`, and a predicate *p*. An example triple is (`James Ivory`, */film/director/film*, `A Room with a View`).

Topic (entity, node): Freebase objects can be divided into topics and non-topics. Topics are distinct entities, e.g., `James Ivory`, `A Room with a View`, and `BAFTA Award for Best Film` in Figure 1. In viewing Freebase as a graph, these topics correspond to nodes in the graph. However, not every node in the Freebase graph is a topic. CVT (Compound Value Type) nodes are examples of non-topic objects used to represent n-ary relations (details in Section 4.1). Each topic and non-topic node has a

unique *machine identifier* (MID), which consists of a prefix (either `/m/` for Freebase Identifiers or `/g/` for Google Knowledge Graph Identifiers) followed by a base-32 identifier. For example, the MID of `James Ivory` is `/m/04ld94`.

Type: Types are used to group topics semantically. A topic may have multiple types, e.g., `James Ivory`'s types include */people/person* and */film/director*. Types are further grouped into *domains*. For instance, domain *film* includes types such as */film/actor*, */film/director*, and */film/editor*.

Property (predicate, relation, edge): Properties are used in Freebase to provide facts about topics. A property of a topic defines a relationship between the topic and its property value. The property value could be a literal or another topic. For example, topic `James Ivory` has the property */people/person/date_of_birth* with value 1928-06-07. It also has another property */film/director/film*, on which the value is another topic `A Room with a View`, as shown in Figure 1. When a relationship is represented as a triple, the predicate in the triple is a property of the subject. In viewing Freebase as a graph, a property is a directed edge from the subject node to the object node. The type of an edge or *edge type* can be uniquely identified by the label of the edge. The occurrences of an edge type in the graph are *edge instances*.

Schema: The term schema refers to the way Freebase is structured. It is expressed through types and properties. The schema of a type is the collection of its properties. Given a topic belonging to a type, the properties in the schema of that type are applicable to the topic. For example, the schema of type */people/person* includes property */people/person/date_of_birth*. Hence, each topic of this type (e.g., `James Ivory`) may have the property.

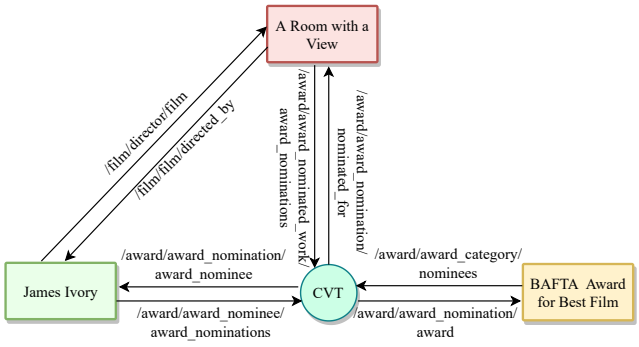

Figure 1: An example of a mediator node in Freebase

3 Useful Idiosyncrasies of Freebase

Freebase is amongst the largest cross-domain common fact KGs that is publicly available. The Freebase raw data dump contains more than 80 million nodes, more than 14,000 distinct relations, and 1.9 billion triples. Datasets that were imported into it (e.g., MusicBrainz [43]) have a huge impact on the number of triples in this knowledge graph. Freebase has a total of 105 domains, 89 of which are diverse *subject matters domains*—domains describing real-world facts—ranging from American music to visual art [13]. This makes Freebase applicable on a broad range of applications and tasks. As stated in [19], Freebase's data is consistent, correct, reliable, semantically valid, and certified free of error to a very admissible degree. Before Google shut down Freebase in 2015, the company announced its plan to help with the transfer of Freebase content to Wikidata [17]. This transfer is yet to be completed [38, 1]. Nonetheless, Freebase has several idiosyncrasies of data modeling design choices, which are rarely found in Wikidata and other comparable datasets. The rest of this section focuses on these idiosyncrasies.

Mediator Nodes *Mediator nodes*, also called CVT nodes, are used in Freebase to represent n-ary relationships [38]. For example, Figure 1 shows a CVT node connected to an `award`, a `nominee`, and a `work`. This or similar approach is necessary for accurate modeling of the real-world. To reduce complexity of algorithmic solutions, one may convert an n-ary relationship centered at a CVT node into $\binom{n}{2}$ binary relationships between every pair of entities connected to the CVT node. Note that such a transformation leads to loss of information [50]. For instance, after such a transformation for Figure 1, the new triples cannot exactly pinpoint to the work that leads to `James Ivory`'s nomination for the BAFTA award.

Reverse Triples When a new fact was included into Freebase, it would be added as a pair of reverse triples (s, p, o) and (s, p^{-1}, o) where p^{-1} is the reverse of p. Freebase actually denotes reverse relations explicitly using a special relation */type/property/reverse_property* [38, 18]. For instance, the triple (`/film/film/directed_by`, */type/property/reverse_property*, `/film/director/film`) denotes that */film/film/directed_by* and */film/director/film* are reverse relations. Therefore, (`A Room With A View`, */film/film/directed_by*, `James Ivory`) and (`James Ivory`, *film/director/film*, `A Room With A View`) form a pair of reverse triples, as shown in Figure 1. Reverse relations help traverse the graph in both directions [38].

Freebase Type System Freebase categorizes each topic into one or more types and each type into one domain. Furthermore, the triple instances satisfy *pseudo* constraints as if they are governed by a rigorous type system. Specifically, 1) given a node, its types set up constraints on the labels of its properties; the type of an edge in most cases belongs to one of the types of the subject node. To be more precise, this is a constraint satisfied by 98.98% of the nodes—we found 610,007 out of 59,896,902 nodes in Freebase (after cleaning the data dump; more to be explained later) having at least one property belonging to a type that is not among its node's types. 2) Given an edge type and its edge instances, there is *almost* a function that maps from the edge type to a type that all subjects in the edge instances belong to, and similarly *almost* such a function for objects. For instance, all subjects of edge *comedy/comedian/genres* belong to type */comedy/comedian* and all their objects belong to */comedy/comedy_genre*. Particularly, regarding objects, the Freebase designers explained that every property has an "expected type" [10]. For each edge type, we identified the most common entity type among all subjects and all objects in its instances, respectively. Out of 2,891 edge types (103,324,039 triples, again, after cleaning the data dump), for 2,011, 2,510, 2,685, and 2,723 edge types, the most common entity type among subjects covers 100%, 99%, 95%, and 90% of the edge instances, respectively. With regard to objects, the numbers are 2,164, 2,559, 2,763, and 2,821, for 100%, 99%, 95%, and 90%, respectively.

Given the *almost* true constraints reflected by the aforementioned statistics, we are interested in creating an explicit type system, which can become useful when enforced in various intelligent tasks. Note that Freebase itself does not explicitly provide such a type system, even though the data appear to be included while following guidelines that approximately form the type system, e.g., the "expected type" mentioned earlier. The goal is to, given an edge type, designate an *required type* for its subjects (and objects, respectively) from a pool of candidates formed by all types that the subjects (objects, respectively) belong to. As an example, consider edge type *film/film/performance* and the entities o at the object end of its instances. These entities belong to types {*film/actor*, *music/artist*, *award/award_winner*, *people/person*, *tv/tv_actor*}, which thus form the candidate pool. We select the required type for its object end in two steps, and the same procedure is applied for the subject/object ends of all edge types. In *step 1*, we exclude a candidate type t if $P(o \in t) < \alpha$, i.e., the probability of the object end of *film/film/performance* belonging to t is less than a threshold α. The rationale is to keep only those candidates with sufficient coverage. In the dataset, $P(o \in$ *film/actor*$) = 0.9969$, $P(o \in$ *music/artist*$) = 0.0477$, $P(o \in$ *award/award_winner*$) = 0.0373$, $P(o \in$ *people/person*$) = 0.998$, and $P(o \in$ *tv/tv_actor*$) = 0.1052$. Using $\alpha = 0.95$, *music/artist*, *award/award_winner* and *tv/tv_actor* were excluded. In *step 2*, we pick the most specific type among the remaining candidates. We use $P(n \in t_1 | n \in t_2)$ for the probability of a Freebase entity n belonging to type t_1 given that it also belongs to type t_2. In the dataset $P(n \in$ *people/person*$|n \in$ *film/actor*$) = 0.9984$ and $P(n \in$ *film/actor*$|n \in$ *people/person*$) = 0.1394$. Thus, we assigned *film/actor* as the required entity type for the object node of edge type *film/film/performance*.

The type system we created can be useful in at least several applications and tasks, as follows. 1) In an interactive graph query system that aids users to formulate queries over Freebase by drawing nodes and edges on a canvas (e.g., [27, 28]), the type system is essential in helping find entity types by filtering through domains and likewise for finding entities through domains and types. The type system that uses edge type to constraint entity types at two ends is also necessary for clearly labeling what entities are to be expected at each node in the query graph. 2) Graph-to-text models [21, 30, 48, 4, 35, 39] can also benefit from a type system. These models narrate triples using natural language sentences.

As evidenced in [16], an entity type mapper can be highly effective in improving a model's quality. Such a mapper replaces entities with their types, e.g., a triple (`James Ivory`, */film/director/film*, `A Room with a View`) will become (*/film/director*, */film/director/film*, */film/film*). The type system we created will facilitate such mapping automatically. 3) A few studies employed type information to improve knowledge graph link prediction [51, 24]. Particularly, embedding models learn vector representations of entities. Exploiting entity types can help these models ensure entities of the same type stay close to each other in the embedding space [24]. Further, type information could be a simple, effective feature of a model or used as a constraint while generating negative training or test examples. For instance, given the task of predicting the objects in (`James Ivory`, */film/director/film*, `?`), knowing the object end type of */film/director/film* is */film/film* can already successfully exclude many candidates; on the other hand, a negative example (`James Ivory`, */film/director/film*, `BAFTA Award for Best Film`) has less value in gauging a model's accuracy since it is a trivial case as `BAFTA Award for Best Film` is not of type */film/film*.

4 Challenges of Directly Using Freebase Data Dump

The latest Freebase data dump constitutes a snapshot of the data and its schema. As discussed earlier, the data modeling idiosyncrasies of Freebase could pose challenges that hamper the advancement of KG-oriented technologies. Specifically, when algorithms and models for intelligent tasks are developed and evaluated agnostically of these characteristics, they may fall into pitfalls without knowing. This section explains such challenges and their impact on tasks such as link prediction and knowledge graph querying.

4.1 Mediator Nodes

Link prediction [11] is almost always conducted on binary relations. While the effectiveness of current link prediction approaches on data with CVT nodes is unknown, blindly applying such approaches when CVT nodes are present has foreseeable problems. For example, random splitting of triples into training/test/validation sets might not be enough. As each CVT node is connected to only a few entities, random splitting could lead to many CVT nodes appearing in the test set without being existent in the training set. Nevertheless, link prediction on CVT nodes could be challenging as they are long-tail nodes with limited structural information.There are a few studies that considered link prediction on n-ary relational facts [50, 56, 23]. These studies represent the mutiary facts in Freebase as (r, e_1, e_2, ..., e_n), e.g., (*/award/award_nominee/award_nominations*, `BAFTA Award For Best Film`, `James Ivory`, `A Room With A View`). They then learn the relatedness between entities or between entity-relation pairs.

It is also difficult to accommodate n-ary relationships in an interactive graph query system [27, 28]. CVT nodes, unlike non-CVT nodes, lack meaningful labels. This makes node creation difficult for users in formulating n-ary relations. Enabling automatic suggestions also becomes more challenging when mediator nodes are included. When a system suggests a mediator node to be included into a query, its lack of meaningful label makes the query graph difficult to comprehend.

There is no available evidence that existing graph-to-text models can successfully handle mediator nodes. Current models were all built on datasets generated from DBpedia or Wikidata [21, 32, 4, 14], none of which contains mediator nodes. It is useful to create Freebase datasets both with and without CVT nodes. On the one hand, a Freebase dataset without CVT nodes will be compatible with current models and also facilitate model comparison across different datasets. On the other hand, a dataset with CVT nodes will create the opportunity to study how such nodes impact graph-to-text models.

Despite the challenges that may arise with mediator nodes, there is not any large-scale dataset created from the latest Freebase data dump with CVT nodes removed. In Freebase86m [57] (a large-scale subset of Freebase), 23% of the nodes are CVT nodes. The FB1 dataset presented in this paper is the first large-scale dataset addressing this gap. Details of the process of separating and removing CVT nodes are discussed in Section 6.

4.2 Reverse Triples

Several previous studies discussed the problems of including the reverse relations in datasets used for link prediction [15, 45, 5, 6]. Link prediction is the task of predicting the missing `s` in triple (`?`, p, `o`) or missing `o` in (`s`, p, `?`). The popular benchmark dataset FB15k (a subset of Freebase), created by Bordes et al. [11], is almost always used for knowledge graph link prediction. Toutanova and Chen [45] noted that FB15k contains many reverse triples. They constructed another dataset, FB15k-237, by only keeping one relation out of any pair of reverse relations. The idiosyncrasies of the link prediction task on datasets such as FB15K with many reverse triples can be summarized as 1) Link prediction becomes much easier on a triple if its reverse triple is available. Hence, the reverse triples led to a substantial over-estimation of the link prediction's accuracy, which is verified by experiments in [6], 2) Instead of complex models, one may achieve similar results by using statistics of the triples to derive simple rules of the form (`s`, p_1, `o`) $\Rightarrow$ (`o`, p_2, `s`) [15], and 3) The link prediction scenario, given such data, is non-existent in the real-world at all. With regard to FB15k, the redundant reverse relations, coming from Freebase, were just artificially created. When a new fact was added into Freebase, it would be added as a pair of reverse triples, denoted explicitly by the relation */type/property/reverse_property* [38, 18]. For such intrinsically reverse relations that always come in pair when the triples are curated into the datasets, there is not a scenario in which one needs to predict a triple while its reverse is already in the knowledge graph. Training a knowledge graph completion model using FB15k is thus a form of *overfitting* in that the learned model is optimized for the reverse triples which cannot be generalized to realistic settings. More precisely, this is a case of excessive *data leakage*—the model is trained using features that otherwise would not be available when the model needs to be applied for real prediction.

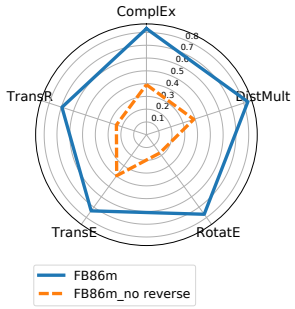

Figure 2: Performance decrease of link prediction after removal of reverse triples (the higher the better)

In a an interactive graph query system with automatic suggestion [27, 28]), including reverse relations would complicate the design. The objective of the automatic suggestion feature is to display a ranked list of edges to users while they are formulating a query. If reverse relations are present in the graph, the system has to treat them specially when assigning a ranking score to an edge and its reverse, and it should refrain from displaying an edge as suggestion if its reverse edge has already been suggested. Thus, removal of reverse edges will help simplify the system design.

Redundancy of reverse triples poses some challenges for the task of graph-to-text as well. In this task, for the automatic alignment between a text corpus and a knowledge graph, we need to determine which relation to preserved when there are multiple relations between two entities. Because between any two entities in the text, only one semantic relation is expressed. Existence of reverse relations may make this process more complicated as it increases the number of candidate relations. Furthermore, if one triple is in the training set and its reverse is in the test set, the same semantic meaning of these relations may cause bias in evaluating the performance of graph-to-text models. Therefore, removing the reverse triples makes the task less complicated.

The dataset FB1 presented in this paper is the first large-scale Freebase dataset without reverse triples. More than 38% of the triples in Freebase86m form reverse pairs. As mentioned above, including these triples will lead to overestimation of link prediction results. Figure 2 compares the link prediction results before and after removal of reverse triples from Freebase86m. We can observe that the performance has decreased remarkably. Similar results were observed on FB15k vs. FB15k-237 [6]. The process of removal of reverse relations is discussed in detail in Section 6.

4.3 Metadata and Administrative Data

As stated in [13], Freebase domains can be divided into 3 groups: implementation domains, Web Ontology Language (OWL) domains, and subject matter domains. Freebase implementation domains

such as */dataworld/* and */freebase/* include triples that convey schema and technical information used in creation of Freebase. For instance, the two aforementioned domains */dataworld/* and */freebase/* are described as "a domain for schema that deals with operational or infrastructural information" and "a domain to administer the Freebase application" respectively. As an example type in */freebase/* domain, is */freebase/mass_data_operation*. It is defined as a type to track the large scale data tasks carried out by the data team. OWL Domains contain properties such as *rdfs:domain* and *rdfs:range*. These properties provide information on how to use the predicate p; domain shows to which class the subject of any triple that uses p as its predicate belongs, and range shows the type of the object of any such triple [7]. For example, the domain and range of the predicate *film/director/film* are *director* and *film* respectively. Subject matter domains on the other hand contain triples about real-world facts in different areas such as Music, People, and Food.

Among all these three groups of domains, subject matter domains are the only group that contains triples describing real-world facts. Real computational tasks need to be applied on this group rather than the whole data which contains the other two groups including information about the data itself. However, in Freebase86M, just OWL domains are removed. As shown in Table 1, around 31% of triples in this dataset fall under non-subject matter domain. We believe including these triples could be irrelevant to many tasks and will make the dataset unrealistically large. We have created 4 datasets in which all the triples belonging to subject matter domains are retained and the rest of the triples are removed. We also provide the information related to type system for all datasets. The details of this process is discussed in Section 6.

Table 1: Statistics of implementation domains in Freebase86m

Domains	#Triples	%Total
/common/	48,610,556	14.4
/type/	26,541,747	7.8
/base/	14,253,028	4.2
/freebase/	7,705,605	2.3
/dataworld/	6,956,819	2.1
/user/	322,215	0.1
/pipeline/	455,377	0.1
/kp_lw/	1,034	0.0003

5 Overview of Existing Freebase Datasets

In the past few years, several datasets were created from Freebase. In this section, we review some of these datasets and briefly discuss why none of them meet the criteria required for conducting intelligent tasks on them.

FB15k: FB15k is a subset of Freebase generated by Bordes et al. [11] for the task of *link prediction*. FB15k retains entities with at least 100 appearances in Freebase that were also available in Wikipedia based on the *wiki-links* database [2]. Each included relation has at least 100 instances. 14,951 entities and 1,345 relations satisfy these criteria, which account for 592,213 triples included into FB15k. These triples were randomly split into the training, validation and test sets. This dataset suffers from data redundancy in the forms of reverse triples, duplicate and reverse-duplicate relations. The details of these issues are discussed thoroughly in [6].

FB15k-237: FB15k-237 [45], with 14,541 entities, 237 relations, and 309,696 triples, was created from FB15k in order to mitigate the aforementioned data redundancy. Only the most frequent 401 relations from FB15k are kept. Near-duplicate and reverse-duplicate relations were detected, and only one relation from each pair of such redundant relations is kept. This process further decreased the number of relations to 237. This step could incorrectly remove useful information. For example, *place_of_birth* and *place_of_death* may have many overlapping subject-object pairs, but they are not semantically redundant. Furthermore, the creation of FB15k-237 did not resort to the accurate reverse relation information encoded by *reverse_property* in Freebase. They also removed all triples in the test and validation sets whose entity pairs were directly linked in the training set through any relation.

Freebase86m: This dataset is created from the latest Freebase data dump and is employed in evaluating multiple large-scale knowledge graph embedding frameworks [57, 36]. It includes 86,054,151 entities, 14,824 relations, and 338,586,276 triples. No information is available on how this dataset was created from Freebase. We carried out an extensive investigation on this dataset to assess its quality and found out that it includes: 1) some of the Freebase implementation domains such as */common/* and */type/*, 2) many mediator nodes, and 3) abundant data redundancy in the form of reverse

triples. As discussed in Sections 4 and 3, including such information could be beneficial for some models and algorithms but one should be wary of the pitfalls associated with them.

6 Data Cleaning

URI Simplification As mentioned in Section 2, Freebase is stored in N-Triples RDF format. Each component of a triple (`subject`, *predicate*, `object`) if not a literal can be identified by a URI (uniform resource identifier) [31]. For simplification, we removed the first segment of each URI, *"<http://rdf.freebase.com/ns/>"*, and only retained the segments that correspond to domains, types, properties' labels, and MIDs. These segments are dot-delimited in the URI. To make it more readable, we replaced the dots by a *"/"*. For example, the URI *<http://rdf.freebase.com/ns/film.director.film>* will be simplified to */film/director/film* in which *film* is the domain of this property, *director* is its type, and *film* is its property label. Likewise, *<http://rdf.freebase.com/ns/award.award_winner>* and *<http://rdf.freebase.com/ns/m.0zbqpbf>* which are the URIs of a Freebase type and an MID are simplified to */award/award_winner* and `/m/0zbqpbf`.

Table 2: Statistics of the four variants of Freebase

Variant	CVT-nodes	Reverse-pairs	#entities	#properties	#Triples
FB1	removed	removed	39,732,008	2,891	103,324,039
FB2	removed	retained	39,745,618	4,894	235,307,422
FB3	retained	removed	59,894,890	2,641	134,213,735
FB4	retained	retained	59,896,902	4,425	244,112,599

Addressing Metadata and Administrative Data Problem First, subject matter triples and non-subject matter triples are separated. We call (`s`, *p*, `o`) a subject matter triple if the domains of `s`, *p* and `o` belong to subject matter domains (discussed in Section 4.3). Otherwise we call it a non-subject matter triple. Second, we organize the non-subject matter triples that are used to organize the metadata. We created two different mappings, one for a Freebase object (not to be confused with RDF object) to its type(s), another for a Freebase object to its label. We used the Freebase predicates */type/object/types* and */type/object/name* respectively, to create these mappings. The mappings are used to quickly look up what types an object belongs to and identifying the label of that object. Moreover, for each Freebase concept $domain$, $type$, $property$ and $entity$, we created lookup tables mapping a Freebase object MID to its label. These lookup tables came in handy in different applications.

Eliminating Reverse Triples As discussed in 3, Freebase has a reverse property which determines whether two relations are reverse of each other. To remove reverse triples from all the subject matter triples, we picked *r2* from each pair of reverse relations *r1* and *r2*. We then discarded all the triples with property *r2* to remove redundant information.

Eliminating Mediator Nodes Our goal in this step is to identify all CVTs and remove them from the graph. Identifying CVT nodes is challenging as Freebase does not directly disclose them. There are 2,868 types which are specified as *mediator types*. According to our empirical analysis, CVT can be ascribed as a Freebase object that belongs to at least one mediator type but was given no label in the data set. For example, object `/m/011tzbfr` belongs to the mediator type */comedy/comedy_group_membership* and does not have any label. Thus, we identified this object as a mediator node.

Once we found all CVTs in the graph, we created concatenated edges collapsing the CVTs and merging the intermediate edges (edges with at least one CVT endpoint). For instance, the triples (`BAFTA Award For Best Film`, *award_category/nominees*, `CVT`) and (`CVT`, *award_nomination/nominated_for*, `A Room With A View`) in Figure 1 would be concatenated to form a triple between the `award` and the `work` nominated for the award.

Variants of the Freebase Dataset We provided four variants of the Freebase dataset by inclusion/exclusion of reverse triples and CVT nodes. These variants can be beneficial since one can leverage or avoid these features based on the nature of their task. In all variants, the type system is included and the metadata and administrative triples are detached from the subject matter triples. Table 2 presents the statistics of these variants.

The datasets and data preprocessing scripts are made publicly available at `https://github.com/idirlab/freebases`.

Table 3: Link prediction results on FB1vs FB2

	FB1				FB2			
Model	MRR$^{\uparrow}$	MR$^{\downarrow}$	H1$^{\uparrow}$	H10$^{\uparrow}$	MRR$^{\uparrow}$	MR$^{\downarrow}$	H1$^{\uparrow}$	H10$^{\uparrow}$
TransE	0.958	3.857	0.944	0.980	0.686	41.810	0.625	0.799
DistMult	0.965	6.059	0.956	0.979	0.709	69.388	0.670	0.780
ComplEx	0.970	5.567	0.964	0.981	0.717	68.798	0.681	0.783
TransR	0.952	4.655	0.939	0.974	0.636	52.251	0.584	0.733
RotatE	0.938	13.513	0.926	0.956	0.455	143.688	0.399	0.559

Table 4: Link prediction results on FB15kvs FB15k-237

	FB15K				FB15K-237			
Model	MRR$^{\uparrow}$	MR$^{\downarrow}$	H1$^{\uparrow}$	H10$^{\uparrow}$	MRR$^{\uparrow}$	MR$^{\downarrow}$	H1$^{\uparrow}$	H10$^{\uparrow}$
TransE	0.639	45.805	0.509	0.844	0.249	247.627	0.150	0.445
DistMult	0.677	59.240	0.562	0.867	0.247	386.749	0.151	0.444
ComplEx	0.747	68.876	0.664	0.881	0.240	404.196	0.150	0.426
TransR	0.666	67.371	0.581	0.802	0.576	196.994	0.527	0.671
RotatE	0.685	50.202	0.582	0.849	0.248	288.630	0.160	0.426

7 Experiments

We conducted several link prediction experiments on FB1 and FB2. As it was mentioned in the previous sections, blindly applying link prediction models when CVT nodes are present may lead to foreseeable problems. Hence, we decided to do our experiments on versions of the Freebase dataset without CVT nodes. These two datasets were randomly divided into training, validation and test sets with the split ratio of 90/5/5. We trained TransE [11], DistMult [54], ComplEx [46], TransR [34], and RotatE [42] embedding models on these datasets and evaluated their performance using common evaluation metrics MRR$^{\uparrow}$(Mean Reciprocal Rank), MR$^{\downarrow}$(Mean Rank), Hits@1$^{\uparrow}$, and Hits@10$^{\uparrow}$(replaced by H1$^{\uparrow}$and H10$^{\uparrow}$in the tables to save space) [11]. An upward/downward arrow beside a measure indicates that methods with greater/smaller values by that measure possess higher accuracy. Recently, some multi-processing multi-GPU distributed training frameworks have been proposed to scale embedding models [33, 58, 57]. To conduct our experiments, we used one of these frameworks called DGL-KE [57] with the same hyperparameters and settings that was suggested by them. The experiments were conducted on an Intel-based machine with an Intel Xeon E5-2695 processor running at 2.1GHz, Nvidia Geforce GTX1080Ti GPU, and 256 GB RAM.

The results of our experiments can be observed in Table 3. As discussed in 4.2, previous studies show substantial over-estimation of the link prediction's accuracy when reverse triples included in the dataset. Results on FB1 and FB2 with and without reverse relations present a similar observation in the large-scale. On the other hand, link prediction results on FB15k-237, the only dataset created from Freebase that excluded reverse relations, is poor as can be observed in Table 4. The low accuracy could also be a consequence of the small size of this dataset which does not provide enough information for training of effective models. We can observe that results on FB2 has improved in comparison to results on FB15k-237 and we can attribute this to the larger size of our dataset. Our results show that our datasets can provide the opportunity to evaluate embedding models more realistically.

8 Conclusion

We laid out a comprehensive analysis of the challenges associated with Freebase data modeling idiosyncrasies (CVT nodes, reverse properties, and type system). To tackle these challenges and thus to facilitate scalable and robust development of AI and machine learning technologies fully leveraging Freebase, we provide four variants of the Freebase dataset by inclusion/exclusion of these idiosyncrasies. Therefore, one can grasp the opportunity to leverage or avoid such features based on the nature of their tasks. Furthermore, the datasets underwent thorough cleaning in order to improve their utility and data quality. In all these variants, the Freebase type system is included. This is the first time that a group of datasets are created by carefully considering Freebase idiosyncrasies. We believe they can be a valuable resource to researchers and practitioners in developing technologies by and for knowledge graphs.

References

[1] Wikidata:Wikiproject Freebase. https://www.wikidata.org/wiki/Wikidata:WikiProject_Freebase#noSuchAnchor. Accessed: 2022-06-09.

[2] Wikipedia links data. https://code.google.com/archive/p/wiki-links/. Accessed: 2022-06-09.

[3] Open knowledge network: Summary of the big data IWG workshop. https://www.nitrd.gov/open-knowledge-network-summary-of-the-big-data-iwg-workshop/, 2018.

[4] Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688*, 2020.

[5] Farahnaz Akrami, Lingbing Guo, Wei Hu, and Chengkai Li. Re-evaluating embedding-based knowledge graph completion methods. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1779–1782, 2018. doi:10.1145/3269206.3269266.

[6] Farahnaz Akrami, Mohammed Samiul Saeef, Qingheng Zhang, Wei Hu, and Chengkai Li. Realistic re-evaluation of knowledge graph completion methods: An experimental study. In *Proceedings of the 2020 ACM Special Interest Group on Management of Data International Conference on Management of Data*, page 1995–2010, 2020. doi:10.1145/3318464.3380599.

[7] Dean Allemang and James Hendler. *Semantic web for the working ontologist: effective modeling in RDFS and OWL*. Elsevier, 2011.

[8] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBPedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.

[9] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM Special Interest Group on Management of Data international conference on Management of data*, pages 1247–1250, 2008.

[10] Kurt Bollacker, Patrick Tufts, Tomi Pierce, and Robert Cook. A platform for scalable, collaborative, structured information integration. In *Intl. Workshop on Information Integration on the Web*, pages 22–27, 2007.

[11] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 2787–2795, 2013.

[12] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI conference on artificial intelligence*, 2010.

[13] Niel Chah. Freebase-triples: A methodology for processing the freebase data dumps. *arXiv preprint arXiv:1712.08707*, 2017.

[14] Anthony Colas, Ali Sadeghian, Yue Wang, and Daisy Zhe Wang. Eventnarrative: A large-scale event-centric dataset for knowledge graph-to-text generation. *arXiv preprint arXiv:2111.00276*, 2021.

[15] Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 1811–1818, 2018.

[16] Bayu Distiawan, Jianzhong Qi, Rui Zhang, and Wei Wang. GTR-LSTM: A triple encoder for sentence generation from RDF data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1627–1637, 2018.

[17] Jason Douglas. Announcement: From freebase to wikidata. Accessed: 2015-02-17. URL: https://groups.google.com/g/freebase-discuss/c/s_BPoL92edc/m/Y585r7_2E1YJ.

[18] Michael Färber. *Semantic Search for Novel Information*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 2017.

[19] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. Linked data quality of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web*, 9(1):77–129, 2018.

[20] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building Watson: An overview of the DeepQA project. *Artificial Intelligence magazine*, 31(3):59–79, 2010.

[21] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, 2017.

[22] Jan Grant and Dave Beckett. RDF test cases. 2004. URL: https://www.w3.org/TR/rdf-testcases/.

[23] Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. Link prediction on n-ary relational data. In *The World Wide Web Conference*, pages 583–593, 2019.

[24] Shu Guo, Quan Wang, Bin Wang, Lihong Wang, and Li Guo. Semantically smooth knowledge graph embedding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 84–94. Association for Computational Linguistics, 2015. URL: https://aclanthology.org/P15-1009, doi:10.3115/v1/P15-1009.

[25] Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231, 2017.

[26] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[27] Nandish Jayaram, Rohit Bhoopalam, Chengkai Li, and Vassilis Athitsos. Orion: Enabling suggestions in a visual query builder for ultra-heterogeneous graphs. *arXiv preprint arXiv:1605.06856*, 2016.

[28] Nandish Jayaram, Sidharth Goyal, and Chengkai Li. Viiq: auto-suggestion enabled visual interface for interactive graph query formulation. *Proceedings of the VLDB Endowment*, 8(12):1940–1943, 2015.

[29] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[30] Mihir Kale and Abhinav Rastogi. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland, December 2020. Association for Computational Linguistics. URL: https://aclanthology.org/2020.inlg-1.14.

[31] Graham Klyne. Resource description framework (RDF): Concepts and abstract syntax. 2004. URL: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

[32] Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. Text generation from knowledge graphs with graph transformers. *arXiv preprint arXiv:1904.02342*, 2019.

[33] Adam Lerer, Ledell Wu, Jiajun Shen, Timothee Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. PyTorch-BigGraph: A Large-scale Graph Embedding System. In *Proceedings of the 2nd SysML Conference*, Palo Alto, CA, USA, 2019.

[34] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 2181–2187, 2015.

[35] Diego Marcheggiani and Laura Perez-Beltrachini. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9, Tilburg University, The Netherlands, November 2018. Association for Computational Linguistics. URL: `https://aclanthology.org/W18-6501`, `doi:10.18653/v1/W18-6501`.

[36] Jason Mohoney, Roger Waleffe, Henry Xu, Theodoros Rekatsinas, and Shivaram Venkataraman. Marius: Learning massive graph embeddings on a single machine. In *15th USENIX Symposium on Operating Systems Design and Implementation*, pages 533–549, 2021.

[37] Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale knowledge graphs: lessons and challenges. *Communications of the ACM*, 62(8):36–43, 2019.

[38] Thomas Pellissier Tanon, Denny Vrandečić, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. From Freebase to Wikidata: The great migration. In *Proceedings of the 25th International Conference on World Wide Web*, pages 1419–1428, 2016.

[39] Leonardo FR Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. Modeling global and local node contexts for text generation from knowledge graphs. *Transactions of the Association for Computational Linguistics*, 8:589–604, 2020.

[40] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data*, 15(2):1–49, 2021.

[41] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A large ontology from wikipedia and wordnet. *Journal of Web Semantics*, 6(3):203–217, 2008.

[42] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the International Conference on Learning Representations*, pages 926–934, 2019.

[43] Aaron Swartz. Musicbrainz: A semantic web service. *IEEE Intelligent Systems*, 17(1):76–77, 2002.

[44] Niket Tandon, Aparna S Varde, and Gerard de Melo. Commonsense knowledge in machine intelligence. *The ACM Special Interest Group on Management of Data Record*, 46(4):49–52, 2018.

[45] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015. `doi:10.18653/v1/W15-4007`.

[46] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2071–2080, 2016.

[47] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[48] Qingyun Wang, Semih Yavuz, Xi Victoria Lin, Heng Ji, and Nazneen Rajani. Stage-wise fine-tuning for graph-to-text generation. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing Student Research Workshop*, pages 16–22, Online, August 2021. Association for Computational Linguistics. URL: `https://aclanthology.org/2021.acl-srw.2`.

[49] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[50] Jianfeng Wen, Jianxin Li, Yongyi Mao, Shini Chen, and Richong Zhang. On the representation and embedding of knowledge bases beyond binary relations. In *Proceedings of the International Joint Conference on Artificial Intelligence*, page 1300–1307, 2016.

[51] Ruobing Xie, Zhiyuan Liu, Maosong Sun, et al. Representation learning of knowledge graphs with hierarchical types. In *Proceedings of International Joint Conference on Artificial Intelligence*, volume 2016, pages 2965–2971, 2016.

[52] Chenyan Xiong, Russell Power, and Jamie Callan. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*, pages 1271–1279, 2017.

[53] Bishan Yang and Tom Mitchell. Leveraging knowledge bases in lstms for improving machine reading. *arXiv preprint arXiv:1902.09091*, 2019.

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

[55] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 353–362. ACM, 2016.

[56] Richong Zhang, Junpeng Li, Jiajie Mei, and Yongyi Mao. Scalable instance reconstruction in knowledge bases via relatedness affiliated embedding. In *Proceedings of the 2018 World Wide Web Conference*, pages 1185–1194, 2018.

[57] Da Zheng, Xiang Song, Chao Ma, Zeyuan Tan, Zihao Ye, Jin Dong, Hao Xiong, Zheng Zhang, and George Karypis. DGL-KE: Training knowledge graph embeddings at scale. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 739–748. Association for Computing Machinery, 2020.

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