Realistic Re-evaluation of Knowledge Graph Completion Methods: An Experimental Study

ABSTRACT

Embedding models have gained popularity in knowledge graph completion, particularly for the task of link prediction. A benchmark dataset FB15k has been extensively used to evaluate such models in prior studies. Most triples in FB15k belong to a few special types of relations which exist due to artificial reasons. There are reverse relations that represent the same relationship in reverse directions, and there are Cartesian product relations for which every triple formed by the Cartesian product of applicable subjects and objects is a valid triple. Link prediction on such relations can be achieved with even better accuracy using straightforward approaches instead of the sophisticated embedding models. A more fundamental defect of this line of study is that the link prediction scenario, given such data, is non-existent in the real-world at all. Therefore, in evaluating link prediction models, it is misleading to mix the aforementioned artificial, straightforward cases with more realistic, challenging cases.

This is the first systematic study with the main objective of assessing the true effectiveness of embedding models when the unrealistic triples are removed. The results of our extensive experiments show that these models are much less accurate than what we used to perceive. Furthermore, methods that were supposed to substantially outperform TransE—the first such model—only attained similar or even worse performance on realistic cases. The results also show the inadequacy of the standard evaluation measures in penalizing correct predictions that do not exist in the labeled dataset. These results put into questions the foundational principles of such models. Hence, it calls for re-investigation of truly effective approaches for completing knowledge graphs. At a time when increasingly more AI-powered applications build their capacities relying on knowledge graphs as a pillar, this work can lead to profound impacts.

1 INTRODUCTION

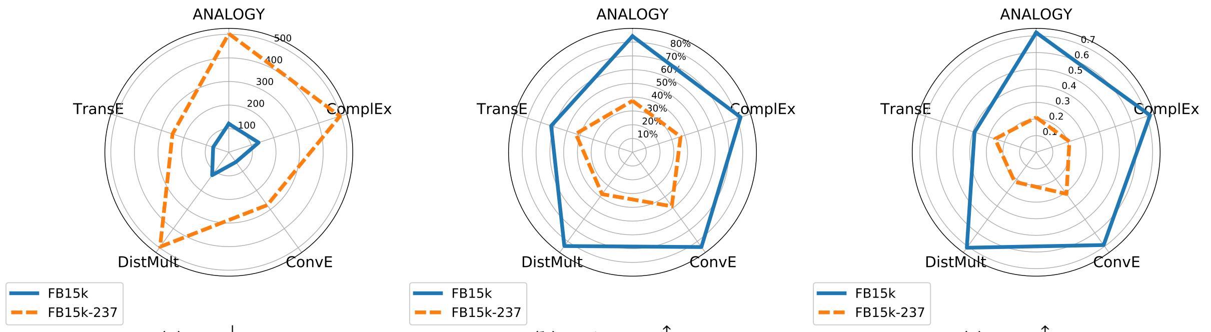
Large-scale knowledge graphs such as Freebase [3], DBpedia [2] and NELL [5] store real-world facts as triples in the form of (head entity (subject), *relation*, tail entity (object)), denoted (h, *r*, t), e.g., (Ludvig van Beethoven, *profession*, Composer). They are an important resource for many AI applications, such as question answering [31, 34, 35], search [7], and smart healthcare [24], to name just a few. Despite their large sizes, knowledge graphs are far from complete in most cases, which hampers their usefulness in these applications.

To address this important challenge, various methods have been proposed to automatically complete knowledge graphs. Existing methods in this active area of research can be categorized into two groups [21]. One group is based on *latent feature models*, also known as *embedding models*, represented by methods such as TransE [4] and RESCAL [22]. The other group is based on *observed feature models* that exploit observable properties of a knowledge graph. Examples of such methods include rule mining systems [9] and path ranking algorithms [15].

Particularly, the latent feature models are extensively studied. They embed each entity h (or t) into a multi-dimensional vector $\mathbf{h}$ (or $\mathbf{t}$). A relation r, also represented as a vector $\mathbf{r}$, is an operation on $\mathbf{h}$ and $\mathbf{t}$, e.g., *translation* [4] which is a geometric transformation between the head and tail entities in the embedding space. In TransE [4]—the first and one of the simplest and most efficient embedding models—embedding is learned in such a way that, if (h, *r*, t) holds, then $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. The learned latent representations are then used to infer new triples and find missing head or tail entities. Since its introduction in 2013, many models have been proposed to further improve TransE, e.g., TransH [29] and TransR [17].

Embedding models have been extensively evaluated on *link prediction*, a task that predicts the missing h in triple (?, *r*, t) or missing t in (h, *r*, ?). In such evaluation, a benchmark dataset named FB15k, a subset of Freebase created by Bordes et al. [4], has been extensively used. Toutanova and Chen [27] noted that FB15k contains many reverse triples, i.e., it includes many pairs of (h, *r*, t) and (t, r^{-1}, h) where r and r^{-1} are reverse relations. They constructed another dataset, FB15k-237, by only keeping one relation out of any pair of reverse relations. The community has started to use FB15k-237 in evaluating models and noted significant performance degeneration of existing models in comparison with their performance on FB15k [1, 6, 14, 27, 36].

Impact of reverse relations: In this paper we thoroughly examined the impact of reverse triples in FB15k (details in Section 4.2.1). The idiosyncrasies of the link prediction task on such data can be alarmingly summarized as follows. A1) Link prediction becomes much easier on a triple if its reverse triple is available. A2) For reverse triples, a straightforward method could be even more effective than complex machine learning models. We discovered that 70%



(a) FMR$^{\downarrow}$ **(b) FHits@10$^{\uparrow}$** **(c) FMRR$^{\uparrow}$**

Figure 1: Performance of embedding models on FB15k vs. FB15k-237 using several accuracy measures

of the triples in the training set of FB15k form reverse pairs. Similarly, for 70% of the triples in its test set, reverse triples exist in the training set. The abundant reverse triples suggest that TransE and other embedding models would have been biased toward learning whether two relations r_1 and r_2 form a reverse pair. Instead of complex machine learning models, one may achieve this goal by using statistics of the triples to derive simple rules of the form (h, r_1, t) $\Rightarrow$ (t, r_2, h). In fact, we generated such a simple model which attained 71% on FHits@1$^{\uparrow}$, a commonly used accuracy measure for embedding models. This is as accurate as an oracle and more accurate than the best-performing embedding model.

The above analysis suggests that the reverse triples led to a substantial over-estimation of the accuracy of embedding models, which is verified by our experiments on a wide range of models. While Section 5 thoroughly examines the results, the performance comparison of a few representative models using several measures is illustrated in Figure 1.[1] The results show that R1) the performance of all existing embedding models degenerates significantly after reverse triples are removed. R2) Many successors of the original TransE model were empirically shown to outperform TransE by far on FB15k, but they only attained similar or even worse performance on FB15k-237. For example, the FHits@10$^{\uparrow}$—an evaluation metric used in [4] and many other studies—of ANALOGY vs. TransE is 37.4% vs. 43.3% on FB15k-237, in stark contrast to 84.3% vs. 62.1% on FB15k. R3) The absolute accuracy of all models is poor, rendering them ineffective for real-world link prediction task. For example, using another popular measure FMRR$^{\uparrow}$, ConvE [6] is still the method with the best results on FB15k-237 (0.31). However, its performance on FB15k was considerably stronger (0.69).

The existence of excessive reverse triples in FB15k—the de facto benchmark dataset for link predication—actually presents a defect of this line of study that is far more fundamental than one may perceive: A3) the link prediction scenario, given such data, is non-existent in the real-world at all. These 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 a special relation *reverse_property*. There is never 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 models are 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 become available when the model needs to be applied for real prediction.

Impact of other data redundancy and Cartesian product relations: The data leakage due to reverse triples is a form of data redundancy that unrealistically inflates the models' accuracy. We identified other types of data redundancy in FB15k, by considering overlap of subject-object pairs in relations (Section 4.2.2). More specifically, some relations are *duplicate* as their subject-object pairs substantially overlap, and some are *reverse duplicate* when one relation's subject-object pairs overlap a lot with another relation's object-subject pairs.

We also discovered another type of relations, which we call *Cartesian product relations* (Section 4.3), that unrealistically inflate a model's link prediction accuracy. Given such a relation, there are a set of subjects and a set of objects, and the relation is valid from every subject in the first set to every object in the second set. In a May 2013 snapshot of Freebase, close to 15% of the relations are Cartesian product relations. In FB15k, 142 out of the 1345 relations are such relations. One example is *position*, since every team in a certain professional sports league has the same set of positions. The link prediction problem for such relations thus becomes predicting, say, whether an NFL team has the quarter-back

[1]Throughout the paper, we always place an upward (downward, resp.) arrow beside a measure to indicate that methods with greater (smaller, resp.) values by that measure possess higher accuracy.

position. Apparently such a prediction task is not very meaningful in the real-world. Moreover, when a substantial subset of the aforementioned subject-object Cartesian product is available in the training set, it is relatively easy for a model to attain a strong prediction accuracy.

The same analyses A1-A3 on reverse relations are also applicable on duplicate and Cartersian product relations. A1) In evaluating prediction models, it is misleading to mix such straightforward relations with more realistic, challenging relations. In the test set of FB15k, the numbers of reverse relations, duplicate relations, Cartesian product relations, and the remaining relations are 798, 118, 78, and 106, respectively. The FMRR$^{\uparrow}$ of ConvE on such relations is 0.79, 0.95, 0.82, and 0.51, respectively. A2) Instead of learning complex embedding models, a simpler approach can be more effective on Cartesian product relations. For instance, given a relation, by observing that a large percentage of possible subject-object pairs for the relation exist in the labeled dataset, one can derive the relation is a Cartesian product relation and thus the same relation should exist in all such pairs. Our experiments on 9 Cartesian product relations in FB15k obtained an average FHits@1$^{\uparrow}$ of 75% using this method, which is much higher than the 47.8% FHits@1$^{\uparrow}$ of TransE on these relations. A3) A vast majority of the duplicate and reverse duplicate relations in FB15k were artificially created. The existence of Cartesian product relations in FB15k is also quite artificial. In fact, 75% of them are due to special mediator nodes in Freebase that represent multiary relationships and simplification in FB15k for removing such nodes through concatenating edges. That is how FB15k entails the less meaningful prediction tasks mentioned above. Just like reverse triples, they render a link prediction scenario largely nonexistent in the real-world and lead to unrealistically strong prediction accuracy.

This paper aims to shed light on whether the embedding models are still effective for link prediction in real-world scenarios. There is no prior systematic study with the main objective of assessing the true effectiveness of embedding models in real-world settings. Other studies continue to evaluate models using both FB15k and FB15k-237, merely viewing the latter as a more challenging dataset. However, based on our analyses A1-A3 and experiment results R1-R3, we argue that FB15k is completely misleading and should not be used anymore.

Moreover, the much weaker performance of embedding models on FB15k-237 drove us to compare them with observed feature models, specifically using rules discovered by the rule mining system AMIE [9]. For instance, by observing that most persons in a knowledge graph have the same citizenship as their parents, AMIE may possibly generate rule (a, *is_citizen_of*, c) $\wedge$ (a, *has_child*, b) $\Rightarrow$ (b, *is_citizen_of*, c). Our experiment results show that an observed feature model such as AMIE also degenerate significantly on the more realistic FB15k-237. Furthermore, the results show it can outperform embedding models, as the FHits@10$^{\uparrow}$ of AMIE is the highest on FB15k among all methods. Regarding FB15k-237, it is harder to conclude a definitive winner.

This study can have **large potential impacts**. The original TransE paper [4] has been cited more than 1,200 times in less than 6 years and hundreds of publications focusing on improvements and variants of TransE have been produced, using the same FB15k dataset. This study puts into questions the foundational principles and the results of the field of knowledge graph completion using embedding models. It raises awareness of the inadequacy of the current de facto benchmark dataset and performance measures. It depicts a realistic picture of embedding models being much less accurate in link prediction than what we used to perceive. By that it thus renders link prediction a task without effective automated solution. Hence, it calls for re-investigation of possible effective approaches for completing knowledge graphs. At a time when increasingly more AI-powered applications build their capacities relying on knowledge graphs as a pillar [10], this work can lead to profound impacts on science, economy and society.

To sum up, this paper made these **contributions**:

- It provides a thorough investigation of the data redundancy problem in how existing embedding models for knowledge graph completion were trained, due to reverse and duplicate triples in the de facto benchmark dataset FB15k (Section 4).
- For the first time, It identifies the existence of Cartesian product relations in FB15k which, together with the data redundancy problem, makes previous performance measures of embedding models unrealistic (Section 4).
- It presents the results of a comprehensive evaluation of these defects' impacts on the performance of many representative embedding models (Section 5).
- It compares in detail the performance of embedding models with an observed feature model (AMIE) (Section 6).
- All codes, experiment scripts, datasets, and results are in a public repository https://github.com/Knowledge-Graph/KnowledgeGraph-completion . It will help ensure the reproducibility of this research and the repository will become a valuable resource to the community.

2 BACKGROUND: KNOWLEDGE GRAPH COMPLETION METHODS

In this section, we provide a brief summary of the two broad categories of knowledge graph completion methods, namely the observed feature models and the latent feature models, and we explain some of the most important methods in these two categories. Throughout the explanation, our approach of notation is as follows. Vectors are represented as bold

Table 1: Scoring functions of embedding models

Model	Scoring function $f_r(\mathsf{h},\mathsf{t})$
TransE [4]	$-\|\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|\|^2_{\ell_1/\ell_2}$
TransH [29]	$-\|\|(\mathbf{h}-\mathbf{v}_r^\top \mathbf{h}\mathbf{v}_r)+\mathbf{d}_r-(\mathbf{t}-\mathbf{v}_r^\top \mathbf{t}\mathbf{v}_r)\|\|$
TransR [17]	$-\|\|\mathbf{M}_r\mathbf{h}+\mathbf{r}-\mathbf{M}_r\mathbf{t}\|\|^2_{\ell_2}$
TransD [13]	$-\|\|(\mathbf{r}_p\mathbf{h}_p^\top+\mathbf{I})\mathbf{h}+\mathbf{r}+(\mathbf{r}_p\mathbf{t}_p^\top+\mathbf{I})\mathbf{t}\|\|$
RESCAL [22]	$\mathbf{h}^\top\mathbf{W}_r\mathbf{t}$
DistMult [32]	$\langle\mathbf{h},\mathbf{w}_r,\mathbf{t}\rangle$
ComplEx [28]	$\mathrm{Real}(\langle\mathbf{h},\mathbf{w}_r,\mathbf{t}\rangle)$
ANALOGY [18]	$\mathbf{h}^\top\mathbf{W}_r\mathbf{t}$
ConvE [6]	$f(vec(f([\bar{\mathbf{h}};\bar{\mathbf{r}}]*w))\mathbf{W})\mathbf{t}$

lower case letters such as $\mathbf{x}$ and $\mathbf{x}\in\mathbb{R}^d$ means the vector $\mathbf{x}$ is of dimensionality d. A matrix is denoted by a bold upper case letter such as $\mathbf{M}$. $[\mathbf{x}]_i$ represents the ith element of $\mathbf{x}$. A knowledge graph $\mathcal{G}$ consists of a set of entities $\mathcal{E}$ and a set of relations $\mathcal{R}$. Triples are represented as (h, r, t) where h, $\mathsf{t}\in\mathcal{E}$ are the head and tail entities, and relationship $r\in\mathcal{R}$ exists from the head to the tail. $\langle\mathbf{x},\mathbf{y},\mathbf{z}\rangle=\sum_i[\mathbf{x}]_i.[\mathbf{y}]_i.[\mathbf{z}]_i$ is the component-wise multi-linear dot product.

2.1 Latent Feature Models

Embedding-based methods employ two crucial components: (1) a scoring function to measure the plausibility of triples (h, r, t), and (2) a process to learn the representations (i.e., embeddings) of entities and relations by solving an optimization problem of maximizing the scores of correct triples while minimizing the scores of incorrect ones. Table 1 summarizes the scoring functions of the embedding models discussed in this section. In TransE [4], the scoring function is as follows.

$$f_r(\mathsf{h},\mathsf{t})=-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|^2_{\ell_1/\ell_2} \tag{1}$$

TransE is a scalable method with a small number of model parameters, but it has limitations in modeling 1-to-n, n-to-1, and m-to-n relations [29].

TransH [29] is similar to TransE but it aims to address TransE's limitations by not using the same embedding of an entity in different relations. Consider the following example. In principle the films produced by Universal Studios should be represented as similar vectors when we focus on a relation about film production. When the relation in focus becomes film genre instead, the vectors for films in different genres should be far from each other. Albeit intuitive, such is not possible in TransE. TransH models each relation as a vector $\mathbf{d}_r$ on a relation-specific hyperplane $\mathbf{v}_r$ (i.e., the normal vector where $\|\mathbf{v}_r\|=1$), and the entity embeddings $\mathbf{h}$ and $\mathbf{t}$ are projected to this hyperplane to obtain $\mathbf{h}_\perp=\mathbf{h}-\mathbf{v}_r^\top\mathbf{h}\mathbf{v}_r$ and $\mathbf{t}_\perp=\mathbf{t}-\mathbf{v}_r^\top\mathbf{t}\mathbf{v}_r$. In this way, the embeddings of entities are learned differently for each relation.

Lin et al. [17] proposed TransR which learns the embeddings in two different vector spaces $\mathbb{R}^d$ and $\mathbb{R}^k$ for entities and relations, in which the dimensions k and d are not necessarily identical. They argued that using the same semantic space for entities and relations, as in TransE and TransH, is insufficient because they are two completely different types of objects. Instead, TransR defines a projection matrix $\mathbf{M}_r$ to map entity embeddings to the vector space for each relation. By this approach, similar entities near each other in the entity space may become far apart after being mapped to the relation space because they are different regarding particular aspects. The scoring function of TransR is defined as

$$f_r(\mathsf{h},\mathsf{t})=-\|\mathbf{M}_r\mathbf{h}+\mathbf{r}-\mathbf{M}_r\mathbf{t}\|^2_{\ell_2} \tag{2}$$

In TransD [13], which improves over TransR, the entities and the relation in a triple (h, r, t) are represented by two groups of vectors: one being $\mathbf{h}$, $\mathbf{r}$ and $\mathbf{t}$, and the other being $\mathbf{h}_p$, $\mathbf{r}_p$ and $\mathbf{t}_p$. The latter group defines the projection matrices as follows.

$$\mathbf{M}_{rh}=\mathbf{r}_p\mathbf{h}_p^\top+\mathbf{I} \qquad \mathbf{M}_{rt}=\mathbf{r}_p\mathbf{t}_p^\top+\mathbf{I} \tag{3}$$

As in TransR, these projection matrices are used to map entity vectors $\mathbf{h}$ and $\mathbf{t}$ to the relation vector space. However, in contrast to TransR, there is a unique projection matrix for each entity-relation pair. The argument given by [13] is that different types of entities connected to a relation should have different mapping matrices and one projection matrix per relation, as in TransR, is insufficient. Furthermore, computationally expensive matrix-vector multiplication in TransR poses a challenge in scaling it up. TransD tackles this challenge by using vector operations instead.

To learn the entity and relation representations, a loss function is minimized. Two of the most-frequently used loss functions in embedding models are margin-based loss function and logistic loss, as in Equations 4 and 5, respectively.

$$L=\sum_{(\mathsf{h},r,\mathsf{t})\in S}\ \sum_{(\mathsf{h}',r,\mathsf{t}')\in S'} max(0,f_r(\mathsf{h},\mathsf{t})+\gamma-f_r(\mathsf{h}',\mathsf{t}')) \tag{4}$$

$$L=\sum_{(\mathsf{h},r,\mathsf{t})\in S\cup S'} log(1+exp(-y_{hrt}.f_r(\mathsf{h},\mathsf{t}))) \tag{5}$$

In these equations, y_{hrt} is the sign of a training example (+1 for positive example and -1 for negative example), γ is the margin, S is the set of positive triples (h, r, t) in the training set and S' is the set of negative triples (h', r, t'). Since a real-world knowledge graph contains only positive triples, negative triples in FB15k were generated by *corrupting* the positive triples—a process that replaces the head or tail entity of each positive triple by other entities in the knowledge graph [4].

While these embedding methods employ additive operations for composition of head and tail entity vectors, there are bilinear approaches such as RESCAL [22], DistMult [32] and ComplEx [28] which use multiplicative operations (i.e., element-wise dot product). RESCAL is a collective matrix factorization model which represents a relation as a matrix $\mathbf{W}_r \in \mathbb{R}^{d\times d}$ that describes the interactions between latent representations of entities. The score of a triple in this method is defined as:

$$f_r(\mathsf{h}, \mathsf{t}) = \mathbf{h}^\top \mathbf{W}_r \mathbf{t} \tag{6}$$

DistMult is similar to RESCAL but it restricts relations to be diagonal matrices $\mathbf{w}_r \in \mathbb{R}^d$ in order to reduce the number of relation parameters:

$$f_r(\mathsf{h}, \mathsf{t}) = \langle \mathbf{h}, \mathbf{w}_r, \mathbf{t} \rangle \tag{7}$$

However, this simplification comes at a cost—DistMult can only model symmetric relations. ComplEx is an extension of DistMult. It uses complex numbers instead of real numbers to handle symmetric and anti-symmetric relations. Therefore the diagonal matrices are $\mathbf{w}_r \in \mathbb{C}^d$. ANALOGY [18] further optimizes the embeddings of entities and relations with respect to their analogical properties. They argue that, if two subgraphs g_1 and g_2 are analogous, missing triples in g_1 can be inferred by their counterparts in g_2, and vice versa. ConvE [6] is a neural network model that uses 2D convolutional layers over embeddings and interactions between entities and relations are modeled by convolutional and fully connected layers. In Table 1, f is a nonlinear function, $*$ represents the convolution operator, Real(.) returns the real part of a complex number. Due to space limitations, we will not further explain the details of ComplEx and ConvE which can be found in their corresponding publications.

2.2 Observed Feature Models

In contrast to embedding models which employ latent features of knowledge graphs, observed feature models directly exploit observable features. For example, one can derive the ancestor-descendant relationship between two persons captured by the rule (a, *father_of*, b) $\wedge$ (b, *father_of*, c) $\Rightarrow$ (a, *ancestor*, c) which can be also represented as *father_of*(a,b) $\wedge$ *father_of*(b,c) $\Rightarrow$ *ancestor*(a,c). This section focuses its discussion on a representative of the observed feature models, the rule mining system AMIE [9]. Compared with the aforementioned embedding-based methods, rule-mining systems have the advantage of being more interpretable.

AMIE mines Horn rules in the aforementioned form. Mining Horn clauses from data has long been extensively studied in the field of *inductive logic programming* (ILP) [20]. However, methods proposed in this field are not applicable to knowledge graphs due to two reasons. First, ILP systems are not efficient enough to cope with large knowledge graphs. Second, in addition to positive examples, they also rely on negative examples which are non-trivial to come by as general knowledge graphs explicitly record only known facts. Under the *open-world assumption*, which is far more realistic than the *closed-world assumption* [23] in the context of knowledge graphs, facts nonexistent or cannot be proven based on available knowledge are not necessarily *false* and can be just interpreted as *unknown*. In fact, the Web Ontology Language (OWL) follows the open-world assumption [12]. AMIE tackles these challenges with an efficient rule mining algorithm and a method to generate negative examples using *partial completeness assumption*.

In AMIE, a rule has a body (antecedent) and a head (consequent), represented as $B_1 \wedge B_2 \wedge \ldots \wedge B_n \Rightarrow H$ or in simplified form $\overrightarrow{B} \Rightarrow H$. The body consists of multiple *atoms* $B_1, \ldots, B_n$ and the head H itself is also an atom. In an atom r(h,t), which is another representation of fact triple (h, r, t), the subject and/or the object are variables to be instantiated. The prediction of the head can be carried out when all the body atoms can be instantiated in the knowledge graph.

As in ILP systems, AMIE uses constraints to restrict the hypotheses (the set of clauses in first-order logic) because the hypothesis spaces of logic programs are usually very large. The first constraint is that all the atoms in a rule must be connected. In more detail, if every atom is modeled as an edge and the edges connect via common variables and entities, all the atoms must form a single connected component. This constraint excludes rules with completely unrelated atoms. The second constraint is to consider only closed rules in which every variable appears at least twice.

One popular measure of the quality of a mined rule is *confidence*, defined as follows in which $z_1, \ldots, z_m$ are variables.

$$conf(\overrightarrow{B} \Rightarrow r(\mathsf{h}, \mathsf{t})) := \frac{support(\overrightarrow{B} \Rightarrow r(\mathsf{h}, \mathsf{t}))}{\#(\mathsf{h}, \mathsf{t}) : \exists z_1, \ldots, z_m : \overrightarrow{B}} \tag{8}$$

The numerator in Equation (8) is the support of the rule, i.e., "the number of distinct pairs of subjects and objects in the head of all instantiations" that appear in the knowledge graph [9]. The denominator is the support of the body.

3 EXISTING BENCHMARK DATASET AND MEASURES FOR EVALUATION OF EMBEDDING MODELS

3.1 Freebase and FB15k

Most embedding models in literature have been evaluated on a subset of Freebase named FB15k that was generated by Bordes et al. [4]. Freebase [3] is one of the largest public domain knowledge bases. Its final 2015 version records 1.9 billion triples of common facts. This knowledge base, available in various data formats including RDF, can be modeled

Table 2: Basic statistics about FB15k and FB15k-237

	FB15k	FB15k-237
# of relations	1,345	237
# of entities	14,951	14,541
# of triples in training set	483,142	272,115
# of triples in validation set	50,000	17,535
# of triples in test set	59,071	20,466

as a graph in which the entities and relations correspond to nodes and edges, respectively.

According to [4], FB15k contains only those Freebase entities that were also available in Wikipedia based on the *wiki-links* database [2] and have at least 100 appearances in Freebase. The relations included into FB15k must also have 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 training, validation and test sets, as shown by the statistics in Table 2.

3.2 Evaluation Methods and Measures

Embedding models have been evaluated using several highly-related knowledge graph completion tasks such as triple classification [26, 29], link prediction [16], relation extraction [17, 30], and relation prediction [25]. Triple classification is the binary classification task of determining whether a given triple is correct. Relation extraction seeks to extract the relation between two given entities using text corpus whereas relation prediction aims at predicting the relation between two entities by exploiting knowledge graph itself.

The *link prediction* task as described in [4] is particularly widely used for evaluating different embedding methods. Its goal is to predict the missing h or t in a triple (h, *r*, t). For each test triple (h, *r*, t), the head entity h is replaced with every other entity h' $\in \mathcal{E}$ in the dataset, to form *corrupted* triples. The original test triple and its corresponding corrupted triples are ranked by their scores according to the score functions (Section 2.1) and the rank of the original test triple is stored as $rank_{\mathsf{h}}$. The same procedure is repeated by replacing the tail entity t to calculate $rank_{\mathsf{t}}$. A method with the ideal ranking function should rank the test triple at top.

The accuracy of different embedding models is measured using Mean Rank (MR$^{\downarrow}$), Hits@1$^{\uparrow}$, Hits@10$^{\uparrow}$, and Mean Reciprocal Rank (MRR$^{\uparrow}$), as in [4]. MR$^{\downarrow}$ is the mean of the test triples' ranks, as follows:

$$MR = \frac{1}{2\,|T|} \sum_{(\mathsf{h},r,\mathsf{t}) \in T} (rank_{\mathsf{h}} + rank_{\mathsf{t}}) \tag{9}$$

in which $|T|$ is the size of the test set. Hits@k$^{\uparrow}$ is the percentage of top k results that are correct. MRR$^{\uparrow}$ is the average multiplicative inverse of the ranks of the test triples:

$$MRR = \frac{1}{2\,|T|} \sum_{(\mathsf{h},r,\mathsf{t}) \in T} (\frac{1}{rank_{\mathsf{h}}} + \frac{1}{rank_{\mathsf{t}}}) \tag{10}$$

Besides these raw metrics, we also used their corresponding *filtered* metrics [4], denoted FMR$^{\downarrow}$, FHits@1$^{\uparrow}$, FHits@10$^{\uparrow}$, and FMRR$^{\uparrow}$, respectively. In calculating these measures, corrupted triples that are already in training, test or validation sets do not participate in ranking. In this way, a model is not penalized for ranking other correct triples higher than a test triple. For example, consider the task of predicting tail entity. Suppose the test triple is (Tim Burton, *film*, Edward Scissorhands) and the training, test, or validation set also contains another triple (Tim Burton, *film*, Alice in Wonderland). If a model ranks Alice in Wonderland higher than Edward Scissorhands, the filtered metrics will remove this film from the ranked list so that the model would not be penalized for ranking (Tim Burton, *film*, Edward Scissorhands) lower than (Tim Burton, *film*, Alice in Wonderland), both correct triples.

We note that, by definition, higher Hits@1$^{\uparrow}$ (FHits@1$^{\uparrow}$), Hits@10$^{\uparrow}$ (FHits@10$^{\uparrow}$) and MRR$^{\uparrow}$ (FMRR$^{\uparrow}$), and lower MR$^{\downarrow}$ (FMR$^{\downarrow}$) indicate better accuracy.

4 ANALYZING THE INADEQUACY OF FB15K AND EVALUATION MEASURES

In this section we investigate the existence and impact of a few types of relations in FB15k, including reverse relations, redundant relations, and Cartesian product relations. The outcome is alarming. It suggests that triples in these relations led to a substantial over-estimation of the accuracy of embedding models.

4.1 Identifying the Most Probable Freebase Snapshot Used for Producing FB15k

In order to understand the various defects in FB15k and their root cause, we searched for the same Freebase snapshot that was used to create FB15k. When it was active, Freebase maintained periodic snapshots, more frequent than monthly. It is unclear from [4] which snapshot was used to create FB15k.[3] To derive the most probable timestamp of the snapshot from which FB15k was produced, we compared the snapshots available at https://commondatastorage.googleapis.com/freebase-public/. Particularly, we considered the snapshots in the several months before [4] was published. We analyzed to what degree these snapshots overlap with FB15k. The May 5, 2013 snapshot (i.e., https://commondatastorage.googleapis.com/freebase-public/rdf/freebase-rdf-2013-05-05-00-00.gz) has the largest overlap among these snapshots, as it contains 99.54% of the

[2]https://code.google.com/archive/p/wiki-links/

[3]We also had email communication with the authors of [4]. They could not remember the exact date of the Freebase snapshot used for producing FB15k.

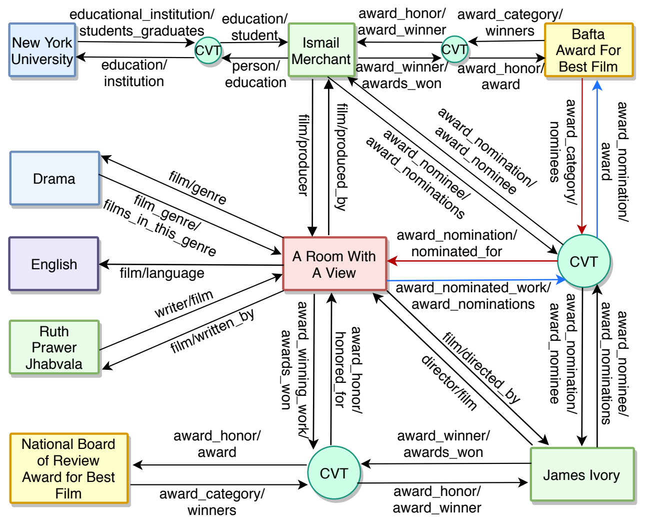


Figure 2: A Freebase fragment with mediator (CVT) nodes

triples in FB15k, including 99.2% of the non-concatenated regular triples and 99.69% of the concatenated triples. (We explain below what the concatenated triples are.) Based on these statistics, we concluded that FB15k was most likely drawn from a snapshot around May 5, 2013, which can be approximated by the snapshot on that exact date.

Concatenated Edges

Mediator nodes, also called *compound value type* (CVT) nodes, are used in Freebase to represent multiary relationships. For example, Figure 2 shows several CVT nodes. The rightmost CVT node is connected to an award, two nominee nodes and a work through various relations. In the May 2013 Freebase snapshot, for many (but not all) CVT nodes, additional *concatenated edges* were created. Specifically, for such a CVT node, multiple triples were created, each a concatenation of two edges connected through the CVT node. These binary relationships partly capture the multiary relationship represented by the CVT node. 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 2 would be concatenated to form a triple between the award and the work nominated for the award. The concatenation of two relations r_1 and r_2 is written as $r_1.r_2$. There are 54,541,700 concatenated triples in the May 2013 snapshot.

FB15k does not include any CVT nodes or edges adjacent to such nodes from Freebase. However, it kept most concatenated edges (or maybe all, although we have no absolute way to verify, since nowhere we can find all Freebase snapshots that have ever been created). All the 707 concatenated relations in FB15k exist in the May 2013 snapshot. Among the 592,213 triples in FB15k, 396,418 are concatenated and 394,947 of them could be found in the snapshot.

4.2 Data Redundancy

4.2.1 Data Leakage Due to Reverse Triples. Toutanova and Chen [27] noted that the widely-used benchmark dataset FB15k contains many reverse triples, i.e., it includes many pairs of (h, r, t) and (t, r^{-1}, h) where r^{-1} is the reverse of r. Freebase actually denotes reverse relations explicitly using a special relation *reverse_property*.[4] For instance, the triple (film/directed_by, *reverse_property*, director/film) in Freebase denotes that *film/directed_by* and *director/film* are reverse relations.[5] Therefore, (A Room With A View, *film/directed_by*, James Ivory) and (James Ivory, *director/film*, A Room With A View) form a pair of reverse triples, as shown in Figure 2. The reverse relation of a concatenated relation $r_1.r_2$ is the concatenation of the corresponding reverse relations, i.e., $r_2^{-1}.r_1^{-1}$. For example, in Figure 2 *award_nominated_work/award_nominations . award_nomination/award* (blue edges) and *award_category/nominees . award_nomination/nominated_for* (red edges) are two concatenated relations which are reverse of each other.

Using the reverse relation information from the May 2013 snapshot of Freebase, out of the 483,142 triples in the training set of FB15k, 338,340 triples form 169,170 reverse pairs. Furthermore, for 41,529 out of the 59,071 triples (i.e., about 70.3%) in the test set of FB15k, their reverse triples exist in the training set. These data characteristics suggest that TransE and other embedding models would have been biased toward learning reverse relations for link prediction. More specifically, the task can largely become inferring whether two relations r_1 and r_2 form a reverse pair. Given the abundant reverse triples in the dataset, this goal could potentially be achieved without using a machine learning approach based on complex embeddings of entities and relations. Instead, one may aim at deriving simple rules of the form (h, r_1, t) $\Rightarrow$ (t, r_2, h) using statistics about the triples in the dataset. In fact, Dettmers et al. [6] generated such a simple model which attained a 68.9% accuracy by the measure FHits@1$^\uparrow$. We generated a similar model by finding the relations that have more then 80% intersections. It attained an FHits@1$^\uparrow$ of 71%. This is even slightly better than the 70.3% accuracy

[4]The relation's full name is */type/property/reverse_property*. Freebase prefixes relation names with domains and entity types, i.e., the full name of a relation follows the template of /domain/entity type/relation. For instance, */tv/tv_genre/programs* represents a relation named *programs* belonging to domain *tv*. The subject of any instance triple of this relation belongs to type *tv_genre*. For simplicity of presentation in this paper, by default we omit the prefixes. In various places we retain the entity type to avoid confusions, e.g., different relations having identical names after prefixes are removed.
[5]Note that relations *film/directed_by* and *director/film* are also special entities in (film/directed_by, *reverse_property*, director/film).

one may achieve using an oracle based on the reverse relations denoted in the May 2013 Freebase snapshot. It is also stronger than the 67% `FHits@1`$^\uparrow$ from the best performing embedding model (more details in Table 10 of Section 5).

The problem with evaluating embedding models using FB15k is far more fundamental than one may perceive. Not only the existence of reverse triples makes link prediction more straightforward, but also it represents a scenario of prediction that does not exist in the real-world at all. The redundant reverse relations in Freebase are just artificially created. When a new fact is added into the knowledge graph, it would be added as a pair of reverse triples, using the reverse relations denoted by the aforementioned *reverse_property*. There is never a scenario in which one needs to predict a triple while its reverse is already there. Either both triples are already in the knowledge graph or none of them. In training a knowledge graph completion model using FB15k, we fall into a form of overfitting in that the learned models are 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 become available when the model needs to be applied for real prediction.

4.2.2 *Other Redundant Triples.*

Reverse relations denoted by *reverse_property* are absolutely reciprocal, as triples in such relations were added into Freebase in pairs. In addition to that, there are other kinds of semantically redundant relations in FB15k. While it is infeasible to manually verify such semantic redundancy, we used a simple method to automatically detect it. Given two relations r_1 and r_2, we calculate how much their subject-object pairs overlap. Suppose $|r|$ is the number of instance triples in relation r and T_r denotes the set of subject-object pairs in relation r, i.e., $T_r = \{(\mathrm{h},\mathrm{t}) \mid r(\mathrm{h},\mathrm{t}) \in \mathcal{G}\}$. We say r_1 and r_2 are *near-duplicate relations*, simplified as *duplicate relations*, if they satisfy the following condition:

$$\frac{|T_{r_1} \cap T_{r_2}|}{|r_1|} > 0.8 \quad \text{and} \quad \frac{|T_{r_1} \cap T_{r_2}|}{|r_2|} > 0.8 \qquad (11)$$

Moreover, T_r^{-1} denotes the reverse entity pairs of T_r, i.e., $T_r^{-1} = \{(\mathrm{t},\mathrm{h}) \mid (\mathrm{h},\mathrm{t}) \in T_r\}$. We say r_1 and r_2 are *reverse duplicate relations* if they satisfy the following condition:

$$\frac{|T_{r_1} \cap T_{r_2}^{-1}|}{|r_1|} > 0.8 \quad \text{and} \quad \frac{|T_{r_1} \cap T_{r_2}^{-1}|}{|r_2|} > 0.8 \qquad (12)$$

For example *football_position/players* (r_1) and *sports_position /players.football_roster_position/player* (r_2) are duplicate based on this definition, since $\frac{|T_{r_1} \cap T_{r_2}|}{|r_1|} = 0.87$ and $\frac{|T_{r_1} \cap T_{r_2}|}{|r_2|} = 0.97$. These two relations are displayed in Figure 3 using red and green edges, respectively. The first relation records each football player's position considering their overall career. For

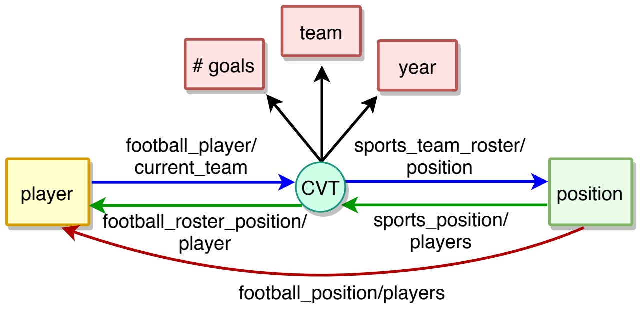


Figure 3: Duplicate relations

the second relation, each instance triple is a concatenation of two edges connected through a mediator node, representing a multiary relationship about the position a player plays for a team as shown in Figure 3. Since most players play at the same position throughout their careers, these two relations are redundant. Another example of similar nature is that r_1 and *football_player/current_team . sports_team_roster/position* (r_3) are reverse duplicate relations, because $\frac{|T_{r_1} \cap T_{r_3}^{-1}|}{|r_1|} = 0.87$ and $\frac{|T_{r_1} \cap T_{r_3}^{-1}|}{|r_3|} = 0.97$. In Figure 3 they are highlighted in red and blue, respectively.

For each triple in the test set of FB15k, we use the methods explained to determine whether it has 1) reverse triples, 2) duplicate or reverse duplicate triples in the training set, and whether it has 3) reverse triples, 4) duplicate or reverse duplicate triples in the test set itself. A triple may have redundant triples in any of these four categories. We use bitmap encoding to represent different cases of redundancy for a triple. For example, 1100 is for a triple that has both reverse triples and (reverse) duplicate triples in the training set. Hence, there are 16 possible different combinatorial cases. Considering the test triples in FB15k, not all 16 cases exist, instead, 12 different cases exist. Figure 4 shows the percentages of triples in different cases. The 7 cases smaller than 1% are combined in one slice. The biggest three slices are 1000 (triples with only reverse triples in the training set), 0000 (triples without any redundant triples), and 0010 (triples with only reverse triples in the test set). In total, 41529, 1847 and 2701 test triples have reverse, reverse duplicate and duplicate triples in the training set, and 4992, 249, and 328 test triples have these categories of redundant triples in the test set itself. The data redundancy causes overestimation of embedding models' accuracy. For instance, the `FMR`$^\downarrow$, `FHits@10`$^\uparrow$, `FHits@1`$^\uparrow$, and `FMRR`$^\uparrow$ of ConvE are 1.67, 99.1, 92.1, and 0.95 on such relations, whereas its performance using these measures on relations without any redundancy is only 225.1, 69.3, 42.6, and 0.51, respectively.

Table 3: The strong FMRR$^\uparrow$results on a few Cartesian product relations in FB15k-237

relation	# of triples	TransE	DistMult	ComplEx	ANALOGY	ConvE
olympic_games/medals_awarded . olympic_medal_honor/medal	16	1	1	1	1	1
food/nutrients . nutrition_fact/nutrient	105	0.71	0.73	0.72	0.72	082
travel_destination/climate . travel_destination_monthly_climate/month	60	0.85	0.98	0.98	0.99	0.98

Table 4: Link prediction using Cartesian product property

	TransE results								Prediction using Cartesian product property											
	FB15k as ground truth								FB15k as ground truth								Freebase as ground truth			
	MR$^\downarrow$	H10$^\uparrow$	H1$^\uparrow$	MRR$^\uparrow$	FMR$^\downarrow$	FH10$^\uparrow$	FH1$^\uparrow$	FMRR$^\uparrow$	MR$^\downarrow$	H10$^\uparrow$	H1$^\uparrow$	MRR$^\uparrow$	FMR$^\downarrow$	FH10$^\uparrow$	FH1$^\uparrow$	FMRR$^\uparrow$	FMR$^\downarrow$	FH10$^\uparrow$	FH1$^\uparrow$	FMRR$^\uparrow$
r1	19.97	43.33	3.33	13.66	1.39	99.17	83.33	0.9	16.83	56	5	0.16	1.46	100	68	0.82	1.43	100	79	0.83
r2	5	100	0	0.24	2.5	100	0	0.42	4	100	0	0.33	2	100	50	0.67	1	100	100	1
r3	6.5	100	0	0.22	2.5	100	0	0.42	3	100	0	0.38	1	100	100	1	1	100	100	1
r4	1784.25	25	0	0.09	1777	25	0	0.09	1642.38	75	25	0.47	1638	88	88	0.88	1638	88	88	0.88
r5	12.74	58.82	11.76	0.31	1	100	100	1	566.32	56	26	0.41	552	94	94	0.94	552	94	94	0.94
r6	10.72	62.5	9.38	0.27	1.03	100	96.88	0.98	7.13	72	19	0.38	1	100	100	1	1	100	100	1
r7	7.75	50	25	0.35	2.75	100	25	0.51	2271.25	50	0	0.22	2263.5	75	25	0.41	2262.5	75	50	0.55
r8	10.75	62.5	0	0.19	4.63	75	25	0.44	7.88	63	0	0.21	3.38	100	50	0.59	2.5	100	50	0.64
r9	15.59	53.13	18.75	0.34	1	100	100	1	10.25	63	22	0.38	1	100	100	1	1	100	100	1

r1	*travel_destination/climate . travel_destination_monthly_climate/month*
r2	*computer_videogame/gameplay_modes*
r3	*gameplay_mode/games_with_this_mode*
r4	*educational_institution/sexes_accepted . gender_enrollment/sex*
r5	*olympic_medal/medal_winners . olympic_medal_honor/olympics*
r6	*x2010fifaworldcupsouthafrica/world_cup_squad/current_world_cup_squad . x2010fifaworldcupsouthafrica/current_world_cup_squad/position*
r7	*dietary_restriction/compatible_ingredients*
r8	*ingredient/compatible_with_dietary_restrictions*
r9	*olympic_games/medals_awarded . olympic_medal_honor/medal*

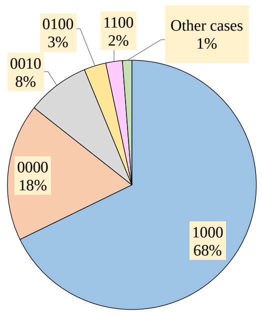


Figure 4: Redundancy in the test set of FB15k

4.3 Cartesian Product Relations

We also discovered another issue with FB15k which makes existing performance measures of embedding models unrealistic. This problem manifests in what we call *Cartesian product relations*. Given such a relation, the subject-object pairs from all instance triples of the relation form a Cartesian product. In other words, there are a set of subjects and a set of objects, and the relation exists from every subject in the first set to every object in the second set. One example Cartesian product relation is *climate*, since (a, *climate*, b) is a valid triple for every possible city a and month b. Another example is *position*, since every team in a certain professional sports league has the same set of positions. The link prediction problem for such relations thus becomes predicting whether a city has a climate in, say, January, or whether an NFL team has the quarter-back position. Apparently such a prediction task is not very meaningful.

A few notes can be made about Cartesian product relations. (1) Similar to reverse relations and other forms of data redundancy, the existence of these relations unrealistically inflates a model's link prediction accuracy. When a substantial subset of the aforementioned subject-object Cartesian product is available in the training set, it is relatively easy for a model to attain strong accuracy. However, it is problematic to mix such straightforward test cases with more realistic, challenging cases. At least, the performance of a model should be separately evaluated on Cartesian product relations and non-Cartesian product relations.

(2) Albeit not always meaningful, one may still perform link prediction on Cartesian product relations. However, a simpler approach can be more effective than learning complex embedding models. For instance, by examining all instance triples of a relation r, a method can identify the sets of all subjects $S_r = \{h \mid \exists r(h,t) \in \mathcal{G}\}$ and objects $O_r = \{t \mid \exists r(h,t) \in \mathcal{G}\}$ in the instance triples. By observing a large

percentage of possible subject-object pairs existing in the relation, the method can derive the relation might be a Cartesian product relation. More specifically, if $|r| \ / \ (|S_r| \times |O_r|)$ is greater than a pre-determined threshold (0.8 in our study), we consider r a Cartesian product relation. There are a total of 35,084 relations (371,200,214 instance triples) in the aforementioned May 2013 Freebase snapshot, of which 4,698 relations have only one instance triple each. Among the 30,386 remaining relations (371,195,516 instance triples), we detected 4,683 Cartesian product relations (2,480,392 instance triples) using this method. We also identified 142 Cartesian product relations in FB15k, with 13,038 triples. Although there are not as many Cartesian product relations as reverse relations, we discovered that among the relations on which embedding models attained the highest accuracy there are Cartesian product relations. Table 3 shows such results on the relations using FMRR$^{\uparrow}$ on FB15k after reverse triples are removed. [6] These are the Cartesian product relations among the top-12 relations ranked by FMRR$^{\uparrow}$ of ConvE on FB15k.

Once a relation r is detected as a Cartesian product relation, for link prediction, we can predict triple (h, r, t) to be valid, given any h $\in S_r$ and t $\in O_r$. We can further extend this approach. If an entity type system exists (which is the case with Freebase), we can identify the common type of all entities in S_r and O_r, respectively, and then predict (h, r, t) valid for all h and t belonging to the corresponding types.

(3) The existence of Cartesian product relations in FB15k is quite artificial. In fact, many such relations are due to mediator nodes in Freebase and simplification in FB15k for removing such nodes. The majority of relationships in Freebase are multiary relationships connected through mediator nodes. For instance, a mediator node is connected to Tokyo with an edge labeled *climate* and to January with an edge labeled *month*. It is further connected to 34 with an edge labeled *average_min_temp_c*, indicating that the average low temperature in Tokyo is 34 degrees Fahrenheit in January. In fact, it is also connected to other nodes for capturing the maximal temperature, rain fall, and so on. The more realistic and useful prediction task is to predict the average temperature, rather than whether a city has a temperature. Even though most real-world relationships are multiary, the studies on link prediction have always simplified it as multiple binary relationships (which is lossful as the multiple binary relationships cannot be used to restore the identical original relationship). That is how FB15k entails the less meaningful prediction tasks. In fact, out of the 4,683 Cartesian product relations mentioned in (2), 3506 are concatenated relations by removing mediator nodes. In the specific example above, the concatenated edge is between Tokyo and January, connecting the original *climate* and *month* edges.

(4) The performance measures in Section 3.2 are based on the closed-world assumption and thus have flaws when a model correctly predicts a triple that does not exist in the labeled dataset. More specifically, if a corrupted triple of a given test triple does not exist in the training or test set, it is considered incorrect. However, the corrupted triple might be correct as well. If a model ranks the corrupted triple higher than the test triple itself, its accuracy measures will be penalized, which contradicts with the exact goal of ink prediction—finding correct triples that do not already exist in the dataset. While this defect of the accuracy measures is applicable on all types of relation, [7] it is more apparent in evaluating a method that is capable of leveraging the characteristics of Cartesian product relations. Such a method would mark many triples non-existent in the labeled dataset as correct and further rank many of them higher than the test triples.

Table 4 uses several measures to show the accuracy of the prediction method based on the Cartesian product property, as explained in (2). [8] The method's accuracy is evaluated using both FB15k and the larger Freebase snapshot as the ground truth. The table also shows the results of using TransE. It presents these results on 9 Cartesian product relations in FB15k. Some of these relations are detected as Cartesian product relations by applying the aforementioned process over the training set of FB15k and some are detected over the Freebase snapshot. We can make several observations regarding the results in Table 4. First, the performance of using Cartesian product property is higher when Freebase instead of FB15k is the ground truth. This is because Freebase subsumes FB15k and thus is affected less by the defect mentioned in (4). Second, using Cartesian product property is more accurate than embedding models such as TransE, especially when Freebase snapshot is used as the ground truth to calculate filtered measures. (Note that using Freebase as the ground truth will not affect the regular measures such as MR$^{\downarrow}$. Therefore we do not repeat those measures in the table.) For example, consider predicting triples in relation *r2* in Table 4. We observed that Cartesian product has a FMRR$^{\uparrow}$ of 0.67 while using FB15k as ground truth in comparison to 0.42 for TransE, however this result can be improved to 1 if the Freebase snapshot is used as the ground truth.

[6] This dataset is called FB15k-237 and will be further discussed in Section 5. We want to inspect the impact of Cartesian product relations after removing reverse relations, because the latter also causes over-estimation of embedding models' accuracy and they dominate Cartesian product relations in terms of number of triples.

[7] Fundamentally it is because the models are evaluated by ranking instead of binary classification of triples.

[8] For coping with space limitations, we shortened the names of some measures, e.g., FHits@10$^{\uparrow}$is shortened as FH10$^{\uparrow}$.

Table 5: Link prediction results

	FB15k						FB15k-237					
	Raw measures			Filtered measures			Raw measures			Filtered measures		
Model	MR$\downarrow$	Hits@10$\uparrow$	MRR$\uparrow$	FMR$\downarrow$	FHits@10$\uparrow$	FMRR$\uparrow$	MR$\downarrow$	Hits@10$\uparrow$	MRR$\uparrow$	FMR$\downarrow$	FHits@10$\uparrow$	FMRR$\uparrow$
TransE [4]	243.0	34.9	–	125.0	47.1	–	–	–	–	–	–	–
	200.1	43.9	0.23	69.7	62.1	0.39	440.5	30.6	0.17	250.9	43.3	0.26
TransH [29]	211.0	42.5	–	84.0	58.5	–	–	–	–	–	–	–
	234.7	45.5	0.18	84.0	69.0	0.35	575.4	19.5	0.07	365.5	32.7	0.14
TransR [17]	226.0	43.8	–	78.0	65.5	–	–	–	–	–	–	–
	231.8	48.8	0.24	78.2	72.9	0.47	544.5	28.8	0.15	336.3	43.8	0.27
TransD [13]	211.0	49.4	–	67.0	74.2	–	–	–	–	–	–	–
	234.4	47.4	0.18	85.4	70.9	0.35	586	22.8	0.09	276.3	45.5	0.22
RESCAL [22]	828.0	28.4	–	683.0	44.1	–	–	–	–	–	–	–
	378.4	30.8	0.15	224.0	46.6	0.28	837.5	20.3	0.10	627.9	32.4	0.18
DistMult [32]	–	–	–	–	57.7	0.35	–	–	–	–	–	–
	313.0	45.0	0.21	159.6	71.4	0.42	1063.7	12.1	0.06	852.7	25.6	0.13
	269.6	50.6	0.25	112.3	83.3	0.65	708.8	18.0	0.08	494.0	35.2	0.18
	279.0	50.0	0.26	120.4	84.2	0.71	708.4	22.1	0.12	495.4	37.6	0.22
ComplEx [28]	–	–	0.24	–	84.0	0.69	–	–	–	–	–	–
	350.3	43.8	0.21	192.3	73.0	0.52	1150.3	8.4	0.04	936.1	20.9	0.11
	266.2	48.5	0.23	106.0	82.6	0.68	630.7	18.7	0.08	415.7	36.9	0.18
	292.7	49.2	0.25	132.9	82.5	0.72	708.5	21.1	0.11	495.1	36.7	0.21
ANALOGY [18]	–	–	0.25	–	85.4	0.73	–	–	–	–	–	–
	279.4	50.5	0.26	120.9	84.3	0.72	715.9	21.9	0.12	502.7	37.4	0.21
ConvE [6]	–	–	–	64.0	87.3	0.75	–	–	–	246.0	49.1	0.32
	190.8	52.5	0.27	51.2	85.1	0.69	489.3	28.4	0.15	277.0	48.5	0.31
NLFeat [27]	–	–	–	–	87.0	0.82	–	–	–	–	34.7	0.23
NeuralLp [33]	–	–	–	–	83.7	0.76	–	–	–	–	36.2	0.24
AMIE [9]	–	–	–	–	88.05	–	–	–	–	–	47.67	–

• Published results • OpenKE (https://github.com/thunlp/OpenKE) • ComplEx (https://github.com/ttrouill/complex) • ANALOGY (https://github.com/quark0/ANALOGY) • ConvE (https://github.com/TimDettmers/ConvE)

Table 6: Most test triples in FB15k on which various models outperformed TransE have reverse and duplicate triples in training set

Model	FMR$\downarrow$	FHits@10$\uparrow$	FHits@1$\uparrow$	FMRR$\uparrow$
DistMult	77.03 %	79.94 %	81.31 %	72.71 %
ComplEx	77.08 %	79.67 %	81.29 %	78.46 %
ANALOGY	77.06 %	79.74 %	81.34 %	78.42 %
ConvE	75.18 %	78.14 %	79.81 %	75.22 %

Table 7: Number of relations on which each model is the most accurate on FB15k-237

Model	MR$\downarrow$	H10$\uparrow$	H1$\uparrow$	MRR$\uparrow$	FMR$\downarrow$	FH10$\uparrow$	FH1$\uparrow$	FMRR$\uparrow$
TransE	140	114	103	101	100	50	43	38
DistMult	13	37	37	11	23	41	29	19
ComplEx	6	34	27	11	15	36	28	14
ANALOGY	10	38	33	17	16	35	30	24
ConvE	60	122	94	90	81	162	137	139

5 RE-EVALUATING LINK PREDICTIO N METHODS

5.1 FB15k-237

Among the first to document the data redundancy in FB15k, Toutanova and Chen [27] created FB15k-237 from FB15k by removing such redundancy. They first limited the set of relations in FB15k to the most frequent 401 relations. Their approach of removing redundancy is essentially the same as the equations for detecting duplicate and reverse duplicate relations in Section 4.2.2, likely with different thresholds. For each pair of such redundant relations, only one was kept. This process decreased the number of relations to 237. They also removed all triples in test and validation sets whose entity pairs were directly linked in the training set. 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 absolutely accurate reverse relation information encoded by *reverse_property*. Finally, it does not identify Cartesian product relations. Nevertheless, we used both FB15k-237 and FB15k in our experiments. This allows us to corroborate the experiment results with those from a number of recent studies.

5.2 Experiment Setup

Our experiments were conducted on an Intel-based machine that is composed of an Intel Xeon E5-2695 processor running at 2.1GHz, Nvidia Geforce GTX1080 GPU, and 251 GB RAM. The experiments used source codes of various methods from several places, including the OpenKE [11] repository[9] which covers implementations of TransE, TransH, TransR, TransD, RESCAL, DistMult, and ComplEx, as well as the

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

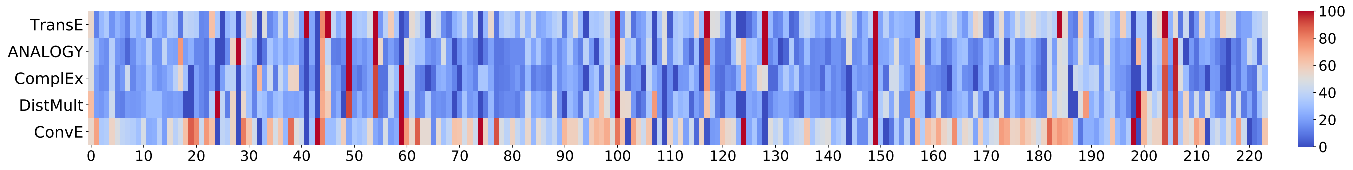
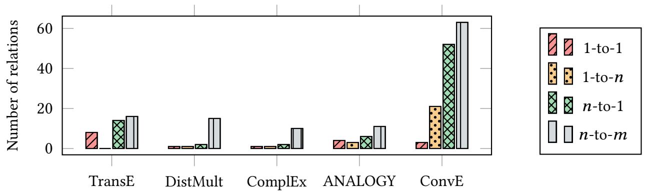
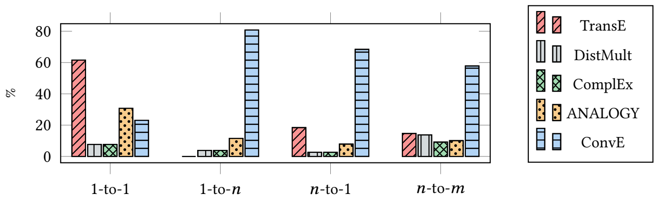


Figure 5: Percentage of triples on which each method outperforms others, separately for each relation



(a) Categorizing the relations on which each method has the best result



(b) Break-down of methods achieving best performance on each type of relations

Figure 6: Best performing models on FB15k-237 by FMRR↑

source codes released for ANALOGY[10] (which covers DistMult and ComplEx too), ComplEx[11] (which also covers DistMult), and ConvE[12]. All source codes and data used in our experiments as well as results are available at https://github.com/Knowledge-Graph/KnowledgeGraph-completion. The different models used in our experiments have different hyperparameters. We used the same settings that were used by the developers of the source codes for both FB15k and FB15k-237. The details can be found in the source codes.

5.3 Results

Table 5 displays the results of link prediction on FB15k and FB15k-237 for all compared methods, using both raw and filtered metrics explained in Section 3.2. For each method, the table shows the original publication where it comes from. The values in black color are the results listed in the original publication, while a hyphen under a measure indicates that the original publication did not list the corresponding value.

[10]https://github.com/quark0/ANALOGY
[11]https://github.com/ttrouill/complex
[12]https://github.com/TimDettmers/ConvE

The values in other colors are obtained through our experiments using various source codes: blue for results obtained by implementations from the OpenKE repository [11]; green for results from codes provided by Trouillon et al. [28] which introduced ComplEx and also supplied an implementation of DistMult; red for Liu et al. [18] which introduced ANALOGY and also provided implementations of DistMult and ComplEx; and orange for code from Dettmers et al. [6] which introduced ConvE. For ComplEx and DistMult, we have presented the results from several different implementations which have quite different performance values. Similarly many of the implementations in our experiments produced better results than those reported in the original publications. We attribute these differences to different dimensionalities of vectors representing entities and different optimization methods that were employed in the implementations.

Below we summarize and explain the results in Table 5. (1) The overall observation is that the performance of all methods worsens considerably after removal of reverse relations, as demonstrated by results on FB15k-237. For instance, the FMRR↑ of ConvE—one of the best performing methods under many of the metrics—has decreased from 0.69 (on FB15k) to 0.31 (on FB15k-237), and its FMR also became much worse, from 51.2 to 277. This result verifies that embedding-based methods may only perform well on reverse relations. However, a straightforward approach based on detection of reverse relations can achieve even better accuracy, as explained in Section 4.2.1.

(2) Many successors of TransE (e.g., DistMult, ComplEx, and ANALOGY, ConvE) were supposed to significantly outperform it. This was indeed verified by our experiment results on FB15k. However, by MR↓, Hits@10↑ and FMR↓, they all have worse results than TransE on FB15k-237, except for almost equal performance in a couple of cases. By FMRR↑, their margin over TransE became much smaller on FB15k-237: TransE with an FMRR↑ of 0.26, in comparison with DistMult (0.22), ComplEx (0.21), ANALOGY (0.21), and ConvE (0.31). We hypothesize that these models only improved the results on reverse and duplicate triples and hence, after removing those triples, they do not exhibit clear advantage. This hypothesis can be verified by our finding that most of the test triples on which these models outperformed TransE have reverse or duplicate triples in training set, as shown in Table 6.

3) We have further analyzed how the most-accurate models perform on FB15k-237. There are 224 distinct relations in the test set of FB15k-237. Table 7 shows, for each metric and each model, the number of distinct test relations on which the model is the most accurate.[13] Furthermore, the heatmap in Figure 5 shows, for each of the 224 relations, the percentage of test triples on which each model has the best performance (i.e., the highest rank) in comparison with other models, using `FMRR`$^{\uparrow}$ as the performance measure. Both Table 7 and Figure 5 show that TransE clearly outperformed other models, with the exception of ConvE.

4) To better understand the strengths and weaknesses of each model, we further brake down the numbers in the column `FMRR`$^{\uparrow}$of Table 7. The relations on which a method has the best result are categorized into 4 different classes 1-to-1, 1-to-*n*, *n*-to-1 and *n*-to-*m* based on the average number of heads per tail and tails per head. An average number less than 1.5 is marked as "1" and "*n*" otherwise [4]. Figure 6a shows the break-down of relations on which each method has the best result. For example, among the 38 relations on which TransE has outperformed others, 8 are 1-to-1, 0 are 1-to-*n*, 14 are *n*-to-1, and 16 are *n*-to-*m* relations. Among the 224 distinct relations in the test set of FB15k-237, 5.8% are 1-to-1, 11.6% are 1-to-*n*, 33.9% are *n*-1, and 48.7% are *n*-to-*m* relations. The number of test triples belonging to these 4 types of relations are 192, 1293, 4285 and 14696, respectively. Figure 6b shows the break-down of best performing models within each type of relations. It shows that ConvE has the best result on 1-to-*n*, *n*-to-1 and *n*-to-*m* types, however it has shortcomings in dealing with 1-to-1 relations. Since the number of 1-to-1 relations in test set of FB15k-237 is much less than other relations, this weakness of ConvE doesn't affect its overall result that much, but if a dataset is dominated by 1-to-1 relations it will encounter problems for conducting link prediction. TransE has the best performance on 1-to-1 relations and it outperforms multiplicative models DistMult, ComplEx and ANALOGY on *n*-to-1 relations while it has similar results on *n*-to-*m* relations, this means that TransE actually better performs on some of the more complicated relations as well.

6 COMPARISON WITH RULE-BASED METHODS

Our experiment results in Table 5 show that the performance of rule-mining system AMIE [9], similar to that of embedding models, substantially degenerates in the absence of data redundancy. Table 5 also shows similar results from NLFeat [27] and NeuralLP [33]. NLFeat utilizes simple observed features of entities and relations such as direct links between candidate entity pairs. NeuralLP presents a completely differentiable system which learns first-order logical rules from knowledge graphs.

AMIE rules were generated by applying the AMIE+ [14] code released by the authors of [8] on the training set of FB15k or FB15k-237. For any link prediction task (h, *r*, ?) or (?, *r*, t), all the rules that have relation *r* in the rule head are employed. The instantiations of these rules are used to generate the ranked list of results. For example, if the test case is (Bill Gates, *place_of_birth*, ?), the following rule will be employed:

$$(?a, \textit{places\_lived.location}, ?b) \Rightarrow \quad (?a, \textit{place\_of\_birth}, ?b)$$

Then the instantiations of variable ?b are used to find the list of predictions. Several rules may generate the same answer entity. It is imperative to combine the confidence of those rule in some way in order to score the answer entities. We ranked the answer entities by the maximum confidence of the rules instantiating them and broke ties by the number of applicable rules [19]. `FHits@10`$^{\uparrow}$ and `FHits@1`$^{\uparrow}$ are used to calculate the performance of link prediction.[15]

Table 8 compares AMIE with TransE, DistMult, ComplEx, ANALOGY, and ConvE using `FHits@10`$^{\uparrow}$ on FB15k-237. More specifically, it lists the number of relations on which each model is the most accurate. (Table 7 shows different numbers since AMIE was not included in its comparisons.) The table shows AMIE and ConvE significantly outperformed other models. Figure 8 displays the results of head-to-head comparisons between AMIE and other models, under each type of

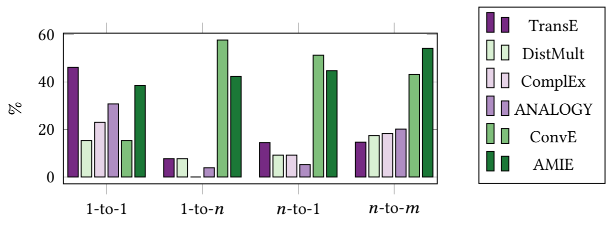


Figure 7: Break-down of methods achieving best performance on each type of relations, AMIE included, FB15k-237, `FHits@10`$^{\uparrow}$

Table 8: Number of relations on which each model is the most accurate on FB15k-237, AMIE included

	TransE	DistMult	ComplEx	ANALOGY	ConvE	AMIE
`FH10`$^{\uparrow}$	35	30	30	31	103	109

[13]We rounded accuracy measures to the nearest hundredth. Since there are ties in best performing models, the summation of each column can be greater than 224.

[14]https://bit.ly/2Vq2OIB

[15]Measures such as `FMR`$^{\downarrow}$ and `FMRR`$^{\uparrow}$ are not always applicable for AMIE. This is because it is possible that no rule can instantiate an answer entity and thus we cannot obtain its rank.

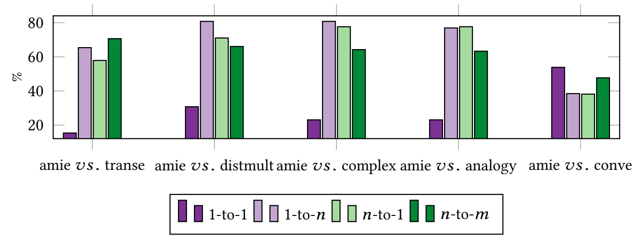


Figure 8: Head-to-head comparisons of AMIE with other models on FB15k-237 using FHits@10$^{\uparrow}$(names of models in lower case to save space)

relations. It shows that AMIE outperformed DistMult, ComplEx and ANALOGY on 1-to-n and n-to-1 relations, ConvE on 1-to-1 relation, and TransE on n-to-m relations. For instance, it is more accurate than TransE on more than 70% of the n-to-m relations.

We further computed the FHits@10$^{\uparrow}$ for head and tail predictions separately for each relation type, as in Table 9. The first and second best results are shown with boldface and underline, respectively. All the methods performed better at predicting the "1" side of 1-to-n and n-to-1 relations, except for ConvE which is better at predicting the side n. TransE is the best performing model on 1-to-1 relations. Both AMIE and TransE are the best ones in predicting the side 1 of 1-to-n and n-to-1 relation types. AMIE's performance is weaker than ConvE in predicting the side n of 1-to-n and n-to-1 relations. However, it is more accurate than other embedding embedding methods in these cases. AMIE and CovE have better results than other models on n-to-m relations. Overall, AMIE has the first or second best accuracy in almost all the cases.

Table 10 compares various methods using FHits@1$^{\uparrow}$, which is a more demanding measure than FHits@10$^{\uparrow}$, since it only considers whether a model ranks a correct answer at the very top. The different colors indicate the implementations used, as explained for Table 5. The purple color is for the results of our approach of utilizing AMIE rules in link prediction, as explained earlier. The results show that AMIE clearly outperformed all other strong-performing models on both FB15k and FB15k-237, with the only challenger in ConvE on FB15k-237.

7 CONCLUSIONS

In this paper, we did an extensive investigation of data redundancy available in the widely-used benchmark dataset FB15k and showed its impact on the performance of several embedding models. Our experiments showed that in the absence of the straightforward prediction tasks, the performance of embedding models will degenerate significantly. We also identified Cartesian product relations in FB15k for which the link prediction can be easily done by the Cartesian product of the head and tail entities. It was also showed how some simple approaches can achieve better performance than embedding models in these cases. We believe the existence of this kind of data will cause the overestimation of the models. Moreover, they present unrealistic cases of link prediction nonexistence in the real world. We also showed the inadequacy of evaluation metrics when a method can generate correct predictions not available in the labeled dataset. All the aforementioned problems show the necessity of a new benchmark dataset and also some more adequate evaluation metrics for the assessment of the knowledge graph completion methods.

REFERENCES

[1] Farahnaz Akrami, Lingbing Guo, Wei Hu, and Chengkai Li. 2018. Re-evaluating Embedding-Based Knowledge Graph Completion Methods. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1779–1782.
[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.
[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*. 1247–1250.
[4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*. 2787–2795.
[5] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.
[6] Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *AAAI*.
[7] Jeffrey Scott Eder. 2012. Knowledge graph based search system. (June 21 2012). US Patent App. 13/404,109.
[8] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2015. Fast Rule Mining in Ontological Knowledge Bases with AMIE$$+$$+. *The VLDB Journal* 24, 6 (Dec. 2015), 707–730. https://doi.org/10.1007/s00778-015-0394-1
[9] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *WWW*. ACM, 413–422.
[10] Big Dats Interagency Working Group. 2018. Open Knowledge Network: Summary of the Big Data IWG Workshop of October 2017. (2018).
[11] Xu Han, Shulin Cao, Xin Lv, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. OpenKE: An Open Toolkit for Knowledge Embedding. In *EMNLP: System Demonstrations*. 139–144.
[12] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F Patel-Schneider, and Sebastian Rudolph. 2009. OWL 2 web ontology language primer. *W3C recommendation* 27, 1 (2009), 123.
[13] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *ACL*. 687–696.
[14] Timothee Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical Tensor Decomposition for Knowledge Base Completion. In *ICML (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, StockholmsmÃď ssan, Stockholm Sweden, 2863–2872.

Table 9: `FHits@10`$^\uparrow$ by category of relations on FB15k-237

	1-to-1		1-to-*n*		*n*-to-1		*n*-to-*m*	
Model	Left FH10$^\uparrow$	Right FH10$^\uparrow$	Left FH10$^\uparrow$	Right FH10$^\uparrow$	Left FH10$^\uparrow$	Right FH10$^\uparrow$	Left FH10$^\uparrow$	Right FH10$^\uparrow$
TransE	**51.04**	**52.6**	**54.22**	8.28	10.04	80.05	37.35	50.78
ANALOGY	44.79	43.75	38.05	6.11	9.59	71.02	31.79	43.17
DistMult	42.19	41.15	40.84	7.73	9.45	71.09	30.98	43.6
ComplEx	43.75	43.75	36.58	7.42	9.24	68.78	31.03	42.15
ConvE	26.56	27.6	11.29	**59.63**	**85.34**	14.38	**56.73**	41.2
AMIE	43.23	44.27	46.17	13.07	18.2	**80.47**	42	**55**

Table 10: `FHits@1`$^\uparrow$ results

	analogy	**complex**	**distmult**	**conve**	**transe**	**amie**
FB15k	64.6	59.9	-	67	-	-
	66.1	66.06	62.88	60.68	26.88	75.13
FB15k-237	-	-	-	23.9	-	-
	13.4	13.0	13.95	22.13	18.22	22.49

[15] Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning* 81, 1 (2010), 53–67.

[16] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 705–714. https://doi.org/10.18653/v1/D15-1082

[17] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion.. In *AAAI*. 2181–2187.

[18] Hanxiao Liu, Yuexin Wu, and Yiming Yang. 2017. Analogical Inference for Multi-relational Embeddings. In *ICML*. 2168–2178.

[19] Christian Meilicke, Manuel Fink, Yanjie Wang, Daniel Ruffinelli, Rainer Gemulla, and Heiner Stuckenschmidt. 2018. Fine-grained evaluation of rule-and embedding-based systems for knowledge graph completion. In *International Semantic Web Conference*. Springer, 3–20.

[20] Stephen Muggleton and Luc De Raedt. 1994. Inductive logic programming: Theory and methods. *The Journal of Logic Programming* 19 (1994), 629–679.

[21] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[22] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *ICML*. 809–816.

[23] Raymond Reiter. 1981. On closed world data bases. In *Readings in artificial intelligence*. Elsevier, 119–140.

[24] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Alexander Sontag. 2017. Learning a Health Knowledge Graph from Electronic Medical Records. *Scientific Reports* 7 (12 2017). https://doi.org/10.1038/s41598-017-05778-z

[25] Baoxu Shi and Tim Weninger. 2017. ProjE: Embedding projection for knowledge graph completion. In *AAAI*.

[26] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

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

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

[29] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI*. 1112–1119.

[30] Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting Language and Knowledge Bases with Embedding Models for Relation Extraction. In *EMNLP*. Association for Computational Linguistics, 1366–1371. https://www.aclweb.org/anthology/D13-1136

[31] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question Answering on Freebase via Relation Extraction and Textual Evidence. In *ACL*. Association for Computational Linguistics, Berlin, Germany, 2326–2336. https://doi.org/10.18653/v1/P16-1220

[32] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

[33] Fan Yang, Zhilin Yang, and William W Cohen. 2017. Differentiable learning of logical rules for knowledge base reasoning. In *NIPS*. 2316–2325.

[34] Xuchen Yao and Benjamin Van Durme. 2014. Information extraction over structured data: Question answering with freebase. In *ACL*, Vol. 1. 956–966.

[35] Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base. 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)*. ACL, Beijing, China, 1321–1331. https://doi.org/10.3115/v1/P15-1128

[36] Wen Zhang, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen. 2019. Interaction Embeddings for Prediction and Explanation in Knowledge Graphs. In *WSDM*. ACM, 96–104.