Predicting Instance Type Assertions in Knowledge Graphs Using Stochastic Neural Networks

Tobias Weller
Data and Web Science Group, University of Mannheim
Mannheim, Germany
tobi@informatik.uni-mannheim.de

Maribel Acosta
Ruhr University Bochum
Bochum, Germany
maribel.acosta@rub.de

ABSTRACT

Instance type information is particularly relevant to perform reasoning and obtain further information about entities in knowledge graphs (KGs). However, during automated or pay-as-you-go KG construction processes, instance types might be incomplete or missing in some entities. Previous work focused mostly on representing entities and relations as embeddings based on the statements in the KG. While the computed embeddings encode semantic descriptions and preserve the relationship between the entities, the focus of these methods is often not on predicting schema knowledge, but on predicting missing statements between instances for completing the KG. To fill this gap, we propose an approach that first learns a KG representation suitable for predicting instance type assertions. Then, our solution implements a neural network architecture to predict instance types based on the learned representation. Results show that our representations of entities are much more separable with respect to their associations with classes in the KG, compared to existing methods. For this reason, the performance of predicting instance types on a large number of KGs, in particular on cross-domain KGs with a high variety of classes, is significantly better in terms of F1-score than previous work.

CCS CONCEPTS

• **Applied computing**; • **Information systems** → *Data mining*; *Data analytics*; • **Computing methodologies** → *Machine learning*;

KEYWORDS

Entity Type Prediction, Entity Classification, Knowledge Graphs, Stochastic Networks

ACM Reference Format:
Tobias Weller and Maribel Acosta. 2021. Predicting Instance Type Assertions in Knowledge Graphs Using Stochastic Neural Networks. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3459637.3482377

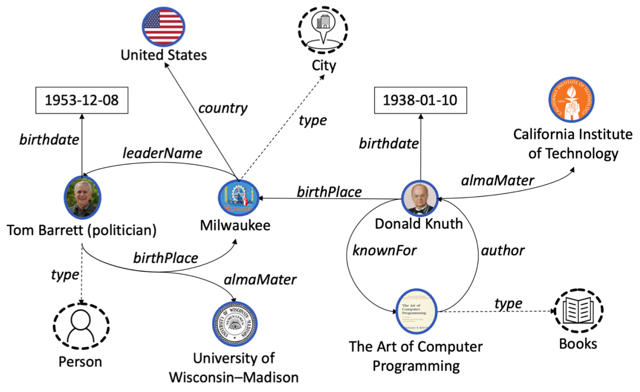


Figure 1: Motivating example. Entities from the same classes use the same predicates in their descriptions. We leverage this to predict missing type information for Donald Knuth.

1 INTRODUCTION

Knowledge graphs (KGs) use schema assertions or axioms to model concepts and relations that serve as the foundation for describing entities and their connections in the KG. These schema assertions typically define the meaning and associations between classes, relations, and class memberships. Furthermore, reasoners can use schema axioms to logically deduce additional facts from the KG or to detect inconsistencies between statements encoded in the KG. Therefore, having complete schema assertions in KGs is key to fully exploit the power of semantics in graphs, yet, these assertions may be incomplete due to several reasons. For example, KGs created in a pay-as-you-go fashion can suffer from this kind of incompleteness since new instances, classes, and properties are sometimes added to the KG with partial information that is available at the time of insertion. Similar situation may occur when building KGs from unstructured or incomplete sources. Furthermore, some schema assertions require domain-specific knowledge provided by experts, but manually completing them is not a feasible solution for KGs with a large number of classes.

Despite the relevance of schema assertions, recent advances in KG embeddings that have been proposed are tailored to completing instance-level statements [5, 23, 32], i.e., enhancing the descriptions of entities or instances in the KG. To address the incompleteness of schema assertions, other solutions have been proposed that focus on the problem of instance type prediction. For this, either specific approaches [24] have been devised or some of the aforementioned solutions [21] have also been applied. Nonetheless, the recent work by Jain et al. [12] shows that state-of-the-art approaches are not effective for predicting instance types, in particular, when considering specialized classes in class hierarchies.

In this paper, we introduce an approach called Ridle that learns representations of entities, tailored to predict instance type assertions. The hypothesis of this work is that instances from the same class are described with the same predicates in the KG which, in turn, allows for predicting the types of similar instances. To illustrate this, consider the KG from Figure 1, where the entity *Donald Knuth* is not associated with any class. Yet, the description of *Donald Knuth* has some relations in common with the entity *Tom Barret* of type *Person*, e.g., *birthDate*, *birthPlace* and *almaMater*. In contrast, *Donald Knuth* does not have relations in common with the entity *Milwaukee* of type *Country*. Based on this information, *Donald Knuth* might also belong to the class *Person*. Note that the objects of the predicates are different for *Donald Knuth* and *Tom Barret*, still, by just looking at the relations used in their descriptions we can predict the class affiliation for *Donald Knuth*. This example shows that the occurrences of instances and relations can be an effective predictor for instance types. The underlying idea of Ridle is to exploit these characteristics and compute a target distribution over the occurrence of relations to learn a compressed representation of the KG, in which this distribution is latently encoded. We use a stochastic factorization model, namely Restricted Boltzmann Machine (RBM), to learn this target distribution. In a downstream stage, we use these representations combined with information about existing schema statements to predict missing instance type assertions using a supervised neural network architecture. While most methods are inherently transductive and therefore do not efficiently generalize to unseen entities, our method is inductive, allowing it to generate representations and predictions for schema assertions, even in the presence of unknown entities.

We conducted an extensive evaluation on 20 KGs, including 4 cross-domain and 16 category-specific KGs, allowing for a detailed analysis of the features of the proposed approach. We compare Ridle with state-of-the-art methods with different learning paradigms [5, 23, 24, 28, 32]. We report on F1-score, as similar studies have done before [24, 28]. The results show that Ridle achieves on average a better performance in predicting instance type assertions, particularly in cross-domain knowledge graphs, than current state-of-the-art methods, thus providing a new baseline for predicting schema assertions.

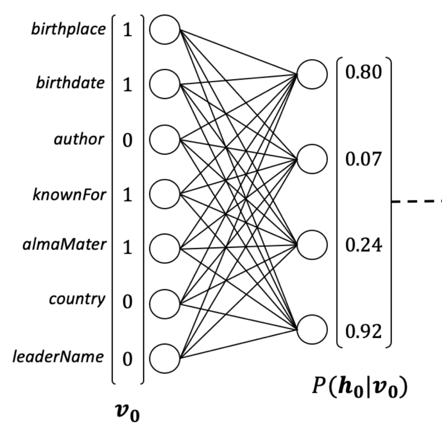
2 RELATED WORK

Our work focuses on learning a representation for entities in knowledge graphs for predicting instance type assertions. SDType [24] is a heuristic link-based type inference mechanism focusing on instance type prediction. As pointed out in related works [13, 26] traditional reasoning methods tend to struggle with noisy data, false or unforeseen schemas. The heuristic method SDType uses a statistical distribution of the actually used scheme in the computation and thus making it more robust. This characteristic of robustness of statistical distributions is used in this paper as well, in order to deal with noise in the data. In contrast to SDType, we use a stochastic factorization model to learn representations of entities on the use of relations of instances. Based on the representations, we learn a model for predicting instance types assertions. RDF2Vec [28] is an adaptation of the language model Word2Vec [19, 20] by using random walks and Weisfeiler-Lehman Subtree RDF Graph Kernels to create sequences of nodes that are passed to Word2Vec for learning low-dimensional numerical representations of entities and relations. Despite the promising results of previous studies on instance type prediction using RDF2Vec [4, 14, 31], we consider RDF2Vec especially suitable for measuring the semantic similarity of entities [6, 28, 29], as well as the entity alignment between knowledge graphs [1], due to the use of Word2Vec and thus the ability to represent relationships between entities accurately. Yet, using a random walk approach, the performance of RDF2Vec varies significantly depending on the topology of the knowledge graphs and the random walk strategy. In contrast, we learn features based on a target distribution and, thus, it is not negatively affected by the topology of the knowledge graph or the chosen random walk strategy. We prefer the low-dimensional representation of entities as learned by RDF2Vec, but consider that the complete triple is not relevant for the prediction of schema assertions and, thus, consider RDF2Vec to be too complex for this task. In a further related work on instance type prediction, entities were classified exclusively on the basis of the relations used [18]. For the sole use of the relation of entities, the results are very promising. However, this approach neglects the semantic relationship, as well as latent features expressed in the correlations between the graph relations. In contrast, we want to provide representations of entities by using a target distribution over the usage of relations of entities and thus the identification of latent features based on relation usage of entities.

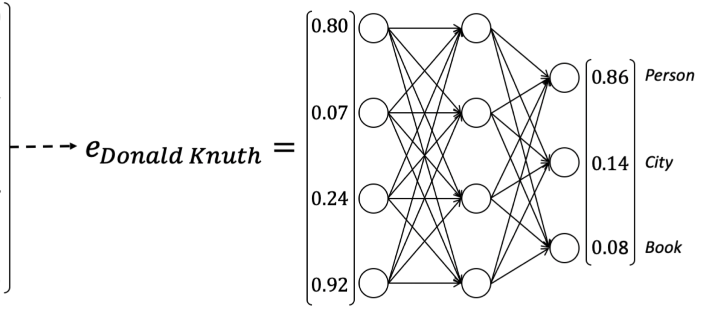
In this work, we use Restricted Boltzman Machines (RBMs) to learn a target distribution over the usage of relations of entities allowing to use the hidden layer as latent features for representation of entities. RBMs have been applied in the past especially for dimensionality reduction [11], learning and reconstructing sparse representations of the input [27], collaborative filtering [2, 30] and link prediction [16, 33]. Although first attempts were made to apply RBMs for feature learning [15, 22], it has not been yet applied for learning features in knowledge graphs. In addition, we want to mention that in this context, RBMs share a similar idea as auto-encoders, but use stochastic units. Instead of reconstructing the exact input as done with auto-encoders, we are trying to identify the distribution of the used relations to determine latent features, which we will use as representation of entities.

3 PROPOSED APPROACH: RIDLE

For the downstream task of predicting instance type assertions, we need a semantically meaningful representation of entities. Our hypothesis is that the relations used by the instances are most relevant for their classification. For example, considering the knowledge graph in Figure 1, we notice that *Donald Knuth* has among others the relations *birthplace* and *birthdate*. Knowing only these relations, already gives a hint, that this instance is most likely of type person, as both relations are in general used only by entities of the class *Person*. The complete statements, i.e., in which place exactly *Donald Knuth* was born or at which date, is less relevant to predict the instance type. Likewise, we notice in this example, that certain relations are more likely to co-occur in the context of certain types of instances than others and, thus, the distribution of used relations can be used to predict class memberships.



(a) Vector representation of the entity *Donald Knuth*. Learning representation for target distribution of the used relations using a RBM.



(b) Using latent features to perform instance type prediction with a supervised neural network.

Figure 2: Overview of Ridle: proposed solution for instance type prediction. Entities are first encoded as a binary vector, representing the usage of relations. An RBM is used to learn a target distribution over the used relations. Afterwards, the compressed vector representations of the RBM hidden layer $P(\mathbf{h}_0|\mathbf{v}_0)$ are used as the representation of the entities to train a supervised 2-layer neural network for instance type prediction. Labels for training are obtained from the knowledge graph.

We devise an approach to exploit the aforementioned two characteristics about associations between instances and relations in knowledge graphs. Our proposed approach, Ridle (Relation-Instance Distribution Learning), is able to learn the distribution of relations in the knowledge graph which, in turns, allow for predicting instance types. Figure 2 depicts the two components of Ridle: (a) a representation model based on instance-relation occurrences in the knowledge graph, (b) a neural network for predicting instance types based on the learned entity representations. In the following, we describe each of the two components.

3.1 Learning Instance-Relation Representation

The goal of the first component of Ridle is to learn a representation of a given knowledge graph (KG). We define a KG as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{L}, \mathcal{T})$, where the pair-wise disjoint sets $\mathcal{E}$, $\mathcal{R}$, $\mathcal{L}$, and $\mathcal{T}$ correspond to the set of entities, relations, literals, and types or classes, respectively. A statement in $\mathcal{G}$ is modelled as a triple (s, p, o), with $s \in \mathcal{E} \cup \mathcal{R} \cup \mathcal{T}$, $p \in \mathcal{R}$, and $o \in \mathcal{E} \cup \mathcal{R} \cup \mathcal{L} \cup \mathcal{T}$. We denote the predicate *rdf:type*, defined in the RDF [34] and RDFS [9] specifications, as the relations $type \in \mathcal{R}$. In addition, we denote $\mathcal{R}^- \subset \mathcal{R}$ the set of domain-specific relations, which excludes *type*, and further predicates from meta-models or general-purpose ontologies.

To learn an effective instance-relation representation, our approach encodes each entity of the knowledge graph $\mathcal{G}$ based on its properties. Ridle focuses on domain-specific properties $\mathcal{R}^-$, which allows for uncovering entities with similar semantic descriptions. General-purpose predicates are left out from the entity representation as they might hinder the learning process for two reasons: (i) these predicates alone do not provide information that allow for distinguishing entities from different classes,[1] and (ii) these predicates typically occur in the majority of entities. Therefore, Ridle models an entity $s \in \mathcal{E}$ as a binary $|\mathcal{R}^-|$-vector $\mathbf{v}$, where:

$$\mathbf{v}[i] = \begin{cases} 1 & \text{if } (s, r_i, o) \in \mathcal{G}, \text{ for some } o \in \mathcal{E} \cup \mathcal{R}^- \cup \mathcal{L} \cup \mathcal{T} \\ 0 & \text{otherwise} \end{cases}$$

Note that we choose a binary representation as the frequency of the used relations is irrelevant for the downstream classification of instance types. Figure 2a shows an example of this representation of the instance *Donald Knuth* encoded as the vector $\mathbf{v}$.

Then, the binary vector $\mathbf{v}$ serves as input for a Restricted Boltzmann Machine (RBM), for which the relation distribution in a knowledge graph $\mathcal{G}$ is learned. An RBM is a generative model to simulate input distributions of binary data, consisting of one visible layer, denoted as $\mathbf{v}$, and one hidden layer $\mathbf{h}$ with size h. In RBMs, there is no explicit output layer, since the unsupervised model tries to approximate the distribution of the input data. The distribution is used to compute latent features in the hidden layer, which can be seen as a compressed representation of the input data.

In RBMs, the weights or parameters between the layers represent the impact of the individual input nodes on the latent features in the hidden layer. The learning of the parameters in our approach is done by means of Gibbs-sampling. Therefore, in the following notations, the indexing is used to indicate the step of the Gibb-sampling process. The input $\mathbf{v}_0$ is multiplied with a weight matrix $\mathbf{W}$ and added with a bias $\mathbf{a}$. Similar to a feed-forward neural network, a sigmoid activation function σ is used to compute the hidden values, denoted as $P(\mathbf{h}_0|\mathbf{v}_0)$.

[1]Note that the object of triples are not considered in our representation. Therefore, encoding that e.g. the predicate *type* occurs in an entity is not informative to predict the class to which the entity belongs to.

$$P(\mathbf{h}_0|\mathbf{v}_0) = \sigma(\mathbf{W}\mathbf{v}_0 + \mathbf{a}) \tag{1}$$

Afterwards, samples based on the computation of the hidden layer $P(\mathbf{h}_0|\mathbf{v}_0)$ are taken from a Bernoulli distribution to compute the hidden state $\mathbf{h}_0$.

$$\mathbf{h}_0 \sim \text{Bernoulli}(P(\mathbf{h}_0|\mathbf{v}_0)) \tag{2}$$

Introducing a stochastic distribution function extends the neurons to stochastic neurons. While a high $P(\mathbf{h}_0|\mathbf{v}_0)$ results in a high probability of having a positive hidden state $\mathbf{h}_0$, a low probability results in zero output. Based on the hidden state $\mathbf{h}_0$, the input data $\mathbf{v}_0$ will be reconstructed by using the hidden state $\mathbf{h}_0$ as input and backwarded in the neural network. Hereby, the hidden state $\mathbf{h}_0$ is multiplied with the same weight matrix $\mathbf{W}$ as it was computed, but transposed, and a bias value $\mathbf{b}$ added. Afterwards, the sigmoid activation function is applied to this weighted sum. The resulting vector, denoted as $P(\mathbf{v}_1|\mathbf{h}_0)$, in the visible layer can be seen as an approximation of the original input.

$$P(\mathbf{v}_1|\mathbf{h}_0) = \sigma(\mathbf{W}^T\mathbf{h}_0 + \mathbf{b}) \tag{3}$$

RBMs are energy-based probabilistic models, using a probability distribution through an energy function to measure the quality, similar to cost functions of machine learning models. The hidden layer serves as latent variable to increase the expressiveness of the model, therefore the following energy-based probabilistic function (Gibbs distribution) specifies that a certain state $\mathbf{v}$ can be observed:

$$P(\mathbf{v}) = \frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{4}$$

where Z is the sum from all possible states and called the normalizing factor:

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{5}$$

With Eq. 4, we can conclude that a low energy $E(\mathbf{v},\mathbf{h})$ leads to a high probability $P(\mathbf{v})$, while a high energy leads to a low probability $P(\mathbf{v})$. In order to increase the probability $P(\mathbf{v})$ we, therefore, have to minimize the energy function $E(\mathbf{v},\mathbf{h})$. The energy function $E(\mathbf{v},\mathbf{h})$ for an RBM with its input $\mathbf{v}$ and hidden state $\mathbf{h}$ is the following:

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{v}^T\mathbf{W}\mathbf{h} - \mathbf{a}^T\mathbf{v} - \mathbf{b}^T\mathbf{h} \tag{6}$$

The aim is to approximate the distribution, therefore the difference in the distribution of the input data $\mathbf{v}_0$ and the reconstructed input data $P(\mathbf{v}_1|\mathbf{h}_0)$ should be minimized. Thus, the energy functions described in Eq. 6 of these two distributions are to be aligned. In previous work it has been shown that contrastive divergence with Gibbs-sampling, as an approximation of the log-likelihood gradient, is a very efficient method to learn the parameters of the RBM to compute the target distribution [8]. The number how often the Gibbs chain is applied for a single sample is denoted by the parameter k. In related studies as well as in preliminary conducted experiments, it has been shown that $k = 1$ already achieves sufficient results in the approximation of the target distribution. Therefore, similar to related work, we use k-step contrastive divergence to learn the parameters of the RBM. The gradient w.r.t. log-likelihood for one sample $\mathbf{v}_0$ is then approximated by the following formula [3]:

$$\begin{aligned} CD(\theta,\mathbf{v}_0) = &-\sum_{\mathbf{h}} P(\mathbf{h}_0|\mathbf{v}_0)\frac{\partial E(\mathbf{v}_0,\mathbf{h}_0)}{\partial\theta} \\ &+\sum_{\mathbf{h}} P(\mathbf{h}_0|\mathbf{v}_1)\frac{\partial E(\mathbf{v}_1,\mathbf{h}_0)}{\partial\theta} \end{aligned} \tag{7}$$

Based on Eq. 7, we get the following updates of the parameters:

$$\Delta\mathbf{W} = P(\mathbf{h}_0 = 1|\mathbf{v}_0)\cdot\mathbf{v}_0 - P(\mathbf{h}_0 = 1|\mathbf{v}_1)\cdot\mathbf{v}_1 \tag{8}$$

$$\Delta\mathbf{a} = \mathbf{v}_0 - \mathbf{v}_1 \tag{9}$$

$$\Delta\mathbf{b} = P(\mathbf{h}_0 = 1|\mathbf{v}_0) - P(\mathbf{h}_0 = 1|\mathbf{v}_1) \tag{10}$$

Using the update rules the parameters converge so that the distribution of the reconstructions $\mathbf{v}_1$ corresponds to the distribution of the input $\mathbf{v}_0$.

3.2 Predicting Instance Types

We model the problem of instance type prediction as a multi-label classification problem, since entities can belong to several classes in $\mathcal{G}$. To perform the predictions, Ridle exploits the latent features of entities learned with the RBM, which are fed into a supervised learning algorithm. For every entity $s \in \mathcal{E}$ modelled as $\mathbf{v}_0$ in the RBM, Ridle obtains the learned representation $\mathbf{e}_s = P(\mathbf{v}_0|\mathbf{h}_0)$ (c.f. Figure 2b). I.e., instances in Ridle are represented as the learned probabilities of activating a hidden state. This representation was chosen over the binary vector $\mathbf{h}_0$ obtained after the Bernoulli sampling, as $P(\mathbf{v}_0|\mathbf{h}_0) \in [0,1]^h$ corresponds to the likelihood of membership to the latent features that encode classes in the KG and, therefore, carries more in information than binary values. Then, Ridle constructs a vector with the classes to which the entity s belongs to, i.e., $\mathbf{t}_s[i] = 1$ if $(s, type, t_i) \in \mathcal{G}$ for some $t_i \in \mathcal{T}$, $\mathbf{t}_s[i] = 0$ otherwise. The vector $\mathbf{t}_s$ is used as labels in the classification problem.

For the supervised learning algorithm, Ridle implements a 2-layer neural network,[2] with input layer size h and output layer size $|\mathcal{T}|$. In the hidden layer, Ridle uses an approximation of the GELU activation function. We select the GELU activation function due to its excellent performance in related machine learning tasks, as shown in BERT [7]. For an input x in the network, the used approximation of the GELU function, as described in more detail in the corresponding paper [10], is defined as follows:

$$\text{GELU}(x) = \frac{1}{2}x\left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right]\right) \tag{11}$$

In the output layer Ridle applies a sigmoid function, thus the results can be interpreted as the probability that an instance belongs to a certain class or type.

[2]We conducted a preliminary evaluation using neural networks with varying number of layers. However, the 2-layer setting exhibited the best F1 performance.

Table 1: Characteristics of the studied KGs. For each KG $\mathcal{G}$, $|\mathcal{G}|$=number of triples, $|\mathcal{E}|$=number of subjects, $|\mathcal{R}|$=number of relations, $|\mathcal{T}|$=number of classes

KG	$\|\mathcal{G}\|$	$\|\mathcal{E}\|$	$\|\mathcal{R}\|$	$\|\mathcal{T}\|$
DBp_3.8	3,246,924	31,952	10,200	294
DBp_2016-04	2,457,561	49,004	11,070	354
WD_2017-03-13	3,141,087	49,884	1,763	1,939
YAGO4	2,230,760	147,464	109	823
UMLS	6,029	135	45	46
DBLP	2,712,914	136,485	26	11
Pers(DBp)	333,296	64,423	2,239	126
Pers(WD)	249,059	8,400	1,509	53
Books(DBp)	242,989	13,361	619	17
Books(WD)	285,757	59,819	519	461
Chem(DBp)	58,952	9,674	265	5
Chem(WD)	268,534	16,872	339	1,008
Comp(DBp)	162,887	9,531	1,274	40
Comp(WD)	14,943	6,456	330	217
Movies(DBp)	416,834	69,761	959	13
Movies(WD)	410,295	8,807	382	74
Songs(DBp)	115,833	6,200	332	9
Songs(WD)	204,542	41,990	321	230
Uni(DBp)	183,700	9,029	2,021	13
Uni(WD)	66,182	12,133	472	274

4 EXPERIMENTS

4.1 Experimental Setup

Datasets. Following related work [17, 24], we use well-known public KGs such as the English DBpedia (3.8 and 2016-04) , Wikidata (WD_2017-03-13), YAGO4, UMLS and DBLP. In the DBpedia graphs, we removed common relations including *prov:wasDerivedFrom*, *dbo:wikiPageRevisionID*, and *dbo:wikiPageID*. All those relations occur in most instances and, therefore, do not provide class-specific information. Furthermore we have removed schema assertions, i.e. *rdf:type*, to not bias the downstream prediction tasks. All KGs except Wikidata use the property *rdf:type* to specify instance type assertions. Wikidata uses *wd:P31* as class membership property. Given the size of the KGs and limited computational resources, we performed a data pre-processing step on DBpedia, Wikidata and YAGO4, where only a subset of entities that occur in at least 10 triples and at most in $1,000$ triples are considered.[3] In addition, we extracted category-based subgraphs from DBp_2016-04 (DBp) and WD_2017-03-13 (WD) to study the performance of approaches in KGs limited to specific topics: persons (Pers), books (Books), chemical compounds (Chem), companies (Comp), movies (Mov), songs (Songs), and universities (Uni). Table 1 summarizes the datasets.

Metrics. We use the F1-score for measuring the effectiveness of instance type predictions, as done in related works before [24, 28]. We report on both F1-macro and F1-micro for the aggregation of multi-label performance in order to show the impact of prediction errors in more detail. We conducted each experiment by using 10-fold cross-validation and report on the average F1-score. The results can be reproduced using the k-fold cross-validator implemented in scikit-learn [25] using a random seed of 42.

Baselines. We compare our approach, Ridle, with current state-of-the-art models in instance type prediction. The models include strong baselines, e.g., RDF2Vec [28]. The selection of the models was based on their focus on instance type prediction in RDF data e.g. SDType [24], excellent performance, e.g., TransE [5] and RESCAL [23], as well as latest developments in link predictions, e.g. InteractE [32]. The state-of-the-art methods were used to learn a KG representation for each of the datasets described in Table 1. Then, for a direct comparison, each learned representation was fed to the same neural network architecture detailed in Sect. 3.2. The only exception is SDType, as it does not learn a KG representation, but directly produces an instance type prediction.

Implementation. Ridle is implemented in Python3. We used the same hyperparameter settings on every knowledge graph. We chose a learning rate $\alpha = 0.01$ with a hidden layer size of 50 and 100 iterations for learning the representations. The experiments were performed on a server with Intel(R) Xeon(R) Gold 6142 CPU@2.60GHz, 32 physical cores and 188GB RAM. For the baselines, we used the standard hyperparameters recommended by the authors[4].

4.2 F1-Score Performance

The effectiveness of the approaches in terms of the F1-macro and F1-micro is presented in Table 2. Overall, we can observe that none of the methods completely outperforms the other methods throughout all the studied knowledge graphs.

Considering the cross-domain KGs (cf. Table 2a), Ridle significantly outperforms the state-of-the-art methods with respect to the metric F1-macro. This indicates that, even in the presence of large KGs with a high number of classes and relations like is the case of DBpedia, Wikidata, and YAGO, our proposed solution is still able to produce accurate predictions. The main reason for this is that the KG representation learned with the RBM model (cf. Sect. 3.1) is able to capture the distribution of relations across the entity in the KG. This, in turn, enables the identification of entities that belong to the same classes based on the relations used to describe the entity. In contrast, the studied baselines are mostly tailored to learn statement-level representations which cannot effectively encode the knowledge about instance types when considering large KGs with a high number of classes and relations. With respect to the metric F1-micro, we can observe that Ridle clearly outperforms the other approaches except for the Wikidata KG. In this case, Ridle cannot correctly predict entities for the most popular classes, i.e., classes with a large number of entities like *human settlement (wd:Q486972)*. The reason for this behavior is that the entities (e.g. *wd:Q13071219*) in the most popular Wikidata classes contain a few class-specific relations, thus, affecting the performance of Ridle. In contrast, the baseline methods use additional object information while learning the KG representations, which allows for differentiating subjects from different classes. These results confirm that Ridle achieve a high performance in scenarios where entities in the KG are described with sufficient class-specific relations. In the rest

[3]This pre-processing step removes *noisy* entities with too many or too few descriptions.

[4]https://github.com/TobiWeller/Ridle

Table 2: Results for predicting instance types specified with the predicates *rdf:type* (DBpedia) and *wd:P31* (Wikidata). Bold values represent best average results.

(a) Results for cross-domain knowledge graphs

	F1-Macro						F1-Micro					
KG	Ridle	SDType	RDF2Vec	RESCAL	IntE	TransE	Ridle	SDType	RDF2Vec	RESCAL	IntE	TransE
DBp_3.8	**.840±.01**	.224±.02	.331±.02	.370±.01	.098±.01	.376±.01	**.965±.00**	.662±.01	.000±.00	.688±.00	.002±.00	.716±.00
DBp_2016-04	**.846±.01**	.222±.01	.209±.02	.317±.02	.188±.02	.371±.02	**.968±.00**	.595±.01	.000±.00	.624±.00	.000±.00	.715±.00
WD_2017-03-13	**.805±.01**	.115±.01	.774±.01	.784±.01	.784±.01	.779±.01	.590±.01	.563±.01	.000±.00	.751±.00	.752±.00	**.801±.00**
YAGO4	**.727±.01**	.056±.00	.693±.02	.623±.01	.657±.01	.621±.01	**.965±.00**	.888±.00	.643±.00	.889±.00	.725±.00	.890±.00

(b) Results for category-specific knowledge graphs

	F1-Macro						F1-Micro					
KG	Ridle	SDType	RDF2Vec	RESCAL	IntE	TransE	Ridle	SDType	RDF2Vec	RESCAL	IntE	TransE
UMLS	**.669±.04**	.064±.02	.555±.06	.617±.05	.557±.06	.387±.18	**.598±.08**	.281±.08	.315±.06	.524±.09	.318±.05	.508±.12
DBLP	**.803±.04**	.018±.00	.198±.05	.593±.00	.134±.03	.645±.01	**.995±.00**	.051±.00	.630±.01	.970±.00	.504±.01	.970±.00
Pers(DBp)	**.680±.03**	.329±.01	.210±.02	.331±.03	.212±.03	.375±.02	**.943±.00**	.880±.00	.735±.00	.842±.00	.743±.01	.879±.00
Pers(WD)	.844±.08	.322±.23	**.848±.10**	**.848±.10**	**.848±.10**	**.848±.10**	**.997±.00**	**.997±.00**	**.997±.00**	**.997±.00**	**.997±.00**	**.997±.00**
Books(DBp)	.859±.10	.603±.18	**.865±.12**	.770±.24	**.865±.12**	**.865±.12**	**.999±.00**	**.999±.00**	**.999±.00**	**.999±.00**	**.999±.00**	**.999±.00**
Books(WD)	**.734±.01**	.036±.01	.712±.02	.720±.02	.712±.02	.720±.02	**.932±.00**	.901±.00	.912±.00	.914±.00	.912±.00	.916±.00
Chem(DBp)	**.820±.19**	.729±.18	**.820±.19**	**.820±.19**	**.820±.19**	**.820±.19**	**.999±.00**	.994±.00	**.999±.00**	**.999±.00**	**.999±.00**	**.999±.00**
Chem(WD)	.765±.01	.012±.00	**.766±.02**	**.766±.02**	**.766±.02**	**.766±.02**	**.847±.01**	.816±.01	.797±.01	.798±.01	.797±.01	.831±.01
Comp(DBp)	**.762±.07**	.386±.14	.681±.13	.737±.13	.681±.13	.753±.13	**.993±.00**	.980±.00	.969±.00	.981±.00	.969±.00	.990±.00
Comp(WD)	**.828±.03**	.070±.02	.819±.03	.819±.03	.819±.03	.819±.03	**.939±.01**	.892±.01	.935±.01	.935±.01	.935±.01	.935±.01
Movies(DBp)	**.650±.18**	.331±.11	.608±.13	.608±.13	.608±.13	.608±.13	**.999±.00**	.998±.00	**.999±.00**	**.999±.00**	**.999±.00**	**.999±.00**
Movies(WD)	.785±.08	.197±.08	**.787±.07**	.685±.22	**.787±.07**	**.787±.07**	**.989±.00**	**.989±.00**	**.989±.00**	.985±.01	**.989±.00**	**.989±.00**
Songs(DBp)	**.854±.10**	.739±.10	.731±.10	.837±.10	.733±.10	.842±.10	**.990±.00**	.989±.00	.952±.00	.986±.00	.952±.00	.989±.00
Songs(WD)	**.745±.02**	.062±.01	.731±.03	.736±.03	.731±.03	.740±.03	**.917±.00**	.806±.00	.889±.00	.895±.00	.889±.00	.911±.00
Uni(DBp)	**.766±.16**	.613±.20	.708±.11	.683±.14	.708±.11	.708±.11	**.998±.00**	**.998±.00**	**.998±.00**	**.998±.00**	**.998±.00**	**.998±.00**
Uni(WD)	**.710±.03**	.047±.01	.701±.03	.704±.03	.701±.03	.704±.03	**.854±.01**	.790±.01	.824±.01	.828±.01	.824±.01	.831±.01

of the KGs, Ridle can correctly classify entities for both large and small classes as shown with both metrics. Another important result is the low F1-micro values achieved by RDF2Vec in DBpedia and Wikidata. In particular, RDF2Vec is always predicting *foaf:Agent* as class, which is considered a false positive according to the test data.

Next, we look at the performance of the approaches in the category-specific KGs (cf. Table 2b). We can observe that, in some KGs, the performance of all approaches is significantly higher in comparison to the cross-domain KGs. This indicates that the correct classification of entities is easier in certain classes of a given KG. In terms of average F1-macro, Ridle outperforms the state of the art in 12 out of the 16 studied KGs. Still, in the other 4 KGs – Pers(WD), Books(DBp), Chem(WD), and Movies(WD) – we can conclude that Ridle achieves competitive performance in comparison to the best approaches when considering the difference between the average F1-macro (in the order of 10^{-2}) and the standard deviation (in the order of 10^{-1}). In terms of F1-micro, Ridle achieves a very high performance on average. Furthermore, the other approaches also achieve a high performance for the datasets Pers(WD), Books(DBp), Chem(DBp), Movies(DBp), Movies(WD), and Uni(DBp). These KGs are mostly characterized by having a low to moderate number of classes ($|\mathcal{T}|$ between 5 and 53), and hundreds of relations ($|\mathcal{R}|$ between 265 and 959) with the exception of Pers(WD) with $|\mathcal{R}| = 1509$. Yet, even in category-specific KGs with hundreds of classes or few relations, our approach outperforms the state-of-the-art.

4.3 Visualizing Entities from the Learned KG Representations.

To get insights into the effectiveness of our solution in instance type predictions, we computed PCA projections of the learned entity representations into a two-dimensional space for the DBpedia, Wikidata, and YAGO KGs. Besides Ridle, we present the results for RDF2Vec, as an exemplary approach that exhibits a good performance in the category-specific KGs. In the following, we analyse the results for selected classes in the KGs in Figure 3.

For the DBpedia KG, we focus on the top four most popular classes[5]. We can observe that, although the class *Agent* is broad in both representations, Ridle allows for better distinguishing the classes *Work* and *Place* from *Agent*. This separation in vectors from different classes is essential to achieve a high performance in the downstream task for predicting instance types. The representations of RDF2Vec, in contrast, are suggesting that the classes *Work* and *Place* are (semantically) related to *Agent*, which does not hold in

[5]The DBpedia Ontology hierarchy is available at http://mappings.dbpedia.org/server/ontology/classes/

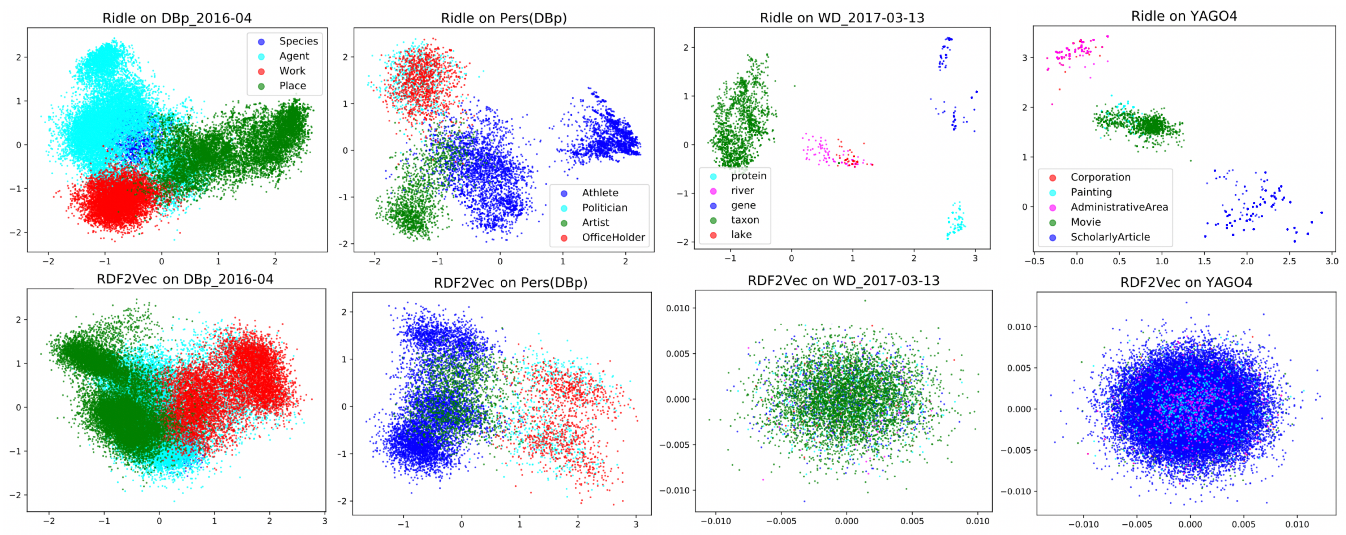


Figure 3: PCA projections for the learned entity representations. Popular classes from cross-domain KGs were selected for visualization. Ridle (top) allows for a better separation of the instances into their respective classes in comparison to the state-of-the-art RDF2Vec approach (bottom).

DBpedia. Furthermore, the class *Species* is not visible in RDF2Vec since it is covered by the representation of *Agent*.

Next, we analyse the learned representations of the top four classes from the Pers(DBp) KG. In both approaches, we observe that the instances of the classes *OfficeHolder* and *Politician* are strongly interwoven and are difficult to distinguish from each other. A closer look at the instances of *OfficeHolder* and *Politician* revealed that these instances frequently use the same relations and that there is no variety of specific relations for these classes. In regards to the classes *Athlete* and *Artist*, Ridle achieves a greater separation of the computed vectors in comparison to RDF2Vec. The reason for this is that RDF2Vec considers the object values in the triples, therefore, under this representation the classes *Athlete* and *Politician* are considered similar as their instances share in some cases the same object. This behaviour, however, negatively affects the instance type prediction capabilities of RDF2Vec. To analyse the results for Wikidata and YAGO, we selected some similar and some dissimilar classes to show the behavior of Ridle in different scenarios. In Wikidata, Ridle clearly distinguishes distant classes – i.e., *protein* and *river* – and represents closely those with semantic proximity – i.e., *river* and *lake*, which often use similar relations e.g. *located in the administrative territorial entity (P131)* and *tributary (P974)*. Similar to previous results, the YAGO instance representations of Ridle allow for distinguishing the different classes,[6] although entities using similar relations, e.g, entities of the classes *Painting* and *Movie*, are closer to each other due to the frequent use of the same relations. In contrast, RDF2Vec cannot effectively distinguish the entity types in these datasets.

Using the insights gained from the PCA projections, we can further comprehend the results for predicting the instance types presented in Table 2. In the cross-domain KGs like DBp_2016-04 and YAGO4, there exists a larger number of classes whose instances are mostly described by class-specific relations. By computing a target distribution over the usage of these relations, the representation of Ridle can classify the entities more accurately on average than the baseline methods. Including the object information of the statements, as done by the baseline methods, a closer proximity of the instances is caused, leading to a more difficult classification of the instances into their correct classes. This was observed, for example, in RDF2Vec with the classes *Agent* and *Place* in DBp_2016-04. By using very few class-specific relations to describe instances, Ridle can no longer distinguish between classes, causing a loss of performance with respect to the F1-score. Overall, we can conclude that when entities are described with class-specific relations, Ridle is able to obtain a representation that effectively encodes both semantically similar and dissimilar classes.

5 CONCLUSION AND FUTURE WORK

In this paper, we presented an inductive stochastic factorization model to represent entities of knowledge graphs (KGs), suitable for predicting instance type assertions. Our approach, Ridle, first implements an unsupervised learning model based on Restricted Boltzmann Machines (RBMs) to leverage the distribution over the usage of relations in instances of KGs. We then devise entity representations based on the latent features learned with the RBM. Using the learned representations, Ridle implements a neural network architecture for predicting instance type assertions.

The experimental results showed that, on average, Ridle outperforms current state-of-the-art models in several KGs, which sets a new baseline in the tasks of predicting instance type assertions. The visualization of the learned KG representation shows that Ridle is able to correctly group entities with similar distribution of relations, whereas dissimilar entities are represented far away. This property of Ridle is key for achieving a high performance in entity

[6]The plot includes all the entities, but they are superimposed. Ridle maps many of the entities from the class *Scholarly Article* to the same point in space.

prediction. Likewise, Ridle was able to reconstruct semantic associations between relations from instance-relation distributions, even though ontological information was not available during training. This learned semantic associations in the instance representation of Ridle is a decisive factor for the downstream performance of the instance type prediction task.

Future work may focus on studying the effectiveness of the learned instance representations for constructing class taxonomies. As shown in this paper, instances using a similar target distribution are closed in the dimensional space, which could be mined to predict containment relations between classes. Future work may also study the completion of other types of schema knowledge, e.g., the domain and range of properties.

REFERENCES

[1] Michael Azmy, Peng Shi, Jimmy Lin, and Ihab F. Ilyas. 2019. Matching Entities Across Different Knowledge Graphs with Graph Embeddings. *CoRR* abs/1903.06607 (2019).

[2] Robert M. Bell and Yehuda Koren. 2007. Lessons from the Netflix Prize Challenge. *SIGKDD Explor. Newsl.* 9, 2 (2007), 75–79.

[3] Yoshua Bengio and Olivier Delalleau. 2009. Justifying and Generalizing Contrastive Divergence. *Neural Comput.* 21, 6 (June 2009), 1601–1621.

[4] Russa Biswas, Rima Türker, F. B. Moghaddam, Maria Koutraki, and H. Sack. 2018. Wikipedia Infobox Type Prediction Using Embeddings. In *DL4KGS@ESWC*.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 2787–2795.

[6] Michael Cochez, Petar Ristoski, Simone Paolo Ponzetto, and Heiko Paulheim. 2017. Biased Graph Walks for RDF Graph Embeddings. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics (WIMS '17)*. Association for Computing Machinery, Article 21, 12 pages.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018).

[8] Asja Fischer and Christian Igel. 2012. An Introduction to Restricted Boltzmann Machines. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Springer Berlin Heidelberg, 14–36.

[9] Ramanathan Guha and Dan Brickley. 2014. *RDF Schema 1.1.* W3C Recommendation. W3C. https://www.w3.org/TR/2014/REC-rdf-schema-20140225/.

[10] Dan Hendrycks and Kevin Gimpel. 2016. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *CoRR* abs/1606.08415 (2016).

[11] G. E. Hinton and R. R. Salakhutdinov. 2006. Reducing the Dimensionality of Data with Neural Networks. *Science* 313, 5786 (2006), 504–507.

[12] Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. 2021. Do Embeddings Actually Capture Knowledge Graph Semantics?. In *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 12731)*. Springer, 143–159. https://doi.org/10.1007/978-3-030-77385-4_9

[13] Qiu Ji, Zhiqiang Gao, and Zhisheng Huang. 2011. Reasoning with Noisy Semantic Data. In *The Semanic Web: Research and Applications*. Springer Berlin Heidelberg, 497–502.

[14] Mayank Kejriwal and Pedro A. Szekely. 2017. Supervised Typing of Big Graphs using Semantic Embeddings. *CoRR* abs/1703.07805 (2017).

[15] Hugo Larochelle and Yoshua Bengio. 2008. Classification Using Discriminative Restricted Boltzmann Machines. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. Association for Computing Machinery, 536–543.

[16] Feng Liu, Bingquan Liu, Chengjie Sun, Ming Liu, and Xiaolong Wang. 2013. Deep Learning Approaches for Link Prediction in Social Network Services. In *Neural Information Processing*. Springer Berlin Heidelberg, 425–432.

[17] André Melo, Heiko Paulheim, and Johanna Völker. 2016. Type prediction in rdf knowledge bases using hierarchical multilabel classification. In *Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics*. 1–10.

[18] Nandana Mihindukulasooriya and Mariano Rico. 2018. Type Prediction of RDF Knowledge Graphs Using Binary Classifiers with Structural Data. In *Current Trends in Web Engineering*. Springer International Publishing, 279–287.

[19] Thomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* (Lake Tahoe, Nevada) *(NIPS'13)*. Curran Associates Inc., 3111–3119.

[21] Changsung Moon, Paul Jones, and Nagiza F. Samatova. 2017. Learning Entity Type Embeddings for Knowledge Graph Completion *(CIKM '17)*. Association for Computing Machinery, 2215–2218.

[22] Tu Dinh Nguyen, Truyen Tran, Dinh Phung, and Svetha Venkatesh. 2013. Latent Patient Profile Modelling and Applications with Mixed-Variate Restricted Boltzmann Machine. In *Advances in Knowledge Discovery and Data Mining*. Springer Berlin Heidelberg, 123–135.

[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-way Model for Collective Learning on Multi-relational Data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML'11)*. Omnipress, USA, 809–816.

[24] Heiko Paulheim and Christian Bizer. 2013. Type Inference on Noisy RDF Data. In *The Semantic Web – ISWC 2013*. Springer Berlin Heidelberg, 510–525.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[26] Axel Polleres, Aidan Hogan, Andreas Harth, and Stefan Decker. 2010. Can we ever catch up with the Web? *Semantic Web* 1, 1, 2 (2010), 45–52.

[27] Marc' Aurelio Ranzato, Y-Lan Boureau, and Yann LeCun. 2007. Sparse Feature Learning for Deep Belief Networks. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS'07)*. Curran Associates Inc., 1185–1192.

[28] Petar Ristoski and Heiko Paulheim. 2016. RDF2Vec: RDF graph embeddings for data mining, In The Semantic Web - ISWC 2016 : 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I. *Lecture Notes in Computer Science* 9981, 498–514.

[29] Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. 2019. RDF2Vec: RDF graph embeddings and their applications. *Semantic Web* 10, 4 (2019), 721–752.

[30] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann Machines for Collaborative Filtering. In *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*. Association for Computing Machinery, 791–798.

[31] Radina Sofronova, M. Alam, and H. Sack. 2020. Entity Typing based on RDF2Vec using Supervised and Unsupervised Methods.

[32] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Nilesh Agrawal, and Partha Talukdar. 2020. InteractE: Improving Convolution-based Knowledge Graph Embeddings by Increasing Feature Interactions. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. AAAI Press, 3009–3016.

[33] Yuhao Wang and Jianyang Zeng. 2013. Predicting drug-target interactions using restricted Boltzmann machines. In *Bioinform.*

[34] David Wood, Richard Cyganiak, and Markus Lanthaler. 2014. *RDF 1.1 Concepts and Abstract Syntax.* W3C Recommendation. W3C. https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/.