Knowledge Reconciliation with Graph Convolutional Networks: Preliminary Results

Pierre Monnin, Chedy Raïssi, Amedeo Napoli, Adrien Coulet

Knowledge Reconciliation with Graph Convolutional Networks: Preliminary Results*

Pierre Monnin[1], Chedy Raïssi[1,2], Amedeo Napoli[1], and Adrien Coulet[1,3]

[1] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
{pierre.monnin, chedy.raissi, amedeo.napoli, adrien.coulet}@loria.fr
[2] Ubisoft, Singapore
[3] Stanford Center for Biomedical Informatics Research, Stanford University, 94305 Stanford, California, USA

Abstract. In this article, we investigate the task of identifying nodes that are identical, more general, or similar within and across knowledge graphs. This task can be seen as an extension of instance matching or entity resolution and is here named knowledge reconciliation. In particular, we explore how Graph Convolutional Networks (GCNs), previously defined in the literature, can be used for this task and evaluate their performance on a real world use case in the domain of pharmacogenomics (PGx), which studies how gene variations impact drug responses. PGx knowledge is represented in the form of n-ary relationships between one or more genomic variations, drugs, and phenotypes. In a knowledge graph named PGxLOD, such relationships are available, coming from three distinct provenances (a reference database, the biomedical literature and Electronic Health Records). We present and discuss our preliminary attempt to generate graph embeddings with GCNs and to use a simple distance between embeddings to assess the similarity between relationships. By experimenting on the 68,686 PGx relationships of PGxLOD, we found that this approach raises several research questions. For example, we discuss the use of the semantics associated with knowledge graphs within GCNs, which is of interest in the considered use case.

Keywords: Knowledge Reconciliation · N-ary relationships · Graph Embeddings · Graph Convolutional Networks.

1 Introduction

Data and knowledge can be accessed extensively on the Web and interpreted by both human and software agents. Because these elements of knowledge are of various provenances, spread in various places and published following distinct standards, it is challenging to compare and conjointly use their content. Semantic Web and Linked Open Data (LOD) [2] provide standards and technologies to

* Supported by the *PractiKPharma* project, founded by the French National Research Agency (ANR) under Grant ANR15-CE23-0028, by the IDEX "Lorraine Université d'Excellence" (15-IDEX-0004) and by the *Snowball* Inria Associate Team.

facilitate the interoperability of knowledge spread over the Web, such as Uniform Resource Identifiers (URIs) and the Resource Description Framework (RDF) format. URIs identify nodes that can represent entities of the real world (*e.g.*, places, persons, drugs), while RDF statements represent edges, using predicates to link entities to each others or to literals (*e.g.*, strings, integers). Such predicates express the semantics of the relationship that connects two nodes or a node and a literal (*e.g.*, *is-born-in*, *has-firstname*). Therefore, URIs and RDF statements enable to represent knowledge in the form of a directed and labeled multigraph, loosely called a *knowledge graph*.

Because datasets are independently published on the Web, possibly with some overlap, it happens that different URIs are used to identify the same resource. For example, `dbpedia:Warfarin` and `wikidata:Q407431` are two URIs representing the chemical compound *Warfarin* in DBpedia and Wikidata. As a consequence, identifying different URIs possibly referring to the same resource is necessary to use various and initially independent datasets together. This task, called *instance matching* [6] or *entity resolution*, can be extended to identify not only identical resources but also more general or somehow similar ones, a task we call *knowledge reconciliation* (by analogy with reconciliation in databases [1]).

In this work, we illustrate this task with a real world application in the field of pharmacogenomics (abbreviated PGx). This field studies the influence of genomic variations in drug response phenotypes. Knowledge in PGx is typically composed of n-ary relationships between one or more genomic variations, drugs and phenotypes, stating that a patient having the specified genomic variations, and being treated with the specified drugs will be more likely to experience the given phenotypes. PGx relationships can be found in different sources: reference databases, biomedical literature, or by mining Electronic Health Records (EHRs). Therefore, there is a need to reconcile these PGx relationships from different sources, for example to confirm state-of-the-art knowledge found in the literature with clinical counterpart found in EHRs [5].

Several existing works use Semantic Web technologies to represent PGx knowledge, as they allow to easily relate resources to other nodes in the knowledge graph that can enrich their semantics (*e.g.*, *partOf* resources, classes of ontologies). For example, we built PGxLOD [10], a large knowledge graph containing 68,686 PGx relationships from the three aforementioned sources. As Semantic Web technologies only allow binary predicates, to be represented, PGx relationships are reified: the relationship itself is a node, linked by predicates to its components. For example, Figure 1 depicts the reification as the node `pgx_rel_1` of a ternary relationship between gene `CYP2C9`, drug `warfarin` and phenotype `cardiovascular_diseases`. A PGx relationship is fully defined by its components and, accordingly, two relationships involving the same sets of components are identical. Hence, reconciliation techniques based on the relational structure of nodes [6] are well-suited to reconcile PGx relationships represented using Semantic Web technologies.

In this paper, we investigate how the task of knowledge reconciliation can be achieved using graph embeddings [4], *i.e.*, low-dimensional vectors representing

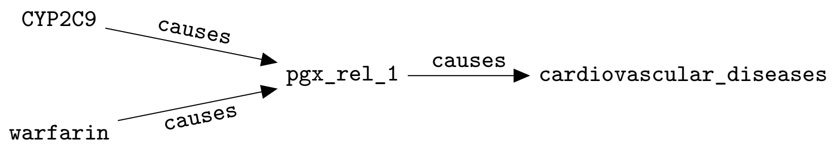


Fig. 1. Representation of a reified ternary PGx relationship between gene `CYP2C9`, drug `warfarin` and phenotype `cardiovascular_diseases`. The relationship is reified as the node `pgx_rel_1`, which is connected to its components by the `causes` predicate.

graph structures (*e.g.*, nodes, edges, subgraphs) while preserving variously the properties of the graph. Particularly, we present the preliminary results of an original experiment using Graph Convolutional Networks (GCNs) [8, 15] that have already been successfully used for link prediction, a task somehow similar to knowledge reconciliation. GCNs compute an embedding for each node considering its neighbors, and, thus, are well adapted to our task in which the relational structure is of prime importance. Similarity between n-ary relationships could be represented by ensuring a low distance between their respective embeddings. Inspired by recent works [13, 16], we use definitions of inverses of predicates to illustrate how semantics of knowledge graphs could be used in GCNs. We experimented by reconciling the 68,686 PGx relationships from PGxLOD [10]. The remainder of this article is organized as follows. Section 2 presents related works. Section 3 details GCNs and the proposed general setting for knowledge reconciliation of n-ary relationships. Section 4 describes our experiment with the biomedical knowledge graph PGxLOD. Finally, we discuss our results and future directions in Section 5.

2 Related Works

Numerous works exist on ontology matching. The interested reader could refer to [6] for a detailed presentation of approaches. In the following, we focus on graph embeddings techniques, that have been investigated in multiple works and successfully applied on knowledge graphs for tasks such as node classification or link prediction [8, 14, 15]. Works differ in the considered type of graphs (*e.g.*, homogeneous graphs, heterogeneous graphs such as knowledge graphs) or in the graph embedding techniques used (*e.g.*, matrix factorization, deep learning with or without random walk), as listed in the taxonomies of problems and techniques in Cai *et al.* survey [4]. In the following, few specific examples are detailed but a more thorough overview can be found in some of the existing surveys [4, 11].

A first example is TransE [3], which computes for each triple $\langle s, p, o \rangle$ of a knowledge graph, embeddings h_s, h_p, h_o, such that $h_s + h_p \approx h_o$, *i.e.*, the translation vector from the subject to the object of a triple corresponds to the embedding of the predicate. This approach is adapted for link prediction but, according to the authors, it is unclear if it can model adequately relationships of distinct arities, such as *1-to-Many*, or *Many-to-Many*. Another example is RDF2Vec [14], which first extracts, for each node, a set of sequences of graph sub-structures starting from this node. Elements in these sequences can be edges, nodes or even subtrees. Then, sequences feed algorithms from the word2vec neural language model that compute embeddings for each element in a sequence by either maximizing the probability of an element given the other elements of the sequence (Continuous Bag of World architecture) or maximizing the probability of the other elements given the considered element (Skip-gram architecture). A third approach, adopted in this article, is GCNs that have been introduced in [8] for semi-supervised classification on graphs and extended in [15] for entity classification and link prediction in knowledge graphs. Contrasting TransE and RDF2Vec that respectively work at the triple and sequence levels, GCNs compute the embedding of a node by considering its neighborhood. Therefore, GCNs seem more suited to the task of reconciling n-ary relationships, that are entirely defined by their neighboring nodes representing their components.

However, previous methods do not consider the semantics associated with predicates and nodes. Alternatively, Logic Tensor Networks [16] are used to learn groundings. The grounding of a logical term is a vector of real numbers (*i.e.*, an embedding) and the grounding of a logical clause is a real number in the interval $[0, 1]$ (*i.e.*, the confidence in the truth of the clause). The learning process tries to minimize the satisfiability error of a set of clauses, while trying to ensure the logical reasoning. This work can interestingly be compared to graph embeddings if knowledge graphs are considered in their logical form, *i.e.*, considering nodes as logical terms and edges linking two nodes as logical formulae. We adopted such consideration by exploring in this work a first manner to include (limited) semantics within GCNs, for the knowledge reconciliation task.

3 Knowledge Reconciliation with GCNs

3.1 Learning Task

We consider that we have at our disposal a knowledge graph, with specific nodes representing reified n-ary relationships. Our task consists in learning embeddings for these relationships such as their distance reflects their similarity or dissimilarity. The learning task relies on two elements: the learning of embeddings associated with each node of the graph and the assessment of the similarity between nodes representing n-ary relationships by computing the distance between their respective embeddings.

To train our GCN model, we constitute a training set and a test set made of a balanced number of positive and negative examples. Regarding positive examples, we assume that some n-ary relationships are already labeled as similar from a manual labeling or from the execution of another method, such as the similarity rules validated by an expert we use in Section 4. This similarity labels may have different levels (*e.g.*, very high, high) or may reflect different semantics (*e.g.*, identical relationships, more general ones). However in this preliminary setting, we do not take into account such detailed semantics and only consider a coarse-grained similarity: similar or not labeled. Indeed, as knowledge graphs are built under the *Open World Assumption*, absent statements from a knowledge graph are only unknown and not false. For this reason, we consider that we have at our disposal only positive similarity labeling for n-ary relationships. To generate "negative" examples, an approach similar to the one used in TransE [3] is considered: for each pair of similar n-ary relationships (i, j), another n-ary relationship k is found such as it is not labeled as similar either to i or to j. The triple (i, j, k) representing a training example is then added to the training set S. The same method is used to generate the test set.

3.2 Using GCNs to generate graph embeddings

In the following, we adopt the notations defined in [15]. As such, $\mathcal{R}$ denotes the set of predicates in the considered knowledge graph. Considering a node i and a predicate $r \in \mathcal{R}$, we denote $\mathcal{N}_i^r$ the set of nodes reachable from i by r. Only nodes and predicates linking nodes are considered. Literals and predicates linking nodes to literals are discarded.

GCNs can be seen as a message-passing framework of multiple layers, in which the embedding $h_i^{(l+1)}$ of a node i at layer $(l+1)$ depends on the embeddings of its neighbors at level (l), as stated in Equation (1).

$$h_i^{(l+1)} = \sigma\left(\sum_{r \in \mathcal{R}} \sum_{j \in \mathcal{N}_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)} + W_0^{(l)} h_i^{(l)}\right) \tag{1}$$

The convolution over the neighboring nodes of i is computed with a specific weight matrix $W_r^{(l)}$ for each predicate $r \in \mathcal{R}$ and each layer (l). This convolution is regularized by a constant $c_{i,r}$, that can be set for each node and each predicate. Additionally, to ensure that the embedding of i at layer $(l+1)$ also depends on its embedding at layer (l), a self connection is allowed, with the weight matrix $W_0^{(l)}$. σ is a non-linear function such as ReLU, used in our experiment (Section 4).

Authors in [15] consider that every predicate $r \in \mathcal{R}$ has an inverse $r_{inv} \in \mathcal{R}$. As this is not always true in knowledge graphs, and to illustrate how the semantics of predicates could be used in GCNs, we leverage potentially defined inverse predicates. We consider the three following cases for a predicate r:

(i) If r is defined as symmetric, we do not consider an inverse but ensure that the adjacency matrix for r is symmetric;
(ii) If r has a defined inverse r^{-1}, we use r and r^{-1} and ensure their adjacency matrices are indeed representing inverse relations;
(iii) Otherwise, we generate an inverse r_{inv} such as its adjacency matrix represent the inverse of r.

By doing so, we avoid generating an abstract inverse r_{inv} for predicates r having a defined inverse r^{-1} or being symmetric, which would add unnecessary messages. Indeed, consider a predicate r, its defined inverse r^{-1}, and two nodes i and j such that edges $i \xrightarrow{r} j$ and $j \xrightarrow{r^{-1}} i$ are in the knowledge graph. By always generating an abstract inverse, two edges would be added, $j \xrightarrow{r_{inv}} i$ and $i \xrightarrow{r_{inv}^{-1}} j$, duplicating the existing edges and adding two unnecessary messages.

To train GCNs, we minimize the loss function presented in Equation (2), inspired from the one used in TransE [3].

$$\mathcal{L} = \sum_{(i,j,k)\in S} \max\Big(||h_i - h_j||_2 + \gamma - ||h_i - h_k||_2,\ 0\Big) \tag{2}$$

Given a training example (i, j, k) from the training set S, minimizing the loss function aims at minimizing the distance between h_i and h_j, *i.e.*, ensuring their similarity, while maximizing the distance between h_i and h_k. The constant γ is a margin hyperparameter aiming at increasing the difference between the two distances.

4 Experimentation on PGx Knowledge

4.1 Input Knowledge Graph: PGxLOD

We experimented our approach on PGxLOD [10], a large knowledge graph containing PGx relationships from three distinct sources: PharmGKB (a reference database), the biomedical literature and results from studies on Electronic Health Records. Main statistics of PGxLOD are presented in Table 1.

Table 1. Main statistics of PGxLOD. The line "Predicates" only counts predicates used to link two nodes together, excluding literals. As our experiment uses a 3-layer network, only the 3-hop neighborhood of PGx relationships is considered to compute their embeddings.

Triples	59,136,400
Nodes and literals	25,226,599
Predicates	378
PGx relationships	68,686
↳ Nodes in their 3-hop neighborhood	2,943,613
↳ Edges in their 3-hop neighborhood	32,773,429
Similarity links between PGx relationships	283,248
↳ `owl:sameAs` links	109,226
↳ `skos:broadMatch` links	136,264
↳ `skos:relatedMatch` links	37,758

In PGxLOD, some similar PGx relationships are already labeled by being linked together with one of the three following predicates: `owl:sameAs` expressing identical relationships, `skos:broadMatch` expressing more general ones and `skos:relatedMatch` expressing related ones to some extent. Such labels result from the application of logical reconciliation rules that are described in [10].

They are used to constitute the training and test sets, in a "knowledge graph as silver standard" perspective [12]. Even if `owl:sameAs`, `skos:broadMatch` and `skos:relatedMatch` links express different similarity semantics, here we indifferently consider the three predicates as expressing a coarse-grained similarity. Thus, two PGx relationships are labeled as similar if they are linked by one of these three predicates. Additionally, we consider their adjacencies in an undirected perspective, *i.e.*, having (i, j) as a similarity edge is equivalent to having (j, i). As `owl:sameAs` and `skos:relatedMatch` are symmetric, numbers of available links for training and test sets are consequently half of those presented in Table 1. For each predicate, the training set is constituted by $\frac{2}{3}$ of the links and the test set by $\frac{1}{3}$. To form triples (i, j, k) in these sets, k is chosen such as it is not directly linked via a similarity predicate either to i or to j.

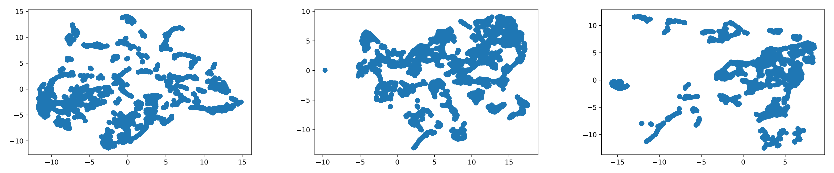
4.2 Experimental Setting

For our preliminary experiment, we use a standard architecture and hyperparameters previously reported in the literature with successful uses of GCNs. We consider a 3-layer network where each layer uses ReLU as the activation function. In such a 3-layer architecture, only neighboring nodes up to 3 hops of PGx relationships will have an impact on their embeddings, output at layer 3 (Equation (1)). The input layer consists in a featureless approach as in [8, 15], *i.e.*, the input is just a one-hot vector for each node of the graph. Both the input layer and the hidden layer have an output dimension of 16 while the output layer has an output dimension of 10. Therefore, embeddings for all nodes in the knowledge graph are in $\mathbb{R}^{10}$. Only the embeddings for nodes representing the reified PGx relationships are of interest in our reconciliation task and are considered in the loss function. As in [15], the constant $c_{i,r}$ is set to $|\mathcal{N}_i^r|$ and we use the basis-decomposition with 10 basis to avoid the growth in number of parameters. For the learning process, we use the Adam optimizer [7] with a starting learning rate of 0.01 and a L2-regularization coefficient of 0.0005. The margin hyperparameter γ is set to 2. Our experiment was implemented using PyTorch and the Deep Graph Library.

4.3 Results

We trained our model during 60 epochs. The last layer outputs embeddings for all nodes in the graph but we only consider the ones representing reified PGx relationships. The mean and standard deviation of distances between embeddings of similar relationships in the training set were respectively $\mu_{train} = 1.93$ and $\sigma_{train} = 4.18$. As a simple evaluation, for each example (i, j, k) from the test set, (i, j) or (i, k) were considered as similar if their embeddings were distant of less than $\mu_{train} + \sigma_{train}$. These results were compared with existing similarity links, obtaining a precision of 0.92, a recall of 0.94 and a F1-score of 0.93.

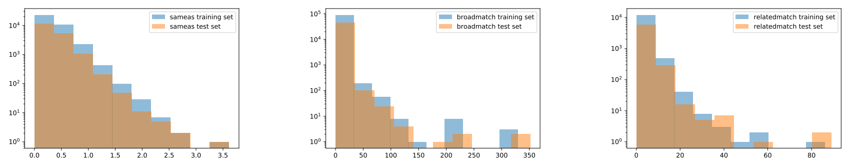
Then, we investigated differences between the three similarity predicates. 2D projections of embeddings using UMAP [9] are depicted in Figure 2. We can see that clusters of nodes are appearing but seem still close. More epochs or a wider neighborhood may allow to emphasize the difference between such clusters.



(a) `sameAs` links (b) `broadMatch` links (c) `relatedMatch` links

Fig. 2. 2D projections using UMAP [9] of embeddings of PGx relationships involved in the given similarity predicates.

Figure 3 depicts the distributions of distances between embeddings of similar PGx relationships depending on their similarity link. The low means for the three predicates illustrate that similar relationships have indeed embeddings with low distances. We notice that the mean distances for `owl:sameAs` are the lowest while the ones for `skos:relatedMatch` are the greatest. This could indicate the ability of the network to learn the close similarity expressed by `owl:sameAs` links and the more fuzzy one expressed by `skos:relatedMatch` links. Regarding `skos:broadMatch` links, the distance ranges and variances are more important than for the two other predicates. This could also illustrate the ability to learn the semantics of the predicate. Additionally, `skos:broadMatch` links are directed, and thus, could be more difficult to fit correctly, mixed with symmetric similarity predicates. Also, only `skos:broadMatch` links connect PGx relationships across the three considered sources [10]. Therefore, they link together relationships that may have more diversity in the semantics and vocabularies of their components, potentially explaining the higher distance ranges and variances.



(a) `sameAs` links
$\mu_{train} = 0.33$
$\sigma_{train} = 0.26$
$\mu_{test} = 0.33$
$\sigma_{test} = 0.26$

(b) `broadMatch` links
$\mu_{train} = 2.27$
$\sigma_{train} = 4.90$
$\mu_{test} = 2.30$
$\sigma_{test} = 5.12$

(c) `relatedMatch` links
$\mu_{train} = 4.16$
$\sigma_{train} = 2.77$
$\mu_{test} = 4.35$
$\sigma_{test} = 3.31$

Fig. 3. Distributions of distances between embeddings of similar PGx relationships linked by the given similarity predicates. μ and σ respectively denote the mean and the standard deviation.

5 Discussion and Conclusion

In this paper, we investigated the task of generating graph embeddings for knowledge reconciliation using Graph Convolutional Networks and ensuring that embeddings associated with similar reified n-ary relationships have a low distance. We experimented our approach on the real world use case of reconciling PGx n-ary relationships from three distinct sources. Our preliminary results found this approach to be suitable and to raise several research questions.

First, the network output different distances for the three considered predicates: `owl:sameAs`, `skos:broadMatch` and `skos:relatedMatch`. This could indicate that it was able to learn their different similarity semantics or had difficulties to adequately fit some of them. Possible improvements would be to increase the number of epochs or test other values for hyperparameters. The loss function could integrate the different semantics of the predicates linking similar relationships i and j, for example by considering three different weighted sums. Separate models could be learned: one per predicate or one for `owl:sameAs` and `skos:relatedMatch` and another for `skos:broadMatch` which is not symmetric, which could make easier interpreting the semantics of the output similarity.

Because we used a 3-layer network, only nodes in the 3-hop neighborhood of PGx relationships were considered for the computation of their embeddings. However, nodes in further neighborhoods may bring additional semantics. In particular, phenotypes extracted from the biomedical literature are frequently complex and formed by several simpler phenotypes, indicated by *dependsOn* links. Therefore, we could benefit from using a network with more layers.

We also illustrated how semantics associated with a knowledge graph can be used in GCNs by considering the definitions of inverses of predicates. This could be improved, for example by considering the semantics of `owl:sameAs` links between nodes. Indeed, these links indicate identical nodes that are currently considered as neighboring nodes and used as such in the embeddings computation. Thus, a pre-processing step could consist in mapping nodes linked by `owl:sameAs` links into a unique node. Additionally, the generation of negative examples could be improved by considering ontologies. In such case, PGx relationships whose components instantiate classes in different parts of an ontology could be more interesting negative examples.

Our model was evaluated using a manually-defined threshold on distances between embeddings of relationships to assess their similarity. Other methods such as (multi-)classification machine learning models could also be investigated. Advanced performance results could also be computed on knowledge graphs from other domains as well as be compared with other state-of-the-art methods presented in Section 2. Finally, we should manually check on a few examples whether relationships considered as close given the distance between their embeddings but not linked by any of the similarity predicates are indeed similar.

To conclude, these results constitute solely an initial attempt to use graph embeddings for the non-trivial task of reconciling n-ary relationships. Several future directions are considered, among which is the further integration of the semantics associated with knowledge graphs in GCNs.

References

1. Abiteboul, S., Manolescu, I., Rigaux, P., Rousset, M., Senellart, P.: Web Data Management. Cambridge University Press (2011)
2. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. Int. J. Semantic Web Inf. Syst. **5**(3), 1–22 (2009)
3. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States. pp. 2787–2795 (2013)
4. Cai, H., Zheng, V.W., Chang, K.C.: A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Trans. Knowl. Data Eng. **30**(9), 1616–1637 (2018)
5. Coulet, A., Smaïl-Tabbone, M.: Mining electronic health records to validate knowledge in pharmacogenomics. ERCIM News **2016**(104) (2016)
6. Euzenat, J., Shvaiko, P.: Ontology Matching, Second Edition. Springer (2013)
7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014), http://arxiv.org/abs/1412.6980
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. CoRR **abs/1609.02907** (2016), http://arxiv.org/abs/1609.02907
9. McInnes, L., Healy, J., Saul, N., Grossberger, L.: Umap: Uniform manifold approximation and projection. The Journal of Open Source Software **3**(29), 861 (2018)
10. Monnin, P., Legrand, J., Husson, G., Ringot, P., Tchechmedjiev, A., Jonquet, C., Napoli, A., Coulet, A.: PGxO and PGxLOD: a reconciliation of pharmacogenomic knowledge of various provenances, enabling further comparison. BMC Bioinformatics **20-S**(4), 139:1–139:16 (2019)
11. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs: From multi-relational link prediction to automated knowledge graph construction. Proceedings of the IEEE **104**(1), 11–33 (2016)
12. Paulheim, H.: Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web **8**(3), 489–508 (2017)
13. Paulheim, H.: Make embeddings semantic again! In: Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th - to - 12th, 2018. (2018)
14. Ristoski, P., Paulheim, H.: 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. pp. 498–514 (2016)
15. Schlichtkrull, M.S., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings. pp. 593–607 (2018)
16. Serafini, L., d'Avila Garcez, A.S.: Learning and reasoning with logic tensor networks. In: AI*IA 2016: Advances in Artificial Intelligence - XVth International Conference of the Italian Association for Artificial Intelligence, Genova, Italy, November 29 - December 1, 2016, Proceedings. pp. 334–348 (2016)