Abstract Graphs and Abstract Paths for Knowledge Graph Completion

Vivi Nastase
University of Heidelberg
Heidelberg, Germany
nastase@cl.uni-heidelberg.de

Bhushan Kotnis
NEC Laboratories Europe GmbH
Heidelberg, Germany
bhushan.kotnis@gmail.com

Abstract

Knowledge graphs, which provide numerous facts in a machine-friendly format, are incomplete. Information that we induce from such graphs – e.g. entity embeddings, relation representations or patterns – will be affected by the imbalance in the information captured in the graph – by biasing representations, or causing us to miss potential patterns. To partially compensate for this situation we describe a method for representing knowledge graphs that capture an intensional representation of the original extensional information. This representation is very compact, and it abstracts away from individual links, allowing us to find better path candidates, as shown by the results of link prediction using this information.

1 Introduction

Knowledge graphs have become a very useful framework to organize and store knowledge. Their interconnected nature is not just a natural way to represent facts, but it has potential that the separate storage of facts does not have, such as: (i) we can use it as a relational model of meaning, and derive jointly representations for nodes (entities) and edges (relations); (ii) the structure can be explored to discover systematic patterns that reveal interesting and exploitable regularities, such as paths connecting nodes in direct relations, (iii) discovering and inducing new connections.

Link prediction methods in knowledge graphs (see (Nickel et al., 2016) for an overview) predict additional edges in the graph, based on induced node and edge representations that encode the structure of the graph and thus capture regularities (such as homophily).

Lao and Cohen (2010) introduced a new method that predicts direct links based on paths that connect the source and target nodes. Such paths are not only useful for link prediction (Lao et al., 2011; Gardner et al., 2014), but also for finding explanations for direct links and help with targeted information extraction to fill in incomplete knowledge repositories (Yin et al., 2018; Zhou and Nastase, 2018).

These approaches rely on the structure of the knowledge graph, which is inherently incomplete. This incompleteness can affect the process in different ways, e.g. it leads to representations for nodes with few connection that are not very informative, it can miss relevant patterns/paths (or derive misleading patterns/paths).

In this paper we investigate whether a higher-level view of a graph – an abstract graph that captures an intensional view of the original extensional graph – can help derive more robust and informative patterns. Such patterns are paths (i.e. sequences of relations) that could be used not only for link prediction, but also for targeted information extraction for completing the graph with external information. This abstract graph will contain only one edge for each relation type, that will connect a node representing the relation's domain (or source) to its range (or target). Additional edges will link the nodes to capture set relations (intersection, subset, superset) information between the different relations' domains and ranges. This step drastically reduces the graph size, making many different graph processing approaches more tractable. We investigate whether in this graph that represents a more general version of the information in the original KG, good patterns/paths are stronger and easier to find, because the aggregated view compensates for individual missing edges throughout the graph. We test the extracted paths through the link prediction task on Freebase (Bollacker et al., 2008) and NELL (Carlson et al., 2010a), using Gardner et al. (2014)'s experimental set-up: pairs of nodes are represented using their connected paths as fea-

*Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 147–157
Minneapolis, June 6–7, 2019.

tures, and a model for predicting the direct relations is learned and tested on training and test sets for 24 relations in Freebase and 10 relations in NELL. Our analysis shows that we find different and much fewer paths than the PRA method does (mostly because the abstract paths do not contain back-and-forth sequences of generalizing or type relations). The paths found in the abstract graphs lead to better performance on NELL than the PRA paths, which could be explained by the fact that NELL's relation inventory was designed to capture interdependencies (Carlson et al., 2010a). On Freebase the results we obtain are lower, but this could be due to a different negative sampling process. Inspection of the paths produced reveal that they seem to capture legitimate dependencies.

2 Related Work

Representing facts in a knowledge graph has multiple advantages: (i) they provide knowledge in an easily accessible and machine-friendly format; (ii) they facilitate various ways of encoding this information and deriving representations for nodes and edges that reflect their connectivity in the graph; (iii) they allow for the discovery of connectivity patterns, and possibly more.

In recent years, projecting the knowledge graph in an n-dimensional vector space, or learning embeddings for predicting missing facts has attracted a lot of interest. Embedding models aim to map entities, relations and triples to vector space such that additional facts can be inferred from known facts using notions of vector similarity. A class of embedding models that aim to factorize the graph are termed as latent factor models. Neural network based models such as ER-MLP (Dong et al., 2014), NTN (Socher et al., 2013), RNNs (Neelakantan et al., 2015; Das et al., 2016) and Graph CNNs (Schlichtkrull et al., 2018) are examples of embedding models while RESCAL (Nickel et al., 2012), DistMult (Yang et al., 2015), TransE (Bordes et al., 2013), ComplEx (Trouillon et al., 2017) are examples of latent factor models.

Lao and Cohen (2010) introduced a novel way to exploit information in knowledge graphs: using weighted extracted paths as features in four different recommendation tasks, which can be modeled as typed proximity queries. The idea of using paths in the graph has then been applied to the task of link prediction (Lao et al., 2011), and extended to incorporate textual information (Gardner et al., 2014). Lao et al. (2011) obtain paths for given node pairs using random walks over the knowledge graph. To be used as features shared by multiple instances, the information about nodes on the paths is removed, transforming the actual paths into "meta-paths".

The paths themselves can be incorporated in different ways in a model – as features (Lao et al., 2011; Gardner et al., 2014), as Horn clauses to provide rules for inference in KGs whether directly or through scores that represent the strength of the path as a direct relation (Neelakantan et al., 2015; Guu et al., 2015), also taking into account information about intermediary nodes (Das et al., 2017; Yin et al., 2018). Gardner and Mitchell (2015) perform link prediction using random walks but do not attempt to connect a source and target node, but rather to characterize the local structure around a (source or target) node using such localized paths. Using these *subgraph features* leads to better results for the knowledge graph completion task.

We focus here on discovering useful and explanatory paths, not on optimizing or further improving the KGC task. Using paths can lead to interpretable models because the paths can help explain the predicted fact. Meng et al. (2015) present a method to automate the induction of meta-paths in large heterogeneous information networks (a.k.a. knowledge graphs) for given node pairs, even if the given node pairs are not connected by a direct relation.

Path information is also found to improve performance since paths help the model learn logical rules. However, mining paths from a large knowledge graph is often computationally expensive since it involves performing a traversal through the graph. To overcome this limitation (Das et al., 2017) proposed deep reinforcement learning and (Chen et al., 2018) proposed RNNS for generating paths. However, many datasets suffer from paths sparsity, lack of enough paths connecting source target pairs, resulting in poor performance for many relations.

Wang et al. (2013) have a different approach – they start with patterns in the form of first-order probabilistic rules, which they then ground in a small subgraph of a large knowledge graph.

The approach we present here combines different elements of these previous approaches in a novel way: we build an abstract graph to find pat-

terns that would be similar to those used by (Wang et al., 2013). To test the quality of these paths we ground them using the original KG and use these grounded paths in a learning framework similar to (Gardner et al., 2014).

3 Abstract Graphs and Abstract Paths

Knowledge graphs are incomplete in an imbalanced way. Figures 1a-1b show how much the relation and node frequencies for Freebase 15k and NELL vary, and the fact that numerous nodes and edges have very low frequency (each data point corresponds to a node/relation, and the value is the degree of the node/frequency of the relation respectively). Freebase and NELL have a helpful characteristic: they have strongly typed relations, i.e. the source and target of a relation have a very specific type. NELL for example, has relations such as like *ActorStarredinMovie, StateHasLake*, and Freebase has */film/film/rating, /book/literary_series/author*, whose arguments have type *Person, Movie, State*, etc.

Previous work has shown that using node type information – provided in Freebase through the domain and range types for each relation – can help optimize computation for link prediction by filtering the entity matrix for each relation based on the relation's domain and range types (Chang et al., 2014), improve prediction by adding a factor in the loss function that accounts for the type of the entities involved in a relation (Kotnis and Nastase, 2017), or improve predictions based on paths in the graph by using the types of intermediary entities (Yin et al., 2018).

Entity types and the type of the domain and range of a relation have been proven to be useful for improving link prediction models. We investigate here the hypothesis that by relying on the fact that such strong constraints on the arguments of relations in Freebase exist, we can build an intensional graph of the knowledge repository that is smaller and thus easier to analyze than the full KG. We also hypothesize that at this abstract level we can induce better patterns/paths that are indicative of direct relations, because individual missing relation instances will not obfuscate useful patterns. We verify whether these patterns are good by testing their usefulness for link prediction. Finding qualitative patterns would have additional benefits, as they could be used to explain direct relation, and fill in the KG through targeted information extraction (Zhou and Nastase, 2018).

3.1 Abstract graphs

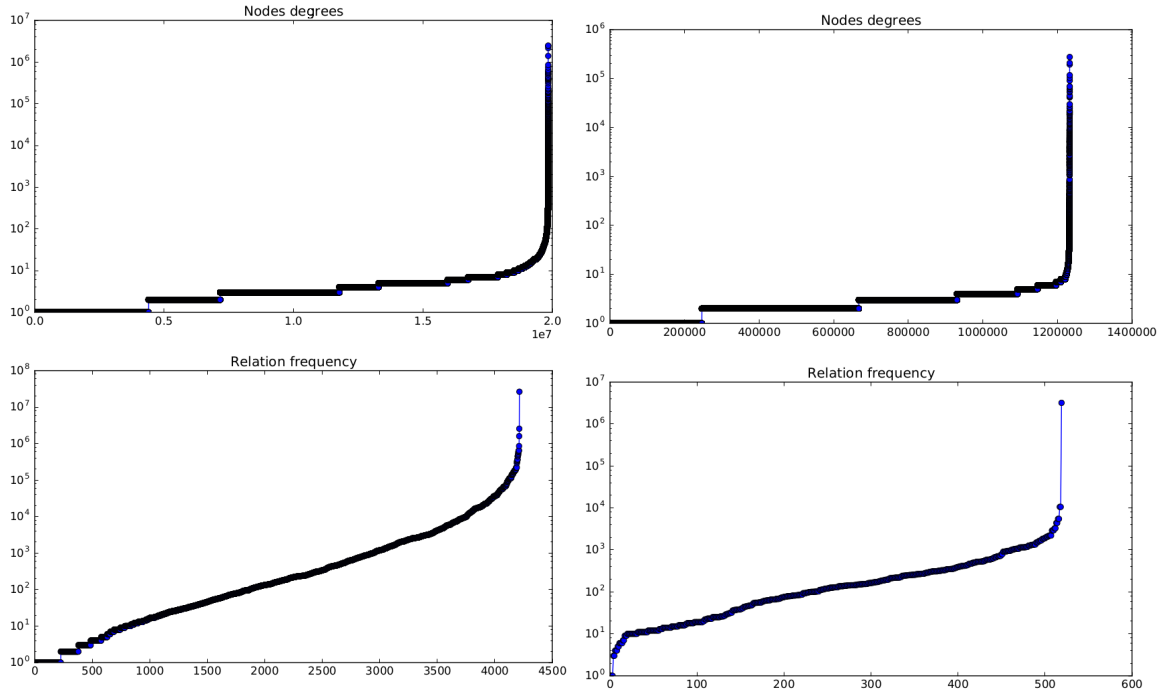
A knowledge graph (KG) is an extensional representation of a relation schema, where each instance of a relation type r corresponds to an edge connecting two nodes, a source s and a target t, usually represented as a triple: $< s, r, t >$. We replace this representation with an intesional representation, where we have only one edge for each relation type, and draw additional edges to capture set relations (intersection, subset, superset) between the (original graph's) relations' domain and ranges. These edges are weighed with the size of the overlap between the sets. Formally:

for a knowledge graph
$\mathcal{KG} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$
with:
vertices $\mathcal{V} = \{v_1, ..., v_n\}$,
relation types $\mathcal{R} = \{r_1, ..., r_k\}$
relation instances (i.e. edges)
$\mathcal{E} = \{(v_i, r_x, v_j) | v_i, v_j \in \mathcal{V}; r_x \in \mathcal{R}\}$,

we build the abstract graph
$\mathcal{KG}_A = (\mathcal{V}_A, \mathcal{E}_A, \mathcal{R}_A)$
with:
vertices $\mathcal{V}_A = \{V_{1,s}, V_{1,t}, V_{2,s}, V_{2,t}, ..., V_{k,s}, V_{k,t}\}$,
where:
the source node of relation r_i in the abstract graph is the *set of source nodes* (the domain) of relation r_i in $\mathcal{KG}$:
$V_{i,s} = \{v_x | (v_x, r_i, *) \in \mathcal{E}\}$
the target node of relation r_i in the abstract graph is *the set of target nodes* (the range) of r_i in $\mathcal{KG}$:
$V_{i,t} = \{v_x | (*, r_i, v_x) \in E\}$
relation types $\mathcal{R}_A = \mathcal{R} \cup \mathcal{R}_{set}$ where:
$\mathcal{R}$ is the set of relation types of $\mathcal{KG}$,
$\mathcal{R}_{set} = \{intersection, subset, superset\}$[1].
weighted edges
$\mathcal{E}_A = \{(V_{i,s}, r_i, V_{i,t}, 1) | r_i \in \mathcal{R}, V_{i,s}, V_{i,t} \in \mathcal{V}_A\}$
$\cup \{(V_{i,x}, r, V_{j,y}, w) | r \in \mathcal{R}_{set}, V_{i,x}, V_{j,y} \in \mathcal{V}_A$
$w = overlap(V_{i,x}, V_{j,y}))\}$

where the weight of a set relation between $\mathcal{KG}_A$'s nodes quantifies the overlap between the two sets:
$overlap(V_{i,x}, V_{j,y}) = \frac{|V_{i,x} \cap V_{j,y}|}{|V_{i,x}|}$

[1]There is no *equal* relation because if two sets are equal there will be only one node to represent them.



(a) Freebase (b) NELL

Figure 1: Knowledge graphs statistics on a logarithmic scale: relation and nodes frequencies for Freebase and NELL (the version used by (Gardner et al., 2014) and in this paper). Every data point is the degree of a node (top plots), or frequency of a relation (bottom plots). The data points are ordered monotonically, the x axis is just an index.

Building such a graph makes sense only for knowledge repositories that have strongly typed relations – like Freebase and NELL – but we do not require knowledge of the types of the relations' domains and ranges. Such information is not fine-grained enough: for example, the relation *capital* has a type *City* as a domain, but capital cities are a very small subset of the set of all cities. Using an "atomic" node to represent the domain/range of a relation would not allow us to make finer grained connections and distinctions between the domains and ranges of the existing relations.

Figure 2 shows a subset of the abstract graph built from the Freebase dataset. The blue edges are set relations – intersection, superset, subset – between the domains and ranges of a subset of the relations in the dataset. The black edges correspond to the actual relations in the dataset.

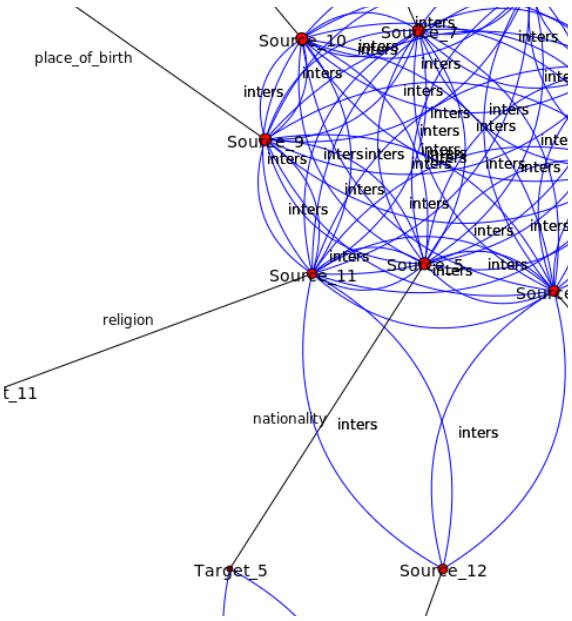


Figure 2: An abstract graph built on a subset of the Freebase dataset. The blue edges are set relations between the domains and ranges of the included relations, the black edges are the actual relations from the dataset.

3.2 Abstract paths

The Path Ranking Algorithm formalism originally proposed by (Lao and Cohen, 2010) performs two main steps to represent of a pair of nodes in a graph: (i) feature selection – adding paths that connect the node pair; (ii) feature computation –

KB variation	Freebase				NELL			
	Original graph		Abstract graph		Original graph		Abstract graph	
	# nodes	# edges	# nodes	# edges	# nodes	# edges	# nodes	# edges
KB	20M	67M	4086	22,946	1.2M	3.4M	587	2746
KB + SVO	30M	97M	35,905	1.7M	20M	71M	68,149	512,503
KB + Vector SVO	30M	97M	4112	23,257	1.3M	4.3M	613	3383
KB + Clustered SVO	30M	125M	4138	24,098	1.3M	3.9M	639	3818

Table 1: Graph statistics on the datasets used by (Gardner et al., 2014), and their abstract versions

associating a value for each added path.

Obtaining paths from a large graph is a computationally intensive problem, particularly in graphs that have numerous nodes with high degrees. Figure 1a shows that about 60% of Freebase nodes have degree higher than 10, which leads to an exponential growth in the number of paths starting in a node. Algorithms that harness path information often mine paths either by performing costly random walks (Guu et al., 2015), traversals (Gardner et al., 2014; Neelakantan et al., 2015; Das et al., 2016) or by constructing paths through generative models (Das et al., 2017; Ding et al., 2018). Here, we adopt a different approach, by abstracting the graph first, then finding paths in this graph through traversal algorithms.

For a relation r_i, we start at its domain (source) node $V_{i,s}$ and search for a path to its range (target) node $V_{i,t}$ using breadth first search. We constrain this path to contain at most k "proper" relations[2], and we do not allow consecutive set relations, thus forcing the algorithm to move from one "proper" relation to another through a set relation that connects the range of one with the domain of the next. An abstract path, just like a meta-path extracted by previous work, is a sequence of relation types: $\pi_j =< r_{j,1}, r_{j,2}, ...r_{j,m} >$, some of which are "proper" relations, some are set relations.

Because of the more general view of the graph, we lose information about individual paths (i.e. instances of a path in the original graph). Because of this, the paths we extract are hypothetical, but will have associated a confidence score based on the frequency of occurrence of relations in the original KG, and the strength of the connection of the range of one relation on the path with the domain of the next one. The weight of an abstract path π_j is computed as:

$$w(\pi_j) = \prod_{i=1}^{m} w(r_{j,i})$$

[2]In our experiments we used $k = 5$

where the weight $w(r_{j,i})$ of an individual relation is defined based on whether $r_{i,j}$ is a "proper" relation or a set relation as:

$$w(r_{j,i}) = \begin{cases} \frac{|\{<*,r_{j,i},*>\in\mathcal{E}\}}{|\mathcal{E}|} & \text{if } r_{j,i} \in \mathcal{R} \\ overlap(r_{j,i}) & \text{if } r_{j,i} \in \mathcal{R}_{set} \end{cases}$$

We use this weight to rank abstract relations for potential filtering, and to compute the weight of its grounding for specific node pairs.

3.3 Grounded paths

The abstract paths are hypothetical paths that could connect the source s and target t of a $< s, r, t >$ tuple. They can be used in different ways, e.g. (i) as features in a link prediction system (e.g. (Gardner et al., 2014)), (ii) to fill in larger portions of the graph by producing, rather than finding, groundings of the path for specific instances.

In the work presented here we test the abstract paths through the link prediction task, so we will try to ground abstract paths for relation instances in the training and test data. After finding the set of abstract paths $\{\pi_{i,r}\}$ associated with a relation r, for a given instance of the relation $r - < s, r, t > -$ we can (try to) ground the paths as follows: (i) we first eliminate set relations from the abstract paths: at this point set relations between relation types domain and ranges are not useful (they were necessary only for the connectivity and search process in the abstract graph). Set relations have no counterpart in the extensional graph, as at this level nodes themselves make the connection between successive relations (ii) starting at the source node, we follow again a breadth first traversal, constraining at each step the type of relation to follow based on the "cleaned up" abstract path.

We compute the weight of a grounded path $gp =< v_0, r_{x_1}, v_1, ..., v_{l-1}, r_{x_l}, v_l >$ (where $v_0 = s$ and $v_l = t$) as a combination of the weight of the corresponding abstract path $\pi =< r_1, ..., r_m >$ ($r_{x_i} \in \pi$) and specific information for the current node pair (s, t):

$w(\pi) = \prod_{i=1}^{l} w(v_{i-1}, r_i, v_i)$

where the weights of the relations on the grounded path reflect the specificity of the relation to its source node:

$$w(v_{i-1}, r_i, v_i) = \begin{cases} \frac{1}{|\{<v_{i-1}, r_i, *> \in E\}|} & \text{if } r_i \in gp \\ 1 & \text{if } r_i \in \mathcal{R}_{set} \end{cases}$$

4 Experiments

Because we want to compare the abstract paths found using the abstract graph with paths found using PRA, we use the experimental set-up of (Gardner et al., 2014), where we replace the feature selection and feature computation steps with the approach presented here. A big difference will be caused by the negative sampling, which also makes the results not directly comparable. The issues are explained in the **negative sampling** paragraph below. The data thus obtained is used for training a linear regression model (similarly to (Gardner et al., 2014)), and tested on the provided test sets and evaluated using mean average precision (MAP).

4.1 Data

We build abstract graphs and paths from the Freebase and NELL data described in (Gardner et al., 2014). We then use the extracted paths for link prediction.

The graphs built by Gardner et al. (2014) cover several variations, where the KGs were enhanced with $< subject, verb, object >$ triples extracted from dependency parses of ClueWeb documents. Table 1 shows the statistics for each original and abstract graph. The generated abstract graph is several degrees of magnitude smaller compared to the original KG. The abstract graph approach we present here does not fit well the combination of the knowledge base (Freebase or NELL) with unstructured SVO triples, because we rely on strongly typed relations to build node sets. The SVO triples bring in numerous low frequency relations, that without additional processing are not beneficial. The results presented by Gardner et al. (2014) show that this configuration very rarely (and never overall) leads to better results than the other graph variations. The numerous relation types brought in by the SVO triples also lead to high computation time for the abstract graph: its shortcoming is the computation of set relations between the different relations' domains and ranges, which grows quadratically with the number of relation types. We will skip this graph variation in the rest of the experiments presented here.

KG	Avg. no. inst	min	max
NELL train	650.7	81	1468
NELL test	163.2	21	367
Freebase train	122.9	10	600
Freebase test	41.6	4	200

Table 2: Statistics on the size of the training and test sets

Gardner et al. (2014) use these graphs to generate paths for augmenting the representation of node pairs, for link prediction, for a subset of 24 relation types from Freebase's inventory, and 10 relations from NELL. Each relation has a training and test set, whose numbers vary quite a bit, as shown through the statistics in Table 2.

Negative sampling The number of negative instances used in (Gardner et al., 2014) is not clearly stated. Both the number and methods of generating the negative samples can impact the results (Kotnis and Nastase, 2018). We use (up to) 200 negative samples for each positive pair: for a pair (s, t) in the provided training or test sets for each relation r, we make 100 negative samples by corrupting the source s, and 100 negative samples by corrupting the target t. The corrupted s' and t' are chosen from r's domain $V_{r,s}$ and range $V_{r,t}$ respectively, such that these corrupted triples are not part of the training, test or graph. If 100 instances do not exist, we extract as many as possible.

$$Neg(s, r, t) = \{(s', r, t) | s' \in V_{r,s}, (s', r, t) \notin \mathcal{E}\}$$
$$\cup \{(s, r, t') | t' \in V_{r,t}, (s, r, t') \notin \mathcal{E}\}$$

Because the relations are strongly typed, producing negative instances by corrupting the source/target nodes from the relation's domain and range leads to difficult negative instances. Instances with source and target nodes that don't match the argument types of the direct relation we want to predict can be filtered out before the link prediction.

Representing instances For each of these 24 Freebase and 10 NELL relations we mine paths in the abstract graph using depth first traversal. An example of abstract path found for the NELL rela-

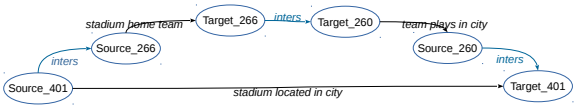


Figure 3: An abstract path for relation *StadiumLocatedInCity* from NELL

tion *StadiumLocatedInCity* is shown in figure 3.

Each of the 24 Freebase and 10 NELL relations has a set of training and test examples. After building abstract paths, for each instance $< s, r, t >$ in these datasets we will ground the corresponding abstract paths as described in Section 3.3. For each relation type the set of features representing the corresponding data will be twice the number of abstract paths. We produce two features for each abstract path: one that is the weight of this path, and one that is the weight of its grounding for a given relation instance. If a relation instance does not have a grounding for an abstract path, the values of these features will be 0.

4.2 Results and discussion

The overall results of the experiments are presented in Table 3, and the relation-level results are in Tables 4 for NELL, and 5 for Freebase.

graph	Freebase		NELL	
	MAP_G	MAP_{KG_A}	MAP_G	MAP_{KG_A}
KB	0.278	0.186	0.193	0.246
KBCl	0.326	0.233	0.276	0.411
KBVec	0.350	0.223	0.301	0.306

Table 3: Results on the three graph variations of Freebase and NELL as reported by (Gardner et al., 2014) (G) and using abstract graphs (KG_A).

Overall, the results indicate that enhancing Freebase and NELL with additional facts from textual sources leads to better results, particularly when these additional facts ($< subject, verb, object >$ triples) are processed and clustered using low dimensional dense representations (Gardner et al.; Gardner et al. (2014; 2013) use embeddings obtained by running PCA on the matrix of SVO triples).

Freebase has 4200+ relation types, and NELL 500+. More than 500 relation types in Freebase have less than 10 instances, wheres NELL does not have this issue (see Figures 1a and 1b). Because we test the approach for knowledge graph completion using classification based on the patterns as features, having features that appear too few times will not help the system find a robust model. For the purpose of the presented experiments we filter the Freebase abstract graph to use only relation types that have at least 10 instances (Table 1 shows the statistics for this configuration).

Relation	PRA best	KB	KB Cl	KB Vec
ActorStarredInMovie	**0.037**	0	0	0
AthletePlaysForTeam	**0.589**	0.145	0.089	0.136
CityLocatedInCountry	**0.347**	0.078	0.071	0.057
JournalistWritesForPub.	0.319	0.317	**0.515**	0.436
RiverFlowsThroughCity	0.076	0.027	**0.146**	0.058
SportsTeamPos.ForSport	0.217	0	**0.615**	0
StadiumLocatedInCity	0.321	0.316	**0.414**	0.110
StateHasLake	0.000	0	**0.688**	0.681
TeamPlaysInLeague	**0.947**	0.910	0.916	0.917
WriterWroteBook	0.202	**0.661**	0.659	0.661

Table 4: Relation results for the NELL KB. The second column is the best result for each relation reported by (Gardner et al., 2014).

It is not surprising that overall the results for NELL are higher – NELL has been designed on the principle of coupled learning, where connections between different relations are the basis of the resource and its continuous growth (Carlson et al., 2010b). It also has more training data for each relation (see table in Section 4.1). There is no consistent trend – for some relations using the paths extracted with this approach leads to better results, for others it does not (although, as we frequently mentioned, the fact that we used different negative sampling methods, the results are not directly comparable).

A more complete picture emerges when we look at the paths found, and compare them with the paths obtained with the PRA approach[3]. For all Freebase KG configurations, Gardner et al. (2014) have 1000 paths for most relations (approx. 6 of the relations have between 230 and 973). For NELL the number varies more, between 58 and 5509, 6 of the relations have more than 1000 meta-paths. With the abstract graphs the numbers are much lower. For Freebase we find between 1 and 258 abstract paths, most of the relations (21) having fewer than 30 abstract paths for all KG configurations. For NELL we find between 1 and 157 paths, 5 of the relations having more than 100 ab-

[3]We used the archive shared by Matt Gardner https://github.com/matt-gardner/pra, in particular the translated paths for each relation.

Relation	PRA best	KB	KB Cl	KB Vec
/amusement/parks/park_rides	0.013	**0.503**	0.503	0.503
/arch./arch./ struc._designed	**0.376**	0	0	0
/astronomy/constel./contains	0.017	**0.503**	0.503	0.503
/autom./auto._class/examples	0.006	0	0	0
/autom./model/auto._class	**0.768**	0.009	0.009	0.009
/aviation/airline/hubs	**0.336**	0.279	0.279	0.330
/book/literary_series/author	**0.830**	0.461	0.461	0.461
/comp./sw_genre/sw._in_genre	0.001	**0.002**	0.002	0.002
/edu./field_of_study/ journals_in_this_disc.	0.003	**0.005**	0.005	0.005
/film/film/rating	**0.914**	0.087	0.096	0.136
/geo./island/body_of_water	**0.602**	0.286	0.286	0.286
/geo./lake/basin_countries	**0.437**	0.083	0.075	0.112
/geo./lake/cities	0.177	0.003	**0.442**	0.442
/geo./river/cities	0.066	0	**0.127**	0.127
/ice_h./h._player/h._position	**0.364**	0.007	0.007	0.007
/loc./adm._division/ country	**0.991**	0.189	0.199	0.199
/medicine/disease/symptoms	0.078	0.035	**0.088**	0.060
/medicine/drug/drug_class	0.169	0	**0.212**	0.002
/people/ethnicity/lang._spoken	**0.226**	0.128	0.135	0.115
/spaceflight/astronaut/miss.	**0.848**	0.272	0.272	0.272
/transp./bridge/body_of_water_spanned	**0.727**	0.190	0.384	0.384
/tv/tv_prog._cr./prog._created	0.181	**0.646**	0.646	0.646
/vis._art/art_period_movement/assoc._artists	0.046	0.318	**0.340**	0.340
/vis._art/vis._artist/assoc._per._or_mov.	0.295	0.474	0.509	**0.516**

Table 5: Statistics of the number of instances in the training and testing sets for the relations analyzed, and the number of paths extracted for each set (in parentheses the number of abstract paths for each graph).

stract paths. The overlap between the sets of paths discovered with the two methods is very small: for Freebase the average overlap with respect to PRA is around 0.004 (for the different graph configurations), and with respect to the abstract paths around 0.2; for NELL around 0.003 relative to PRA and 0.27 relative to the abstract paths.

We note that overall, the system found more paths than what could be grounded for the given training instances for both Freebase and NELL. Another general observation is that relations for which we found the most patterns (*AthletePlaysForTeam* and *StateHasLake* for NELL, */medicine/disease/symptoms* and */film/film/rating* for Freebase) do not necessarily perform the best.

NELL The results for each relation in terms of average precision are presented in Table 4. We include the best result on PRA (on any variation of the graph), as reported by (Gardner et al., 2014), although since we used different negative instances the results are not directly comparable. Several of the NELL target relations have interesting patterns in the abstract graph, in particular StadiumLocatedInCity, TeamPlaysInLeague. In several cases, the algorithm has discovered "parallel" relations. For the relation *WriterWroteBook*, the most useful feature is the relation *AgentCreated*, which connects many of the source-target pairs in the *WriterWroteBook* relation. We found a similar situation with the relation *JournalistWritesForPublication*, which has *WorksFor* paralleling it in the graph.

Looking at specific relations, the paths extracted from the abstract graph are more focused. An example of this is the relation *StadiumLocatedInCity*. Numerous paths detected by PRA seem irrelevant, as illustrated by the following (highest frequency) paths:

$$\begin{array}{l} \underline{generalizations \rightarrow generalizations^{-1}} \\ generalizations \rightarrow generalizations^{-1} \\ \quad \rightarrow \underline{generalizations \rightarrow generalizations^{-1}} \\ generalizations \rightarrow generalizations^{-1} \\ \quad \rightarrow \underline{CityHotels} \\ generalizations \rightarrow generalizations^{-1} \\ \quad \rightarrow \underline{StadiumLocatedInCity} \\ generalizations \rightarrow generalizations^{-1} \\ \quad \rightarrow BuildingLocatedInCity \end{array}$$

The paths found in the abstract graph, as the example in Figure 3 shows, seem to capture more informative relation interdependencies.

Our system does not always find high quality patterns. It also finds surprising and most probably idiosyncratic patterns. In particular, for the *StateHasLake* relation, from the paths found, some very unexpected ones had groundings for the given training data:

$$\begin{array}{l} Agric.Prod.GrowingInStateOrProv.^{-1} \\ \quad \rightarrow Agric.Prod.GrowingInStateOrProv. \\ \quad\quad \rightarrow StateHasLake \end{array}$$

$$\begin{array}{l} MaleMovedToStateOrProv.^{-1} \\ \quad \rightarrow MaleMovedToStateOrProv. \\ \quad\quad \rightarrow StateHasLake \end{array}$$

While the first rule could be justified (having lakes may favour the growing of certain types of agricultural products), the second one seems completely accidental. With a stronger filtering method based on the computed path scores we could eliminate some of these false patterns.

Paths extracted using PRA
$/type/object/type \rightarrow /type/object/type^{-1} \rightarrow /film/content\_rating/film^{-1}$
$/film/performance/film^{-1} \rightarrow /type/object/type \rightarrow /type/object/type^{-1} \rightarrow /film/performance/film \rightarrow /film/film/rating$
$/type/object/type \rightarrow /type/object/type^{-1} \rightarrow /film/film/rating$
$/film/performance/film^{-1} \rightarrow /type/object/type \rightarrow /type/object/type^{-1} \rightarrow /film/performance/film \rightarrow /film/content\_rating/film^{-1}$
$/film/film\_genre/films\_in\_this\_genre^{-1} \rightarrow /film/film/genre^{-1} \rightarrow /film/film/rating$
$/film/film/genre \rightarrow /film/film/genre^{-1} \rightarrow /film/film/rating$
$/film/film/language \rightarrow /film/film/language^{-1} \rightarrow /film/film/rating$
Paths extracted using abstract graphs
$/film/film/edited\_by \rightarrow /film/editor/film \rightarrow /film/film/rating$
$/film/film/directed\_by \rightarrow /film/producer/film \rightarrow /film/film/rating$
$/film/film/cinematography \rightarrow /film/cinematographer/film \rightarrow /film/film/rating$
$/film/film/costume\_design\_by \rightarrow /film/film/costumer\_designer\_costume\_design\_for\_film \rightarrow /film/film/rating$
$/film/film/music \rightarrow /film/music/contributor\_film \rightarrow /film/film/rating$
$/film/film/film\_production\_design\_by \rightarrow /film/film\_prod.\_designer/films\_prod.\_designed \rightarrow /film/film/rating$

Table 6: Sample relations extracted using PRA and abstract graphs, respectively

Freebase The fine-grained results for Freebase, in terms of average precision, are presented in Table 5. We make the same observation as for NELL – for several relations, the paths obtained from the abstract graph are different and more focused than the PRA ones. For the relation */film/film/rating* for which the PRA approach gives very high results with the abstract graph has lower scores, some of the highest scoring paths found by the PRA are presented in Table 6. For comparison we also include the highest rated paths obtained using the abstract graph. While some of these paths were also found by the PRA, they are much lower in the list of extracted paths. The highest weighted paths found in the abstract graph connect specific properties of films with their rating.

An archive containing the abstract graphs, the abstract paths, the train/test data, negative samples and the groundings of the abstract paths for these relations for the variations of Freebase and NELL presented here is available from the University of Heidelberg[4].

[4] https://www.cl.uni-heidelberg.de/english/research/downloads/resource_pages/AbstractGraphs/AbstractGraphs.shtml

5 Conclusions

We proposed and evaluated a method for obtaining paths from large knowledge graphs by compressing them into their intensional versions. We relied on the fact that these graphs have strongly typed relations, such that their domain and ranges consist of homogeneous sets that have overlaps only with the domains and ranges of a small number of other relations. This compression step leads to a smaller graph to work with, where we found paths that seem to capture qualitative patterns in the data. The results on link prediction on Freebase and NELL show the advantage of using such paths for some of the relations, but the task does not showcase the full potential of this representation. Further work will explore the potential of such patterns as explanatory links between directly connected nodes, or as a source of additional patterns for filling in the knowledge graphs not only with missing links, but also missing nodes, either by predicting intermediate nodes or by using the paths as patterns for targeted information extraction.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010a. Toward an architecture for never-ending language learning. In *AAAI*.

Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka, Jr., and Tom M. Mitchell. 2010b. Coupled semi-supervised learning for information extraction. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 101–110, New York, NY, USA. ACM.

Kai-Wei Chang, Wen-tau Yih, Bishan Yang, and Christopher Meek. 2014. Typed tensor decomposition of knowledge bases for relation extraction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1568–1579. Association for Computational Linguistics.

Wenhu Chen, Wenhan Xiong, Xifeng Yan, and William Wang. 2018. Variational knowledge graph reasoning. *arXiv preprint arXiv:1803.06581*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2017. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv preprint arXiv:1711.05851*.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2016. Chains of reasoning over entities, relations, and text using recurrent neural networks. *arXiv preprint arXiv:1607.01426*.

Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving knowledge graph embedding using simple constraints. *arXiv preprint arXiv:1805.02408*.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *KDD*.

Matt Gardner and Tom Mitchell. 2015. Efficient and expressive knowledge base completion using subgraph feature extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1488–1498. Association for Computational Linguistics.

Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 397–406, Doha, Qatar. Association for Computational Linguistics.

Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom Mitchell. 2013. Improving learning and inference in a large knowledge-base using latent syntactic cues. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 833–838, Seattle, Washington, USA. Association for Computational Linguistics.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327. Association for Computational Linguistics.

Bhushan Kotnis and Vivi Nastase. 2017. Learning knowledge graph embeddings with type regularizer. In *Proceedings of the Knowledge Capture Conference*, K-CAP 2017, pages 19:1–19:4. ACM.

Bhushan Kotnis and Vivi Nastase. 2018. Analysis of the impact of negative sampling on link prediction in knowledge graphs. In *Workshop on Knowledge Base Construction, Reasoning and Mining (KBCOM)*.

Ni Lao and William W. Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.*, 81(1):53–67.

Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 529–539, Stroudsburg, PA, USA. Association for Computational Linguistics.

Changping Meng, Reynold Cheng, Silviu Maniu, Pierre Senellart, and Wangda Zhang. 2015. Discovering meta-paths in large heterogeneous information networks. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 754–764.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International*

Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 156–166. Association for Computational Linguistics.

Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 1955–1961. AAAI Press.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2012. Factorizing yago: Scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 271–280, New York, NY, USA. ACM.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 926–934. Curran Associates, Inc.

Théo Trouillon, Christopher R Dance, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2017. Knowledge graph completion via complex tensor factorization. arXiv preprint arXiv:1702.06879.

William Yang Wang, Kathryn Mazaitis, and William W. Cohen. 2013. Programming with personalized Pagerank: A locally groundable first-order probabilistic logic. In *Proceedings of the 22Nd CIKM*, pages 2129–2138, New York, NY, USA. ACM.

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 *Proceedings of the 2015 International Conference on Representation Learning*.

Wenpeng Yin, Yadollah Yaghoobzadeh, and Hinrich Schütze. 2018. Recurrent one-hop predictions for reasoning over knowledge graphs. In *Proceedings of the 27th ACL*, pages 2369–2378. Association for Computational Linguistics.

Mengfei Zhou and Vivi Nastase. 2018. Using patterns in knowledge graphs for targeted information extraction. In *Workshop on Knowledge Base Construction, Reasoning and Mining (KBCOM)*.