Knowledge base completion using distinct subgraph paths

Knowledge Base Completion Using Distinct Subgraph Paths

Sameh K. Mohamed
Insight Centre for Data Analytics
National University of Ireland Galway
sameh.kamal@insight-centre.org

Vít Nováček
Insight Centre for Data Analytics
National University of Ireland Galway
vit.novacek@insight-centre.org

Pierre-Yves Vandenbussche
Fujitsu Ireland Ltd.
pierre-yves.vandenbussche@ie.fujitsu.com

ABSTRACT

Graph feature models facilitate efficient and interpretable predictions of missing links in knowledge bases with network structure (*i.e.* knowledge graphs). However, existing graph feature models—*e.g.* Subgraph Feature Extractor (SFE) or its predecessor, Path Ranking Algorithm (PRA) and its variants—depend on a limited set of graph features, connecting paths. This type of features may be missing for many interesting potential links, though, and the existing techniques cannot provide any predictions at all then. In this paper, we address the limitations of existing works by introducing a new graph-based feature model – Distinct Subgraph Paths (DSP). Our model uses a richer set of graph features and therefore can predict new relevant facts that neither SFE, nor PRA or its variants can discover by principle. We use a standard benchmark data set to show that DSP model performs better than the state-of-the-art – SFE (ANYREL) and PRA – in terms of mean average precision (MAP), mean reciprocal rank (MRR) and Hits@5, 10, 20, with no extra computational cost incurred.

CCS CONCEPTS

• **Semantic networks**; • **Machine Learning**; • **Machine learned ranking**;

KEYWORDS

Knowledge Graphs, Path Ranking, Knowledge Base Completion

ACM Reference Format:
Sameh K. Mohamed, Vít Nováček, and Pierre-Yves Vandenbussche. 2018. Knowledge Base Completion Using Distinct Subgraph Paths. In *SAC 2018: SAC 2018: Symposium on Applied Computing , April 9–13, 2018, Pau, France.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3167132.3167346

1 INTRODUCTION

Large scale knowledge graphs (*i.e.* graph-structured knowledge bases) have been used as convenient means for modelling information in many different domains, including general human knowledge [12], biomedical information [4] and language lexical information [15]. Knowledge graphs are now used by different applications such as enhancing semantics of search engine results [22, 26], biomedical discoveries [18], or powering question answering and decision support systems [5]. Despite the huge volume of information stored in knowledge graphs, they are still incomplete [23]. For example, 65% to 99% of entities in most of the popular knowledge bases like Freebase [1] and YAGO3 [14] lack at least one property possessed by all other entities in the same class [23]. Incompleteness of knowledge bases can substantially affect the efficiency of systems relying on them, which has motivated research in knowledge base completion via automatic prediction of new, implicit facts.

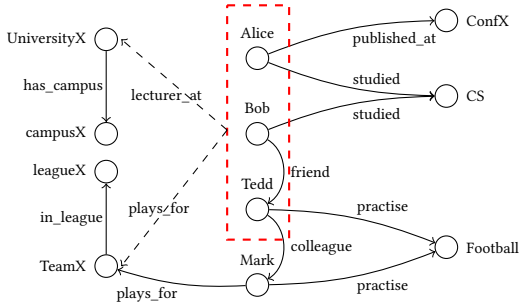

Figure 1: A sample of a graph about people and their professions.

This work addresses a family of knowledge base completion models known as graph feature models, which use graph patterns as features. One of the early models in this family is Path Ranking Algorithm (PRA) [10], which uses paths connecting pairs of nodes as indicating features for predicting new direct links between nodes. For example, in Fig. 1, PRA can predict the fact that *Tedd* is playing for *TeamX* using the path $\langle colleague, plays\_for \rangle$ along with the path $\langle practise, practise^{-1}, plays\_for \rangle$ where $practise^{-1}$ is the inverse of relation *practise*. PRA extracts these connecting path features using random walks linking the subject and object nodes. Then, it uses each random walk probability as a value in a feature vector corresponding to the subject and object. This technique is able to provide expressive prediction for new facts. However, it suffers from low efficiency, and high computational cost of computing random walk probabilities. An extension of PRA uses backward random walks [11] to extract paths originating from object and reaching the subject node like the path $\langle plays\_for^{-1}, colleague^{-1} \rangle$ in Fig. 1, which resulted in an efficiency improvement over traditional PRA path features. Other extensions suggest using latent feature representations as a support, like latent syntactic cues [8] which introduces a latent feature representation of combination of relations to infer new ones. Another approach suggests incorporating similarity between latent representation of relations as support features for knowledge base completion [9], leading to significant efficiency improvements for models based on random walk inference. However, they suffer from the same computational problems as for PRA. Later, Neelakantan et

SAC 2018, April 9–13, 2018, Pau, France

ACM ISBN 978-1-4503-5191-1/18/04…$15.00
https://doi.org/10.1145/3167132.3167346

al. introduce path bigram features [19] for connecting paths. This work leads to significant improvement in the efficiency of PRA. Furthermore, Subgraph Feature Extraction (SFE) [7] – the state of the art model – drops random walk probabilities, and uses a binary representation of paths. SFE also proposes using ANYREL features, which is a set of subgraph paths built from connecting paths by replacing relation instances with a wild card. This supports a richer representation of connecting paths, allowing for more intersections of similar connecting paths through the ANYREL wild card that replaces relation instances in connecting paths.

Despite the improvements achieved by SFE in path extraction and representation, all reviewed methods still suffer from two interrelated problems. Firstly, they represent facts using a limited feature set, *i.e.* connecting paths. This only captures interaction between subject and object entities, and neglects information describing entities themselves (like other subgraph paths to neighbour nodes that can capture entity attributes and properties). This can lead to sub-optimal predictions. The second problem is that the current methods totally disregard entity pairs with no connecting paths in between as a consequence of using a limited feature set. This means that the methods cannot make certain predictions at all.

We propose a new model called Distinct Subgraph Paths (DSP) model. The model uses a new set of features that describe distinct properties of entities using disjoint sets of subgraph paths for both subject and object entities. For example, in Fig. 1, while investigating the fact ⟨*Alice*, *lecturer_at*, *UniversityX* ⟩, we propose using subgraph paths to express properties of subject and object entities *Alice* and *UniversityX*. *Alice* can be expressed using the path set: [⟨*SUB:published_at*⟩, ⟨*SUB:studied*⟩, ⟨*SUB:studied,studied*$^{-1}$⟩], and *UniversityX* using: [⟨*OBJ:has_campus*⟩]. These disjoint path sets provide distinct description of both subject and object entities for a given fact. This makes our model capable of:

(1) Employing a richer set of features that can describe properties of candidate fact entities.
(2) Providing ranking scores for node pair candidates in the absence of connecting paths.
(3) Providing better results than PRA and SFE (ANYREL) in terms of mean average precision (MAP), mean reciprocal rank (MRR) and Hits@5, 10, 20 with no extra computational cost.

Organization. The rest of this paper is organized as follows. Section 2 motivates the use of the new feature set. Section 3 presents the proposed model. Section 4 describes experiments validating the model. Section 5 discusses the experimental results. Section 6 reviews related works and highlights the contributions of this paper. Section 7 concludes the paper and outlines the future work.

2 MOTIVATING EXAMPLE

The task of knowledge base completion is a ranking task by nature, since the aim is to find the most probable absent true facts in a knowledge base [27]. For example, in Fig. 1, a knowledge base completion task would aim at ranking possible facts about people and their workplaces. In the absence of connecting paths between entities, such as between {*Alice*, *Bob*, *Tedd*} and *UniversityX*, approaches like SFE, PRA and its variants would not be able to provide a corresponding relation score. Indeed these methods rely on the assumption that since non-connected nodes have no connecting paths, they have no direct relationship. However, absence of connecting paths can be a result of knowledge incompleteness.

Candidates of relation: *lecturer_at*				
Rank	**Subject**		**Object**	
#	**Entity**	**Rel.**	**Entity**	**Rel.**
1	Alice	High	UniversityX	High
2	Bob	Med.	UniversityX	High
3	Tedd	Low	UniversityX	High
4	Alice	High	TeamX	Low
5	Bob	Med.	TeamX	Low
6	Tedd	Low	TeamX	Low

Table 1: Example of candidates' ranking for the relation *lecturer_at* as per knowledge graph from Fig. 1 and their relevance (Rel.) to the relation.

In our approach we consider using a set of features that we call distinct subgraph paths (DSP) as support features for ranking candidate absent facts. Distinct subgraph paths are the union of the two sets of subgraph paths originating from subject and object nodes of a triple in a knowledge graph, each prefixed with a distinct label corresponding to its origin ("*SUB:*" for subject or "*OBJ:*" for object). For example, when investigating the relation *lecturer_at* between persons group of entities and corresponding workplace entities, our model will be able to use the presence of path features like ⟨*SUB:published_at*⟩ and ⟨*SUB:studied*⟩ to predict that *Alice* has a high probability of being a subject for the relation. Similarly for other entities in our example, subgraph paths can be used to rank candidate facts as in Table 1. This provides a rank of most relevant entities to be connected with a specific relation. As per our example, *Alice* who has *studied*, and *published_at* a conference, is more likely to be lecturing than *Bob* who only has *studied* or *Tedd* who has none of these relative attributes. Also, the fact that each one of them can be lecturing at *UniversityX* is more probable than lecturing at *TeamX*, as attributes of *UniversityX* like *has_campus* is more relevant to objects of relation *lecturer_at* than those for *TeamX*. Therefore, this approach of using distinct sets of subgraph paths[1] for subject and object entities for a specific relation can support the construction of ranking scores for node pair candidates even in the absence of connecting paths as shown in Table 1.

However, unlike connecting paths features used in PRA or SFE, subgraph paths do not capture the interactions (connecting paths) between entity pairs. Therefore, we use a combination of both DSP and ANYREL connecting paths features to support ranking node pair candidates whether they are connected or not in the knowledge graph.

3 DISTINCT SUBGRAPH PATHS MODEL

In this section, we focus on the technical description of DSP model. We present how DSP model extracts feature paths from knowledge graphs, how the model is learned, and how DSP model predict a

[1] In the context of feature extraction for knowledge graph triples, we define a subgraph path as any path originating from candidate fact subject or object nodes.

Algorithm 1 EXTRACT SUBGRAPH PATHS

Input: v node, depth d, knowledge graph $\mathcal{G}$, root path R
Output: v^{sg} Subgraph paths

if d = 0 **then**
  return $\langle v \rangle$
else
  v^{sg} = [ ]
  for $(r, u) \in \Gamma(v, \mathcal{G})$ **do**
    $R_u = R + \langle r_k, u_k \rangle$
    v_u^{sg} =ExtractSubgraphPaths(u, d-1, $\mathcal{G}$, R_u)
    $v^{sg} = v^{sg} \cup v_u^{sg} \cup R_u$
  return v^{sg}

score for new absent facts. DSP model training operates in two phases. First, it extracts path features for each node pair instance consisting of both connecting path (ANYREL) and subgraph path (DSP) features. Both of these features sets capture different properties of candidate facts. Subgraph paths of subject and object entities capture the relevance between these two entities and the considered relation, while connecting paths between subject and object entities capture their interactions. DSP model uses these two types of features to build a feature matrix with binary representation of path features for each relation type. In the second phase, DSP model trains a binary classifier for each feature matrix and uses this model for later predictions. Further description of the how it works follows in the next subsections.

3.1 Feature Extraction

Let $\mathcal{G}$ be a knowledge graph, $\Gamma(e, \mathcal{G})$ be the set of neighbour links of entity e in a knowledge graph $\mathcal{G}$, where a link is a combination of a neighbour relation r and neighbour node u reached by this relation, $(l_1 + l_2)$ be the concatenation of two lists l_1 and l_2, and p_u be a path p going through node u.

First, DSP model extracts subgraph paths of both subject and object nodes using Depth-First Search as in Algorithm 1, then it labels these paths with distinct prefixes corresponding to their origin node ("*SUB:*" for subject or "*OBJ:*" for object) using a labelling function $\gamma(p, label)$. Then, DSP model combines subgraph path originating from subject and object entities that share a common target node to build connecting paths. Let $\tau(p)$ be the target node of path p that if p starts from node v to node u, then $\tau(p) = u$, $P_{s \rightsquigarrow t}$ be a path from node s to node t, and p^{-1} be the inverse of path p. An inverse of a path is obtained by inversing the order of path relations and changing their direction, that is if $p = \langle r_1, r_2^{-1}, r_3 \rangle$, then $p^{-1} = \langle r_3^{-1}, r_2, r_1^{-1} \rangle$. DSP model combines subgraph paths originating from subject node with the inverse of path originating from object node (providing they share a common target node) to build a connecting path from subject to object. For example, a subject subgraph path $p_{s \rightsquigarrow t} = \langle r_1, r_2^{-1} \rangle$ and object subgraph path $p_{o \rightsquigarrow t} = \langle r_5, r_3^{-1} \rangle$ are combined to generate a connecting path $p^{cp}_{s \rightsquigarrow o} = p_{s \rightsquigarrow t} \oplus p^{-1}_{o \rightsquigarrow t} = \langle r_1, r_2^{-1}, r_3, r_5^{-1} \rangle$. After, DSP model builds ANYREL paths corresponding to extracted connecting paths and label them with prefix label "*ANYREL:*". This procedure of extracting DSP and ANYREL path features is described in Algorithm 2.

For example, when DSP model extracts features for the fact (*Tedd, plays_for, TeamX*) from Fig. 1, it extracts subgraph paths around

Algorithm 2 EXTRACT FEATURE PATHS

Input: (s,t) node pair, path length l, knowledge graph $\mathcal{G}$
Output: $P^{ANYREL}_{s \rightsquigarrow t}$, $P^{DSP}_{s \rightsquigarrow t}$ feature paths

s^{sg} = ExtractSubgraphPaths(s, $\lceil l/2 \rceil$, $\mathcal{G}$, [])
t^{sg} = ExtractSubgraphPaths(t, $\lceil l/2 \rceil$, $\mathcal{G}$, [])
$P^{DSP}_{s \rightsquigarrow t} = \gamma(s^{sg}, "sub") \cup \gamma(t^{sg}, "obj")$
$T_s = \{\ \tau(p) \mid p \in s^{sg}\ \}$
$T_t = \{\ \tau(p) \mid p \in t^{sg}\ \}$
$T_c = T_s \cap T_t$
$P^{cp}_{s \rightsquigarrow t}$ = [ ]
$P^{ANYREL}_{s \rightsquigarrow t}$ = [ ]
for $t \in T_c$ **do**
  for $p_s \in s^{sg} \wedge \tau(p_s) = t$ **do**
    for $p_t \in t^{sg} \wedge \tau(p_t) = t$ **do**
      $P^{cp}_{s \rightsquigarrow t} = P^{cp}_{s \rightsquigarrow t} \cup (p_s \oplus p_t^{-1})$
$P^{ANYREL}_{s \rightsquigarrow t} = \bigcup_{p \in P^{cp}_{s \rightsquigarrow t}}$ AnyrelPaths(p)
return $P^{ANYREL}_{s \rightsquigarrow t} \cup P^{DSP}_{s \rightsquigarrow t}$

subject and object entities *Tedd* and *TeamX*. The union of these two sets of subgraph paths constitutes DSP feature paths. After, DSP model uses common target nodes to subject and object entities *Tedd* and *TeamX* such as *Mark* and *Football* to extract connecting paths. For example, DSP model combines paths to common target node *Football*: ⟨*practise*⟩ that originates from subject node and ⟨*plays_for,practise*⟩ that originates from object node, by appending the subgraph path from subject entity ⟨*practise*⟩to the inverse of object entity path ⟨$practise^{-1}$,$plays\_for^{-1}$⟩. This results in a connecting path ⟨*practise*,$practise^{-1}$,$plays\_for^{-1}$⟩.

DSP model uses the same connecting paths extraction process as SFE, and extends it with distinct subgraph paths for subject and object nodes as discussed. Despite its high complexity, the feature extraction process is embarrassingly parallel and can therefore be distributed to minimise computational cost. It is important to note that feature extraction phase of DSP model requires no extra computational overhead compared to SFE since proposed DSP subgraph features are already extracted while generating connecting paths as shown in Algorithm 2.

3.2 Model Learning

For each candidate fact, the DSP model extracts distinct subgraph and ANYREL paths features to build a feature matrix based on the union of the two paths feature sets. These features represent the column names of the feature matrix, and for each fact, DSP model populates corresponding feature column with 1 when the feature is extracted for the fact and 0 otherwise.

DSP generates a separate feature matrix for each relation, where feature matrices are generated from a set of both positive and negative facts. Then, for each feature matrix DSP model trains a logistic regression model to learn a binary classification model for each relation. This model is then used to predict scores for candidate facts such that learned path feature weights differ for each relation type corresponding to its extracted path feature matrix.

Logistic regression is a binary classifier *i.e.* it can discriminate between *true* and *false* facts in case of knowledge base completion. It is used to predict scores of candidate facts corresponding to both classes. DSP model uses the difference between these scores

corresponding to *true* and *false* facts to generate a single score for candidate facts in the following manner:

$$s(f)_{DSP} = s(f)_{lr}^{true} - s(f)_{lr}^{false}$$

where $s(f)_{DSP}$ is DSP model's score of candidate fact f, $s(f)_{lr}^{true}$ is logistic regression score of class "*true facts*" and $s(f)_{lr}^{false}$ is logistic regression score of class "*false facts*", that:

$$s(f)_{lr}^{true} = \frac{1}{1 + \exp(-(X \cdot A + b))}$$

for feature row X and learnt coefficients A, and intercept b and $s(f)_{lr}^{false} = 1 - s(f)_{lr}^{true}$, that DSP model scoring function can be defined as:

$$s(f)_{DSP} = 2 * s(f)_{lr}^{true} - 1$$

Since the output is an ordered rank, we can simplify the scoring function to be

$$s(f)_{DSP} = s(f)_{lr}^{true} = \frac{1}{1 + \exp(-(X \cdot A + b))}$$

Using the difference of both logistic regression scores associated to true fact and false fact classes the DSP model is able to transform the output of the classification into a rank. In case of using a different learning model than logistic regression, classes score difference can have different interpretation. We use logistic regression following previous state-of-the-art path feature models, however, we aim to investigate the performance of different learning models *e.g.* SVM classifier or decision trees in future work.

High ranked elements reflect high positive difference of class scores in favour of the true facts' class, and low ranked elements reflect a high negative difference of class scores in favour of the false facts' class.

3.3 Model Interpretability

Expressiveness of machine learning models is a key aspect in their evaluation, as understanding the behaviour of a model empowers both users and designers of the model, and it can help assessing the trust in it [24]. Graph feature models use graph components as features, and these components can be used as a meaningful explanation of their prediction. Usually, predictions of graph feature models are expressed by features they extract, that represent prior knowledge parts *e.g.* subgraph paths, connecting paths or neighbour nodes. While other approaches *e.g.* association rule mining [6], and relation path pattern mining [17] extract rules and patterns from knowledge graphs and use them as evidence for existence of candidate paths and triples.

DSP model expresses its predictions using the set of features it uses: distinct subgraph paths and ANYREL paths. It uses weights of the learned path features coefficients as an explanation. Each feature coefficient in DSP learned model represents how important is the corresponding path feature for the predicted score. I For example, predicting a score for candidate fact (*Tedd*, *plays_for*, *TeamX*) from Figure 1 can be expressed in a series of path features and associated weights as shown in Table 2. Among the features that contributed the most to the prediction is ⟨*ANYREL:ANYREL,plays_for*⟩which can correspond in our example to the path: ⟨*colleague,plays_for*⟩.

Table 2 presents a set of possible path features that can be extracted by DSP model for candidate fact (*Tedd*, *plays_for*, *TeamX*) with their possible learnt weights for DSP and SFE. The table shows that DSP model can extract and learn coefficients for richer set of path features. For example, it can learn weights for first four path feature types, distinct subgraph paths, while SFE model only extracts and learns ANYREL path feature types.

#	Path feature	DSP Weight	SFE Weight
1	⟨*SUB:practise*⟩	0.32	N/A
2	⟨*SUB:friend*$^{-1}$⟩	-0.21	N/A
3	⟨*OBJ:in_league*⟩	0.45	N/A
4	⟨*OBJ:plays_for*$^{-1}$⟩	0.53	N/A
5	⟨*ANYREL:colleague,ANYREL*⟩	0.75	0.8
6	⟨*ANYREL:ANYREL,plays_for*⟩	1.45	1.2

Table 2: Example of DSP and SFE model interpretation of a prediction score of fact (*Tedd, plays_for, TeamX*).

4 EXPERIMENTAL SETUP

In this section, we present the benchmarking dataset, NELL, and we discuss its curation sources, its properties, and its method for generating negative instances. Then, we discuss our evaluation protocol and ranking metrics. We also discuss setup and implementation details of our experiments.

4.1 Benchmark dataset

NELL benchmark dataset. To evaluate DSP model and compare it with prior art, we reuse the NELL benchmark dataset[2] proposed by Gardner and Mitchell [7] and used to compare SFE, PRA and its variants. NELL dataset was automatically created by scraping the Web then extracting general knowledge information from web pages [16]. The dataset itself uses knowledge base completion models for assessment of new candidate facts that can be learned from present facts. The NELL benchmark dataset consists of three elements: graph triples, the knowledge graph including all entities and relations; training triples and testing triples, sets of positive and negative instances of 10 relations used for evaluation purpose. Statistics of NELL benchmark dataset used in experiments are detailed in Table 3.

Negative sample generation. Generating negative example facts is an important issue for training the model. In the NELL benchmark dataset, negative facts are generated using constrained version of closed world assumption. Typically, knowledge graphs contain only true facts. However, under the open world assumption, all absent facts can not be assumed to be false as this absence can happen due to knowledge graph incompleteness. Using a constrained version of closed world assumption allows using facts that does not exist in the knowledge graph as negative examples while using heuristics to minimise the chances of those newly asserted negative facts to be true. Gardner and Mitchell [7] applied the following strategy to generate negative facts as part of the NELL benchmark dataset: for each subject and object nodes in the set of positive facts, a score is

[2]Description for the NELL benchmark dataset can be found at http://rtw.ml.cmu.edu/emnlp2015_sfe/

NELL benchmark dataset	
Element	**# Triples**
Graph relations	≈ 110K
Graph entities	≈ 1.2M
Graph fact triples	≈ 3.8M
Training triple instances	≈ 54K
Testing triple instances	≈ 13.5K

Table 3: Statistics of the NELL benchmark dataset used in experiments.

computed for similar nodes of same class (NELL meta information) using personalised page rank [25]. Then, negative examples are generated by creating absent facts from most similar nodes from the personalised page rank with a ratio of 1:10 positives to negatives. We use the NELL benchmark dataset which consists of both positive and negative fact instances with 1:10 positives to negatives ratio to be able to compare to previous works.

4.2 Evaluation

Although using the same NELL benchmark dataset, PRA and SFE models make use of a limited instance set corresponding to limitations of their systems. In the following, we discuss the evaluation configuration and evaluation metrics with regard to approaches used by prior art.

Configuration. Since PRA and SFE based models are only taking into account connecting paths as features, they only consider a subset of the evaluation dataset for which node pairs have a connecting paths between them. DSP model can handle both instance node pairs with and without connecting paths. Ergo, we run our experiments in two different configurations: set of instances with connecting paths (referred to as "connected nodes") and set of all instances (referred to as "all nodes"). We evaluate DSP model, PRA [10], SFE with different features like plain connecting paths *i.e.* PRA feature paths, bigram feature paths introduced by Neelakantan et al. [19], and combination of PRA and ANYREL feature paths [7] over these two different configurations. Further discussion of these approaches and their path feature types is presented by Gardner and Mitchell [7].

Since PRA, its variants and SFE do not handle non-connecting paths, we assume 0 as a score for instance node pairs with no connecting paths. The scoring function of these models depends on the accumulation of weights corresponding to connecting path features. Therefore, 0 score of non-connecting paths which do not belong to their feature set will not impact the scoring function.

Metrics. Similarly to prior-art, the evaluation metrics we use are the mean average precision (MAP) and mean reciprocal rank (MRR). We introduce as well a new metric – Hits@k – that is the number of correct elements predicted among the top-k elements. MAP is the mean of a set of average precision (AP) scores, and average precision is the average of Precision@k scores for positive elements in the rank [13]. Precision@k is defined as:

$$P@k(\pi, l) = \frac{\sum_{t \leq k} I_{\{l_{\pi^{-1}(t)}=1\}}}{k}$$

where π is a list, l is label function, $I_{\{.\}}$ is an indicator function of a element which equals to 1 when the element is relevant and 0 otherwise, and $\pi^{-1}(j)$ denotes the element ranked at position j of the list π. Let n be the total number of rank elements and m be the total number of true elements, we can define Average Precision as:

$$AP(\pi, l) = \frac{\sum_{k=1}^{n} P@K(\pi, l) \cdot I_{\{l_{\pi^{-1}(t)}=1\}}}{m}$$

Mean average precision is the mean of a set of average precision scores[3]. Mean reciprocal rank is the harmonic mean of the rank position of the first relevant element defined as:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

where $rank_i$ refers to the rank position of the first relevant element for the i-th query.

4.3 Pre-processing

For comparison purposes, we apply the same pre-processing strategy as for SFE [7]. NELL benchmark dataset includes semantic information about relations such as inverse relation property. For example, relation *concept:riverflowsthroughcity* is declared as an inverse relation of *concept:cityliesonriver*, and both exist in the dataset. Following the pre-processing applied in SFE work, we discard these inverses while extracting path features so to disable direct inference form inverse relations. For instance, if there is a candidate fact (*river:wye*, *concept:riverflowsthroughcity*, *city:hay*) in the knowledge graph, we discard the corresponding inverse relation fact (*city:hay*, *concept:cityliesonriver*, *river:wye*).

Nonetheless, to allow for bidirectional exploration of the knowledge graph, we append inverses of all facts in the knowledge graph used. That is if (e_1, r_1, e_2) is a fact in the dataset, we also append (e_2, r_1^{-1}, e_1). While predicting a given fact using the model, we disregard candidate facts' inverse.

4.4 Implementation

In our experiments we use Python3 as a language. Version 0.17.1 of the scikit-learn python library is used for the implementation of logistic regression [21]. We use logistic regression with default configuration of L1 regularization, where inverse of regularization strength $C = 1.0$. We choose adjacency matrices as a data structure to represent a knowledge graph. During feature extraction, we extract subgraph paths of depth 2. We only use 50 neighbour instances per relation in order to avoid dense neighbourhood of nodes and keep a sample of all neighbour nodes. We do not sample the set of neighbour all together, as this may result in discarding neighbour relations with fewer instances in dense nodes. We therefore sample neighbour nodes per relation instance. All experiments run over a machine with 40 Gb of RAM and 10 CPU processing cores of 2.2 GHz.

[3]During our experiments, we have found out that the published code of PRA and SFE uses an inaccurate implementation of the AP metrics. After confirming this with the authors in a private communication, we have reimplemented the metric using the presented formula before computing the values discussed in this paper. Note that the reimplemented metric does not introduce any dramatic changes when comparing the existing techniques among themselves or with our results. We only wanted to make sure the results we present are as accurate as possible.

Connected Nodes		
Model (features)	**MAP**	**MRR**
PRA (PRA)	0.447	0.792
SFE (PRA)	0.557	0.806
SFE (Bigrams)	0.638	**1.000**
SFE (PRA+ANYREL)	0.675	0.933
DSP (ANYREL+DSP)*	**0.690**	0.950
All Nodes		
Model (features)	**MAP**	**MRR**
PRA (PRA)	0.569	0.783
SFE (PRA)	0.540	0.806
SFE (Bigrams)	0.654	**1.000**
SFE (PRA+ANYREL)	0.655	0.933
DSP (ANYREL+DSP)*	**0.698**	0.950

Table 4: Evaluation of DSP model over set of connected/all node pair instances.

Model	MRR	Hits@k		
		@5	**@10**	**@20**
SFE (Bigrams)	**1.000**	4.5	8.5	16.9
SFE (PRA+AR)	0.933	**4.6**	8.9	17.0
DSP (DSP +AR)*	0.950	**4.6**	**9.0**	**17.5**

Table 5: Average Hits@k of DSP and other models.

5 RESULTS AND DISCUSSION

The outcome of our experiments in both connected nodes and all nodes configurations is presented in Table 4. DSP model achieves a mean average precision of 0.692 and 0.698 for connected nodes configuration and all nodes respectively, outperforming SFE with MAP of 0.675 and 0.655, and PRA with 0.557 and 0.54. Considering also non-connected paths (all nodes configuration), the mean average precision of DSP model shows a slight improvement of 1%, while other approaches like SFE with different features shows a decrease of mean average precision. On the contrary, PRA shows an improvement of 12% in all node configuration, as in connected nodes configuration PRA has high percentage of discarded positive candidate facts due to absence of connecting path as it uses random walks, where scoring non connected instances with 0 enables PRA to gain this improvement.

In terms of mean reciprocal rank (MRR), SFE with bigrams features provides best results with a score of 1.0. While MRR provides a useful information when a user only wishes to see one relevant element, it may be more suited in the context of knowledge graph completion to look at the number of top-k relevant elements. We present in Table 5 the measure of Hits at k for $k = 5, 10, 20$. For that measure, DSP model outperforms all other models demonstrating a better ability to rank higher relevant elements within top-5, 10, 20.

Table 6 details models' performance in the "all nodes" configuration per relation. Column "NCI" represents the percentage of non-connecting pair nodes instances for a given relation[4]. The results show that models like SFE or PRA are affected negatively by the absence of non-connected node pair instances. On the contrary and following our intuition, DSP model improvement is greater for relations with high percentage of non-connected instances. DSP model's highest improvement $\Delta_{MAP} \approx 0.21$ is observed for relation *concept:sportsteampositionforsport* which has the highest percentage of non-connected instances of 13%. The lowest DSP model's relative performance $\Delta_{MAP} \approx -0.04$ is observed for relation *concept:statehaslake* with the lowest percentage of non-connected instances of 0%.

In our experiments, we have found that DSP model is able to provide a high rank[5] for true candidate facts even in the absence of connecting paths, hence not in the result set of any previous graph feature models. For example, considering the relation *concept:citylocatedincountry*, DSP model is able to predict that *city:abu_dhabi* is located within *country:the_united_arab_emirates* (ranked at top 2.87%) while there are no connecting paths between them. Similarly, DSP model was able to provide a high rank (top 2.19%) for the fact that *river:wye* and *city:hay* are connected with *concept:riverflowsthroughcity* relation, despite they have no connecting paths between them.

6 RELATED WORK

Many relational learning models were developed to predict new facts in knowledge bases. In recent years, latent feature models witnessed a rapid development providing a variety of models using methods such as tensor factorization [20] or latent distance embeddings [2]. Although these models excel in the task of link prediction in knowledge graphs, their predictions are hard to interpret. They act as a black box relying on latent representation of features that are hard to trace back to original knowledge [28].

In contrast, graph feature models provide more expressive predictions. They use graph-based features like subgraphs, connecting paths and neighbourhood information, which corresponds to intelligible parts of prior knowledge. This makes these techniques more suited to use cases where interpretability of the predictions matters (*e.g.* in life sciences).

Recent development of graph feature models encompasses Path Ranking Algorithms (PRA) [10] and its variations that used backward random walks [11], latent syntactic cues [8], incorporating vector similarity in inference [9] or bigram feature path [19]. The most recent improvement of the PRA-based techniques is SFE [7] that uses ANYREL path features. This set of models relies exclusively on connecting paths between nodes as features. They provide expressive and interpretable predictions, but they still lack in terms of efficiency and ability to predict scores for relationship between non-connected nodes [7].

In their work, Gardner and Mitchell [7] investigated the use of non-connected subgraph paths called "One-Sided features." In their experiment, the authors only considered these features with higher expressivity on data with connected entity pairs and concluded that such features yield inferior results compared to PRA

[4]Depending on the sampling method used (random walk, DFS), pair nodes can be considered connected or non-connected.

[5]Positive candidate facts with high rank are in the top 10% elements of the rank, where positive to negative ratio is 1:10.

Relation	NCI	DSP (DSP + ANYREL)					SFE (PRA + ANYREL)				
		AP	RR	H@5	H@10	H@20	AP	RR	H@5	H@10	H@20
concept:riverflowsthroughcity	1%	**0.503**	**1.00**	5	9	19	0.480	**1.00**	5	9	19
concept:sportsteampositionforsport	13%	**0.768**	**1.00**	5	10	14	0.558	**1.00**	5	9	9
concept:citylocatedincountry	1%	**0.548**	**1.00**	3	8	18	0.495	0.333	3	8	17
concept:athleteplaysforteam	1%	**0.776**	**1.00**	5	10	20	**0.776**	**1.00**	5	10	20
concept:writerwrotebook	10%	**0.828**	**1.00**	5	10	20	0.783	**1.00**	5	10	20
concept:actorstarredinmovie	9%	**0.869**	**1.00**	5	10	20	0.786	**1.00**	5	10	20
concept:journalistwritesforpublication	8%	**0.879**	**1.00**	5	10	20	0.838	**1.00**	5	10	20
concept:stadiumlocatedincity	1%	**0.628**	**1.00**	5	10	19	0.624	**1.00**	5	10	20
concept:statehaslake	0%	0.237	0.50	3	3	5	**0.278**	**1.00**	3	3	5
concept:teamplaysinleague	1%	**0.942**	**1.00**	5	10	20	0.936	**1.00**	5	10	20
Average	-	**0.698**	**0.950**	4.60	9.00	17.50	0.655	0.933	4.60	8.90	17.00

Table 6: Evaluation of DSP and SFE over NELL 10 relations using all nodes pairs, with percentage of non-connected instances for each relation.

	PRA	**SFE**	**DSP**
Negatives	Failed RW.	PPR	PPR
F. types	CP	CP & ANYREL	DSP & ANYREL
F. weights	RW prob.	Binary	Binary
Model	LogReg	LogReg	LogReg
Scoring	$\sum_{f_i \in X} A_i$	$\sum_{f_i \in X} A_i$	$\frac{1}{1+\exp(-(X \cdot A+b))}$
Scope	Connected	Connected	All

Table 7: Properties of current graph feature models.

and ANYREL features. In our work we consider the effect of non-connected subgraph paths to better model relationships even in the absence of connecting paths. Our experiments have demonstrated the relevance of this contribution.

Despite the recent focus on latent feature models, it has been observed experimentally that neither latent feature models nor graph models are superior for learning over knowledge graphs; they are complementary [3]. The former models harness global graph patterns, while the latter capture local and quasi-local graph patterns [28].

In this work, we focus on enhancing the predictive capabilities of graph feature models and utilising their expressiveness in the task of knowledge base completion, where DSP model extends the work accomplished by PRA and SFE. Table 7 shows a comparison between properties of DSP model compared to previous models like SFE and PRA, where *Negatives* represent negative generation technique, *F.Types* represent feature types, *F. weights* represent representation of feature weights, and *Scoring* represent model's scoring function, where A is learnt coefficients, and b is learnt intercept.

SFE and DSP model use PPR for generating negative instances while PRA uses failed random walks. Also, they use binary representation of feature weights in the feature matrix while PRA uses random walk probabilities. On the other hand, DSP model uses DSP & ANYREL features while PRA and SFE use connected path and ANYREL path features. Also, its prediction scope target all candidate triples while PRA and SFE target triples with connected subject and object entities. These are our main technical contributions over the state of the art, and our experiments have shown these contributions are reflected in tangible gains in performance, without sacrificing computational efficiency.

7 CONCLUSION

We introduced a new graph feature model, Distinct Subgraph Paths (DSP) model, that uses a combination of distinct subgraph paths for subject and object nodes and connecting paths in order to predict new facts in knowledge graphs. Using a richer set of features, the model is able to provide a ranking score for node pairs candidates in the absence of connecting paths addressing an important limitation of current approaches. Moreover, the extraction of distinct subgraph path features comes with no computational overhead compared to SFE. We showed experimentally that DSP model outperforms both PRA and SFE (ANYREL) in terms of mean average precision (MAP), mean reciprocal rank (MRR) and Hits@5, 10, 20.

Moving forward, we plan to explore the use of graph path features in two different directions: (a) injecting graph path feature in latent feature models; (b) using path features to extract horn clauses in knowledge graphs as a support for rule mining models.

REFERENCES

[1] 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)*. ACM, New York, NY, USA, 1247–1250. https://doi.org/10.1145/1376616.1376746

[2] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. 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.* 2787–2795. http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data

[3] 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 *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014.* 601–610. https://doi.org/10.1145/2623330.2623623

[4] Michel Dumontier, Alison Callahan, Jose Cruz-Toledo, Peter Ansell, Vincent Emonet, François Belleau, and Arnaud Droit. 2014. Bio2RDF Release 3: A larger, more connected network of Linked Data for the Life Sciences. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.* 401–404.

[5] David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building Watson: An Overview of the DeepQA Project. *AI Magazine* 31, 3 (2010), 59–79.

[6] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2015. Fast Rule Mining in Ontological Knowledge Bases with AMIE+. *The VLDB Journal* 24, 6 (Dec. 2015), 707–730. https://doi.org/10.1007/s00778-015-0394-1

[7] Matt Gardner and Tom M. 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, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* 1488–1498. http://aclweb.org/anthology/D/D15/D15-1173.pdf

[8] 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, EMNLP 2013.* Association for Computational Linguistics, Seattle, Washington, USA, 833–838. http://www.aclweb.org/anthology/D13-1080

[9] Matt Gardner, Partha Pratim Talukdar, Jayant Krishnamurthy, and Tom M. 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 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL.* 397–406. http://aclweb.org/anthology/D/D14/D14-1044.pdf

[10] Ni Lao and William W. Cohen. 2010. Relational Retrieval Using a Combination of Path-constrained Random Walks. *Machine Learning* 81, 1 (Oct. 2010), 53–67. https://doi.org/10.1007/s10994-010-5205-8

[11] Ni Lao, Einat Minkov, and William W. Cohen. 2015. Learning Relational Features with Backward Random Walks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers.* 666–675. http://aclweb.org/anthology/P/P15/P15-1065.pdf

[12] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Chris Bizer. 2014. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal* (2014).

[13] Tie-Yan Liu. 2011. *Learning to Rank for Information Retrieval.* Springer. 1–285 pages. https://doi.org/10.1007/978-3-642-14267-3

[14] Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. 2014. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *7th Biennial Conference on Innovative Data Systems Research.* CIDR Conference.

[15] George A. Miller. 1995. WordNet: A Lexical Database for English. *Commun. ACM* 38, 11 (Nov. 1995), 39–41. https://doi.org/10.1145/219717.219748

[16] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. 2015. Never-Ending Learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15).*

[17] Sameh K. Mohamed, Emir Muñoz, Vít Novácek, and Pierre-Yves Vandenbussche. 2017. Identifying Equivalent Relation Paths in Knowledge Graphs. In *Language, Data, and Knowledge - First International Conference, LDK 2017, Galway, Ireland, June 19-20, 2017, Proceedings.* 299–314. https://doi.org/10.1007/978-3-319-59888-8_26

[18] Emir Muñoz, Vít Novácek, and Pierre-Yves Vandenbussche. 2016. Using Drug Similarities for Discovery of Possible Adverse Reactions. In *AMIA 2016, American Medical Informatics Association Annual Symposium, Chicago, IL, USA, November 12-16, 2016.* AMIA. http://knowledge.amia.org/amia-63300-1.3360278/t004-1.3364525/f004-1.3364526/2499657-1.3364713/2500122-1.3364708

[19] 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 of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers.* 156–166. http://aclweb.org/anthology/P/P15/P15-1016.pdf

[20] 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 Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011.* 809–816.

[21] 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.

[22] Richard Qian. 2013. Understand Your World with Bing. (2013). http://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/ Bing Blogs.

[23] Simon Razniewski, Fabian M. Suchanek, and Werner Nutt. 2016. But What Do We Actually Know?. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction, AKBC@NAACL-HLT 2016, San Diego, CA, USA, June 17, 2016.* 40–44. http://aclweb.org/anthology/W/W16/W16-1308.pdf

[24] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16).* ACM, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[25] Franco Scarselli, Ah Chung Tsoi, and Markus Hagenbuchner. 2004. Computing personalized pageranks. In *Proceedings of the 13th international conference on World Wide Web - Alternate Track Papers & Posters, WWW 2004, New York, NY, USA, May 17-20, 2004.* 382–383. https://doi.org/10.1145/1013367.1013486

[26] Amit Singhal. 2012. Introducing the Knowledge Graph: things, not strings. (2012). "https://googleblog.blogspot.ie/2012/05/introducing-knowledge-graph-things-not.html" Google Official Blog.

[27] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning with Neural Tensor Networks for Knowledge Base Completion. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'13).* Curran Associates Inc., USA, 926–934. http://dl.acm.org/citation.cfm?id=2999611.2999715

[28] Kristina Toutanova and Danqi Chen. 2015. Observed Versus Latent Features for Knowledge Base and Text Inference. In *3rd Workshop on Continuous Vector Space Models and Their Compositionality.* ACL – Association for Computational Linguistics. https://www.microsoft.com/en-us/research/publication/observed-versus-latent-features-for-knowledge-base-and-text-inference/