Data Science 0 (0) 1
IOS Press


# Application of Concepts of Neighbours to Knowledge Graph Completion[1]

Sébastien Ferré
*Univ Rennes, CNRS, IRISA, France*
*E-mail: ferre@irisa.fr; ORCID: https://orcid.org/0000-0002-6302-2333*

**Abstract.** The open nature of Knowledge Graphs (KG) often implies that they are incomplete. Knowlege graph completion (aka. link prediction) consists in infering new relationships between the entities of a KG based on existing relationships. Most existing approaches rely on the learning of latent feature vectors for the encoding of entities and relations. In general however, latent features cannot be easily interpreted. Rule-based approaches offer interpretability but a distinct ruleset must be learned for each relation. In both latent- and rule-based approaches, the training phase has to be run again when the KG is updated. We propose a new approach that does not need a training phase, and that can provide interpretable explanations for each inference. It relies on the computation of Concepts of Nearest Neighbours (C-NN) to identify clusters of similar entities based on common graph patterns. Different rules are then derived from those graph patterns, and combined to predict new relationships. We evaluate our approach on standard benchmarks for link prediction, where it gets competitive performance compared to existing approaches.

Keywords: Knowledge Graph, Multi-Relational Data, Knowledge Graph Completion, Link Prediction, Graph Pattern, Concepts of Nearest Neighbours, Inference Rules, Analogical Inference, Explainable AI

## 1. Introduction

There is a growing interest for knowledge graphs (KG) as a way to represent and share data on the Web. The Semantic Web [2] defines standards for represention (RDF), querying (SPARQL), and reasoning (RDFS, OWL), and thousands of open KGs are available: e.g., DBpedia, Wikidata (formerly Freebase), YAGO, WordNet. The open nature of KGs often implies that they are incomplete, and a lot of work have studied the use of machine learning techniques to complete them.

The task of *knowledge graph completion*, aka. *link prediction* [3], consists in predicting missing edges or missing parts of edges. It is therefore a form of machine learning. Suppose that film *Avatar* is missing a director in the KG, one wants to predict it, i.e. identify it among all KG nodes. The idea is to find regularities in the existing knowledge, and to exploit them in order to rank KG nodes. The higher the correct node is in the ranking, the better the prediction is. Link prediction was originally introduced for social networks with a single edge type (a single *relation*) [4], and was later extended to multi-relational data and applied to KGs [3]. Compared to the classical task of supervised classification, knowledge graph completion faces several challenges. First, there are as many classification problems as there are relations, which count in the hundreds or thousands in KGs. Second, for each relation, the number of "classes" is the number of different *entities* in the range of the relation, which typically counts in the

[1] This paper is an extended version of a paper published at ESWC'19 [1]. It is a revised version of submission 622-1602.

thousands for relations like *spouse* or *birthPlace*. Third, some relations can be multi-valued, like the relation from films to actors.

In this paper, we report on extensive experimental results about a novel approach to knowledge graph completion that is based on *Concepts of Nearest Neighbours (C-NN)*, which were introduced in [5], and applied to *query relaxation* in [6]. This paper is an extended version of [1] that reports on first results about the application of C-NNs to link prediction. In particular, the extension includes an analogical form of inference, and an extensive and deeper experimental evaluation. The C-NN approach introduces a symbolic form of k-NN (k Nearest Neighbours) where numerical distances are replaced by graph patterns that provide an intelligible representation of how similar two entities are.

Our hypothesis is that the partitioning of the KG entities into concepts of neighbours (see Section 5) provides a valuable basis for different kinds of inferences. We here focus on knowledge graph completion, i.e. the inference of the missing node of an incomplete edge. **The combinatorial aspect of graph patterns over knowledge graphs is tackled by delaying their computation to inference time, which enables to bound the number of C-NNs by the number of entities, and in practice it is generally much smaller. The intensive knowledge-based computation at inference time is managed with in-memory KG indices (as is common in RDF stores), and an any-time algorithm for both C-NN computation and inference.**

The contribution of this work is a novel approach to knowledge graph completion that has the following properties:

(1) it is a form of *instance-based learning*, i.e. it can perform inferences without training, and hence accomodate new data without re-training;
(2) it is a *symbolic approach*, i.e. it can provide explanations for each inference;
(3) it shows *competitive performance* on standard link prediction benchmarks.

The rest of the paper is organized as follows. Section 2 discusses related work on knowledge graph completion. Section 3 contains preliminaries about knowledge graphs and queries. Section 4 presents an overview of our approach. Section 5 recalls the definition of C-NNs, and their efficient computation. Section 6 presents our method to perform C-NN-based knowledge graph completion. Section 7 reports on positive experimental results on standard benchmarks (WN18, WN18RR, FB15k, FB15k-237), along with an in-depth analysis of results. Finally, Section 8 concludes and sketches future work.

## 2. Related Work

Nickel *et al* [3] published a "review of relational machine learning for knowledge graphs", where link prediction is the main inference task. They identify two kinds of approaches that differ by the kind of model they use: *latent feature models*, and *graph feature models*. The former is by far the most studied one. Before going into details, it is useful to set the vocabulary as it is used in the domain. Nodes are called *entities*, edge labels are called *relations*, and edges are called *triples* $(e_i, r_k, e_j)$, where $e_i$ is the *head* entity, $e_j$ is the *tail* entity, and $r_k$ is the relation that links the head to the tail.

Latent feature models learn *embeddings* of entities and relations into low-dimensional vector spaces, and then make inferences about a triple $(e_i, r_k, e_j)$ by combining the embeddings of the two entities and the embedding of the relation. The existing methods vary by how they learn the embeddings, and how they combine them. Those methods are based on a range of techniques including: matrix factorization, tensor factorization, neural networks, and gradient descent. For example, one of the first method

for KGs, TransE [7], models a relation as a translation in the embedding space of entities, and scores a candidate triple according to the distance between the translated head and the tail. The authors of TransE, Bordes *et al*, introduced two datasets, FB15k and WN18, respectively derived from Freebase and WordNet, which became becnhmarks in the evaluation of link prediction methods. Toutanova and Chen [8] however showed that a very simple method was able to outperform previous methods because of a flaw in the datasets: many test triples have their inverse among the training triples. They introduced a challenging subset of FB15k, called FB15k-237, where all inverse triples are removed. Dettmers [9] introduced a similar subset of WN18, called WN18RR. Lately, performance was significantly improved on the challenging FB15k-237 by using convolutional architectures to learn embeddings [10] or to combine the embeddings in scoring functions [9]. The state-of-the-art results are obtained by ComplEx-N3, combining low-rank tensor decomposition with the addition of inverse relations, and various optimizations about hyper-parameters and regularization [11]. The task of link prediction has also been extended with the embedding model RAE to *multi-fold relations* (aka. n-ary relations), and to *instance reconstruction* where only one entity of an n-ary relation is known, and all other entities have to be infered together [12]. In this work, we limit ourselves to binary relations, and let those advanced cases to future work.

The latent feature models face two major drawbacks for their application to real settings. First, as they rely on the learning of entity embeddings, they cannot be applied to unseen entities, and the model has to be retrained whenever new entities or relations are introduced in the knowledge graph. Second, the predictions come without any explanation, and the meaning of learned latent features generally remain obscure. Given that it is unrealistic to expect high precision knowledge graph completion, simply because some of the missing information cannot be inferred from existing knowledge, human curation is necessary in practice. An explanation can comfort the curator in the veracity of the prediction, and could even be used to define automated inference rules.

Graph feature models, also called *observed feature models*, make inferences directly from the observed edges in the KG. Random walk inference [13] takes relation paths as features, and sets the feature values through random walks in the KG. The feature weights are learned by logistic regression for each target relation, and then used to score the candidate triples. The method has shown improvement over Horn clause generation with ILP (Inductive Logic Programming) [14]. AMIE+ [15] manages to generate such Horn clauses in a much more efficient way by designing new ILP algorithms tailored to KGs. They also introduce a novel rule measure that improves the inference precision under the Open World Assumption (OWA) that holds in KGs. A fine-grained evaluation [16] has shown that rule-based approaches are competitive with latent-based approaches, both in performance and in running time. AnyBURL [17] restricts to path-like Horn clauses, and employs an anytime bottom-up strategy for learning rules. It significantly improves performance compared to AMIE+, and gets close to the state-of-the-art of latent feature models. Graph feature models, especially rule-based approaches, offer the advantage to produce intelligible explanations for inferences, unlike the latent feature models. Moreover, they can be applied to unseen entities because inferences are based on the neighborhood of the entity in the knowledge graph, rather than on a learned embedding. However, they require a distinct training phase for each of the possibly many target relations, whereas the latent feature models are generally learned in one training phase. A drawback of both kinds of approaches is that new embeddings and new rulesets need to be learned whenever the KG is updated, which becomes a challenge for dynamic data.

Our approach relies on graph features but a key difference is that there is no training phase, and all the learning is done at inference time. It is therefore an instance-based approach rather than a model-based approach. Given an incomplete triple $(e_i, r_k, ?)$ we compute Concepts of Nearest Neighbours (C-

$$
\begin{aligned}
E &= \{\mathit{Charles}, \mathit{Diana}, \mathit{William}, \mathit{Harry}, \mathit{Kate}, \mathit{George}, \mathit{Charlotte}, \mathit{Louis}, \mathit{male}, \mathit{female}\} \\
R &= \{\mathit{parent}, \mathit{spouse}, \mathit{gender}\} \\
T &= \{(\{\mathit{William}, \mathit{Harry}\}, \mathit{parent}, \{\mathit{Charles}, \mathit{Diana}\}), \\
&\quad (\{\mathit{George}, \mathit{Charlotte}, \mathit{Louis}\}, \mathit{parent}, \{\mathit{William}, \mathit{Kate}\}), \\
&\quad (\mathit{Charles}, \mathit{spouse}, \mathit{Diana}), (\mathit{Diana}, \mathit{spouse}, \mathit{Charles}), \\
&\quad (\mathit{William}, \mathit{spouse}, \mathit{Kate}), (\mathit{Kate}, \mathit{spouse}, \mathit{William}), \\
&\quad (\{\mathit{Charles}, \mathit{William}, \mathit{Harry}, \mathit{George}, \mathit{Louis}\}, \mathit{gender}, \mathit{male}), \\
&\quad (\{\mathit{Diana}, \mathit{Kate}, \mathit{Charlotte}\}, \mathit{gender}, \mathit{female})\}
\end{aligned}
$$

Fig. 1. Example knowledge graph describing part of the British royal family.

NN) from the observed features of head entity $e_i$, where C-NNs have a representation equivalent to the bodies of AMIE+'s rules. From there, we infer a ranking of candidate entities for the tail of relation $r_k$. In fact, as $r_k$ is not involved in the computation of C-NNs, many target relations can be inferred at nearly the same cost as a single relation. Indeed the main cost is in the computation of C-NNs, which is easily controlled because the computation algorithm is anytime. Compared to rule-based approaches, it amounts to compute only rule bodies that match the head entity $e_i$, and to add the rule head according to relation $r_k$.

## 3. Preliminaries

A *knowledge graph* (KG) is defined by a structure $K = \langle E, R, T \rangle$, where $E$ is the set of nodes, also called *entities*, $R$ is the set of edge labels, also called *relations*, and $T \subseteq E \times R \times E$ is the set of directed and labelled edges, also called *triples*. Each triple $(e_i, r_k, e_j)$ represents the fact that relation $r_k$ relates entity $e_i$ to entity $e_j$. This definition is very close to RDF graphs, where entities can be either URIs or literals (or blank nodes) and relations are URIs called properties. It is also equivalent to sets of logical facts, where entities are constants, and relations are binary predicates. As a running example, Figure 1 defines a small KG describing (part of) the British royal family (where $(\{a,b\}, r, \{c,d\})$ is a compact notation for $(a,r,c), (a,r,d), (b,r,c), (b,r,d)$).

*Queries* based on *graph patterns* play a central role in our approach as they are used to characterize the C-NNs, and can be used as explanations for inferences. There are two kinds of *query elements*: triple patterns and filters. A *triple pattern* $(x,r,y) \in V \times R \times V$ is similar to a triple but with variables (taken from $V$) in place of entities. A *filter* is a Boolean expression on variables and entities. We here only consider equalities between a variable and an entity $(x = e)$ and let richer filters about literal values for future work (e.g., inequalities and intervals on numbers and dates, regular expressions on strings). A *graph pattern* $P$ is a set of query elements. $Vars(P)$ denotes the set of variables occuring in $P$. Equality filters are equivalent to allowing entities in triple patterns: e.g., pattern $\{(x, \mathit{parent}, y), (y = \mathit{Kate})\}$ is equivalent to the triple pattern $(x, \mathit{parent}, \mathit{Kate})$. There are two advantages in using filters: (1) simplifying the handling of triple patterns that have a single form (var-var) instead of four (var-var, entity-var, var-entity, entity-entity); (2) opening perspectives for richer filters (e.g., $x > 10$, $x \in [0,10]$). A *query* $Q = (x_1, ..., x_n) \leftarrow P$ is the projection of a graph pattern on a subset of its variables. Such queries find a concrete form in SPARQL with syntax `SELECT ?x1...?xn WHERE { P }`. Queries can be seen as anonymous rules, i.e. rules like those in AMIE+ [15] but missing the relation in the head. For example,

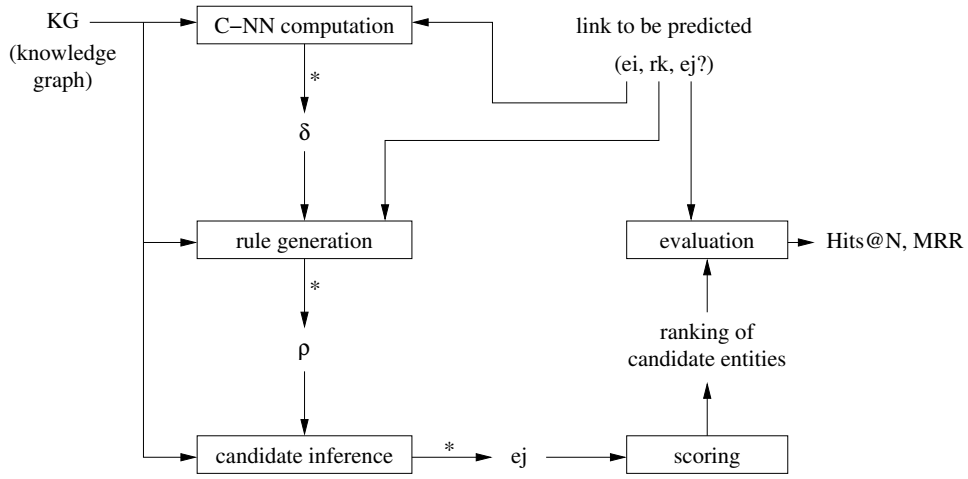


Fig. 2. Overview of link prediction based on concepts of nearest neighbours

the query $Q_{ex} = (x, y) \leftarrow (x, parent, u), (u, parent, v), (y, parent, v), (y, gender, s), (s = male)$ retrieves all (person,uncle) pairs, i.e. all pairs $(x, y)$ where $y$ is a sibling of a parent of $x$, and is male.

We now define the *answer set* that is retrieved by a query. A *matching* of a pattern $P$ on a KG $K = \langle E, R, T \rangle$ is a mapping $\mu$ from $Vars(P)$ to entities in $E$ such that $\mu(t) \in T$ for every triple pattern $t \in P$, and $\mu(f)$ evaluates to true for every filter $f \in P$, where $\mu(t)$ and $\mu(f)$ are obtained from $t$ and $f$ by replacing every variable $x$ by $\mu(x)$. In the example KG, a possible matching for the pattern of the above query is $\mu_{ex} = \{x \mapsto Charlotte, y \mapsto Harry, u \mapsto William, v \mapsto Diana, s \mapsto male\}$. A matching is therefore a homomorphism from the pattern to the graph. Term "matching" is taken from the evaluation of SPARQL queries. In logics, terms "grounding" and "instantiation" are used instead. The *answer set* $ans(Q, K)$ of a query $Q = (x_1, ..., x_n) \leftarrow P$ is the set of tuples $(\mu(x_1), ..., \mu(x_n))$ obtained from all matchings $\mu$ of $P$ on $K$. In the running example, the pair $(Charlotte, Harry)$ is therefore an answer of query $Q_{ex}$. Note that several matchings can lead to the same answer, and that duplicate answers are ignored. In the following, we only consider queries with a single projected variable, whose sets of answers are assimilated to sets of entities.

## 4. Overview of the Approach

**Figure 2 shows a schematic overview of our approach. Given a knowledge graph and a triple $(e_i, r_k, e_j)$, the goal is to predict the link to $e_j$ from $e_i$ through relation $r_k$, i.e. $e_j$ is here the unknown to predict. For example, assume triple $(Charlotte, parent, Kate)$ is missing in the KG of Figure 1, and we want to predict who is the mother of Charlotte. The first step is to compute a collection of Concepts of Nearest Neighbours (C-NN) $\delta$ for entity $e_i$ (see Section 5). In the example where $e_i$ is Charlotte, one C-NN contains Diana and Kate because they have female gender like Charlotte, and another C-NN contains George and Louis because they have William as a parent like Charlotte. The second and third step are to generate a collection of inference rules for each C-NN, and to derive a set of candidate entities from each rule (see Section 6.1). A candidate entity may be inferred by several rules, possibly generated from different concepts. In the example, a rule generated from the latter C-NN is that "persons who have William as a parent also have Kate as a**

**parent", and Kate is therefore a candidate for the unknown entity. Finally, the different candidates can be ranked by scoring them based on rule measures such as support and confidence, and the ranking can be evaluated with standard ranking measures such as Hits@N and MRR w.r.t. the expected entity $e_j$ (see Section 7).**

## 5. Concepts of Nearest Neighbours (C-NN)

In this section, we shortly recall the theoretical definitions underlying Concepts of Nearest Neighbours (C-NN), as well as the algorithmic and practical aspects of their approximate computation under a given timeout. Further details are available in [5, 6]. In the following definitions, we assume a knowledge graph $K = \langle E, R, T \rangle$.

### *5.1. Theoretical Definitions*

**We start by defining *graph concepts*, introduced in Graph-FCA [18], which is a generalization of Formal Concept Analysis (FCA) [19] to knowledge graphs. FCA concepts are a formalization of the classical theory of concepts in philosophy, where each concept has two related sides: the *intension* that characterizes what concept instances have in common, and the *extension* that enumerates those instances.**

**Definition 1.** *A* graph concept *is defined as a pair $C = (A, Q)$, where $A$ is a set of entities and $Q$ is a query such that $A = ans(Q)$ is the set of answers of $Q$, and $Q = msq(A)$ is the most specific query that verifies $A = ans(Q)$. $A$ is called the* extension *$ext(C)$ of the concept, and $Q$ is called the* intension *$int(C)$ of the concept.*

The most specific query $Q = msq(A)$ represents what the neighborhood of entities in $A$ have in common. It is well-defined under graph homomorphism (unlike under subgraph isomorphism). It can be computed from $A$ by using the categorical product of graphs (see PGP intersection $\cap_q$ in [18]), or equivalently Plotkin's anti-unification of sets of facts [20]. In the example KG, William and Charlotte have in common the following query that says that they have married parents:

$$\begin{aligned} Q_{WC} &= msq(\{William, Charlotte\}) \\ &= x \leftarrow (x, gender, s), \\ &\qquad (x, parent, y),\ (y, gender, t),\ (t = male), \\ &\qquad (x, parent, z),\ (z, gender, u),\ (u = female), \\ &\qquad (y, spouse, z),\ (z, spouse, y). \end{aligned}$$

We have $A_{WC} = ans(Q_{WC}) = \{William, Harry, George, Charlotte, Louis\}$ so that $C_{WC} = (A_{WC}, Q_{WC})$ is a graph concept.

A concept $C_1 = (A_1, Q_1)$ is more specific than a concept $C_2 = (A_2, Q_2)$, in notation $C_1 \leqslant C_2$, if $A_1 \subseteq A_2$. For example, a concept more specific than $C_{WC}$ is the concept of *the children of Kate and William*, whose extension is $\{George, Charlotte, Louis\}$, and whose intension is

$$x \leftarrow (x, gender, s), (x, parent, y), (y = William), (x, parent, z), (z = Kate).$$

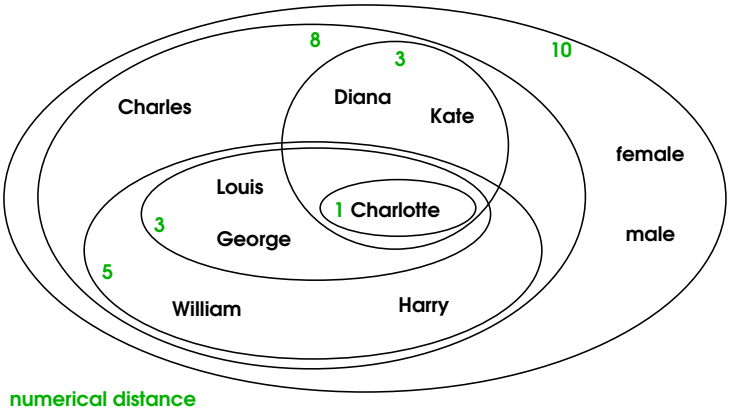


Fig. 3. Venn diagram of the extensions of the 6 C-NNs of Charlotte, labelled by their numerical distance.

**The total number of graph concepts in a knowledge graph is finite but in the worst case, it is exponential in the number of entities. It is therefore not feasible in general to compute the set of all concepts.**

**Instead of considering concepts generated by subsets of entities, we consider concepts generated by pairs of entities, and use them as as a symbolic form of distance between entities.**

**Definition 2.** *Let $e_1, e_2 \in E$ be two entities. The* conceptual distance $\delta(e_1, e_2)$ *between $e_1$ and $e_2$ is the most specific graph concept whose extension contains both entities, i.e. $\delta(e_1, e_2) = (A, Q)$ with $Q = msq(\{e_1, e_2\})$, and $A = ans(Q)$.*

For example, the above concept $C_{WC}$ is the conceptual distance between William and Charlotte. The "distance values" have therefore a symbolic representation through the concept intension $Q$ that represents what the two entities have in common. The concept extension $A$ contains in addition to the two entities all entities $e_3$ that match the common query ($e_3 \in ans(Q)$). Such an entity $e_3$ can be seen as "between" $e_1$ and $e_2$: in formulas, for all $e_3 \in ext(\delta(e_1, e_2))$, $\delta(e_1, e_3) \leqslant \delta(e_1, e_2)$ and $\delta(e_3, e_2) \leqslant \delta(e_1, e_2)$. Note that order $\leqslant$ on conceptual distances is a partial ordering, unlike classical distance measures.

A numerical distance $dist(e_1, e_2) = |ext(\delta(e_1, e_2))|$ can be derived from the size of the concept extension, because the closer $e_1$ and $e_2$ are, the more specific their conceptual distance is, and the smaller the extension is. For example, the numerical distance is 5 between Charlotte and William (see $C_{wc}$), and 3 between George and Charlotte.

**The number of conceptual distances $\delta(e_1, e_2)$ is no more exponential but it is still quadratic in the number of entities. By delaying their computation until we know for which entity $e$ predictions are to be made, we fix one of the two entities in $\delta(e_1, e_2)$, and the number of concepts becomes linear. Those concepts are called *Concepts of Nearest Neighbours (C-NN)*.**

**Definition 3. Let $e \in E$ be an entity. The *Concepts of Nearest Neighbours (C-NN)* (in short, *concepts of neighbours*) of $e$ are all the conceptual distances between $e$ and every entity $e' \in E$.**

$$C\text{-}NN(e, K) = \{\delta(e, e') \mid e' \in E\}$$

Table 1 lists the 6 C-NNs of Charlotte in the running example. **Each row describes a concept of neighbours with its id, its extension, its intension, and its direct sub-concepts among the C-NNs.**

Table 1
**Description of the 6 C-NNs of Charlotte in the royal family KG.**

| $l$ | $ext(\delta_l)$ (proper in bold) | $int(\delta_l)$ | sub-concepts |
|---|---|---|---|
| 1 | {**Charlotte**} | $x \leftarrow (x = Charlotte)$ | - |
| 2 | {**Diana**, **Kate**, *Charlotte*} | $x \leftarrow (x, gender, s), (s = female)$ | 1 |
| 3 | {**George**, **Louis**, *Charlotte*} | $x \leftarrow (x, gender, s), (x, parent, y), (y = William), (x, parent, z), (z = Kate)$ | 1 |
| 4 | {**William**, **Harry**, *George*, *Louis*, *Charlotte*} | $x \leftarrow (x, gender, s), (x, parent, y), (y, gender, t), (t = male), (x, parent, z), (z, gender, u), (u = female), (y, spouse, z), (z, spouse, y)$ | 3 |
| 5 | {**Charles**, *Diana*, *Kate*, *William*, *Harry*, *George*, *Louis*, *Charlotte*} | $x \leftarrow (x, gender, s)$ | 2, 4 |
| 6 | {**female**, **male**, ...} $= E$ | $x \leftarrow \emptyset$ | 5 |

**The bold part of extensions represent the *proper extensions* $proper(\delta_l)$ of C-NNs, i.e. the entities that do not appear in sub-concepts.** The proper extensions of *C-NN*$(Charlotte, K)$ define a partition over the set of entities $E$, where two entities are in the same proper extension if and only if they are at the same conceptual distance to entity *Charlotte*. The intension of the associated graph concept $\delta_l$ provides a symbolic representation of the similarity between every $e' \in proper(\delta_l)$ and *Charlotte*. For instance, concept $\delta_2$ says that *Diana* and *Kate* have in common with *Charlotte* to be female persons. Figure 3 and the last column of Table 1 give the partial ordering between C-NNs: e.g., $\delta_1 \leqslant \delta_2$. Smaller C-NNs contain entities that are closer to the chosen entity, here *Charlotte*. In particular, C-NNs $\delta_2$ and $\delta_3$ contain the nearest neighbours of *Charlotte*. Although their numerical distance are the same (3), their intensions are very different: either being female or having William and Kate as parents.

*Discussion.* Given that *C-NN*$(e, K)$ partitions the set of entities, the number of C-NNs can only be smaller or equal to the number of entities, and in practice it is generally much smaller. This is interesting because, in comparison, the number of graph concepts is exponential in the number of entities in the worst case. Note that the search space of ILP approaches like AMIE+ is the set of queries, which is even larger than the set of all graph concepts. Computing the C-NNs for a given entity is therefore a much more tractable task than mining frequent patterns or learning rules, although the space of representations is the same. The pending questions that we study in this paper is whether those C-NNs are useful for inference, and how they compare to other approaches.

Compared to the use of numerical measures, like commonly done in k-NN approaches, C-NNs define a more subtle ordering of entities. First, because conceptual distances are only partially ordered, it can be that among two entities none is more similar than the other to the chosen entity $e$. This reflects the fact that there can be several ways to be similar to something, without necessarily a prefered one. For instance, who is most similar to Charlotte? Diana because she is also a female (C-NN $\delta_2$) or George because he is also a son of William and Kate (C-NN $\delta_3$)? Second, because conceptual distances partition the set of entities, it can be said that when two entities are at the exact same conceptual distance, they are undistinguishable in terms of similarity (ex. George and Louis in C-NN $\delta_3$). Third, the concept intension provides an intelligible explanation of the similarity to the chosen entity.

**Algorithm 1** Partitioning algorithm for entity $e$ in knowledge graph $K$

**Require:** $K = \langle E, R, T \rangle$ a knowledge graph, and $e \in E$ an entity

$\mathcal{S} :=$ an empty collection of clusters
$\mathcal{S}.add(E, \emptyset, descr(e))$ // the initial cluster (all entities, empty pattern, all edges describing $e$)
**while** no timeout $\wedge\ \exists (S, P, H) \in \mathcal{S} : H \neq \emptyset$ **do**
  $h := chooseEdge(H)$ // choice of the splitting query element
  $P_1 := P \cup \{h\}$ // refined graph pattern
  $S_1 := S \cap ans(x \leftarrow P_1, K)$ // refined cluster
  $S_0 := S \setminus S_0$ // remainder of cluster $S$
  $\mathcal{S}.remove(S, P, H)$
  $\mathcal{S}.add(S_1, P_1, H \setminus \{h\})$ only if $S_1 \neq \emptyset$
  $\mathcal{S}.add(S_0, P, H \setminus \{h\})$ only if $S_0 \neq \emptyset$
**end while**
**return** $\mathcal{S}$

### 5.2. Algorithmic and Practical Aspects

We here sketch the algorithmic and practical aspects of computing the set $C\text{-}NN(e, K)$ of concepts of nearest neighbours of query entity $e$ in a knowlege graph $K$. More details are available in [6]. **The naive approach would be to compute $|E|$ conceptual distances between $e$ and the other entities. However, this is inefficient because in practice different entities will lead to the same concept or to concepts with overlapping intensions, and so some computations will be done again and again. Intuitively, our approach consists in computing the conceptual distances against clusters of entities instead of single entities.** More concretely, our algorithm works by iteratively refining a partition $\{S_l\}_l$ of the set of entities $E$, where each $S_l$ is a cluster of entities, in order to get an increasingly accurate partition converging to the partition induced by the proper extensions of C-NNs.

**Each cluster $S_l$ is associated with a query $Q_l = x \leftarrow P_l$, and a set of candidate query elements $H_l$. The relationship to C-NNs is that when $H_l$ is empty, the pair $\delta_l = (ans(Q_l), Q_l)$ is a C-NN (i.e. $\delta_l \in C\text{-}NN(e, K)$), and $S_l$ is the proper extension of $\delta_l$. When $H_l$ is not empty, $\delta_l$ is a generalization of concepts of neighbours, in the sense that either $ans(Q_l)$ is the union of the extensions of several concepts of neighbours (lack of discrimination), or $Q_l$ is not necessarily the most specific query that matches all entities in $ans(Q_l)$ (lack of precision in the conceptual similarity). In that case, one gets overestimates of conceptual distances for some of the entities in $S_l$.**

Algorithm 1 details the partitioning algorithm. Initially, there is a single cluster $S_0 = E$ with $P_0$ being the empty pattern, and $H_0$ being the *description* of $e$. The description of an entity $e$ is a graph pattern that is obtained by extracting a subgraph around $e$ and, for each entity $e_i$ in the subgraph, by replacing $e_i$ by a variable $y_i$, and by adding filter $y_i = e_i$. Here, we choose to extract the subgraph that contains all edges starting from $e$ up to some depth.

Then at each iteration, any cluster $S$ – with pattern $P$ and set of candidate query elements $H$ – is split in two clusters $S_1, S_0$ by using an element $h \in H$ as a discriminating feature. Element $h$ must be chosen so that $P \cup \{h\}$ defines a connected pattern including variable $x$. Each addition of a query element therefore does one of the following: (a) add a filter contraint on a pattern variable, (b) add an edge between two pattern variables, or (c) add an edge from a pattern variable to a fresh variable. Many strategies are possible for choosing this element. In this work, we only consider two simple strategies:

- **Random**: random choice among the valid elements;

- **Ordered**: choice of equality elements first, then triple patterns whose relation is the least frequently used in $P$ (novelty).

The new clusters are defined as follows:

$$\begin{array}{llll} P_1 = P \cup \{h\} & Q_1 = x \leftarrow P_1 & S_1 = S \cap ans(Q_1, K) & H_1 = H \setminus \{h\} \\ P_0 = P & Q_0 = Q & S_0 = S \setminus S_1 & H_0 = H \setminus \{h\} \end{array}$$

The equations for $S_1, S_0$ ensure that after each split there is still a partition, possibly a more accurate one. The empty clusters ($S_i = \emptyset$) are removed from the partition. As a consequence, although the search space is the set of subgraphs of the description of $e$, which has a size exponential in the size of the description, the number of clusters remains below the number of entities at all time. In the running example, the initial cluster $S_{1-6}$ (the union of concepts $\delta_1$ to $\delta_6$) is split with element $(x = Charlotte)$ into $S_1$ and $S_{2-6}$. Then cluster $S_{2-6}$ is split with element $(x, gender, s)$ into $S_{2-5}$ and $S_6$. Then cluster $S_{2-5}$ is split with element $(s = female)$ into $S_2$ and $S_{3-5}$. Next splits involve elements $(x, parent, y)$ and $(y = William)$ on $S_{3-5}$.

*Handling of RDF Schema (RDFS).* The implementation takes into account the domain knowledge expressed as RDF Schema axioms [21]. A special treatment is done for triples $(e_i, \texttt{rdf:type}, e_j)$, where $e_j$ is a class, i.e. an unary predicate from the logical point of view. Such an unary predicate is simulated by not replacing $e_j$ by a variable in the description of $e$. Taking inspiration from query relaxation [22], the implementation also takes into account hierarchies of classes and properties, and domains and ranges of properties. For instance, this enables to find that two entities with respective type "horse" and "cat" have type "mammal" in common, even if it does not appear in their explicit description. A naive way to achieve this is to saturate entity descriptions with inferred triples. A more efficient way is to refine the above equation defining $H_0$ as $H_0 = H \setminus \{h\} \cup relax(h)$, where $relax(h)$ is a set of relaxations of $h$ [6]. For instance, the relaxation of a class is its set of superclasses.

*Termination and efficiency.* The above algorithm terminates because the set $H$ (or its saturation in case of RDFS axioms) decreases at each split. However, in the case of large descriptions or large knowledge graphs, it can still take a long time. Now, previous experiments [6] show that most concepts are produced early, and that the rest of the time is spent at finding the last few concepts. Moreover, our algorithm is anytime because it can output a partition of entities at any time, and hence concepts of neighbours (possibly generalizations of them). It therefore makes sense to control runtime with a timeout parameter.

Actually, the above algorithm converges to an approximation of the C-NNs, in the sense that the conceptual distance may be still be an overestimate at full runtime for some entities. This is because graph patterns are constrained to be subsets of the description of $e$. In order to get exact results, the duplication of variables and their adjacent edges should be allowed, but this would considerably increase the search space.

Experiments on KGs with up to a million triples have shown that the algorithm can compute all C-NNs for descriptions of hundreds of edges in a matter of seconds or minutes. In contrast, query relaxation does not scale beyond 3 relaxation steps, which is insufficient to identify similar entities in most cases; and computing pairwise symbolic similarities does not scale to large numbers of entities. A key ingredient of the efficiency of the algorithm also lies in the notion of *lazy join* for the computation of answer sets of queries. In short, the principle is to compute $S_1$ from $S$ in an incremental way (rather than computing $ans(Q_1, K)$ from scratch), and to avoid the enumeration of all matchings of $P_1$ by computing joins only as much as necessary to compute the set of query answers (see details in [6]).

**A last point to discuss is the scalability of C-NNs computation w.r.t. the KG size, i.e. what is the impact of doubling the number of entities. If we make the reasonable assumption that the new knowledge follows the same schema as the old knowledge (e.g., adding film descriptions to a film KG), then the impacts are that: a) the query answers are two times larger, and b) the number of C-NNs is somewhere between the same and the double, depending on the novelty of the new knowledge. As a consequence, if our objective is to compute a constant number of C-NNs for further inference, then it suffices to increase the timeout linearly with KG size.**

## 6. Link Prediction

The problem of *link prediction* is to infer a missing entity in a triple $(e_i, r_k, e_j)$, i.e. either infer the tail from the head and the relation, or infer the head from the tail and the relation. Because of the symmetry of the two problems, we only describe here the inference of the tail entity. In the following, we therefore consider $e_i$ and $r_k$ as fixed (we avoid them in indices), and $e_j$ as variable.

### *6.1. C-NN-based Inference*

The principle of C-NN-based inference is to generate a ruleset for each concept of neighbours $\delta_l \in \mathit{C\text{-}NN}(e_i, K)$. Those rules are similar in nature to those of AMIE+ or AnyBURL except that only rules that are matched by the head entity $e_i$ are generated. Indeed, the bodies of generated rules are intensions of C-NNs, which are subsets of $e_i$'s description. From the concept intension $\mathit{int}(\delta_l) = Q_l = x \leftarrow P_l$, we generate two kinds of inference rules $\rho$:

(1) **by-copy rules**: $\rho := P_l \rightarrow (x, r_k, e_j)$, for each $e_j \in E$;
(2) **by-analogy rules**: $\rho := P_l \rightarrow (x, r_k, y)$, for each $y \in \mathit{Vars}(P_l), y \neq x$.

*Inference by copy.* The first kind of rules (**by-copy rules**) state that when an entity matches $P_l$ as $x$, it is related to entity $e_j$ through relation $r_k$. As $e_i$ matches $P_l$ as $x$ by definition of concepts of neighbours, it can be inferred that $e_i$ is related to $e_j$. **In formula, the set of inferred entities is simply $E_\rho = \{e_j\}$.** Of course, the strength of the inference depends on the support and confidence of the rule $\rho$, which are defined as follows.

$$\mathit{supp}(\rho) = |\mathit{ans}(x \leftarrow P_l, (x, r_k, e_j))| \quad \mathit{conf}(\rho) = \frac{|\mathit{ans}(x \leftarrow P_l, (x, r_k, e_j))|}{|\mathit{ans}(x \leftarrow P_l)| + \lambda}$$

The support is defined as the number of entities that match $P_l$ as $x$ and are related to $e_j$ through $r_k$. The confidence is defined as the ratio of the support out of the number of entities that match $P_l$ as $x$. Like this is done in AnyBURL [17], we use a constant $\lambda \geqslant 0$ as a kind of additive Laplace smoothing, in order to favor rules with larger support. The principle of *inference by copy* is that the tail entities $e_j$ to be predicted for $e_i$ can be copied from the tail entities of $e_i$'s neighbour entities. For example, if triple $(\mathit{Charlotte}, \mathit{parent}, \mathit{Kate})$ is missing in the above example graph, it can be inferred from some of her closest neighbours: "George and Louis are similar to Charlotte because they have William as a father, and their mother is Kate, so Charlotte's mother could be Kate too" (support=2, confidence=$2/(3+\lambda)$).

*Inference by analogy.* The second kind of rules (**by-analogy rules**) state that when a pair of entities match $P_l$ as $x$ and $y$, they are related through $r_k$. As $e_i$ matches $P_l$ as $x$ by definition of concepts of neighbours, it can be inferred that $e_i$ is related through $r_k$ to any entity $e_j$ that matches $P_l$ as $y$ when $x = e_i$. **In formula, the set of infered entities is $E_\rho = ans(y \leftarrow P_l, (x = e_i))$.** Like above, the strength of the inference depends on the support and confidence of the rule, which are defined as follows.

$$supp(\rho) = |ans((x,y) \leftarrow P_l, (x, r_k, y))| \quad conf(\rho) = \frac{|ans((x,y) \leftarrow P_l, (x, r_k, y))|}{|ans((x,y) \leftarrow P_l)| + \lambda}$$

The support is defined as the number of pairs of entities that match $P_l$ as $x$ and $y$, and that are related through $r_k$. The confidence is the ratio of the support out of the number of pairs of entities that match $P_l$ as $x$ and $y$, plus Laplace smoothing $\lambda$. The principle of *inference by analogy* here relies on the observation that "$e_i$ is to $e_j$ as $x$ is to $y$". By observing that pairs $(x,y)$ that match pattern $P_l$ often satisfy $(x, r_k, y)$, one can predict the missing entity in $(e_i, r_k, ?)$ to be any entity $e_j$ such that pattern $P_l$ is matched for $x = e_i$ and $y = e_j$. For example, assume in the example graph that George, Charlotte, and Louis are only known to have father William, and that Kate is only known to be William's spouse, i.e. triples $(\{George, Charlotte, Louis\}, parent, Kate)$ are missing. Here, inference by copy for the parents of Charlotte would only produce Charles and Diana, using William and Harry as similar entities (see C-NN $\delta_4$ in Table 1). The intension of the conceptual similarity of Charlotte with William and Harry is

$$\begin{aligned} Q_{WH} &= x \leftarrow P_{WH} \\ P_{WH} &= (x, gender, s), (x, parent, y), (y, gender, t), (t = male), \\ &\quad (y, spouse, z), (z, spouse, y), (z, gender, u), (u = female), \end{aligned}$$

saying that "they have a father married to a woman". From there the following by-analogy rule can be generated: $P_{WH} \rightarrow (x, parent, z)$. This rule states that "any female spouse (wife) of a male parent (father) is a parent". Applying this rule to Charlotte (and equivalently to George and Louis) leads to the inference of the fact $(Charlotte, parent, Kate)$ because Kate is indeed the wife of William, who is the father of Charlotte. It is noteworthy here that Kate is predicted to be a parent although she never appears as a parent in the incomplete graph. This is not possible with inference by copy.

*Comparison with rules in other approaches.* The main difference with other rule-based approaches is that we only generate rules that are specific to the entity $e_i$ and relation $r_k$ for which predictions are to be made. Looking in more detail to the structure of rules, each approach has its own language bias. In AMIE+ [15], only by-analogy rules are considered, and equality filters are excluded in most experiments because they make the number of rules explodes with the length of rules. They only retain *closed* rules, i.e. rules where each variable occurs at least twice. However, non-closed rules still have to be generated in order to reach all closed rules. Another difference is about rule measures. They use *PCA confidence*, a variation of classical confidence under the *Partial Completeness Assumption (PCA)*; and *head coverage*, which is a form of recall of the rule relative to the extension of relation $r_k$. In AnyBURL [17], rule shapes are restricted to chains and cycles (including rule's head), where rule's head and equality filters cannot occur in the middle of a chain. The case of a chain with an equality filter on rule's head is a subset of our by-copy rules. The case of a cycle is a subset of our by-analogy rules. AnyBURL supports a third kind of rule that is not covered in this work: $P \rightarrow (e_i, r_k, y)$, where $y \in Vars(P)$. Here, entities $e_j$ are predicted based on their own description, indepently of entity $e_i$, which acts as a constant here. In RLvLR [23], rules are equivalent to the cyclic rules of AnyBURL.

### 6.2. Aggregation of Inferences

Given an incomplete triple $(e_i, r_k, ?)$, the output of a link prediction system is a ranking of entities $e_j$. Rankings of entities are evaluated with measures such as Hits@$N$ (defined as 1 if the correct entity is among the first $N$ entities, 0 otherwise), and MRR (Mean Reciprocal Rank, the inverse of the rank of the correct entity).

The question addressed in this section is how to aggregate all inferences presented in previous section into a global ranking. **Indeed, the known entity $e_i$ leads to multiple concepts of neighbours, each concept of neighbours $\delta_l$ generates multiple rules, and each rule $\rho$ infers a set $E_\rho$ of candidate entities $e_j$.** The same candidate $e_j$ may be inferred by several rules, possibly generated from different concepts. In order to avoid the handling of too many rules, we exclude rules whose confidence is less than 1%, which is a very weak constraint. As a consequence, it may happen that some entities are not inferred by any rule, and hence that rankings do not include all entities of the KG. The missing entities are those for which there is not a single supporting evidence, and their rank is considered as infinite (Hits@$N$=0, MRR=0). The idea is to combine the measures of rules to give a score to each candidate $e_j$, in order to get a global ranking. In this paper, we consider two scores, which we reuse or adapt from previous work, and detail below:

- **Maximum Confidence (MC)**: introduced in AnyBURL [17];
- **Dempster-Shafer (DS)**: adapted from Denœux's work on k-NN classification [24].

*Maximum Confidence (MC) score.* This score is quite simple, and was shown effective in AnyBURL's work. The score of entity $e_j$ is simply the list of the confidences of the rules that infer triple $(e_i, r_k, e_j)$, in decreasing order. This is a refinement of the score that was used in first experiments of AMIE, where the score was the maximum confidence among rules predicting $e_j$. Using a list of confidences instead of a single confidence allows for a finer ranking. Ranking of lists of confidences is done similarly to lexicographic ordering. Candidates are first compared with the first confidence. If their first confidence are equal, comparison is done with the second confidence, and so on. Then, candidates with longer lists are put first because they have more rules predicting them.

*Dempster-Shafer (DS) score.* This score is inspired by the work of Denœux [24], and adapted to our concepts of neighbours. Denoeux defines a k-NN classification rule based on Dempster-Shafer (DS) theory. Each $k$-nearest neighbour $x_l$ of an instance to be classified $x$ is used as a piece of evidence that supports the fact that $x$ belongs to the class $c_l$ of $x_l$. The degree of support is defined as a function of the distance between $x$ and $x_l$, in such a way that the choice of $k$ is less sensitive, so that large values of $k$ can be chosen. DS theory enables to combine the $k$ pieces of evidence into a global evidence, and to define a measure of *belief* for each class, which can be used as a score.

We adapt Denoeux's work to the inference of $e_j$ in triple $(e_i, r_k, ?)$ in the following way. Each rule $\rho$ that predicts $e_j$ is used as a piece of evidence. The degree of support depends on the extensional distance $dist(\rho) = |ext(\delta_l)|$ of the concept of neighbours $\delta_l$ that generated the rule, and on the confidence of the rule. In order to have comparable pieces of evidence, we instantiate each by-analogy rule $P_l \rightarrow (x, r_k, y)$ for each possible value of $y$ into a by-copy rule: $P_l, (y = e_j) \rightarrow (x, r_k, e_j)$. Indeed, by-analogy rules tend to be more general, and hence to have larger distances.

Because in KGs a head entity can be linked to several tail entities through the same relation, we consider a distinct classification problem for each candidate tail entity $e_j \in E$ with two classes $c_j^1$ ($e_j$ is a tail entity) and $c_j^0$ ($e_j$ is not a tail entity). For each candidate entity $e_j$ and each rule $\rho$ predicting $e_j$, the

**Algorithm 2** Inference and ranking of candidate entities $e_j$ for entity $e_i$ and relation $r_k$ based on C-NNs

**Require:** $K = \langle E, R, T \rangle$ a knowledge graph, $e_i \in E$ an entity, and $r_k$ the relation to predict for $e_i$
**Require:** *getScore*, a function computing an entity score from a set of rules (e.g., max confidence)
**for all** $e_j \in E$ **do**
  $rules[e_j] := \emptyset$ // initialisation of the set of rules inferring each entity
**end for**
**for all** $(S, P, H) \in Partitioning(e_i, K)$ // for each C-NN **do**
  **for all** $e_j \in E$ **do**
    $\rho := P \rightarrow (x, r_k, e_j)$ // for each by-copy rule
    **if** $conf(\rho) \geqslant minConf$ **then**
      $rules[e_j] := rules[e_j] \cup \{\rho\}$
    **end if**
  **end for**
  **for all** $y \in Vars(P)$ s.t. $y \neq x$ **do**
    $\rho := P \rightarrow (x, r_k, y)$ // for each by-analogy rule
    **if** $conf(\rho) \geqslant minConf$ **then**
      **for all** $e_j \in ans(y \leftarrow P, (x = e_i))$ // for each inferred entity **do**
        $rules[e_j] := rules[e_j] \cup \{\rho\}$
      **end for**
    **end if**
  **end for**
**end for**
**for all** $e_j \in E$ **do**
  $score[e_j] := getScore(rules[e_j])$ // aggregating rule scores into an entity score
**end for**
**return** a ranking of entities $e_j \in E$ according to $score[e_j]$

degree of support can therefore be formalized by defining a mass distribution $m_{j,\rho}$ over sets of classes as follows.

$$m_{j,\rho}(\{c_j^1\}) = \alpha_0 \, conf(\rho) \, e^{-dist(\rho)} \qquad m_{j,\rho}(\{c_j^0\}) = 0 \qquad m_{j,\rho}(\{c_j^1, c_j^0\}) = 1 - \alpha_0 \, conf(\rho) \, e^{-dist(\rho)}$$

$m_{j,\rho}(\{c_j^1\})$ represents the degree of belief for $e_j$ being the tail entity, while $m_{j,\rho}(\{c_j^1, c_j^0\})$ represents the degree of uncertainty. $m_{j,\rho}(\{c_j^0\})$ is set to 0 to reflect the OWA (Open World Assumption) of KGs according to which a missing fact is not considered as false. Constant $\alpha_0$ determines the maximum degree of belief, which can be lower than 1 to reflect uncertainty about known triples (e.g. 0.95). The degree of belief decreases exponentially with distance. Finally, we make the degree of belief proportional to the confidence of infering entity $e_j$ with rule $\rho$. In [24], that confidence factor does not exist because it would be 1 for the class of the nearest neighbour, and 0 for every other class.

By applying Dempster-Shafer theory to combine the evidence from all rules $\rho$ that predict entity $e_j$, we arrive at the following equation for the belief of each candidate tail entity $e_j$.

$$Bel_j = m_j(\{c_j^1\}) = 1 - \prod_{\rho} 1 - \alpha_0 \, conf(\rho) \, e^{-dist(\rho)}$$

We can use that belief as a score to rank all predicted entities by decreasing belief. The score used by AMIE+ is a special case of this score, where $\alpha_0 = 1$, $dist(\rho) = 0$ for all rules, and PCA confidence is used instead of classical confidence.

### 6.3. Inference Algorithm and Explanations

**Algorithm 2 summarizes the full inference process in pseudo-code. Lines 1-3 initialises for each candidate entity the set of rules that infer it. Line 4 iterates over the C-NNs of entity $e_i$, after calling the partitioning algorithm (see Algorithm 1). Line 5 iterates over the by-copy rules generated from the current concept, and Line 11 iterates over the by-analogy rules. For each rule, if its confidence is above some threshold (0.01 in our experiments), it is added to the set of rules of all the entities infered by the rule (Line 8 for by-copy rules, and Lines 14-16 for by-analogy rules). Lines 20-22 computes a score for each candidate entity by aggregating each set of rules into a single score. Finally, Line 23 simply returns a ranking of the candidate entities based on their computed score.**

**The algorithm can easily be extended to output explanations for each inferred entity. At Line 21, an explanation can be computed from the set of rules, in addition to the entity score. A simple form of explanation is to select the top-K rules according to the chosen score: confidence in the case of Maximum Confidence, and degree of belief in the case of Demster-Shafer. Each rule is directly interpretable in terms of entities and relationships (see Section 6.1). Their readability can be improved by pretty-printing or graphical representation of the graph pattern. In our implementation, the graph patterns are displayed in a Turtle-like notation by doing a tree traversal starting from variable $x$, which for recall denotes entity $e_i$ and its neighbours. A more advanced form of explanation could post-process rules by removing redundant elements, or by merging similar rules. This is left to future work.**

## 7. Experiments

We here report on experiments comparing our approach to other approaches on several standard benchmarks for link prediction. We first present the methodology, and study the impact on performance of the different strategies and hyper-parameters of our approach. Then we report performance results on benchmarks with an in-depth analysis, and finally present a fine-grained comparison with AnyBURL. The companion page[2] provides access to the source code, the datasets, and the experiment main results and logs.

### 7.1. Methodology

*Datasets.* We use four datasets that are used in many work to evaluate the link prediction task (WN18, WN18RR, FB15k, FB15k-237), plus an additional dataset that we derived from the Mondial dataset [25]. Table 2 provides statistics about datasets (numbers of: entities, relations, train edges, validation edges, and test edges). WN18 and FB15k were introduced by Bordes *et al* [7] for link prediction evaluation. WN18 is derived from a subset of WordNet, and FB15k from FreeBase. In later work, it was observed that many test triples had their inverse in the training set (e.g., *hypernym* is the inverse of *hyponym* in WordNet), and could be solved with very simple rules (e.g., $(y, hypernym, x) \rightarrow (x, hyponym, y)$). More

[2] Companion page: http://www.irisa.fr/LIS/ferre/pub/link_prediction2020/

Table 2
Statistics of datasets.

| Dataset | entities | relations | train edges | valid. edges | test edges |
|---|---|---|---|---|---|
| WN18 | 40 943 | 18 | 141 442 | 5 000 | 5 000 |
| WN18RR | 40 559 | 11 | 86 835 | 3 034 | 3 134 |
| FB15k | 14 951 | 1 345 | 483 142 | 50 000 | 59 071 |
| FB15k-237 | 15 541 | 237 | 272 115 | 17 535 | 20,466 |
| Mondial | 2 473 | 20 | 8 456 | 570 | 701 |

challenging datasets were introduced by removing from the validation and test sets all pairs of entities that occur in the training set. FB15k-237 was introduced by Toutanova and Chen [8], and WN18RR was introduced by Dettmers *et al.* [9].

We introduce Mondial as a subset of the Mondial database [25], which contains facts about world geography. We simplified it to the task of link prediction by removing labelling edges and edges containing dates and numbers, and by unreifying n-ary relations into binary relations. Triples whose relation is `rdf:type` were also removed from the validation and test sets because they do not represent proper links between entities. The dataset is available from the companion page. It is smaller than other datasets but it makes it easier to interpret results, and we use it to study the impact of the different strategies and hyper-parameters without introducing a bias in our evaluation on the classical benchmarks.

*Task.* We follow the same protocol as introduced in [7], and followed by subsequent work. The task is to infer, for each test triple, the tail entity from the head and the relation, and also the head entity from the tail and the relation. We call *test entity* the known entity, and *missing entity* the entity to be infered. We evaluate the performance of our approach by using the same three measures as in [17]: MRR and Hits@{1,10} (given in percents in this paper). Like in previous work, we use filtered versions of those measures to reflect the fact that, for instance, there may be several correct tail entities for a 1-n relation (e.g., the relation from awards to nominees). For example, if the correct entity is at rank 7 but 2 out of the first 6 entities form triples that belong to the dataset (and are therefore considered as valid), then it is considered to be at rank 5.

*Method.* Because our approach has no training phase we can use both train and validation datasets as examples for our instance-based inference. We consider a few alternative strategies:

- Random vs Ordered choice of a query element in the partitionning algorithm (see Section 5.2);
- by-copy vs by-analogy vs both kinds of inference rules (see Section 6.1);
- MC vs DS score for ranking infered entities (see Section 6.2).

Our approach has only three hyper-parameters (and no parameter to learn): the *maximum depth* of the description of the test entity, and two *timeouts*: one for the computation of C-NNs, and another one for the inference of an entity ranking. For the inference of a ranking of entities, we set $\lambda = 2$ in the definition of rule confidence, and $\alpha_0 = 0.95$ in the DS-score.

The implementation of our approach has been integrated to SEWELIS as an improvement of previous work on the guided edition of RDF graphs [26]. A standalone program for link prediction is available from the companion page. We ran our experiments on Fedora 25, with CPU Intel(R) Core(TM) i7-6600U @ 2.60GHz, and 16GB DDR4 memory. So far, our implementation is simple and uses a single core, although our partitioning algorithm lends itself to parallelization. We have observed that in all our experiments the memory footprint, which includes the training and validation triples, remains under a

Table 3
Comparing MRR on Mondial with 12 different strategies

| | by-copy | | by-analogy | | by-copy+analogy | |
|---|---|---|---|---|---|---|
| | MC | DS | MC | DS | MC | DS |
| Random | 28.6 | 16.9 | 42.4 | 41.3 | **45.5** | 44.2 |
| Ordered | 30.1 | 19.2 | 33.7 | 30.8 | 40.2 | 37.3 |

few percents, i.e. a few hundreds MB. An alternative implementation[3] for the computation of C-NNs has recently been developed as a Java library on top of the Jena Framework[4], in order to facilitate their reuse in other applications.

*Baselines.* We compare our approach to a number of historical or state-of-the-art latent-based approaches: DISTMULT [27], ANALOGY [28], KB_LR [29], R-GCN+ [10], ConvE [9], ComplEx-N3 [11], and CrossE [30]. We do so by choosing the same tasks and measures as in previous work because it was not possible for us to run them ourselves (no access to a GPU), and also because it allows for a fairer comparison (e.g., choice of hyper-parameters by authors). We also compare our approach to rule-based approaches: AMIE+ [15], RuleN [16], and AnyBURL [17]. Most performance results of the above approaches are reproduced from Meilicke *et al.* [17] (see Section 7.3 for details). We add yet another baseline *Freq* that simply consists in ranking entities $e_j$ according to their decreasing frequency of usage in $r_k$ over the train+valid dataset, as defined by $freq_j = |ans(x \leftarrow (x, r_k, y), (y = e_j))|$. It is independent of the test entity, and therefore acts as a default ranking.

### 7.2. Impact of Strategies and Hyper-parameters and Consistency of Results

*Impact of strategies.* Table 3 compares the obtained MRR (in percents) on the Mondial dataset with different strategies. The timeout was set to 0.1s for both C-NN computation and inference, and the maximum depth was set to 3. For recall, there are three axes composing the strategy:

- choice of a spliting element: **Random** vs **Ordered**;
- kinds of inference rules: **by-copy** vs **by-analogy** vs both (**by-copy+analogy**);
- aggregation of inferences: **MC** (max. confidence) vs **DS** (Dempster-Shafer).

This results in 12 different strategies, and Table 3 shows that the best performing combines **Random**, **by-copy+analogy**, and **MC**. Looking more in detail, we observe that combining both kinds of rules is always beneficial, as well as aggregating inferences with maximum confidence. However, **Ordered** element choice is better with **by-copy** rules, while **Random** element choice is better with **by-analogy** rules. We can also evaluate the importance of each axis by measuring the MRR decrease when replacing the best option by another. The most important options are in order: **by-analogy** rules (-16.9), **Random** choice (-5.3), **by-copy** rules (-3.1), and finally **MC** aggregation (-1.3). From now on, we only use the best performing strategy.

*Impact of timeout.* Table 4 shows the evolution of a number of measures when timeout exponentially increases from 1ms to 1s. There are three performance measures (Hits@1, Hits@10, MRR), followed by three measures about the inference process, averaged over the test set: number of computed concepts of neighbours, number of inferred entities, and maximum confidence over all inferred entities. It can be

[3] https://bitbucket.org/sebferre/conceptsofneighbours/
[4] https://jena.apache.org/

Table 4
Evolution of performance and internal parameters with timeout

| timeout(s) | Hits@1 | Hits@10 | MRR | nb. concepts | nb. inferred | max. conf |
|---|---|---|---|---|---|---|
| 0.001 | 15.7 | 31.8 | 21.2 | 3.7 | 103 | 0.20 |
| 0.01 | 32.3 | 51.3 | 38.9 | 11.9 | 211 | 0.42 |
| 0.1 | 38.6 | 59.1 | 45.5 | 35.0 | 311 | 0.57 |
| 1 | 39.3 | 63.7 | 47.3 | 89.2 | 351 | 0.66 |

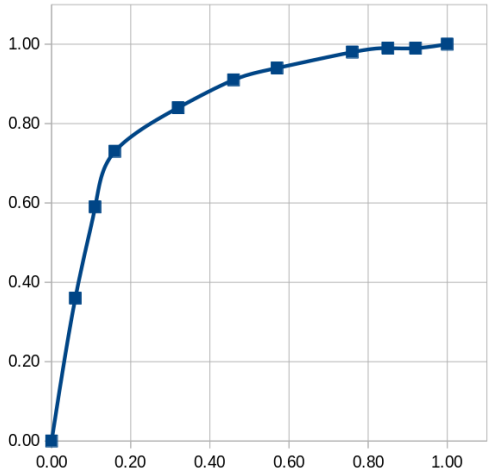


Fig. 4. ROC curve of predicting Hits@1 correctness depending on maximum confidence threshold

observed that with only 1% of the largest timeout (timeout 0.01s vs 1s), the MRR is already at 82% of the larger MRR, and 60% of the inferred entities are obtained, despite the fact that only 13% of the concepts have been computed. This indicates that early approximations of concepts of neighbours are already informative. Furthermore, increasing the timeout and hence the number of concepts does not only improve MRR but also increases confidence in inferences as indicated by the steadily increasing maximum confidence.

*Impact of description depth.* We have observed that increasing description depth beyond 2-3 makes almost no significant difference on performance measures, as well as on the number of inferred entities and on the maximum confidence. In the following, we therefore stick to a maximum description depth equal to 3.

*Consistency of results.* To evaluate the consistency of the results of our approach, we perform two analyses on the Mondial dataset (with timeout=0.1s). First, we analyze the variability of the performance measures by spliting the test set 10-fold. We observe that the standard error accross the 10 folds is very small for all measures: e.g., 0.5% for MRR, around 1% for Hits@1 and Hits@10, and 0.4% for maximum confidence.

Second, we analyze the relevance of the maximum confidence by measuring how predictive it is about the correctness of predictions. Figure 4 shows the ROC curve for the classification task that consists in predicting whether the top entity in the filtered ranking is correct based on the hypothesis that its associated maximum confidence is above a threshold varying from 0 to 1. For example, with threshold at 0.70, the true positive rate is at 73% while the false positive rate is at 16%. The shape of the curve, as well

Table 5
Comparison of performance results on WN18 and WN18RR

| | | WN18 | | | WN18RR | | |
|---|---|---|---|---|---|---|---|
| Approach | source | H@1 | H@10 | MRR | H@1 | H@10 | MRR |
| *Freq* | | 1.8 | 5.0 | 2.9 | 1.5 | 4.4 | 2.6 |
| *Latent-based* | | | | | | | |
| DISTMULT [27] | [10] | 70.1 | 94.3 | 81.3 | - | - | - |
| ANALOGY [28] | [28] | 93.9 | - | 94.2 | - | - | - |
| KB_LR [29] | [29] | - | 95.1 | 93.6 | - | - | - |
| R-GCN+ [10] | [17] | 69.7 | 96.4 | 81.9 | - | - | - |
| ConvE [9] | [17] | 93.5 | 95.5 | 94.2 | 39.0 | 48.0 | 46.0 |
| ComplEx-N3 [11] | [17] | - | 96.0 | 95.0 | - | **57.0** | **48.0** |
| CrossE [30] | [17] | 74.1 | 95.0 | 83.0 | - | - | - |
| *Rule-based* | | | | | | | |
| AMIE+ [15] | [17] | 87.2 | 94.8 | - | 35.8 | 38.8 | - |
| RuleN [16] | [17] | 94.5 | 95.8 | - | 42.7 | 53.6 | - |
| AnyBURL [17] | [17] | 93.9 | 95.6 | 95.0 | **44.6** | 55.5 | **48.0** |
| C-NN (ours) | | **96.7** | **97.2** | **96.9** | 44.4 | 51.9 | 46.9 |
| C-NN − best other | | +2.2 | +0.8 | +1.9 | -0.2 | -5.1 | -1.1 |
| C-NN − best other rule-based | | +2.2 | +1.4 | +1.9 | -0.2 | -3.6 | -1.1 |

as its AUC (Area Under Curve) equal to 84%, demonstrate the relevance of the maximum confidence to estimate the validity of predictions. This comes in complement with interpretable explanations for inferred entities in the form of rules.

### 7.3. Results on Benchmarks

Tables 5 and 6 compare the results of our approach (C-NN) to other approaches presented above as baselines on the four datasets WN18, WN18RR, FB15k, and FB15k-237. The baselines are organized in three groups: naive baseline *Freq*, latent-based approaches, and rule-based approaches. In each group, approaches are sorted by publication year. The *source* indicates which paper the results are taken from. The results for AnyBURL are those where 1000s (largest available time) are allocated to the computation of rules. In the results of C-NN, the timeouts (computation of concepts + inference) are 1+1s for WN datasets, and 1.5+0.5s for FB datasets. The output logs of C-NN predictions and explanations are available from the companion page.

ComplEx-N3 clearly outperforms other approaches on all datasets except WN18 where C-NN outperforms other approaches on the three measures. C-NN comes second on FB15k, and remains close to the best approaches on WN18RR. On FB15k-237, the MRR delta is -7.4 with ComplEx-N3, but only -0.4 with AnyBURL, the best rule-based approach. It is noteworthy that C-NN is competitive with AnyBURL because, whereas AnyBURL rules are computed in a supervised manner (knowing the target relation $r_k$), C-NN concepts are computed in an unsupervised manner (i.e, only knowing the source entity $e_i$). This implies that the concepts of neighbours of an entity can be computed once, and used for many different inference tasks, e.g. predicting links for several target relations.

Looking at the *Freq* baseline, it becomes visible how FB15k-237 is difficult because performance improvements over *Freq* are relatively small while they are huge on other datasets.

Table 6
Comparison of performance results on FB15k and FB15k-237

| | | FB15k | | | FB15k-237 | | |
|---|---|---|---|---|---|---|---|
| Approach | source | H@1 | H@10 | MRR | H@1 | H@10 | MRR |
| *Freq* | | 14.3 | 28.5 | 19.2 | 17.5 | 35.6 | 23.6 |
| *Latent-based* | | | | | | | |
| DISTMULT [27] | [10] | 52.2 | 81.4 | 63.4 | 10.6 | 37.6 | 19.1 |
| ANALOGY [28] | [28] | 64.6 | - | 72.5 | - | - | - |
| KB_LR [29] | [29] | 74.2 | 87.3 | 79.0 | 22.0 | 48.2 | 30.6 |
| R-GCN+ [10] | [17] | 60.1 | 84.2 | 69.6 | 15.1 | 41.7 | 24.9 |
| ConvE [9] | [17] | 67.0 | 87.3 | 74.5 | **23.9** | 49.1 | 31.6 |
| ComplEx-N3 [11] | [17] | - | **91.0** | **86.0** | - | **56.0** | **37.0** |
| CrossE [30] | [17] | 63.4 | 87.5 | 72.8 | 21.1 | 47.4 | 29.9 |
| *Rule-based* | | | | | | | |
| AMIE+ [15] | [17] | 64.7 | 85.8 | - | 17.4 | 40.9 | - |
| RuleN [16] | [17] | 77.2 | 87.0 | - | 18.2 | 42.0 | - |
| AnyBURL [17] | [17] | 80.4 | 89.0 | 83.0 | 23.0 | 47.9 | 30.0 |
| C-NN (ours) | | **82.7** | 89.0 | 84.9 | 22.2 | 44.6 | 29.6 |
| C-NN − best other | | +2.3 | -2.0 | -1.1 | -1.7 | -11.4 | -7.4 |
| C-NN − best other rule-based | | +2.3 | 0.0 | +1.9 | -0.8 | -3.3 | -0.4 |

Table 7
Split of test sets depending on the kind of inference rule (by-copy vs by-analogy).

| | WN18 | | WN18RR | | FB15k | | Fb15k-237 | |
|---|---|---|---|---|---|---|---|---|
| measure | by-c | by-a | by-c | by-a | by-c | by-a | by-c | by-a |
| percent | 2% | 98% | 38% | 57% | 14% | 85% | 73% | 25% |
| Hits@1 | 15.0 | 98.7 | 15.2 | 67.8 | 37.0 | 91.0 | 26.2 | 12.2 |
| Hits@10 | 34.4 | 98.8 | 29.4 | 71.6 | 63.5 | 94.2 | 49.9 | 32.3 |
| MRR | 22.0 | 98.7 | 19.9 | 69.1 | 47.0 | 92.1 | 34.1 | 18.7 |
| max. conf | 0.24 | 0.96 | 0.28 | 0.70 | 0.60 | 0.88 | 0.61 | 0.66 |

### *7.4. In-depth Analysis of Results*

In this section, we refine the performance evaluation of our approach by analysing results w.r.t two criteria: (a) the kind of rule (by-copy vs by-analogy) that inferred the rank-1 predictions, (b) the characteristics of the relation to be predicted.

*Prediction by-copy vs by-analogy.* We here analyze predictions according to the kind of the *best rule*, i.e. the rule with highest confidence, which contributed to decide the first entity in the generated ranking. For recall, there are two kinds of rules: by-copy rules and by-analogy rules (see Section 6.1). Table 7 shows the performance measures, and maximum confidence, on two subsets of the test set for each dataset, depending on the kind of the best rule. Line "percent" gives the relative size of those subsets. The two percentages may not sum up to 100% because for some test cases, no prediction could be made: e.g., all predicted entities are already known to hold, and are therefore filtered out.

The balance between by-copy and by-analogy can be explained by the nature of each dataset. In WN18 and FB15k it is well known that most test triples can be inferred from an inverse triple, which is well captured with by-analogy rules: e.g., $hypernym(y,x) \rightarrow hyponym(x,y)$. In FB15k-237, all inverse

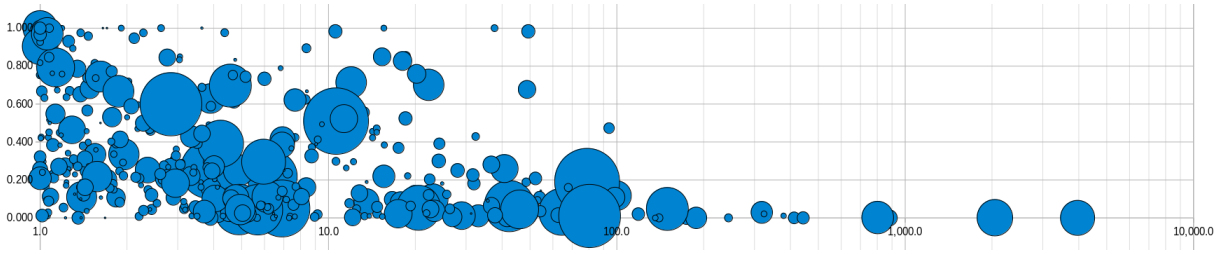


Fig. 5. MRR as a function of relation degree. Bubble sizes are proportional to relation frequency.

triples have been removed, and for many relations, some candidate entities are much more frequent than others: e.g., US nationality for people. Those frequent entities are easily inferred with by-copy rules. In WN18RR, by-analogy rules remain dominant for two reasons. First, inverse relations have been removed (e.g., *hyponym* as inverse of *hypernym*), but other inverse triples have been retained. In particular, symmetric relations such as *similar_to* and *derivationally_related_form* are still present, and account for about one third of test triples. Second, there are much less situations like "US nationality", where a single word would dominate in a relation. The performance measures (Hits@1, Hits@10, MRR) are consistent with the balance between by-copy and by-analogy: measures are better for the most frequent kind of rule. However, it appears that by-analogy rules always have a higher confidence on average. We hypothesize that this is because by-analogy inference relies on the similarity with not only entity $e_i$ but with pair of entities $(e_i, e_j)$, and involves an explicit relationship between $e_i$ and $e_j$, expressed as a graph pattern.

*Results depending on relation characteristics.* Relations are not all alike in knowledge graphs. They vary in frequency, and whether they are functional or multi-valued. We here define the *frequency* of a relation $r$ as the number of test triples using it; and its *degree* as the average number of tail entities per head entity in the training set. When the degree equals (or is closed to) 1, the relation is said *functional*, otherwise it is said *multi-valued*. As the benchmarks imply to predict both the tail from the head, and the head from the tail, we also consider inverse relations. Their frequency is the same, and their degree is the average number of head entities per tail entity. The degree of the inverse relation is unrelated to the degree of the relation, and all combinations are possible, i.e. 1-1, n-1, 1-n, and n-n relations.

Figure 5 shows a bubble plot of the 224 relations and their inverses, which are found in the test set of FB15k-237, the most challenging dataset. Each buble is positioned according to its degree (ranging from 1 to 4000), and to the MRR of its predictions (ranging from 0 to 1). The bubble size reflects the frequency of the relation, and hence its weight in the global performance measures. Several observations can be made. First, the relations cover a wide range of frequencies and degrees, and there are no obvious correlation between those two dimensions. Second, there is a clear tendency that the higher the degree, the lower the MRR, i.e. multi-valued relations are harder to predict than functional relations. However, many relations stand under and above this tendency. Some relations have a low MRR despite a low degree (e.g., *place of birth*); and some relation succeeds to have a high MRR despite degrees up to 50 (e.g., inverse of *food nutrient* with degree=15). Examples of frequent relations with high MRR are: *people nationality* (degree=1.1, MRR=79.3), *people gender* (degree=1, MRR=90.4), *topic webpage category* (degree=1, MRR=100), *marriage type of union* (degree=1.1, MRR=100), *award winners ceremony* (degree=12, MRR=71.4), and its inverse (degree=22, MRR=70.1).

Table 8
Distribution of test instances depending of Hits@1 success of C-NN and AnyBURL. The last two columns give the average Hits@1 first for the best of the two approaches, and second for the best ensemble.

| | | H@1 C-NN / H@1 AnyBURL | | | | H@1 | |
|---|---|---|---|---|---|---|---|
| dataset | nb. test | 1 / 1 | 1 / 0 | 0 / 1 | 0 / 0 | best of two | best ensemble |
| WN18 | 10 100 | 93.2 | **3.4** | 0.9 | 2.5 | 96.6 | 97.5 |
| WN18RR | 6 268 | 41.3 | 3.1 | **3.6** | 52.0 | 44.9 | 48.0 |
| FB15k | 118 142 | 75.3 | **7.4** | 5.5 | 11.8 | 82.7 | 88.2 |
| FB15k-237 | 40 932 | 17.5 | 4.7 | **6.2** | 71.6 | 23.7 | 28.4 |

### *7.5. Detailed Comparison with AnyBURL*

In this section, we detail the comparison with predictions made by AnyBURL[5]. Indeed, AnyBURL is the state-of-the-art among rule-based approaches. The main question we answer here is: How predictions differ between C-NN and AnyBURL. Table 8 gives for the four datasets the distribution of test instances into four cases depending on the Hits@1 success of each approach (C-NN and AnyBURL). If the rank-1 predicted entity is correct, then Hits@1 equals 1, otherwise it equals 0. Case 1/1 means that both approaches are correct, case 1/0 means that C-NN is correct but not AnyBURL, etc. The four cases sum up to 100%.

We consider the null hypothesis that if one approach has lesser performance, then its set of correct predictions is a subset of the correct predictions of the other approach. For instance, on FB15k-237 where AnyBURL is better, we expect that case 0/1 is empty. This is contradicted in all four datasets. For instance, on FB15k-237, C-NN is the only correct approach for 4.7% test instances, which amounts to 1918 test instances. Those results demonstrate that C-NN and AnyBURL can complement each other, and that improvement of state-of-the-art performances remains possible. This is made explicit in the last two columns that allow to compare Hits@1 of the best of the two approaches and the best ensemble of the two (union of correct predictions).

We now detail a few test instances where C-NN and AnyBURL differ (cases 1/0 and 0/1). The analysis is limited by the fact that AnyBURL's code does not output rules as explanations for predictions (although the code could be modified to do so). Here are a few test instances where C-NN succeeds while AnyBURL fails:

- Wichita falls were correctly predicted to have Central Time Zone because it is contained in Texas (by-copy rule, support=31, confidence=0.79);
- Joe Shuster was correctly predicted to have written "Superman II: The Richard Donner Cut" because he has produced a story that was honored for the film, and he won some award (by-analogy rule, support=5, confidence=0.36);
- Joe Shuster was also correctly predicted to live in Cleveland because he has produced the story "Superman II" (by-copy rule, support=1, confidence=0.33);
- Film "Good Will Hunting" was correctly predicted to have two crew roles "make-up artist" and "special effects supervisor" because it was nominated for the "Satellite Award for Best Original Screenplay" (by-copy rule, support=16, confidence=0.78).

Those C-NN rules could be found by AnyBURL, so they do not exhibit real limits of AnyBURL. However, all those rules except the second contain equality filters (constants in AnyBURL's terminology),

[5]Detailed predictions for state-of-the-art latent-based approaches are not available, and specific hardware (GPUs) is required to run them in a reasonable amount of time.

and the number of such rules is extremely large, even for small rules, because of the high number of entities in KGs. C-NN here has the advantage that it only needs to generate rules for a given test instance (instance-based learning), while AnyBURL has to generate the rules before seeing any test instance (model-based learning). We can therefore expect C-NN to have a better coverage of such rules.

Now, here are test instances where C-NN fails while AnyBURL succeeds. In each instance, we give the best C-NN rule, which fails, and then the "successful C-NN rule", i.e. the highest-confidence rule found by our approach that infer the correct entity.

- Lily Tomlin was incorrectly predicted to be nominated for TV series "Murphy Brown", instead of "The West Wing" (predicted 2nd), because he is an actor of the TV series, and won an award (by-analogy rule, support=364, confidence=0.53). The successful rule predicts "The West Wing" because he was nominated along with Joshua Malina (by-copy rule, support=14, confidence=0.50);
- Film "Life is beautiful" was incorrectly predicted to have genre Italian, instead of War Film (predicted 8th), because it appears on the title list of the Netflix genre "Italian" (by-analogy rule, support=393, confidence=0.56). The successful rule predicts War Film because it was produced in Italy (by-copy rule, support=14, confidence=0.25).

It is difficult to derive conclusions without knowing the successful AnyBURL's rules. Still, it can be observed that errors made by C-NN are reasonable ones. In the second example, the successful rule has weaker measures of support and confidence, and is less intuitive than the unsuccessful one. In reality, the film "Life is beautiful" has both genres "Italian" and "War Film", but the gold standard only contains the second. One must keep in mind that what is used as gold standard in those experiments are actually incomplete (and possibly incorrect) knowledge graphs.

## 8. Conclusion

We have shown that a symbolic approach to the problem of knowledge graph completion can be competitive with state-of-the-art approaches, both latent-based and rule-based. This comes with the major advantage that our approach can provide detailed explanations for each inference, in terms of the graph features. Compared to rule-based approaches, which can provide similar explanations, we avoid the need for a training phase that can be costly in runtime and memory (rule mining), in particular with dynamic data. Our approach is analogous to classification with k-nearest neighbours but our distances are defined as partially-ordered graph concepts instead of numbers.

There are many tracks for future work. Extending graph patterns with n-ary relations and richer filters over numbers, dates, etc. Optimizing the computation of C-NNs by finding good strategies to drive the partitioning process, or by parallelizing it. **Improving the selection and display of rules as explanations.** Evaluating our approach on other datasets, and other inference tasks.

## References

[1] S. Ferré, Link Prediction in Knowledge Graphs with Concepts of Nearest Neighbours, in: *The Semantic Web (ESWC)*, P. Hitzler et al., eds, LNCS 11503, Springer, 2019, pp. 84–100. doi:10.1007/978-3-030-21348-0_6.
[2] T. Berners-Lee, J. Hendler and O. Lassila, The Semantic Web, *Scientific American* **284**(5) (2001), 34–43.
[3] M. Nickel, K. Murphy, V. Tresp and E. Gabrilovich, A review of relational machine learning for knowledge graphs, *Proc. IEEE* **104**(1) (2016), 11–33.
[4] D. Liben-Nowell and J. Kleinberg, The link-prediction problem for social networks, *Journal of the American society for information science and technology* **58**(7) (2007), 1019–1031.

[5] S. Ferré, Concepts de plus proches voisins dans des graphes de connaissances, in: *Ingénierie des Connaissances (IC)*, 2017, pp. 163–174.

[6] S. Ferré, Answers Partitioning and Lazy Joins for Efficient Query Relaxation and Application to Similarity Search, in: *Int. Conf. The Semantic Web (ESWC)*, A. Gangemi et al., eds, LNCS 10843, Springer, 2018, pp. 209–224. doi:10.1007/978-3-319-93417-4_14.

[7] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston and O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: *Advances in neural information processing systems*, 2013, pp. 2787–2795.

[8] K. Toutanova and D. Chen, Observed versus latent features for knowledge base and text inference, in: *Work. Continuous Vector Space Models and their Compositionality*, 2015, pp. 57–66.

[9] T. Dettmers, P. Minervini, P. Stenetorp and S. Riedel, Convolutional 2D Knowledge Graph Embeddings, in: *Conf. Artificial Intelligence (AAAI)*, S.A. McIlraith and K.Q. Weinberger, eds, AAAI Press, 2018, pp. 1811–1818.

[10] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. van den Berg, I. Titov and M. Welling, Modeling relational data with graph convolutional networks, in: *The Semantic Web Conf. (ESWC)*, Springer, 2018, pp. 593–607.

[11] T. Lacroix, N. Usunier and G. Obozinski, Canonical Tensor Decomposition for Knowledge Base Completion, in: *Int. Conf. Machine Learning*, J.G. Dy and A. Krause, eds, Proceedings of Machine Learning Research, Vol. 80, PMLR, 2018, pp. 2869–2878. http://proceedings.mlr.press/v80/lacroix18a.html.

[12] R. Zhang, J. Li, J. Mei and Y. Mao, Scalable Instance Reconstruction in Knowledge Bases via Relatedness Affiliated Embedding, in: *Conf. World Wide Web (WWW)*, 2018, pp. 1185–1194.

[13] N. Lao, T. Mitchell and W.W. Cohen, Random walk inference and learning in a large scale knowledge base, in: *Conf. Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2011, pp. 529–539.

[14] S. Muggleton, Inverse Entailment and Progol, *New Generation Computation* **13** (1995), 245–286.

[15] L. Galárraga, C. Teflioudi, K. Hose and F. Suchanek, Fast rule mining in ontological knowledge bases with AMIE+, *Int. J. Very Large Data Bases* **24**(6) (2015), 707–730.

[16] C. Meilicke, M. Fink, Y. Wang, D. Ruffinelli, R. Gemulla and H. Stuckenschmidt, Fine-Grained Evaluation of Rule- and Embedding-Based Systems for Knowledge Graph Completion, in: *The Semantic Web (ISWC)*, D. Vrandecic et al., eds, LNCS 11136, Springer, 2018, pp. 3–20. doi:10.1007/978-3-030-00671-6_1.

[17] C. Meilicke, M.W. Chekol, D. Ruffinelli and H. Stuckenschmidt, Anytime Bottom-Up Rule Learning for Knowledge Graph Completion, in: *Int. Joint Conf. Artificial Intelligence (IJCAI)*, S. Kraus, ed., ijcai.org, 2019, pp. 3137–3143. doi:10.24963/ijcai.2019/435.

[18] S. Ferré and P. Cellier, Graph-FCA: An extension of formal concept analysis to knowledge graphs, *Discrete Applied Mathematics* **273** (2019), 81–102. doi:https://doi.org/10.1016/j.dam.2019.03.003. http://www.sciencedirect.com/science/article/pii/S0166218X19301532.

[19] B. Ganter and R. Wille, *Formal Concept Analysis — Mathematical Foundations*, Springer, 1999.

[20] G.D. Plotkin, Automatic Methods of Inductive Inference, PhD thesis, Edinburgh University, 1971.

[21] P. Hitzler, M. Krötzsch and S. Rudolph, *Foundations of Semantic Web Technologies*, Chapman & Hall/CRC, 2009. ISBN 9781420090505.

[22] C.A. Hurtado, A. Poulovassilis and P.T. Wood, Query Relaxation in RDF, in: *Journal on Data Semantics X*, S. Spaccapietra, ed., LNCS 4900, Springer, 2008, pp. 31–61.

[23] P.G. Omran, K. Wang and Z. Wang, Scalable Rule Learning via Learning Representation, in: *Int. Joint Conf. Artificial Intelligence (IJCAI)*, J. Lang, ed., ijcai.org, 2018, pp. 2149–2155. doi:10.24963/ijcai.2018/297.

[24] T. Denœux, A k-nearest neighbor classification rule based on Dempster-Shafer theory, *IEEE Trans. Systems, Man, and Cybernetics* **25**(5) (1995), 804–813.

[25] W. May, Information Extraction and Integration with FLORID: The MONDIAL Case Study, Technical Report, 131, Universität Freiburg, Institut für Informatik, 1999, Available from http://dbis.informatik.uni-goettingen.de/Mondial.

[26] A. Hermann, S. Ferré and M. Ducassé, An Interactive Guidance Process Supporting Consistent Updates of RDFS Graphs, in: *Int. Conf. Knowledge Engineering and Knowledge Management (EKAW)*, A. ten Teije et al., eds, LNAI 7603, Springer, 2012, pp. 185–199.

[27] B. Yang, W. Yih, X. He, J. Gao and L. Deng, Embedding Entities and Relations for Learning and Inference in Knowledge Bases, in: *Int. Conf. Learning Representations (ICLR)*, Y. Bengio and Y. LeCun, eds, 2015. http://arxiv.org/abs/1412.6575.

[28] H. Liu, Y. Wu and Y. Yang, Analogical Inference for Multi-relational Embeddings, in: *Int. Conf. Machine Learning (ICML)*, D. Precup and Y.W. Teh, eds, Proceedings of Machine Learning Research, Vol. 70, PMLR, 2017, pp. 2168–2178. http://proceedings.mlr.press/v70/liu17d.html.

[29] A. García-Durán and M. Niepert, KBlrn: End-to-End Learning of Knowledge Base Representations with Latent, Relational, and Numerical Features, in: *Conf. Uncertainty in Artificial Intelligence (UAI)*, A. Globerson and R. Silva, eds, AUAI Press, 2018, pp. 372–381. http://auai.org/uai2018/proceedings/papers/149.pdf.

[30] W. Zhang, B. Paudel, W. Zhang, A. Bernstein and H. Chen, Interaction Embeddings for Prediction and Explanation in Knowledge Graphs, in: *Int. Conf. Web Search and Data Mining (WSDM)*, J.S. Culpepper, A. Moffat, P.N. Bennett and K. Lerman, eds, ACM, 2019, pp. 96–104. doi:10.1145/3289600.3291014.