Global Vertex Similarity for Large-Scale Knowledge Graphs

Marco Caballero and Aidan Hogan

DCC, Universidad de Chile & IMFD
{mcaball,ahogan}@dcc.uchile.cl

Abstract. We investigate global measures of vertex similarity for knowledge graphs. While vertex similarity has been explored in the context of directed, unlabelled graphs, measures based on recursive algorithms or learning frameworks can be costly to compute, assume labelled data, and/or provide poorly-interpretable results. Knowledge graphs further imply unique challenges for vertex similarity in terms of scale and diversity. We thus propose and explore global measures of vertex similarity for Knowledge Graphs that (i) are unsupervised, (ii) offer explanations of similarity results; (iii) take into consideration edge labels; and (iv) are robust in terms of redundant or interdependent information. Given that these measures can still be costly to compute precisely, we propose an approximation strategy that enables computation at scale. We compare our measures with a recursive measure (SimRank) for computing vertex similarity over subsets of Wikidata.

Keywords: Similarity, Knowledge Graphs, Wikidata

1 Introduction

Knowledge graphs have become popular for integrating and managing incomplete sources of data at large scale, where nodes represent entities of interest, and edges represent the relations between them. While knowledge graphs can be used for a variety of purposes and applications [20], often they are used to support *exploratory tasks*, where the user may not know precisely what they are looking for but will rather explore the data in order to gain actionable knowledge [26]. In this exploratory context, adopting a graph-based data model opens up opportunities to leverage graph algorithms for the purposes of discovering implicit connections or patterns in a knowledge graph [6].

An important class of graph algorithms is that of *vertex similarity* [17]. Given a graph $G = (V, E)$, which may be directed or undirected, the goal of vertex similarity is to identify pairs of nodes (i.e., vertices) that are similar. Global scenarios involve computing similarity scores for all pairs of nodes in $V \times V$ offline, possibly restricted by a certain criteria, such as a similarity threshold,

or a k-nearest neighbour constraint. Single-source scenarios take a given node $v \in V$ as input and find the nodes most similar to it at runtime. Unlike similarity based (e.g.) on metric spaces, vertex similarity relies on the graph's structure.

As in the case of graph analytics such as *centrality*, *clustering*, etc., there is no one single notion of *similarity* between vertices. Thus a variety of vertex similarity measures have been proposed, ranging from simple local measures that take into consideration a bounded neighbourhood to more complex recursive ones [17]. When considering different similarity measures, a trade-off arises. While recursive algorithms tend to give better results by considering information beyond the local neighbourhood, local similarity measures tend to be more efficient, more scalable and arguably more explainable.

Though vertex similarity has been used productively in various domains [17], knowledge graphs present three characteristic challenges for graph algorithms.

Regarding *diversity*, a knowledge graph can be seen as multiple directed graphs representing different relations, where for the purposes of vertex similarity, it is not immediately clear how the similarity computed for the graphs of individual relations can be combined into an overall measure. Different relations may also encode different topologies of graph. For example, a relation `:citizenship` would yield a bipartite directed graph, where person nodes have low out-degree and zero in-degree, while country nodes have very high in-degree and zero out-degree. On the other hand, a relation `:follows` in a social network would yield a cyclic directed graph consisting of user nodes, whose in-degree and out-degree average to the same value, but follow different distributions. Furthermore, a vertex similarity measure should be able to distinguish two directed labelled edges `:Unforgiven` $\xrightarrow{\texttt{:director}}$ `:CEastwood` and `:Unforgiven` $\xrightarrow{\texttt{:actor}}$ `:CEastwood` as encoding different information about the movie and the person.

Regarding *redundancy*, knowledge graphs may provide redundant (i.e., entailed) or near-redundant (i.e., dependent) data. For example, a knowledge graph may state that `:Jill` $\xrightarrow{\texttt{:mother}}$ `:Anne`, and `:Anne` $\xrightarrow{\texttt{:child}}$ `:Jill`, where though the latter edge is intuitively redundant (i.e., it could be entailed from the former triple with the appropriate ontology), it can affect the results of vertex similarity by changing the structure of the graph. As an example of near-redundant data, a knowledge graph may state `:Honduras` $\xrightarrow{\texttt{:region}}$ `:CentralAmerica` and `:Honduras` $\xrightarrow{\texttt{:language}}$ `:Spanish`, where, while neither triple necessarily follows from the other, there exists a dependency between both edges. Such redundancy is more prevalent in knowledge graphs due to the fact that multiple interdependent relations can be encoded; in fact, encoding such redundancy is a feature of knowledge graphs supported by ontologies. When applying vertex similarity over knowledge graphs, we must then be cautious not to overestimate similarity by considering dependent edges as independent events.

Regarding *scale*, global vertex similarity, when applied to $V \times V$, may generate a similarity relation of quadratic size ($O(|V|^2)$), which will be infeasible to materialise at scale. Even where only the results for one vertex is required, or where thresholds are applied, recursive algorithms may require knowledge of pairwise similarities in order to compute any similarities.

It is in this context that we investigate four measures of vertex similarity for knowledge graphs, two of which are taken from the literature and adapted by us for the knowledge graph setting, and two of which are novel. We will motivate and define these measures, and discuss how they can be optimised and approximated. We will apply these similarity measures to sub-graphs of Wikidata to compare their performance and to see how the results they provide correlate with each other. We also evaluate the measures against two external ground truths: one for similar movies, and another for similar music albums.

2 Background

Vertex similarity. Measures of vertex similarity are mostly proposed in the context of undirected and directed graphs [17]. For this discussion we will assume a directed graph $G = (V, E)$. A vertex similarity measure $\sigma : V \times V \to \mathbb{R}$ is a measure that associates pairs of vertices (v_1, v_2) with a real-valued similarity score. We will assume, without loss of generality, that a higher similarity score indicates that the pair of nodes is more similar; i.e., that $\sigma(v_1, v_2) > \sigma(v_1, v_3)$ indicates that v_1 is more similar to v_2 than it is to v_3.

Vertex similarity measures may be *local*, or may be *non-local*. We say that a particular measure σ is local if the value of $\sigma(v_1, v_2)$ depends on a bounded neighbourhood surrounding v_1 and v_2; different notions of neighbourhood give rise to different levels of locality, such as considering only the nodes themselves, considering the graph induced by the nodes themselves and their direct neighbours in G, or their neighbours' neighbours, and so forth. Otherwise – if $\sigma(v_1, v_2)$ depends on an unbounded neighbourhood – we say that σ is *non-local*.

We also distinguish between *recursive* and *non-recursive* measures. We call a measure σ recursive if the value of $\sigma(v_1, v_2)$ depends on the value of σ for pairs of nodes other than v_1 and v_2. Otherwise we call σ non-recursive. Non-recursive measures tend to be local, while recursive measures tend to be non-local (though recursive measures applied up to a bounded number of iterations are local).

A number of local, non-recursive measures are introduced by Leicht et al. [17], where *nodes are considered similar if they have a similar neighbourhood* (aka. *structural equivalence*). For simplicity, we will consider the neighbourhood of a node v in $G = (V, E)$ generically as $E(v) = \{n \mid (v, n) \in E\}$; we could also, for example, consider neighbours via incoming links in directed graphs. In terms of concrete measures for $\sigma(v_1, v_2)$, we can consider simply a *neighbourhood count* ($|E(v_1) \cap E(v_2)|$), or *neighbourhood Jaccard similarity* ($\frac{|E(v_1) \cap E(v_2)|}{|E(v_1) \cup E(v_2)|}$), or *neighbourhood cosine similarity* ($\frac{|E(v_1) \cap E(v_2)|}{\sqrt{|E(v_1)| \cdot |E(v_2)|}}$)[17].

On the other hand, recursive measures of similarity are generally based on the principle that *nodes are considered similar if they have many similar neighbours*; in this case, we consider the similarity of neighbours. Many recursive similarity measures are based on recursive centrality measures, following loosely the intuition that similar nodes should be well-connected – or easily reachable – from each other. Jeh and Widom [15] propose a vertex similarity measure based

on PageRank centrality, Blondel et al. [5] a vertex similarity measure based on HITS (hubs and authorities) centrality, and Leicht et al. [17] a vertex similarity measure based on Katz centrality. All such measures are recursive, and have proven to give good results in practice. Of these measures, arguably the one that has been most influential is that of Jeh and Widom [15], called *SimRank*, which has led to various follow-up works on accuracy estimations [18], probabilistic graphs [9], similarity joins [28], etc., for the algorithm.

Some more recent approaches define vertex similarity based on latent features of the graph using machine learning techniques. In particular, there are strong connections between the world of *graph embeddings* – whose goal is to develop meaningful numeric representations of the elements of a graph – and vertex similarity, where, on the one hand, embedding techniques can be used to learn similar numeric representations for nodes [23], while on the other hand, off-the-shelf similarity measures can be used to guide the embeddings that are learnt [25]. Such measures can again be local [23] or non-local [25].

Similarity in knowledge graphs. Similarity over knowledge graphs has been used for general applications, such as clustering [1], entity comparison [21], entity linking [14], link discovery [19], ontology matching [10], query relaxation [12,16], semantic relatedness [22], as well as domain-specific use-cases, such as academic search [7], crime reports [12], geographic databases [3], multimedia retrieval [11], protein search [2,4], etc. We identify two forms of similarity used in these works.

Metric similarity involves the use of a (normalised) distance metric δ (Euclidean, Manhattan, Levenshtein etc.), where similarity will typically be defined in terms of the distance as $\sigma(v_1, v_2) = 1 - \delta(v_1, v_2)$. The use of δ thus induces/-supposes a metric space to which nodes must be mapped. Such forms of similarity have been used for geographic databases [3], link prediction [24], multimedia retrieval [11], protein similarity search [2,4] and query relaxation [16]. However, such metric spaces are either domain specific [3,2,4,11], or need to be specified manually [24,16]; they are not inherent to graphs in general.

Vertex similarity, on the other hand, is a measure intrinsic to graphs. Variations of the aforementioned algorithms for vertex similarity have been adapted for use in clustering [1], entity comparison [21], entity linking [14], query relaxation over crime reports [12], semantic relatedness [22], etc.

Novelty. We investigate vertex similarity measures for knowledge graphs, including variants of two existing measures, and two novel measures. Our novel measures are local and non-recursive. We show one to be competitive with SimRank (a representative non-local, recursive measure) in terms of predicting a ground truth, while being more scalable, more efficient, and more explainable.

3 Vertex Similarity Measures

We now define four vertex similarity measures: neighbourhood count (σ_{NC}), neighbourhood selectivity (σ_{NS}), neighbourhood rarity (σ_{NR}) and SimRank (σ_{SR}).

Of these measures, σ_{NS} and σ_{NR} are – to the best of our knowledge – novel (though σ_{NS} is inspired by our prior proposal called *concurrence* [13].). Given a directed graph $G = (V, E)$, we recall the notation $E(v)$ to denote the neighbouring nodes of v in G. We assume that for our similarity measures, $\sigma(v) \in [0, \infty)$, where $\sigma(v_1, v_2) > \sigma(v_1, v_3)$ implies that v_1 is more similar to v_2 than to v_3.

Neighbourhood count Our first vertex similarity measure is the *neighbourhood count* as was defined previously: $\sigma_{\text{NC}}(v_1, v_2) = |E(v_1) \cap E(v_2)|$. The measure simply counts the number of neighbours that v_1 and v_2 have in common.

Neighbourhood selectivity Neighbourhood count treats all neighbours as being equal. However, neighbours with a high (in)degree, like `:UnitedStates`, will be shared by many pairs of nodes; this would still be counted the same as a node with a low (in)degree, such as `:VaticanCity`. Equivalently, we can say that the *probability* that a particular node v has `:UnitedStates` in its neighbourhood is much higher than it having `:VaticanCity` in its neighbourhood. Specifically, for a node n, let $\deg(n) = |\{v \in V \mid n \in E(v)\}|$ denote the degree of n in terms of the number of neighbourhoods it appears in. We denote the probability of n appearing in the neighbourhood of a random node as $P(n) = \frac{\deg(n)}{|V|}$.

Let $\{n_1, \ldots, n_k\} = E(v_1) \cap E(v_2)$ denote the set of neighbours that v_1 and v_2 have in common. Instead of simply counting the neighbours that two nodes have in common (k), we propose *neighbourhood selectivity*, which measures the vertex similarity between v_1 and v_2 as the probability of a random node v not having all of the individual neighbours that v_1 and v_2 share in common:

$$\sigma_{\text{NS}}(v_1, v_2) = P(\neg(n_1 \cap \ldots \cap n_k)) = 1 - \prod_{i=1}^{k} P(n_i) = 1 - \frac{1}{|V|^k} \prod_{i=1}^{k} \deg(n_i)$$

This assumes that having individual neighbours are independent events. The idea behind this measure is that the more neighbours a pair of nodes share, and the rarer those neighbours, then the lower the probability of a random node having all of those neighbours, and thus the higher the similarity measure.

Neighbourhood rarity The previous measure assumes that having different neighbours corresponds to independent events. However, as argued in the introduction, this is often not the case. For example, in a bipartite graph linking movies to actors, having `:SLaurel` in the neighbourhood, and having `:OHardy` in the neighbourhood, should not be considered as being independent events, as these were a famous duo of actors who often starred in movies together. In the presence of redundant or interdependent data – which, as we previously argued is common in the case of knowledge graphs – the previous measure would end up overestimating the similarity of node pairs per its own design criteria.

Rather than combining the probabilities of (supposedly) independent events, we propose *neighbourhood rarity*, which measures the vertex similarity of v_1 and

v_2 as the ratio of other nodes not having (at least) all of the neighbours that v_1 and v_2 share in common. First, for a given set of nodes N, we generalise the definition of degree to a set of nodes: let $\deg(N) = |\{v \in V \mid N \subseteq E(v)\}|$. Defending our slight abuse of notation, note that $\deg(n) = \deg(\{n\})$. We can now define the *neighbourhood rarity* measure as simply:

$$\sigma_{\text{NR}}(v_1, v_2) = 1 - \frac{\deg(E(v_1) \cap E(v_2)) - 2}{|V| - 2}$$

We subtract 2 to exclude v_1 and v_2 from the count. As opposed to the previous measure, if we now have interdependent neighbours in $E(v_1) \cap E(v_2)$, then neighbourhood rarity will take this into account as many other nodes with likewise have those interdependent neighbours together, reducing the above similarity.

SimRank The measures we have presented thus far are non-recursive. We now introduce a representative of a recursive algorithm, namely SimRank [15]. First we need some additional notation. Let $E^-(v) = \{n \mid (n, v) \in E\}$ denote the nodes with incoming edges to v. The SimRank measure σ_{SR} is then defined recursively as follows: if $v_1 = v_2$, then $\sigma_{\text{SR}}(v_1, v_2) = 1$; else if $|E^-(v_1)| = 0$ or $|E^-(v_2)| = 0$, then $\sigma_{\text{SR}}(v_1, v_2) = 0$; otherwise:

$$\sigma_{\text{SR}}(v_1, v_2) = \frac{\gamma}{|E^-(v_1)| \cdot |E^-(v_2)|} \sum_{i=1}^{|E^-(v_1)|} \sum_{j=1}^{|E^-(v_2)|} \sigma_{\text{SR}}(E_i^-(v_1), E_j^-(v_2))$$

where $E_i^-(v)$ and $E_j^-(v)$ denote the i^{th} and j^{th} elements of $E^-(v)$, respectively; and where $\gamma \in [0, 1]$ is a hyperparameter called the *decay constant* (originally $\gamma = 0.8$). Intuitively, the similarity of v_1 and v_2 (where $v_1 \neq v_2$) depends on the sum of the pairwise similarity of their incoming neighbours, normalised by the score they would have if all incoming neighbours had a pairwise similarity of 1 ($|E^-(v_1)| \cdot |E^-(v_2)|$) multiplied by a decay parameter. This measure can be computed iteratively by setting $\sigma_{\text{SR}}(v_1, v_2) = 1$ if $v_1 = v_2$, or $\sigma_{\text{SR}}(v_1, v_2) = 0$ otherwise, and then computing the above equation iteratively.

4 Knowledge Graph Vertex Similarity

We assume a knowledge graph to be a directed, labelled graph $K = (V, L, E)$, where V is a set of nodes, L is a set of edge labels, and $E \subseteq V \times L \times V$ is a set of directed, labelled edges. Thus far our measures have been defined for directed graphs only. This raises the question: how should the proposed measures be applied to knowledge graphs? The first alternative would be to drop the labels from the edges (and remove duplicate edges between the same pair of nodes with different labels) in order to return to a directed graph. For a directed labelled edge $s \xrightarrow{p} o$, this would involve projecting a directed edge $s \to o$. However, as we argued in the case of `:Unforgiven` $\xrightarrow{\texttt{:director}}$ `:CEastwood`

and :Unforgiven $\xrightarrow{\texttt{:actor}}$:CEastwood, edge labels carry important information for vertex similarity measures. Taking a more extreme example, consider :CEastwood $\xrightarrow{\texttt{:citizen}}$:UnitedStates, :DTrump $\xrightarrow{\texttt{:president}}$:UnitedStates, and :BObama $\xrightarrow{\texttt{:president}}$:UnitedStates. Intuitively the similarity of :DTrump and :BObama should benefit more from being presidents of the :UnitedStates.

A second alternative that keeps edge information is to encode a directed labelled edge $s \xrightarrow{p} o$ as two directed edges of the form $(p, s) \rightarrow o$ and $s \rightarrow (p, o)$. For example, the directed labelled edge :Unforgiven $\xrightarrow{\texttt{:director}}$:CEastwood would become two directed edges: :Unforgiven $\rightarrow$ (:director,:CEastwood) and (:director,:Unforgiven) $\rightarrow$:CEastwood. In this case, the resulting directed graph is lossless, meaning that we can from the directed graph reconstruct the full directed edge-labelled graph. While this might seem unusual, particularly in the case of the local neighbourhood measures, this is quite a practical alternative, where elements of the neighbourhood now become pairs of edge labels and nodes, which in turn allows us to distinguish the probabilities associated with, for example, (:president, :UnitedStates) and (:citizen, :UnitedStates).

5 Implementation

We now describe our algorithms for implementing our measures, their complexity, and approximations that we apply in the case of the local neighbourhood measures to enable increased scalability and efficiency.

Neighbourhood measures We implement the local neighbourhood measures on the distributed Apache Spark framework [27]. In comparison with, for example, the Hadoop framework based on MapReduce [8], Spark abstracts storage not only on the hard disk, but also in-memory, which generally allows for more efficient computation across a cluster of machines when sufficient main memory is available. Our implementation assumes an input graph in RDF format.

Implementation. We implement the similarity measures in the natural way, with the exception of the neighbourhood rarity measure, which we compute in a slightly indirect but more efficient way. For reasons of space, we provide a high-level sketch of the details of the algorithms used:

- *Neighbourhood count*: we group the graph by neighbours n, collecting together all nodes $\{v_1, \ldots, v_k\}$ such that $n \in E(v_i)$ for $1 \leq i \leq k$; then for each such group we write out all asymmetric, irreflexive pairs of nodes (v_a, v_b) for $1 \leq a < b \leq k$ sharing that neighbour. We then count occurrences of the resulting pairs, producing the final neighbourhood count.
- *Neighbourhood selectivity*: we apply the same process as before, but instead of writing pairs of the form (v_a, v_b), we output triples of the form (v_a, v_b, k); we can then group these triples by (v_a, v_b) and compute the neighbourhood selectivity over the bag of values $\{\!\{k_1, \ldots, k_m\}\!\}$ for each pair.

– *Neighbourhood rarity*: we apply a neighbourhood count generating pairs of the form $(v_a, v_b, \sigma_{\text{NC}}(v_a, v_b))$; we apply a similar process to count how many neighbours a triple of nodes have in common $(v_a, v_b, v_c, \sigma'_{\text{NC}}(v_a, v_b, v_c))$, where $1 \leq a < b \leq k$ and $a \neq c \neq b$. We can then group both sets of data together by (v_a, v_b), giving us its neighbour count $\sigma_{\text{NC}}(v_a, v_b)$, and a list of pairs $\{(v_{c1}, \sigma'_{\text{NC}}(v_a, v_b, v_{c1})), \ldots, (v_{cm}, \sigma'_{\text{NC}}(v_a, v_b, v_{cm}))\}$ denoting other nodes sharing at least one neighbour with v_a and v_b, along with how many neighbours the three nodes share. Finally we can count how many nodes v_{ci} there are in the set such that $\sigma'_{\text{NC}}(v_a, v_b, v_{ci}) = \sigma_{\text{NC}}(v_a, v_b)$.

Complexity. Letting $\mathsf{E} = |E|$ and $\mathsf{V} = |V|$ for readability, then the worst-case runtime complexities are as follows. Neighbourhood count is $\Theta(\mathsf{V}^3 \log \mathsf{V})$. This worst case is tight, as seen for the V-clique where we first sort the edges of the graph (in time $O(\mathsf{E} \log \mathsf{E})$), and then write $O(\mathsf{V}^2)$ pairs of nodes a total of $O(\mathsf{V})$ times, giving a bag of pairs of nodes of size $O(\mathsf{V}^3)$, which we then sort in time $O(\mathsf{V}^3 \log \mathsf{V}^3)$. This gives us $\Theta(\mathsf{E} \log \mathsf{E} + \mathsf{V}^3 \log \mathsf{V}^3)$, which can be simplified to $\Theta(\mathsf{V}^3 \log \mathsf{V})$ observing that $\mathsf{E} \leq \mathsf{V}^2$ and that $\mathsf{V}^3 \log \mathsf{V}^3 = 3\mathsf{V}^3 \log \mathsf{V}$. Neighbourhood selectivity is likewise $O(\mathsf{V}^3 \log \mathsf{V})$ as it follows a similar procedure. Neighbourhood rarity is rather $\Theta(\mathsf{V}^4 \log \mathsf{V})$; for the V-clique we now write $O(\mathsf{V}^3)$ triples of nodes a total of $O(\mathsf{V})$ times, giving rise to a bag of triples of nodes of size $O(\mathsf{V}^4)$, which we sort in $\Theta(\mathsf{V}^4 \log \mathsf{V})$. However, noting that we produce pairs and/or triples of nodes only for a particular neighbour, we can tighten these bounds if we rather define $\mathsf{D} = \max\{\deg(n) \mid n \in V\}$, where often $\mathsf{D} \ll \mathsf{V}$. This then gives worst-case complexities of $\Theta(\mathsf{E} \log \mathsf{E} + \mathsf{V}\mathsf{D}^2 \log \mathsf{V})$ for neighbourhood count and selectivity, and $\Theta(\mathsf{E} \log \mathsf{E} + \mathsf{V}\mathsf{D}^3 \log \mathsf{V})$ for neighbourhood rarity.

κ-approximation. Still, $\max\{\deg(n) \mid n \in V\}$ can be a prohibitively large value to raise by 3 or 4, particularly for large-scale knowledge graphs. However, this analysis does suggest a flexible mechanism for approximation: we can define a threshold κ such that we ignore neighbours n with $\deg(n) > \kappa$. Intuitively, this means that we will simply ignore high-degree nodes from the similarity computation. While this may affect the final scores produced, we note that in the case of *neighbourhood selectivity* and *neighbourhood rarity*, high-degree neighbours have comparatively less influence on the measure. We will later experimentally check the effect of varying κ on the associated similarity scores. With κ-approximation, the complexity becomes simply $O(\mathsf{E} \log \mathsf{E})$ in all three cases as the quadratic/cubic parameter $\mathsf{D} = \max\{\deg(n) \mid n \in V\} \leq \kappa$ is now bounded. For this reason we believe that these measures are promising for global vertex similarity on large-scale knowledge graphs, with κ allowing to trade precision for scale.

Neighbourhood rarity-s. Finally, we can solve the issue of frequent ties in neighbourhood rarity with negligible practical cost by interleaving the computation of neighbourhood selectivity into the process, where we then define a novel hybrid measure of similarity that we call *neighbourhood rarity-s* as

$$\sigma_{\text{NRS}}(v_1, v_2) = |V| \cdot \sigma_{\text{NR}}(v_1, v_2) + \sigma_{\text{NS}}(v_1, v_2)$$

Table 1. Statistics of the four Wikidata sub-graphs

Sub-graph	Subjects	Predicates	Objects	Triples
Universities	842	787	83,583	94,599
Albums	653	936	165,456	189,653
Countries	179	905	196,318	216,774
Movies	765	896	256,456	365,123

such that the whole part of the similarity measure is given by σ_{NR} (*neighbourhood rarity*), while the decimal part is given by σ_{NS} (*neighbourhood selectivity*); intuitively, σ_{NRS} first ranks similar pairs by σ_{NR}, using σ_{NS} to break ties.

SimRank We use an off-the-shelf implementation of SimRank. The space complexity of SimRank is $O(\mathsf{V}^2)$ for $\mathsf{V} = |V|$ as it needs to store all pairwise similarities (aside from the cases that are trivially 0 or 1 throughout). The worst case time complexity of an iteration of SimRank is $O(\mathsf{V}^4)$ where several such iterations of SimRank are required to approximate the measure. Though optimisations of SimRank have been proposed in the literature, to the best of our knowledge they consider single-source rather than global similarity.

6 Experiments

In this initial work, we perform experiments that consider small-to-medium sized sub-graphs of Wikidata, where each focuses on a different entity type. We show statistics relating to these sub-graphs in Table 1. With respect to our experiments, we wish to address three initial research questions: (1) How do the runtimes and results of local neighbourhood measures compare? (2) How does the value of κ affect runtimes and similarity results? (3) How do the results for the local neighbourhood measures and SimRank compare for ground-truth similarity datasets? Noting that we failed to compute SimRank for any of these subgraphs, we postpone discussion of SimRank results until the end of the section, where we will sample smaller subgraphs for comparison.

6.1 Runtime performance

We first look at the runtime performance of the different measures and for different values of κ. In Table 2, we show the results for the smallest and largest subgraphs. As general observations, we see that, as predicted by the complexity analysis, there is little difference between the runtimes of σ_{NC} and σ_{NS} (being quadratic with respect to κ), while σ_{NRS} is generally slower to execute (being cubic with respect to κ). Furthermore, as κ increases, the runtimes of σ_{NC} and σ_{NS} grow more slowly than in the case of σ_{NRS}. In the case of Movies (and Albums), Spark failed to compute σ_{NRS} with $\kappa = 512$, where a single neighbour n with $\deg(n) = 512$ produces $\frac{(512-2)}{2}(512^2 - 512) = 66,716,160$ quadruples.

Table 2. Runtimes (s) for UNIVERSITIES (left) and MOVIES (right)

κ	σ_{NC}	σ_{NS}	σ_{NRS}
4	145	159	413
8	149	159	411
16	149	160	412
32	152	160	420
64	147	159	431
128	145	161	491
256	147	160	697
512	149	162	710

κ	σ_{NC}	σ_{NS}	σ_{NRS}
4	162	183	433
8	164	200	503
16	175	237	732
32	200	262	1,989
64	283	301	7,303
128	417	367	11,514
256	547	447	20,736
512	1,162	737	—

Table 3. Kendall's-τ correlations for varying κ against results for highest κ on datasets for COUNTRIES, UNIVERSITIES, MOVIES and ALBUMS

	σ_{NC}				σ_{NS}				σ_{NRS}			
κ	C.	U.	M.	A.	C.	U.	M.	A.	C.	U.	M.	A.
4	0.25	0.20	0.25	0.22	0.23	0.29	0.62	0.59	-0.27	0.07	0.34	0.33
8	0.35	0.26	0.35	0.30	0.33	0.34	0.63	0.60	0.13	0.59	0.76	0.60
16	0.58	0.53	0.48	0.52	0.44	0.40	0.66	0.75	0.06	0.59	0.78	0.66
32	0.78	0.72	0.56	0.58	0.79	0.56	0.73	0.79	0.16	0.57	0.83	0.68
64	0.89	0.85	0.75	0.70	0.91	0.76	0.82	0.84	0.56	0.74	0.89	0.92
128	0.95	0.90	0.85	0.83	0.99	0.86	0.88	0.90	0.99	0.86	0.88	0.90
256	1.00	0.99	0.93	0.95	1.00	0.99	0.93	0.95	0.95	0.99	1.00	1.00
512	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	—	—

6.2 Effect of κ on the similarity results

The aforementioned results show the performance benefit of lowering κ, but what effect does this have on the similarity scores? To address this question, we compared the similarity results for varying values of κ against the most accurate approximation ($\kappa = 512$ in all cases, except σ_{NRS} in the case of ALBUMS and MOVIES which did not run, where we compare with $\kappa = 256$). In general, we see that as κ decreases exponentially, the correlations remain strong until $\kappa = 128$, reducing more dramatically at around $\kappa = 32$. In general, we recommend keeping κ as high as feasible to ensure that the results are as faithful to the original measures as possible, but where necessary, lowering κ can still lead to reasonable approximations while tending towards $O(\mathsf{E} \log \mathsf{E})$ worst-case runtime complexity.

6.3 Ground-truth comparisons, full-scale

While the previous results compare the changes in the local neighbourhood results for varying κ, they do not establish how well the measures correspond to a general notion of "similarity". Unfortunately, many of the ground truths considered in the literature consider directed graphs rather than knowledge graphs, as is our setting. For this reason, we develop two novel ground-truths based on

Table 4. Kendall's-τ distributions for ground truths, full-scale

BestSimilar					
Measure	*Min.*	*L. Quar.*	*Median*	*U. Quar.*	*Max*
σ_{NC}	0.3577	0.5018	0.6224	0.7621	0.8558
σ_{NS}	0.2998	0.4347	0.5485	0.6389	0.7982
σ_{NRS}	0.7332	0.8022	0.8630	0.9320	0.9827
Last.fm					
Measure	*Min.*	*L. Quar.*	*Median*	*U. Quar.*	*Max*
σ_{NC}	0.1777	0.3402	0.4601	0.6168	0.7549
σ_{NS}	0.0982	0.2382	0.3836	0.5304	0.6758
σ_{NRS}	0.4902	0.5483	0.6538	0.7319	0.8377

similarity relations scraped from two external websites: BestSimilar for MOVIES and Last.fm for ALBUMS. Both of these websites rely on user inputs, where BestSimilar allows to upvote or downvote similarity pairs, while Last.fm likewise has a vast amount of user data to leverage for similarity. For this reason, we assume that both sites offer reliable ground truths. The question then is: can we replicate these similarity relations based on (costly) user input with our unsupervised vertex similarity measures over the corresponding subgraphs of Wikidata?

On BestSimilar, each movie is associated with a ranked list of 10 similar movies, where we extract 100 such lists. On Last.fm, each album is associated with a ranked list of 20 similar albums, where we again extract 100 lists. We then linked the associated movies and albums with their Wikidata identifiers.We measure the Kendall's-τ correlation between our three local neighbourhood measures and each of the 100 ground truths, using the highest available values for κ. Given that our subgraphs may mention nodes not appearing in the ground truth, and vice versa, we only consider the ordering of elements appearing in both lists when measuring the correlation. In Table 4, we show the distribution of results, where we see that σ_{NRS} provides a clear improvement in both datasets with respect to the other measures, which may justify its cost. We attribute this improvement to the way in which it is more robust to redundant information.

6.4 Ground-truth comparisons, small-scale

Finally, in order to compare with SimRank (σ_{SR}), we reduced the scale of the MOVIES and ALBUMS sub-graph until we could successfully compute the SimRank measure. The new datasets contained 275,123 and 169,963 triples respectively. We show the results in Table 5 where we see that σ_{SR} performs best out of the four measures. This is perhaps to be expected as σ_{SR} is a recursive measure that can incorporate information from the full graph, not just the locality of the two nodes. On the other hand, σ_{NRS} remains quite competitive with σ_{SR}.

Table 5. Kendall's-τ distributions for ground truths, small-scale

BestSimilar

Measure	*Min.*	*L. Quar.*	*Median*	*U. Quar.*	*Max*
σ_{NC}	0.0013	0.0743	0.2137	0.3309	0.4689
σ_{NS}	0.0026	0.0789	0.1416	0.2039	0.3275
σ_{NRS}	0.2291	0.3290	0.4300	0.4990	0.5789
σ_{SR}	0.2739	0.3738	0.4749	0.5438	0.6238

Last.fm

Measure	*Min.*	*L. Quar.*	*Median*	*U. Quar.*	*Max*
σ_{NC}	0.0028	0.0869	0.1482	0.3132	0.4087
σ_{NS}	0.0008	0.0685	0.1441	0.2097	0.2883
σ_{NRS}	0.1763	0.2620	0.3274	0.4330	0.5234
σ_{SR}	0.2317	0.3174	0.3829	0.4884	0.5789

7 Conclusions

We have investigated measures for vertex similarity in the context of knowledge graphs. We proposed two new measures that only consider the local neighbourhoods of the pairs of nodes being compared. One of these measures – neighbourhood rarity-s – shows promising results, being outperformed by non-local SimRank in terms of a comparison with two ground truth datasets, but by a narrow margin. On the other hand, this neighbourhood rarity-s measure, in combination with degree-based (κ) approximation, can be evaluated at larger scale and in a more efficient way than the SimRank alternative. Furthermore, it can avoid "double counting" of redundant information by considering a common neighbourhood as a single event, rather than assuming that it consists of multiple independent events (i.e., individual neighbours). The locality of these measures also have further advantages that we have not explored. For example, unlike SimRank, given a pair of nodes v_1 and v_2, we could in principle write a SPARQL 1.1 query to compute the (local) similarity of the node pair.

More generally, we believe that in knowledge graphs, with the addition of edge labels, the local neighbours of nodes tend to be rich in information: much richer than directed graphs that only consider one relation type. For this reason, we believe that simpler local measures of vertex similarity can compete with more complex non-local measures in knowledge graphs such as Wikidata. As part of future work, we plan to run scale-up experiments to understand what values of κ make it possible to process all of Wikidata with each algorithm, to develop a user interface, and to perform user studies.

Online material. Please see `http://aidanhogan.com/wikidata-sim`.

Acknowledgements. This work was funded by Fondecyt Grant No. 1181896 and ANID Millennium Science Initiative Program ICN17_002.

References

1. Alqadah, F., Bhatnagar, R.: A game theoretic framework for heterogenous information network clustering. In: Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD). pp. 795–804 (2011)
2. Apweiler, R., Bairoch, A., Wu, C.H., Barker, W.C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M.J., Natale, D.A., O'Donovan, C., Redaschi, N., Yeh, L.L.: UniProt: the Universal Protein knowledgebase. Nucleic Acids Research **32**, D115–D119 (01 2004)
3. Battle, R., Kolas, D.: Enabling the geospatial Semantic Web with Parliament and GeoSPARQL. Semantic Web **3**(4), 355–370 (2012)
4. Belleau, F., Nolin, M.A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: towards a mashup to build bioinformatics knowledge systems. Journal of Biomedical Informatics **41**(5), 706–716 (2008)
5. Blondel, V.D., Gajardo, A., Heymans, M., Senellart, P., Dooren, P.V.: A Measure of Similarity between Graph Vertices: Applications to Synonym Extraction and Web Searching. SIAM Review **46**(4), 647–666 (2004)
6. Bonatti, P.A., Decker, S., Polleres, A., Presutti, V.: Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web. Dagstuhl Reports **8**(9), 29–111 (2018)
7. Chiang, M., Liou, J., Wang, J., Peng, W., Shan, M.: Exploring heterogeneous information networks and random walk with restart for academic search. Knowl. Inf. Syst. **36**(1), 59–82 (2013)
8. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2008)
9. Du, L., Li, C., Chen, H., Tan, L., Zhang, Y.: Probabilistic SimRank computation over uncertain graphs. Inf. Sci. **295**, 521–535 (2015)
10. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer (2013)
11. Ferrada, S., Bustos, B., Hogan, A.: IMGpedia: a linked dataset with content-based analysis of Wikimedia images. In: International Semantic Web Conference (ISWC). pp. 84–93. Springer (2017)
12. Hogan, A., Mellotte, M., Powell, G., Stampouli, D.: Towards Fuzzy Query-Relaxation for RDF. In: Extended Semantic Web Conference (ESWC). pp. 687–702 (2012)
13. Hogan, A., Zimmermann, A., Umbrich, J., Polleres, A., Decker, S.: Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora. J. Web Semant. **10**, 76–110 (2012)
14. Hulpus, I., Prangnawarat, N., Hayes, C.: Path-Based Semantic Relatedness on Linked Data and Its Use to Word and Entity Disambiguation. In: International Semantic Web Conference (ISWC). pp. 442–457 (2015)
15. Jeh, G., Widom, J.: SimRank: a measure of structural-context similarity. In: SIGKDD International Conference on Knowledge Discovery and Data Mining (SIDKDD). pp. 538–543 (2002)
16. Kiefer, C., Bernstein, A., Stocker, M.: The fundamentals of iSPARQL: a virtual triple approach for similarity-based Semantic Web tasks. In: International Semantic Web Conference (ISWC), pp. 295–309. Springer (2007)
17. Leicht, E.A., Holme, P., Newman, M.E.: Vertex similarity in networks. Physical Review E **73**(2), 026120 (2006)
18. Lizorkin, D., Velikhov, P., Grinev, M.N., Turdakov, D.: Accuracy estimate and optimization techniques for SimRank computation. VLDB J. **19**(1), 45–66 (2010)

19. Ngomo, A.N., Auer, S.: LIMES – A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. In: International Joint Conference on Artificial Intelligence (IJCAI). pp. 2312–2317 (2011)
20. Noy, N.F., Gao, Y., Jain, A., Narayanan, A., Patterson, A., Taylor, J.: Industry-scale Knowledge Graphs: Lessons and Challenges. ACM Queue **17**(2), 20 (2019)
21. Petrova, A., Sherkhonov, E., Grau, B.C., Horrocks, I.: Entity Comparison in RDF Graphs. In: International Semantic Web Conference (ISWC). pp. 526–541. Springer (2017)
22. Pirrò, G., Euzenat, J.: A Feature and Information Theoretic Framework for Semantic Similarity and Relatedness. In: International Semantic Web Conference (ISWC). pp. 615–630 (2010)
23. Ribeiro, L.F.R., Saverese, P.H.P., Figueiredo, D.R.: *struc2vec*: Learning node representations from structural identity. In: Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD). pp. 385–394 (2017)
24. Sherif, M.A., Ngomo, A.N.: A systematic survey of point set distance measures for link discovery. Semantic Web **9**(5), 589–604 (2018)
25. Tsitsulin, A., Mottin, D., Karras, P., Müller, E.: VERSE: Versatile Graph Embeddings from Similarity Measures. In: World Wide Web Conference (WWW). pp. 539–548 (2018)
26. Yahya, M., Berberich, K., Ramanath, M., Weikum, G.: Exploratory querying of extended knowledge graphs. Proc. VLDB Endow. **9**(13), 1521–1524 (2016)
27. Zaharia, M., Xin, R.S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M.J., Ghodsi, A., Gonzalez, J., Shenker, S., Stoica, I.: Apache Spark: a unified engine for big data processing. Commun. ACM **59**(11), 56–65 (2016)
28. Zheng, W., Zou, L., Chen, L., Zhao, D.: Efficient SimRank-Based Similarity Join. ACM Trans. Database Syst. **42**(3), 16:1–16:37 (2017)