A Novel Metric for Information Retrieval in Semantic Networks

Joshua L. Moore[1,2], Florian Steinke[1], and Volker Tresp[1]

[1] Siemens AG, Corporate Technology, München, Germany
[2] Cornell University, Ithaca, NY
jlmo@cs.cornell.edu,{Florian.Steinke,Volker.Tresp}@siemens.com

Abstract. We propose a novel graph metric for semantic entity-relationship networks. The metric is used for solving two tasks. First, given a semantic entity-relationship graph, such as for example DBpedia, we find relevant neighbors for a given query node. This could be useful for retrieving information relating to a specific entity. Second, we search for paths between two given nodes to discover interesting links. As an example, this can be helpful to analyze the various relationships between Albert Einstein and Niels Bohr. Compared to using the default step metric our approach yields more specific and informative results, as we demonstrate using two semantic web datasets. The proposed metric is defined via paths that maximize the log-likelihood of a restricted round trip and can intuitively be interpreted in terms of random walks on graphs. Our distance metric is also related to the commute distance, which is highly plausible for the described tasks but prohibitively expensive to compute. Our metric can be calculated efficiently using standard graph algorithms, rendering the approach feasible for the very large graphs of the semantic web's linked data.

Keywords: Entity-relationship graph, information retrieval, random walk, commute distance, graph metric, path finding.

1 Introduction

Large entity-relationship (ER) graphs have recently become available on the semantic web. Sources like DBpedia (Auer et al., 2008), YAGO (Suchanek et al., 2007), OpenCyc[1] or Linked Life Data[2] (Momtchev et al., 2009) now encode useful information on a large scale. Simple and efficient information retrieval methods for these data sources are a pressing need.

Here, we focus on two query tasks. First, given a query node in the graph, e.g. a person or a category in DBpedia, which other nodes in the graph represent entities that are useful in the context of the given query node? Answers might be other concepts that could be used to refine or extend interactive search.

[1] http://www.cyc.com/opencyc
[2] http://linkedlifedata.com/

R. García-Castro et al. (Eds.): ESWC 2011 Workshops, LNCS 7117, pp. 65–79, 2012.

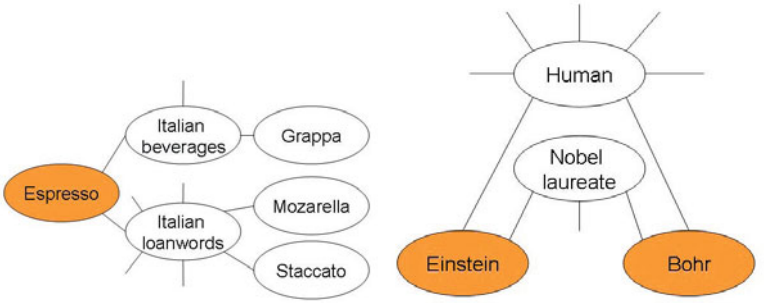

Fig. 1. Stylized examples of the two proposed tasks (data taken from DBpedia) and the associated challenges. In the left example, "Espresso" might be considered to be semantically closer to "Grappa", if compared to "Mozzarella" and "Staccato". But since "Italian loanwords" has a higher degree than "Italian beverages", ranking algorithms, e.g., based on PageRank, might conclude differently. A similar phenomenon occurs on the right side where the link via "Nobel laureates" is more informative than the link via "Human", despite the fact that "Human" has a higher degree than "Nobel laureate".

In the second task, one selects two nodes in the graph and would like to find out how those two are related. For example, a user might be interested to know via which people or concepts Albert Einstein and Niels Bohr are related in DBpedia. In the field of bioinformatics such a link query could be applied to genes and diseases and one would be able to discover novel pathways unknown to the existing literature (Antezana et al., 2009; Bundschus et al., 2008).

In principle, the two described tasks could both be solved with shortest-path search on the ER graph, most simply by assuming a step metric, i.e. one assigns every edge in the graph a unit cost. For the first task, one would compute the shortest-path distance from the query node to all other nodes, e.g. using Dijkstra's algorithm (Dijkstra, 1959). For the second task, one would calculate the k-shortest paths connecting the two given nodes, e.g. by using the k-shortest paths algorithm of (Yen, 1971).

While shortest-path search is straightforward and efficient to implement, it often returns highly irrelevant results. Consider for example a graph which contains a node that is connected to nearly all other nodes, such as a broad category that encompasses most entities in the graph. In the first task, shortest-path search would return the high-degree node for almost any query nodes. This lack of discrimination hinders context search, since a broad topic would lack specific relevance to any one node in the graph. Also, most other graph nodes would be returned with distance two, although many of them would be unrelated to the query node. In the second task, consider a database in which every person is connected to a "Human" category node. In this case, the fact that Einstein and Bohr are both humans is less informative than the fact that they are both Nobel laureates, but the paths via "Human" and "Nobel laureates" nodes would have equal distance. These problems are schematically depicted in Figure 1.

One might consider putting higher weight on interesting nodes in ER graphs using PageRank-like approaches (Brin and Page, 1998). However, a simple implementation of this idea might have an adverse effect in our setting: in PageRank nodes are deemed popular and thus important if many links point to them. Thus high-level, highly connected nodes would become more important, although they are often uninformative from our view point as argued above. A different popularity-based ranking concept (Kasneci et al., 2008) assumes that facts that have many witnesses in a corpus are highly informative. This, however, requires additional input apart from the graph, and the assumption that important facts are expressed more often than others may not hold in curated data stores like wikipedia or company networks.

An approach more interesting in our context is based on the graph structure alone and uses properties of random walks on the graph. It has been argued that the commute time between nodes in an ER graph is a useful distance measure to find relevant neighboring nodes (Baluja et al., 2008). In each step, the random walk jumps from one node with equal probability to any neighboring node. The commute time is the expected number of steps required for a random walk starting from one node to reach another before returning to the first. Using this metric, the problems with the step distance are alleviated in that the commute distance decreases not only if there is a short path between two nodes, but also if there are several short paths between them. Thus, a single link over a high-degree hub is not likely to yield a small distance. Moreover, if two nodes are joined by a path containing a high-degree node, a random walk is likely to "get lost" at the high-degree node, taking steps into unrelated regions of the graph, increasing the commute time between the two nodes.

While these are strong intuitive arguments for the commute distance, it is very expensive to compute. There exists an analytic formula for the commute distance in terms of the pseudoinverse of the graph Laplacian (Klein and Randić, 1993). This, however, is computationally prohibitive for the large graphs encountered in the semantic web: The pseudoinverse of the sparse graph Laplacian matrix is in general not sparse such that storage requirements increase quadratically with the number of graph nodes. More efficient approximations of the commute distance have been developed for citation graphs in Sarkar et al. (2008). Their methods, however, still require 4 seconds for a graph of 600k nodes, which is only a moderate size in the context of the semantic web. Moreover, it is not clear how their methods would perform on more structurally complex graphs such as DBpedia.

Our novel approach combines the simplicity and speed of shortest path finding with the properties of the commute distance. In essence, we perform shortest path finding with a problem-adapted graph metric that assigns to each edge a weight which depends on the degrees of its endpoints. Finding shortest paths in our novel metric can then be interpreted in terms of maximizing the log-likelihood of the path between the two nodes in a random walk on the graph. It can be seen as an optimally adapted first-order approximation to the commute distance, and thus experimentally inherits many of the favorable properties of

the commute distance. At the same time the computations are very efficient since they reduce to purely local shortest path searches that can be performed with standard graph algorithms.

In the next section we introduce the novel metric for solving the two tasks. In Section 3 we justify it in terms of random walks, and discuss how it can be seen as an approximation to the commute distance. In Section 4 we introduce a number of examples and a numeric evaluation on several semantic datasets, demonstrating the superior behavior both in comparison to the step-distance path-finding approach and to another simple approximation of the commute distance.

2 Proposed Approach

Let the semantic ER graph be represented as $G = (V, E)$ where V is the set of nodes or entities and E is the set of edges or relations holding between the entities. We do not distinguish between different relation types for the edges, opting to treat them all equally. Moreover, we also remove edge directions from the graph, since the semantic direction of a relation statement is often not syntactically obvious; for example, "buys" or "is bought by" might both appear in a graph.

For each edge $(u, v) \in E$, we then define a weight

$$w_{(u,v)} = \log(\deg(u)) + \log(\deg(v)),$$

where $\deg(u)$ and $\deg(v)$ are the degrees of the node u and node v, respectively. If G is connected, then the degree $\deg(u)$ of all nodes u is greater than or equal to one and thus $w_{(u,v)} \geq 0$ for all $(u, v) \in E$. The weights therefore define a valid positive semi-definite path metric on G. The two described tasks can now be solved using these novel edge weights in standard shortest paths routines.

In our first task, a node is specified as a query input and we must retrieve a set of other nodes that are ranked based on how relevant or related they are to the query node. The results of this search might include nodes that are related, for example, to topics that are contained within the query topic, to topics which contain the query node, or to topics that are related by common membership within a category or broader topic. In order to solve this task, we find the shortest paths between the query node and all other nodes (considering the weights) and rank the results. We apply Dijkstra's algorithm, which allows us to stop short and directly retrieve the top ranked nodes without computing the shortest paths to all other nodes.

In our second task, we are given two distinct nodes as input and wish to find paths between both that, ideally, provide unique insight into the relationship between the two nodes. This might include interesting or distinct ways that the two nodes are related. We solve this by finding the k shortest paths between the two nodes in the weighted graph, where k is a free parameter. We return the sequence of nodes in each of the k-shortest paths.

The proposed metric can be justified intuitively: In the metric the distance to high-degree nodes often carrying very unspecific information, e.g. the "Human" node, is large. In contrast, the distance to more specific, low degree nodes is small. Effectively, we are searching in compactly connected, local subgraphs, assumed to carry context-specific information.

As we will see in the experimental section, the proposed approach yields matches for query nodes that are highly specific in subject matter and are very appropriate for someone who, for example, wants to explore a particular academic subject in detail. In addition, our metric facilitates the discovery of novel, distinct relations between nodes: nodes that are related to each other in some unique way (i.e. there is a path between them that is connected to relatively few other nodes outside of that path) are closer to each other than nodes that are linked by a very common relationship.

These intuitions can be further motivated by relating our approach to random walks on semantic graphs. Before we do so in the next section, note that the proposed approach only requires standard graph algorithms and is thus simple to implement. It also runs very efficiently even for large graphs. For example with Dijkstra's algorithm, we only have to visit on the order of k nodes to find the k closest neighbors.

3 The Connection to Random Walks

Consider a random walk on G. In each step the walk moves from one node v to an arbitrary adjacent node with probability $\deg(v)^{-1}$, where $\deg(v)$ is the degree of v. Denote the set of paths between two fixed nodes u and v by $\Pi_{u,v}$, i.e. $\pi = (\pi_1, \pi_2, .., \pi_{len(\pi)}) \in \Pi_{u,v}$ iff $\pi_1 = u$ and $\pi_{len(\pi)} = v$. The probability of the random walk following such a path and returning on the same route then is

$$\begin{aligned} p(\pi) &= \left(\prod_{i=1}^{n-1} \deg(\pi_i)^{-1} \right) \left(\prod_{i=2}^{n} \deg(\pi_i)^{-1} \right) \\ &= \deg(\pi_1)^{-1} \prod_{i=2}^{n-1} \deg(\pi_i)^{-2} \deg(\pi_n)^{-1}. \end{aligned}$$

The negative log-likelihood follows as

$$-\log p(\pi) = \log \deg(\pi_1) + 2 \sum_{i=2}^{n-1} \log \deg(\pi_i) + \log \deg(\pi_n) = \sum_{i=1}^{n-1} w_{(\pi_i, \pi_{i+1})}.$$

This result shows that the negative log-likelihood of the path is exactly equal to the path length in our proposed metric. Moreover, shortest path finding between u and v using this metric is thus equivalent to finding that path in $\Pi_{u,v}$ that has minimal negative log-likelihood, or maximal probability, of a random walk following that path back and forth.

3.1 Approximation of the Commute Distance

Random walk probabilities also determine the commute time which has been proposed as an information metric on ER graphs before (Baluja et al., 2008). In contrast to our approach, the commute distance does not only measure whether there is a single high-probability connection between two nodes, but also takes into account how many such paths there are.

Since the commute distance uses more of the structure of the graph, it is potentially more robust. However, this comes at a huge computational cost and, in comparison, our approach is extremely efficient. Still, our approach can be seen as a first order approximation of the commute distance, as we will now discuss.

The commute time $C(u, v)$ between nodes u and v is

$$C(u,v) = \sum_{\pi:(\pi_1=u,\dots\pi_k=v,\dots\pi_{len(\pi)}=v)} len(\pi)\, p(\pi) = \sum_{\pi} len(\pi) \prod_{i=1}^{len(\pi)-1} \deg(\pi_i)^{-1}.$$

The sum is over all paths that start and end at u and visit v in between. Since all terms are positive, a first order lower bound can be achieved by taking into account only a single such path $\hat{\pi}$, i.e.

$$C(u,v) \geq len(\hat{\pi})\, p(\hat{\pi}).$$

Whether this is a tight bound depends on how concentrated the path probabilities are on a single term. While there are certainly situations where the path probabilities are not concentrated, we would argue that for many semantic graphs the approximation might be acceptable. The reason is that the degree of the nodes enters multiplicatively into the sum. Consider query nodes that are both members of two categories of highly different numbers of instances. Then the path through the smaller category and back on the same way is actually quadratically preferred over the one through the larger category.

Given the above lower bound we now try to find the optimal lower bound for the commute distance $C(u, v)$. That is we search for that path $\hat{\pi}$ that contributes the most to the sum above. This then leads to the criterion

$$\max_{\hat{\pi}}\ len(\pi)p(\hat{\pi}) = \min_{\hat{\pi}}\ -\log len(\pi) + \sum_{i=1}^{n-1} \log \deg(\hat{\pi}_i).$$

The second term is additive in the length of the path and quickly dominates the first term whose magnitude increases sub-linearly. At the same time, for paths of equal length only the second term has to be considered for the minimization. Without too large an error we can thus neglect the first term in most cases. Moreover, we restrict the optimization set to those paths that go from u to v and return the same way. The result will still be a lower bound on the commute distance, and it allows us to rewrite the problem using our proposed metric as

$$\min_{\hat{\pi}\in\Pi_{u,v}}\ 2\sum_{i=1}^{n-1} w_{(\hat{\pi}_i,\hat{\pi}_{i+1})}.$$

This is equivalent to our proposed approach up to a constant factor. We can thus interpret our approach as (approximately) finding an optimal lower bound to the commute distance, with the advantage that our bound can be computed very efficiently and with simple standard graph algorithms.

This derivation has involved a number of approximation steps that are not necessarily the tightest ones possible, see the review of Lovász (1993) for other approximations. Yet, this argument still gives some intuition why minimizing our proposed objective might be sensible.

4 Experiments

We demonstrate our methods using two large, real world semantic ER graphs, namely DBpedia and OpenCyc.

For the DBpedia dataset, we use the category (skos) and the article-category data files and create an unweighted, undirected graph neglecting the different relationship types and directions. We ignore literals since they do not add information to the graph structure. In the interest of a fair comparison we discard the "concept" node to which each category is connected.

We similarly define the graph for the OpenCyc dataset. An overview of the properties of both graphs is given in Table 1.

Table 1. Basic statistics of the used datasets

	DBPedia	OpenCyc
nodes	3,660,898	150,088
Edges	8,947,631	554,762
Average degree	4.88	7.39

As baseline methods for our comparisons we use the following two approaches. First, we compare our method to using shortest paths with the step distance. Second, we compute a simple approximation of the commute distance as follows. The exact computation of the commute distance on the full graph is intractable, since it requires the graph Laplacian's pseudoinverse, a matrix that for most graphs is too big to even be stored. Instead, we assume here that the commute distance is moderately local. For each query, we extract the 1000 closest nodes to the query node – in step distance – and only use the subgraph spanned by these nodes and the edges between them to compute the commute distance using the analytic formula of Klein and Randić (1993). If the subgraph has only few edges connecting it to the remaining graph, the approximation is fairly reasonable. However, if a very unspecific node with many neighbors is among the closest nodes to the query node, then it will connect almost any node in the graph to the query by a path of, say, length 2. In this case the selection of the 1000 closest neighbor nodes is arbitrary and not much can be expected from our approximation of the commute distance. The baseline should thus not be regarded as an accurate representative of the true commute distance.

4.1 First Task: Neighborhood Finding

In the following we discuss a number of example results from the two datasets. In Table 2 we list the results of a search for the query node "Espresso." In this case, the step distance gets easily distracted by the high-degree neighbor "Italian loanwords." As a result, the majority of the results listed are unrelated Italian terms which refer mostly to music and food. The commute distance approximation returns highly irrelevant words that are also related mostly to food and music. This is probably due to the nature of the approximation we use – most of the 1000 nodes nearest to the espresso node are probably also connected to the query node via the "Italian loanwords" node. Our method, on the other hand, returns a list of about one third Italian sodas and non-coffee beverages and about two thirds drinks made with espresso or at least coffee, as well as a few other types of terms.

In Table 3 we performed another search for the term "iPod." The step distance mostly gives us various categories relating to hardware or software, and the commute distance mixes these results with a few more specific terms relating to the iPod's function and to the related iPhone. Our method, on the other hand, yields mostly articles relating specifically to variations and functions of the iPod and the iTunes software, which is integral to the use of the iPod.

In Figure 2, we have a graphical representation of the results of the query "Atlanta" with our distance metric. Compare these with the results using the step

Table 2. Top 30 results of neighborhood search for query node "Espresso" in DBpedia, along with the distances from the query node. The first column labeled "Step" contains results for shortest path finding with the step distance; the second column reports results using our proposed approach; the last column shows the results of our simple approximation of the commute distance. Entities marked with (C) represent skos categories, other items are regular DBpedia resources.

Step		Our approach		Approx. Commute	
Espresso	0	Espresso	0	Espresso	0
(C)Italian beverages	1	(C)Italian beverages	4.5	(C)Italian loanwords	1295.75
(C)Italian loanwords	1	(C)Coffee beverages	5.05	(C)Coffee beverages	1296.86
(C)Coffee beverages	1	(C)Italian loanwords	6.09	(C)Italian beverages	1297.35
(C)Italian cuisine	2	Bombardino	7.9	(C)Italian cuisine	1339.5
(C)Italian words and phrases	2	Caffè corretto	8.59	(C)Opera terminology	1401.79
(C)Italian language	2	Grappa	8.59	(C)Italian words and phrases	1452.94
(C)English words foreign origin	2	Torani	8.59	(C)Pasta	1467.75
(C)Romance loanwords	2	Lemonsoda	8.59	(C)Mediterranean cuisine	1529.31
(C)Beverages by region	2	Oransoda	8.59	(C)Cuisine by nationality	1544.99
(C)Italian alcoholic beverages	2	Pelmosoda	8.59	(C)Opera genres	1582.18
(C)Coffee preparation	2	Beverly (drink)	8.59	(C)Opera	1584.02
Castrato	2	Doppio	8.59	(C)Performing arts	1599.79
Da capo	2	Caffè	9	(C)Musical notation	1601.59
Graffiti	2	Chinotto	9	(C)European cuisine	1664.97
Glissando	2	Ammazzacaffè	9	(C)Italian language	1685.92
Macaroni	2	Stappj	9	Turkish coffee	1691.75
Mozzarella	2	Galvanina	9	(C)Beverages by region	1721.4
Opera	2	Irish coffee	9	(C)Dried meat	1737.1
Pasta	2	Cortado	9	(C)Musical theatre	1740.42
Pizza	2	Iced coffee	9	(C)Music	1743.96
Spaghetti	2	Pepsi Kona	9	(C)Articulations	1756.06
Tempo	2	Flat white	9	(C)English words foreign origin	1756.7
Cappuccino	2	Mochasippi	9	(C)Singing	1760.76
Legato	2	Red eye (drink)	9	(C)Salumi	1764.59
Staccato	2	Liqueur coffee	9	(C)Croatian cuisine	1769.7
Operetta	2	Lungo	9	(C)Entertainment	1773.37
Cadenza	2	Caffè Americano	9	(C)Theatrical genres	1788.05
Concerto	2	Espresso con panna	9	(C)Italian culture	1795.82
Cantata	2	Caffè breve	9	(C)Italian prod. protected origin	1799.13

Fig. 2. Graphical representation of the results of the query "Atlanta" in our new distance. In the middle of the figure is the query node, Atlanta. The remaining nodes are the search results. This presentation of the results shows the connection of each of the search results to the query node – that is, the shortest path found from the query to each of the results is preserved in this graph representation.

distance, graphically represented in Figure 3. While the step distance returns mostly generic facts such as the fact that Atlanta is a U.S. city, our distance returns more interesting results including the connection to the U.S. Court of Appeals for the Eleventh Circuit and the connection to Munich – they are both Olympic cities.

We also provide results for the OpenCyc dataset, which is of a slightly different nature. It contains many rather unspecific nodes like "temporally stuff like thing" which are nice examples of how such high-degree nodes are avoided by our algorithm. In Table 4 we show the results of a search for "Machine learning." While the results of the other methods become wildly irrelevant after only the first few matches, nearly the first half of the results of our approach are still relevant to the topic at hand.

To demonstrate the dramatic computational advantage of our method against the described approximation of the commute distance, we picked 1000 query

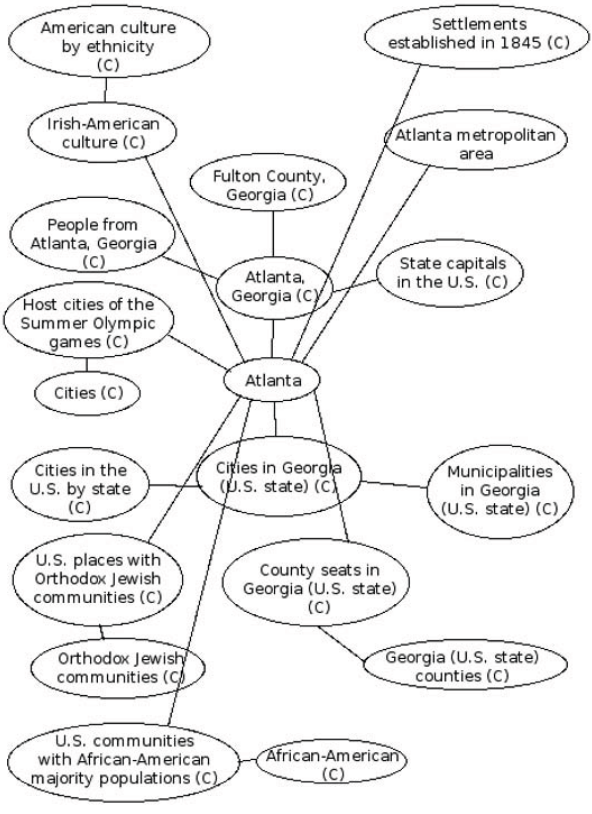

Fig. 3. Graphical representation of the results of the query "Atlanta" in the step distance. As in figure 2, the shortest paths from the query node to each of the results is presented.

nodes at random and performed a query using all three methods. The mean run-times on a standard desktop PC as well as the standard deviation for each method is given below

Step	Our Approach	Approx. Commute
0.13s (0.07s)	0.11s (0.04s)	10.43s (9.51s)

The average run-time for our method was 0.11 seconds, compared to an average of 10.43 seconds for our approximation of the commute time – a difference of two orders of magnitude. As would be expected, our method runs approximately as fast as the step distance method.

It is worth noting that the time taken to approximate the commute time was extremely high in some cases. The longest time taken with the commute distance was over one minute, whereas the longest time taken with our method was only 0.66 seconds. Furthermore, one should also consider that the method we have used to calculate the commute distance is only an approximation using

Table 3. Top 30 results for calculating neighborhood search for query node "iPod" in DBpedia. Labels as in Table 2.

Step		Our approach		Approx. Commute	
IPod	0	IPod	0	IPod	0
(C)2001 introductions	1	(C)IPod	4.97	(C)ITunes	695.93
(C)IPod	1	(C)Industrial designs	5.78	(C)Portable media players	698.52
(C)Portable media players	1	(C)ITunes	5.98	(C)Digital audio players	750.23
(C)ITunes	1	(C)2001 introductions	6.29	(C)IPhone OS software	757.78
(C)IPhone OS software	1	(C)Portable media players	6.49	(C)IPod	784.31
(C)Industrial designs	1	(C)IPhone OS software	6.52	(C)Industrial designs	857.01
(C)2001	2	IPod click wheel	8.15	(C)Smartphones	889.69
(C)Apple Inc. software	2	IPod Photo	8.84	(C)2001 introductions	907.63
(C)Industrial design	2	List of iPod models	8.84	(C)Mac OS X software	929.97
(C)Windows software	2	Dock Connector	8.84	(C)Touchscreen portable media players	955.66
(C)Software by operating system	2	IPod Mini	9.25	(C)Consumer electronics brands	959.29
(C)Apple Inc. hardware	2	IPod advertising	9.25	(C)Apple Inc. software	973.22
(C)Windows media players	2	IPhone Touch	9.25	(C)IPhone	974.07
(C)Mac OS X software	2	IPod Nano	9.53	(C)2007 introductions	1010.71
(C)Digital audio players	2	Neistat Brothers	9.53	IPhone	1025.22
(C)USA PATRIOT Act	2	IPod Classic	9.53	(C)IPhone OS	1031.79
(C)MPEG	2	Ipod+HP	9.53	(C)Web 2.0	1035.86
(C)IPod accessories	2	List of iPhone OS devices	9.53	(C)Windows software	1047.63
(C)IPod software	2	IPod Shuffle	9.76	ITunes	1049.63
(C)21st-century introductions	2	Juicy Salif	9.77	(C)Apple Inc. hardware	1057.46
(C)ITunes-exclusive releases	2	DADVSI	10.09	(C)Software by operating system	1075.57
(C)IPhone OS games	2	NextWorth Solutions	10.09	(C)Online social networking	1096.2
(C)Mac OS X media players	2	IMix	10.17	(C)Mac OS software	1096.22
(C)Apple Inc. peripherals	2	Genius (iTunes)	10.17	(C)Personal digital assistants	1112.79
(C)Apple Inc. services	2	AirTunes	10.17	(C)Brands	1126.36
(C)Vehicles introduced in 2001	2	ITunes law	10.17	(C)Media players	1126.94
(C)IPhone	2	ITunes Music Store	10.17	(C)Creative Technology products	1129.28
(C)2001 comic debuts	2	ITunes U	10.17	(C)IPod software	1151.43
(C)IPhone OS	2	ITunes Applications	10.17	Nimbuzz	1156.45

Table 4. Top 30 results of neighborhood search for query node "Machine learning" in OpenCyc. Labels as in Table 2.

Step		Our approach		Approx. Commute	
machine learning	0	machine learning	0	machine learning	0
temporal stuff also a durative event	1	machine rule induction	2.48	first-order collection	875.61
computer activity	1	discriminative weight learning	2.89	temp stuff also a durative event	887.91
discriminative weight learning	1	generative weight learning	2.89	computer activity	897.63
generative weight learning	1	MLN Generated Using Learning Type	3.18	temporal stuff	921.03
machine rule induction	1	computer activity	6.27	employee computer activity type	1061.05
MLN Generated Using Learning Type	1	markov logic network	6.87	computer activity type	1090.34
alcoholism	2	temporal stuff also a durative event	7.75	athletic activity	1104.59
burning	2	MLN Data File Pathname	9.86	physical information transfer	1115.24
flowing	2	MLN File Pathname	9.86	biological transportation	1138.37
anthem	2	MLN Generated Using Cmd String	9.86	body movement	1152.19
the union of ensemble showman	2	MLN Rule File Pathname	9.86	recreational activity	1169.12
playing	2	MLN Type Const Dec File Pathname	9.86	using a computer	1181.74
halt	2	MLN Represented By Microtheory	10.27	information-accessing event	1195.75
rock climbing	2	Content Of MLN Fn	10.56	physical event	1196.47
snow-skiing	2	computer activity that computer did	11.85	structured information source	1213.39
Iter. Event Scene Fn id veg. 1-3 km	2	computer activity that person did	11.85	type of accomplishment	1236.74
rafting	2	hack	11.85	individual	1239.97
candy making	2	computer thread	11.85	computer editing	1256.46
composting	2	help desk session	11.85	internet activity	1266.56
woodworking	2	network packet filtering	11.85	running computer process	1280.32
diagnosis of Wegeners granulomatosis	2	network packet routing	11.85	locomotion event	1280.92
breast cancer treatment	2	opening presents	11.85	ride	1303.08
AIDS treatment	2	packet sniffing	11.85	CW instantiating	1313.32
acne care	2	partitioning a disk	11.85	unnatural thing	1315.36
affliction procedure	2	placing a residual malicious program	11.85	biological process	1321.43
allergic reaction treatment	2	browser requests a secure connection	12.13	QA clarifying collection type	1338.81
atrial septal aneurysm med treatment	2	locking computer display	12.13	internet communication	1355.33
most autistic procedure	2	website maintenance	12.13	network propagation	1357.71
vision impairment treatment	2	network prop. malicious program	12.13	candidate KB completeness node	1360.95

a graph of 1000 nodes. The most computationally intensive step required of the commute distance is the calculation of the pseudoinverse. Since this step requires cubic time to calculate, an attempt to improve the accuracy of the estimate by adding more nodes to the approximation would drastically increase the time required for computation, while an exact computation would be intractable for most practical problems.

4.2 Task 2: Path Finding

In this section, we present an example of our method as applied to the path finding task, displaying the paths that our method is able to find between two nodes in the graph – i.e., between two concepts in our semantic network. We also compare our method to path finding using the step distance to show the advantage that our method has in discovering truly distinct and specific connections between concepts.

Consider paths between the nodes "Computer vision" and "Machine learning", again with data from DBpedia. The resulting paths are listed in Table 5. Many of the results of our method provide insight into exactly how machine learning is used to solve specific tasks in the computer vision domain. Although insightful, some of the paths returned here by our method have significant intersections with each other. This could, however, be remedied by, for example, modifying the k-shortest paths algorithm to add extra weight to the edges equivalent to the ones traversed in previously discovered paths. Such a modification would lead to increased diversity in the results.

The step distance, on the other hand, gives us only very vague, general connections between the two subjects. The most that we learn from these results is that computer vision and machine learning are both within the subject of artificial intelligence.

Table 5. Path finding between the terms "Computer vision" and "Machine learning" in DBpedia

Our Approach:

- Path 1 (length 15.2407): Computer vision - (C)Computer vision - (C)Learning in computer vision - Machine learning
- Path 2 (length 22.1722): Computer vision - (C)Computer vision - (C)Object recognition and categorization - Boosting methods for object categorization - (C)Learning in computer vision - Machine learning
- Path 3 (length 22.4706): Computer vision - (C)Artificial intelligence - (C)Cybernetics - Machine learning
- Path 4 (length 23.5585): Computer vision - (C)Computer vision - Segmentation based object categorization - (C)Object recognition and categorization - Boosting methods for object categorization - (C)Learning in computer vision - Machine learning
- Path 5 (length 23.5585): Computer vision - (C)Computer vision - Object recognition (computer vision) - (C)Object recognition and categorization - Boosting methods for object categorization - (C)Learning in computer vision - Machine learning

Step Distance:

- Path 1 (length 3): Computer vision - (C)Artificial intelligence - (C)Machine learning - Machine learning
- Path 2 (length 3): Computer vision - (C)Computer vision - (C)Learning in computer vision - Machine learning
- Path 3 (length 3): Computer vision - (C)Artificial intelligence - (C)Cybernetics - Machine learning
- Path 4 (length 4): Computer vision - (C)Artificial intelligence - (C)Machine learning - (C)Learning - Machine learning
- Path 5 (length 4): Computer vision - (C)Computer vision - (C)Artificial intelligence - (C)Machine learning - Machine learning

Table 6. Path finding between the terms "Seattle" and "Quantum mechanics" in DBpedia

Our Approach:

- Path 1 (length 42.957): Seattle - (C)Seattle, WA - Homelessness in Seattle - (C)Articles Created via the Article Wizard - Magnetic translation - (C)Quantum Mechanics - Quantum Mechanics
- Path 2 (length 44.3577): Seattle - (C)Isthmuses - (C)Coastal and oceanic landforms - Sound (geography) - (C)Sound - Amplitude - (C)Fundamental physics concepts - Quantum mechanics
- Path 3 (length 44.933): Seattle - (C)Isthmuses - (C)Coastal and oceanic landforms - Sound (geography) - (C)Sound - Node (physics) - (C)Fundamental physics concepts - Quantum mechanics
- Path 4 (length 45.744): Seattle - (C)Isthmuses - Isthmus - (C)Coastal and oceanic landforms - Sound (geography) - (C)Sound - Amplitude - (C)Fundamental physics concepts - Quantum mechanics
- Path 5 (length 46.1677): Seattle - Seattle, Washington - (C)Education in Seattle, Washington - Washington Large Area Time Coincidence Array - (C)Cosmic-ray experiments - (C)Experimental particle physics - Cherenkov radiation - (C)Fundamental physics concepts - Quantum mechanics

Step Distance:

- Path 1 (length 6): Seattle - (C)Seattle, Washington - Homelessness in Seattle - (C)Articles created via the Article Wizard - Magnetic translation - (C)Quantum mechanics - Quantum mechanics
- Path 2 (length 7): Seattle - (C)Isthmuses - (C)Coastal and oceanic landforms - Archipelago - (C)Greek loanwords - Atom - (C)Fundamental physics concepts - Quantum mechanics
- Path 3 (length 7): Seattle - (C)Seattle, Washington - (C)People from Seattle, Washington - Virgil Bogue - (C)1916 deaths - Kārlis Mīlenbah - (C)Quantum mechanics - Quantum mechanics
- Path 4 (length 7): Seattle - (C)Seattle, Washington - Homelessness in Seattle - (C)Articles created via the Article Wizard - Photomechanical effect - (C)Mechanics - (C)Quantum mechanics - Quantum mechanics
- Path 5 (length 7): Seattle - (C)Seattle, Washington - Homelessness in Seattle - (C)Articles created via the Article Wizard - Magnetic translation - (C)Quantum magnetism - (C)Quantum mechanics - Quantum mechanics

Note that our method is actually able to find informative paths of significant length. While for the step distance the exponential number of possibilities for such paths quickly renders the retrieval infeasible, our method is still able to discriminate between the many choices. This might be an important advantage when applying this framework to biomedical databases, such as for example Linked Life Data. Here, one often tries to find non-obvious rather long distance interactions between different genes and diseases to discover novel pathways. Focusing on the most discriminative ones might save significant research effort in this domain.

In Table 6, we present the results of a shortest paths query of five steps between Seattle and Quantum Physics in our metric and the step metric. In addition to the connection through the interesting semantic ambiguity of the word "Sound," our method yields what could be argued to be the most valuable link between the two concepts: the involvement in the Washington Large Area Time Coincidence Array, a distributed physics experiment. The paths in the step distance are mostly dominated by the tenuous link of articles created with the Article Wizard. It is also interesting (although not vital to the interpretation of the results) to note that although Kārlis Mīlenbah is indeed listed in the category Quantum Physics, at the time of publication, the article for Kārlis Mīlenbah seems to imply that he probably has nothing to do with quantum physics.

5 Conclusion

We have presented a novel metric for solving information retrieval tasks in semantic networks. The metric just depends on the degrees of adjacent nodes and favors paths via low-degree nodes. As our experiments demonstrated, the approach is capable of finding atypical but interesting neighbors of a query node. In addition, the approach is able to find original informative paths between two specified nodes.

Often the authors themselves discovered novel, interesting information when querying the test datasets DBpedia and OpenCyc with different entities. This makes us strongly believe that the proposed approach could also be helpful to others.

A detailed user study is currently under way. From a technical point one could imagine mixing the step metric and the proposed one to obtain a tunable trade-off between the length and the distinctiveness of a path. It would also be interesting to explore ways to learn additional parameters in the metric, e.g. by assigning different weights to specific edge types. Such parametric learning approaches, however, would require a benchmarking dataset which is currently not available to us. In contrast, the proposed approach is parameter free and solely dependent on intuitive arguments.

References

Antezana, E., Kuiper, M., Mironov, V.: Biological knowledge management: the emerging role of the Semantic Web technologies. Briefings in Bioinformatics (2009)

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: Dbpedia: A Nucleus for a Web of Open Data. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)

Baluja, S., Seth, R., Sivakumar, D., Jing, Y., Yagnik, J., Kumar, S., Ravichandran, D., Aly, M.: Video suggestion and discovery for YouTube: taking random walks through the view graph. In: Proceeding of the 17th International Conference on World Wide Web, pp. 895–904. ACM (2008)

Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems 30(1-7), 107–117 (1998)

Bundschus, M., Dejori, M., Stetter, M., Tresp, V., Kriegel, H.-P.: Extraction of semantic biomedical relations from text using conditional random fields. BMC Bioinformatics 9(1), 207 (2008)

Dijkstra, E.: A note on two problems in connexion with graphs. Numerische Mathematik 1(1), 269–271 (1959)

Kasneci, G., Suchanek, F., Ifrim, G., Ramanath, M., Weikum, G.: Naga: Searching and ranking knowledge. In: Proc. of ICDE, pp. 1285–1288 (2008)

Klein, D.J., Randić, M.: Resistance distance. Journal of Mathematical Chemistry 12(1), 81–95 (1993)

Lovász, L.: Random walks on graphs: A survey. Combinatorics, Paul Erdos is Eighty 2(1), 1–46 (1993)

Momtchev, V., Peychev, D., Primov, T., Georgiev, G.: Expanding the pathway and interaction knowledge in linked life data. In: Proc. of International Semantic Web Challenge (2009)

Sarkar, P., Moore, A., Prakash, A.: Fast incremental proximity search in large graphs. In: Proceedings of the 25th International Conference on Machine Learning, pp. 896–903. ACM (2008)

Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A Core of Semantic Knowledge. In: 16th International World Wide Web Conference (WWW 2007). ACM Press, New York (2007)

Yen, J.: Finding the k-shortest loopless paths in a network. Management Science 17(11), 712–716 (1971)