Short Paper

Notable Characteristics Search through Knowledge Graphs

Davide Mottin¹, Bastian Grasnick², Axel Kroschk², Patrick Siegler², Emmanuel Müller¹
Hasso Plattner Institute
¹first.last@hpi.de ²first.last@student.hpi.de

ABSTRACT

Query answering routinely employs knowledge graphs to assist
the user in the search process. Given a knowledge graph that
represents entities and relationships among them, one aims at
complementing the search with intuitive but effective mecha-
nisms. In particular, we focus on the comparison of two or more
to retrieve entities node similar to the query nodes
provided by the user, and then exploits distributional prop-
to understand whether a certain attribute is interesting or not.
Ours preliminary experiments demonstrate the solidity of our
approach and show that we are able to discover notable char-
acteristics that are indeed interesting and relevant for the user.

1 INTRODUCTION

Search engines have greatly evolved from simple indexes of pages
to complex systems that are able to predict user intentions and
answer queries on a variety of data sources. One way to im-
prove the search quality is by using a knowledge graph that
represents entities (e.g., Angela Merkel, Germany) as nodes and
relationships between them (e.g., leaderOf) as edges in a graph.
The great expressiveness of knowledge graphs can complement
the search with more flexible search paradigms. Assume for in-
stance a scholar who requires to know some non-trivial facts
about Angela Merkel and Emmanuel Macron with respect to
other country leaders. It would be interesting to discover for
instance that Angela Merkel studied Physics as opposed to most
of other leaders, and that she has no children. We call this
fact a notable characteristic, to remark the unexpected and non-
trivial aspect of the discovery. To this end, we propose a novel
search called notable characteristics search that allows the
retrieval of such facts from a set of input query entities. Dis-
covering notable characteristics constitutes a ground for targeted
analyses of products (e.g., comparing two cameras effectively) in
electronic commerce or microorganisms in biological networks
(e.g., two influence bacteria) with respect to a set of similars. As
a consequence, in all the cases in which a knowledge graph is
available, the discovery of notable characteristics becomes an
expressive and powerful search type for any user, from experts
and practitioners to novice users.

In our setting, we assume the user provides a set of query
nodes to be compared and the algorithm finds a set of notable
characteristics of these nodes. Given a node, a property is a
relationship with other nodes (e.g., leaderOf). A characteristic or
property is notable, if it deviates from what one would expect for
the kind of nodes (e.g., presidents) into consideration. To the best
of our knowledge, this is the first study of automatic discovery
of notable characteristics (or properties).

The discovery of notable characteristics entails two challenges.
First, given the set of query nodes we need to compare them to
only those nodes that are similar to some extent. Second, we need
to select only those properties that are significantly different
from the one expressed in the query. Note that tackling the first
challenge is very important, as the comparison of the query
nodes has to be performed with a set of similar nodes, which we
call the context of the query. Consider the naïve approach that
returns notable characteristics simply by comparing the query
nodes and assume that the user provides "Angela Merkel" and
"Theresa May" as query. This is a counter example for the naïve
direct comparison, as it will not return the gender as a notable
characteristic. Both query nodes are female, however only in
comparison with other presidents this becomes an interesting
fact. On the other extreme, selecting all the nodes in the graph
as context will mislead the analysis towards non-relevant nodes.
Take our example of "Angela Merkel" and "Emmanuel Macron".
A naïve selection of all humans will not work as context, since
the gender characteristic is not notable among all persons.

It is crucial to provide a thorough context selection to prevent
the above cases. Therefore, we introduce the discovery of context
nodes, i.e., nodes similar to the query nodes. An example of the
proposed approach is depicted in in Figure 1. To this end, we
devises a method that exploits metapaths [12] and random walks
for context discovery. We also propose a generic framework
that efficiently discover notable characteristics through a novel
probabilistic approach based on distribution comparison.

Our contributions are summarized as follows: (1) We formal-
ize the problem of notable characteristics search given a set of
query nodes as input. (2) We show how to effectively compute
metapaths to find the context nodes in knowledge graphs. (3) We
introduce a probabilistic approach to discover notable charac-
teristics given a query node set. (4) We experimentally evaluate
our context selection approach through a user study, and show
evidence of our discovered notable characteristics and the real
time performance of the proposed algorithms.

2 NOTABLE CHARACTERISTICS SEARCH

We are given a set \( \mathcal{A} \) of node labels and a set \( \mathcal{L} \) of edge labels. A knowledge graph is a directed graph \( G : (\mathcal{V}, \mathcal{E}, \phi, \psi) \), where

\( \mathcal{V} \) is a set of nodes, \( \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \) is a set of edges, \( \phi : \mathcal{V} \mapsto \mathcal{A} \),

\( \psi : \mathcal{E} \mapsto \mathcal{L} \) are node and edge labeling functions, respectively.

Figure 1: An example knowledge graph, the query (Merkel
and Macron), and the discovered context nodes (Putin, Xi
Jinping, and Rajoy). The fact that Merkel and Macron do
not have children is a notable characteristic.
For simplicity, we assume that everything is modeled as relationships and nodes. This is the case for attributes such as birthdate: we assume that the date itself is a node connected with a relationship. Additionally, we assume that for every edge $e \in E$ with type $\psi(e) = l$ there exists a reverse edge $e^{-1}$ with $\psi(e^{-1}) = l^{-1}$ to model cases such as presidentOf and hasPresident. The above assumptions do not change the generality of the methods but simplify the notation and the analysis.

We aim at discovering notable characteristics expressed as a set of input query nodes (entities) in relation to their similars.

This intuitive definition entails two questions: (1) what is the set of similars? (2) what are the notable characteristics?

Given a knowledge graph $G = \langle \mathcal{V}, \mathcal{E}, \phi, \psi \rangle$, the set of input nodes, referred to as query set or query in short, is any set $Q \subseteq \mathcal{V}$.

The query is manually provided by the user and therefore considers reasonably small (i.e., at most 10 elements). The first question concerns the definition of a set of similars referred in this work as context nodes. We assume the existence of a similarity function $\sigma : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$ that assigns a high score to nodes that are similar to those in the query set and low otherwise. Given such similarity, the context are the top-$k$ most similar nodes.

Definition 2.1 (Context set). Given a knowledge graph $G = \langle \mathcal{V}, \mathcal{E}, \phi, \psi \rangle$, a query set $Q \subseteq \mathcal{V}$, a similarity function $\sigma : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$, and a parameter $k$, the context set (or simply context) is a set $C \subseteq \mathcal{V}$ such that $Q \cap C = \emptyset$, $|C| = k$, and for each $n_c \in C \land n \in \mathcal{V} \setminus (Q \cup C), \sigma(n_c, n) \leq \sigma(n_c, Q)$.

The second question concerns the notable characteristics. The characteristics are attributes or relationships of a specific node since they implicitly represent a signature of the node itself. We assume the existence of a generic discrimination function $\delta : \mathcal{L} \times \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}^+_0$, which represents how a specific characteristic is discriminative or unexpected comparing two set of nodes. The discrimination function returns 0 if the value is not discriminative. We are now ready to define a notable characteristic.

Definition 2.2 (Notable characteristic). Given a knowledge graph $G = \langle \mathcal{V}, \mathcal{E}, \phi, \psi \rangle$, a query $Q \subseteq \mathcal{V}$, a context $C \subseteq \mathcal{V}$, and a discrimination function $\delta : \mathcal{L} \times \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}^+_0$, a notable characteristic is a relationship $l \in \mathcal{L} \land \psi(e_l) = l \land \phi(e_l) \neq 0$.

The notation $\mathcal{L}_{Q,C} = \{l | \exists x \in Q \cup C, y \in \mathcal{V} \text{ s.t. } (x,y) \in E \land \psi(x,y) = l\}$ denotes the set of edge labels restricted to those found in the edges directly connected to $Q \cup C$.

The general problem we aim to solve is efficiently returning the notable characteristics, given a query, a similarity function and a discrimination function.

Problem 1 (Notable characteristics search). Given a knowledge graph $G = \langle \mathcal{V}, \mathcal{E}, \phi, \psi \rangle$, a query $Q \subseteq \mathcal{V}$, a similarity function $\sigma : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$, and a discrimination function $\delta : \mathcal{L} \times \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}^+_0$, find the set of notable characteristics.

3 A PROBABILISTIC SOLUTION

The problem entails the definition of appropriate $\sigma$ (similarity) and $\delta$ (discrimination) functions. Section 3.1 introduces a graph-principled solution based on random walks for retrieving context nodes, while Section 3.2 describes a probabilistic approach to effectively discover notable characteristics.

3.1 Finding the context

Given the query $Q$, we define a similarity function $\sigma$ to retrieve a set of context nodes. Although many notions of similarity functions have been developed, such as SimRank [4], none seems suitable to our case since they compare two nodes at the time. We devise an algorithm that takes into account edge labels and combines the advantages of random walk and metapath approaches.

In the random walk model, a walker chooses one of the outgoing edges from a node with uniform probability. Motivated by information theoretic notions applied to graphs [10], instead of uniform probability, we favor edge labels with lower frequency. We define $E_l = \{(i, j) \in E | l \in \mathcal{E}, i \in \mathcal{V}, \psi(i,j) = l\}$, the set of edges having label $l \in \mathcal{L}$. The frequency of a label $l$ is the fraction of $l$-labeled edges with respect to the total number of edges. The weighted adjacency matrix is a $|\mathcal{V}||\mathcal{V}|$ matrix, where for each node $i$ and $j$, $A_{ij} = 1 - |E_l|/|E|$ if $(i,j) \in E$ and $A_{ij} = 0$ otherwise.

The PersonalizedPageRank is the vector $p = e^\top \mathbf{A} + (1 - e^\top)v$, where $A_{ij} = A_{ji}/\sum_k A_{jk}$, $e$ is the damping factor, and $v$ is vector considering personalization vector. In our experiments the damping factor is 0.8, in line with previous works. We compute $p$ starting from each node in the query to retrieve the $k$ nodes with the highest score. This is done by setting $v_{|Q|} = 1/|Q|$ for each $n \in Q$. We refer to this baseline as RandomWalk.

The PageRank disregards which type of relationships are involved in the random walk, discarding the valuable information encoded in the surrounding of the query nodes. To this end, we adopt the notion of metapath [6, 12] which generalizes the concept of path. A metapath for a path $(n_1, ..., n_k), n_i \in \mathcal{V}, 1 \leq i \leq t$ is a sequence $(\phi(n_1), \psi(n_1, n_2), ..., \psi(n_{t-1}, n_t), \phi(n_t))$ that alternates node and edge labels along the path.

We mine metapaths as follows. We sample a node in $\mathcal{V} \setminus Q$ uniformly and run a random walk until a query node is reached. The sequence of edge labels $m$ encountered in the random walk is added to the set of metapaths $M$ along with the number of times $c(m)$ the same metapath has been found so far. It has been proved that random walks are effective in mining metapaths [7].

Once the metapaths are retrieved, we compute a score for each node based on the probability that some metapath starting from a query node ends in such node. Given the set of metapaths $M$, $\{n \rightarrow m\}$ is the set of paths from node $n$ to $n'$ matching metapath $m \in M$. The score of a node $n \rightarrow m \in \mathcal{V} \setminus Q$ with respect to node $n \in \mathcal{Q}$ is

$$\sigma(n', Q) = \sum_{m \in M, n \rightarrow m} \frac{|\{n \rightarrow m\}|}{|\{n \rightarrow m' \mid n' \in \mathcal{V} \setminus Q\}|} \Pr(m) \quad (1)$$

where $\Pr(m) = c(m)/\sum_{m \in M} c(m)$ is the probability of choosing metapath $m$. Intuitively, $\sigma$ gives a higher score to nodes that are reachable through frequent metapaths connecting the query nodes or connected through many of these metapaths. Hence, nodes that are reached from infrequent metapaths will have a low score. We refer to this method as ContextXRW.

3.2 Comparing the distributions

We revise the definition of notable characteristics in probabilistic terms. Assume we have computed the distribution of values for each characteristic (i.e., edge label) for both query and context nodes found with the method in Section 3.1. Such distribution of the context represents the expected, or normal behavior, to be evaluated against the notable behaviour of the query set.

Formally, for each characteristic $l \in \mathcal{L}$, we consider two vectors in order to evaluate its notability. The first represents the count of the node labels (e.g., France) connected to a specific edge label (e.g., bornIn). This expresses information about the values in the nodes and can be used to identify cases where different attribute values are relevant. For instance, in the query in Figure 1 all people are European, while in the context half are Europeans and half Asian. We refer to these vectors as instance vectors $I_q(l, C) = (x_1, x_2, ..., x_p)$, $I_q(l, C) = (y_1, y_2, ..., y_q)$ where $x_i$ and $y_i$ are the number of occurrences of node $i$ at the end of an edge labeled $l$ from a node in $Q$ and $C$, respectively. In the example in Figure 1, $I_q(studied, C, Q) = (1, 1, 0)$.
$l_c(studied, C) = (0, 0, 3)$, where the positions in the vector indicate (Physics, Philosophy, Law). Note that both vectors have the same size, so $x_i$ is zero if $i$ appears only in the context.

Similarly, the second vector represents aggregates over the number of occurrences of a specific edge label in the context, which are useful to represent the characteristic "Angela Merkel" has no child, instead of listing the children names. We refer to these vectors as cardinality vectors. \[
K_c(l, C) = (x_1, x_2, \ldots, x_l), \quad \mathcal{K}_c(l, C) = (y_1, y_2, \ldots, y_l)
\]
where $x_i$ and $y_i$ are the number of times a node in $Q$ and $C$ respectively has $i$ edges labeled $l$.

Both vectors can be built by iterating through the nodes in each set and counting the respective occurrences. For a given $l \in L$, this results in two scores $\delta_T^l$ and $\delta_K^l$. The final score $\delta$ is the maximum score between $\delta_T^l$ and $\delta_K^l$. \[
\delta(l, C, Q) = \max(\delta_T(l, C, Q), \delta_K(l, C, Q)) \tag{2}
\]

Many measures have been proposed in statistics to compare two vectors in terms of distributions, such as the Kullback-Leibler (KL) divergence, the $\chi^2$ test, and Earth Mover’s Distance (EMD). However, most of them draw specific assumptions, such as non-zero probabilities, or normality, that are not fulfilled in our case since $I$ and $\mathcal{K}$ have no natural ordering and no distance-function between the values. Therefore, we resort to a more natural multinomial test that better expresses the relationship between our distributions. The multinomial test assumes that a set of observations (the query) is drawn from a multinomial distribution (the context). If the values observed in the query are drawn from the multinomial, than the hypothesis cannot be rejected and the context is marked as non-notable; otherwise, the success of the test tends to denote that the characteristic is notable.

Assume we have a random variable $X_{N, \pi} \sim \text{Mult}(N, \pi)$, with parameters $N$ and distribution $\pi$. We normalize $I_l$ and $\mathcal{K}_c$ to express the probability distributions $\tilde{I}_c = I_c/||I_c||$ and $\tilde{\mathcal{K}}_c = \mathcal{K}_c/||\mathcal{K}_c||$. The significance probability is \[
\Pr_{\pi}(X_{N, \pi} = x) = \sum_{y} \Pr_{\pi}(X_{N, \pi} = y) \quad \text{Pr}_{\pi}(\pi, x) \text{ is the probability of } x \text{ or any equally or less likely outcome being drawn from the probability distribution}.
\]
A difference in distributions is considered significant if the hypothesis is rejected with probability $p > 0.95$.

\[
\text{MT}(%(\pi, x)) = \begin{cases} 1 - \Pr_{\pi}(X_{N, \pi} = x) & \text{if } \Pr_{\pi}(\pi, x) \leq 0.05 \\ 0 & \text{otherwise} \end{cases}
\]

Finally, $\delta$ is defined as $\delta_T(l, C, Q) = \text{MT}(\tilde{I}_c(l, C), I_q(l, C, Q))$ and $\delta_K(l, C, Q) = \text{MT}(\tilde{\mathcal{K}}_c(l, C), \mathcal{K}_q(l, C, Q))$.

## 4 EXPERIMENTAL EVALUATION

We experimentally evaluate our approach on different datasets and show the impact of the parameters on the final results.

### Datasets:
We perform experiments on two real datasets.

- **YAGO** is a large knowledge graph based on Wikipedia, WordNet and Geonames, with 3.3M nodes, 27M edges, 366K node types and 38 edge labels. We downloaded YAGO 2.5 and converted node attributes to edges and attribute values to node labels.
- **LMDB**\cite{randomwalk}: LinkedMDB is a knowledge graph for the movie domain, extracted from the Internet Movie Database (IMDB), with 739K nodes, 1.6M edges, and 18 edge types.

### Experimental Setup:
We implemented our solution in Java 1.8, and ran the experiments on a Intel i5-4210U 1.7 GHz machine with 12GB RAM. All datasets are loaded into Apache Jena triple store. Along our CONTEXTRW described in Section 3.1 for context selection and FINDNC for notable characteristics identification on top of CONTEXTRW, we implement RandomWalk, a baseline for context selection based on Personalized PageRank (see Section 3.1) computed through the power iteration method with 10 iterations and $c = 0.8$.

### 4.1 Evaluating Context selection
We compare the effectiveness of CONTEXTRW with the baseline RandomWalk within different topics. Since no ground truth for finding context nodes given a set of query nodes were available, we generated context nodes via CrowdFlower (https://www.crowdflower.com) for 15 query sets in three domains, namely **politicans, actors, and movie contributors**. For each domain we manually determined 6 entities belonging to the domain and generated queries of increasing size (up to 6 entities). We asked 34 workers to provide a ranked list of related entities given the query, resulting in 7650 entities. From such entities we removed those mentioned only once and obtained 36 to 76 entities per query that are mapped into YAGO.

### Context size
Context size affects the quality of the results, since more context nodes potentially lead to better recall but worse precision. Figure 2a compares RandomWalk and our CONTEXTRW in terms of $F_1$ score at different $|C|$. In all cases, CONTEXTRW performs up to four times better than the RandomWalk, vindicating the effectiveness of our metapath constrained random walk in finding context nodes, while RandomWalk mostly returns nodes that are close to the query nodes, but semantically irrelevant. Quality does not improve for $|C| > 100$ due to a loss in precision. We also experience a lower variance for CONTEXTRW that exploits metapaths for guiding the search.

### Query size
We analyze the performance of the algorithms varying the query size $|Q|$. Figure 2b shows that CONTEXTRW improves in result quality when more query nodes are considered, which means that our method capture semantic relationships between the nodes. On the contrary, RandomWalk is not affected by the size of the query disregarding metapaths.

Figure 3a reports the time to compute the context, showing that RandomWalk is on average up to two orders of magnitude slower than CONTEXTRW, for $|Q| = 5$. Expectedly, CONTEXTRW is faster with larger queries ($<20s$ comprising the database time), since the chance to end up in a query node is larger.

Figure 2c reports the maximum $F_1$ of CONTEXTRW at increasing $|Q|$, comparing YAGO and LMDB datasets within the actors domain. Unsurprisingly, CONTEXTRW performs moderately better than YAGO in LMDB due to the specificity of the dataset; however, YAGO result testifies the generalization ability of CONTEXTRW in larger, more complex datasets.

### Number of paths
The CONTEXTRW algorithm depends on the number of paths. Figure 2d shows the $F_1$ score in relation to the context size and the number of paths. The number of paths does not affect the score; however, as shown in Figure 3b the time increases as the length of the metapaths (and also the number, not reported) increases. Therefore, a reasonable choice for the number of metapaths $|M|$ and maximum length is 5.

### 4.2 Distribution Comparison
Test cases. We show preliminary evidence of the effectiveness of FINDNC with respect to the RandomWalk baseline for context retrieval with the multinomial test. The test case in Figure 4a shows the instance distribution of the query $Q = \{\text{George Clooney}, \}$.
different, for it does not provide explanations on the nodes returned as anomalies, nor such approaches are query-driven.

**Node comparison measures.** Node similarities, typically defined in terms of neighbors, is a centerpiece for community detection, classification, and link prediction. Structural equivalence, such as SimRank [4] defines two nodes similar if the neighbors are similar. Random walk approaches, such as Personalized PageRank [2] and HITS [5] can also be used to find structurally similar nodes. However, node similarities cannot readily explain differences among query nodes and other similar nodes.

**Seed set expansion.** Seed set expansion, or example-based methods, refers to methods that ask the user to provide an initial set of entities or structures and retrieve similar nodes. Seed nodes are used to discover groups of nodes with similar characteristics [6] exploiting the specificity of each node in the seed set. Likewise, seed-based approaches are used to discover dense graph regions [3, 11]. Although these methods provide multiple groups of nodes they cannot properly explain the characteristics and the differences among them; in general, they do not directly compare the query nodes with the others.

**Relevant path summarization.** Our problem is connected to the discovery of metapaths between nodes [8, 12]. Methods have been proposed to automatically learn metapaths from a given seed set [9]. However, metapaths cannot express the lack of an edge (e.g., Angela Merkel has no children), nor they cannot detect notable characteristics: Being born in the same place is notable, only if similar people are born in different places.

### REFERENCES


[12] Yinhao Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Path- sim: Meta path-based top-k similarity search in heterogeneous information network. *PVLDB* 4, 11 (2011), 992–1003.