Visualizing and Exploring Big Datasets based on Semantic Community Detection

Maria Krommyda
School of ECE, NTUA
mariakr@dblab.ece.ntua.gr

Konstantinos Tsitseklis
School of ECE, NTUA
ktsitseklis@netmode.ntua.gr

Verena Kantere
School of ECE, NTUA
verena@dblab.ece.ntua.gr

Vasileios Karyotis
Dept. Informatics, Ionian Univ
karyotis@ionio.gr

Symeon Papavassiliou
School of ECE, NTUA
papavass@mail.ntua.gr

ABSTRACT

The extended use of the RDF model has made available many datasets from heterogeneous sources that are of interest to a wide audience. Their exploration, however, is a highly demanding task requiring extensive training and knowledge of the SPARQL language. In this demo, we present an innovative system, which supports the exploration of large RDF datasets without requiring any knowledge about the RDF or SPARQL. The system is based on a novel algorithm that detects semantically similar communities capitalizing on hyperbolic network embedding and a weighted similarity metric. The detected communities are visualized in a user-friendly way and presented to the user through a two level abstraction interface with a toolbox of exploration functionalities.

1 INTRODUCTION

Many organisations and scientists use the RDF model to share their data. There is currently high availability of linked datasets, that cover a wide range of topics and have a high degree of diversity regarding their size and characteristics [1]. Exploring and visualizing these datasets is a complex task that requires extensive knowledge about RDF and the SPARQL language. Because of their volume and update rate, they require expensive infrastructure for their processing. Therefore, even though the information is of interest to a wide audience, the datasets are, in practice, accessible only to a few data scientists.

In order to allow the exploration of linked datasets by people with no experience with the RDF and the SPARQL language, as well as no access to expensive infrastructure, we need an approach that enables: (a) accessible visualization that is scalable even for very large datasets, (b) exploration that is user-friendly and intuitive. In the following, we discuss these challenges.

Scalable visualization. RDF follows the graph structure, since the entities can be represented as nodes, and the relationships between them as edges of a graph. Systems visualizing datasets compliant with RDF, should use a graph model to represent the data, presenting each entity as a distinct node and their relationships as respective edges. Such systems should be easily accessible through commodity infrastructure and scale efficiently for very large graphs. Actually there is a requirement for efficient and scalable rendering for many devices, such as laptops, tablets and smartphones, and for a high number of simultaneous users.

User-friendly and intuitive exploration. Exploration of the dataset and extraction of relevant information should be user-friendly for users with no knowledge of the RDF model. To this end, the system should offer an easy-to-use toolbox that provides a set of exploration functionalities, such as keyword search, semantic filtering based on labels, path navigation and neighbor retrieval. Also, the system should allow interactive navigation towards the sought information. Finally, the dataset should be explorable through different abstraction levels, allowing the user to determine the degree of detail in the visualized information. Furthermore, exploration should be intuitive for users with little knowledge regarding the information that the specific dataset represents or about what can be of interest to explore.

To this end, many systems use summarization methods [6]. These can be divided in three basic categories: pattern mining, statistical and structural. Pattern mining methods employ aggregations and graph structures to identify trends in the datasets. Due to the strictness of these trends, such methods are ideal for schema identification. Statistical methods provide quantitative results over the data based on targeted queries and available semantic information. Such methods are used for the selection of the proper dataset for the user needs. The structural methods create the summaries based on the graph structure and can be further divided in quotient, which aim to identify equivalent nodes based on an equivalence relation over them, and non-quotient that use other structural measures, such as centrality, to create the summaries. Quotient summaries target indexing and querying, while non-quotient summaries are better suited for visualization and data understanding. Thus, we focus further on them.

The Grouping Nodes on Attributes and Pairwise Relationships (SNAP) [12] method is the most well-known among them. It focuses on the construction of a graph visualization that uses supernodes, nodes that contain multiple nodes of the input graph, to create summarizations based on user input and structural information such as edge values and node connections. The main drawback of this solution is the requirement for the user to select the summarization properties, in order to produce the visualized graph. Such a limitation is hindering for inexperienced users or users that want to explore datasets they are unfamiliar with.

An alternative to summarization, and a promising solution for intuitive exploration of RDF datasets, is community detection. As discussed in [7], community detection has a key role in the analysis of complex networks and the inference of useful insights regarding graph topology. However, although traditional community detection methods are very useful when applied to small networks, they cannot scale for networks of modern size as they rely on heavy computations and require a significant size of main memory. Therefore, they can process networks of up to only a few thousand nodes and edges. Hence, in order to apply community detection to RDF datasets, we need new scalable and efficient algorithms that use persistent memory and data management models to process larger graphs.

Contributions. We propose the demonstration of a robust system that implements a novel visualization technique over a novel community detection algorithm to detect communities that include semantically similar content in large RDF datasets. The system takes as input a semantically annotated dataset, divides it into semantically similar communities using a novel algorithm, stores the communities and their interconnections in a graph database and visualizes the results with respect to the graph structure. The dataset is then presented to the user through a user-friendly interface that offers two abstraction layers, allowing the user to explore both the detected communities and the subgraphs within. The system has strong advantages related to:

Semantic community detection. We have designed and implemented a novel algorithm that detects semantically similar communities [9], by first transforming the RDF dataset into a weighted graph and then employing embedding of the graph in the hyperbolic space. This is a proven "natural" space for embedding large graphs, with a semantic similarity metric aiming to detect semantically similar communities.

Community visualization. The system implements a novel two-level visualization approach: first, the nodes and edges within a community are visualized as an independent two-dimensional graph; second, the connections between communities are visualized to provide a high-level overview.

Scalability. We ensure the scalability of our technique in two ways. First, our algorithm for community detection is the first of its kind to employ a DBMS. We have selected a graph DBMS for indexing and storage of the RDF dataset to facilitate the storage of node coordinates in the hyperbolic space and the computation of the distance for all node pairs. Second, we employ a graph DBMS for storage and indexing of the inner and intra communities graphs, a design decision so that the system can support the retrieval of the information using a client-server architecture. In practice, this ensures that the system can work efficiently on any web browser and device.

Exploration functionalities. We provide an easy-to-use toolbox of functionalities that allows the users to explore the dataset using multiple filtering criteria, and navigation through panning and zooming capabilities [10].

2 SEMANTIC COMMUNITY DETECTION

We propose a new algorithm for semantic community detection. The algorithm takes as input a RDF dataset and pre-processes it in order to map it in a three dimensional (3D) space and store it as a weighted graph. In order to introduce the semantics in the detected communities we need to calculate a weight for pairs of RDF triplets that represents their semantic relation. This is based on a novel metric that encapsulates semantic and lexical similarities. Finally, the new graph is processed by the algorithm we have proposed in [9], which employs hyperbolic space concepts for semantic community detection. Algorithm 1 shows the pseudocode, and in the following we give more details for every step of the algorithm.

Step 1: RDF dataset pre-processing. Each RDF triplet of the input dataset is mapped to one node in a custom 3D space, where each one of the three dimensions correspond to *(subject, predicate, object)*. The nodes of the 3D space are connected with edges, to create a complete graph. This is the representation of the input dataset to the custom 3D space. Next, to ensure the efficiency and scalability of the algorithm for very large datasets, the information is stored and indexed in a graph database where each node has three properties.

Step 2: Weighted graph creation. In order to calculate semantically accurate weights for the graph, we follow a three-step method. First, all the words of the dataset are mapped to semantically similar groups and given a popularity score. Each node has three labels: subject, predicate and object. For each label of the node a semantic metric is calculated. Finally, the metric is used to calculate the weight between two nodes.

Initially, we aim to create groups of semantically similar words. In order to achieve this, each word is examined against all already formed groups in case it exists in one of them. In this case, the word is added to the group. In any other case, the word is examined for lexical similarity with all the words already in each group. If none is found, then, the word is placed in the group that contains a semantically similar word. If no such group exists, then the word is placed in a new group.

As 'lexically similar' we consider two words that share the same lemma. For example, the words 'playing' and 'player' are lexically similar to one other, as well as to the word 'play'. As 'semantically similar' we consider two terms that have common semantic content, based on the likeness of the meaning between them, as defined in dictionaries. Two entities are semantically similar when they are associated with what is commonly refer to as 'is a' semantic relationships which are synonymy, hyponymy and hypernymy [11]. For each group i that has been formed, a score is calculated by dividing the count of words within the group with the total number of words in the dataset.

Definition 2.1. **Popularity Score.** Let $\mathbf{G} = \{\mathcal{G}_1, \mathcal{G}_2, ..., \mathcal{G}_k\}$ be the set of semantic groups, where $\mathcal{G}_i$ for $i = 1, 2, ..., k$ corresponds to the i-th group of words and includes the words $\mathbf{W}_i = \{\mathcal{W}_{i,1}, \mathcal{W}_{i,2}, ..., \mathcal{W}_{i,m}\}$ where each $\mathcal{W}_{i,l}$ for $l = 1, 2, ..., m$ corresponds to the l-th word in the i-th group. Let $\mathbf{C}_i = \{C_{i,1}, C_{i,2}, ..., C_{i,m}\}$ where each $C_{i,l}$ corresponds to the number of times the l-th word of the i-th group is found in the dataset. Then the popularity score for the i-th group is calculated as:

$$popularityScore(G_i) = \frac{\sum_{l=1}^{m} C_{i,l}}{\sum_{i=1}^{k} \sum_{l=1}^{m} C_{i,l}} \quad (1)$$

□

Definition 2.2. **Semantic Similarity.** Each label of the node is split into m words and each word $W_{i,l}$, where $l = 1, 2, ..., m$, is replaced by the popularity score of the group it belongs to, G_i. The sum of the popularity scores divided by the number of words in the label is the semantic metric:

$$semanticMetric(label) = \frac{\sum_{n=1}^{m} popularityScore(G_n)}{m} \quad (2)$$

□

Then, for each pair of nodes the distance between them for each dimension is calculated as well as a total distance between them, the weighted average of the three distance values. The calculated distance is used as the graph weights.

Step 3: Community Detection. The community detection starts with pruning the edges of the complete weighted graph by applying the DMST method [5] to obtain a graph that is dense enough to represent the relationships between nodes and sparse enough to highlight the underlying community structure. This graph is then embedded in the hyperbolic space using Rigel Embedding [15]. In the embedding process, each node is assigned coordinates in the hyperbolic space and these are stored in the database. Based on these coordinates the computation of Hyperbolic Edge Betweenness Centrality (HEBC) follows. The HEBC

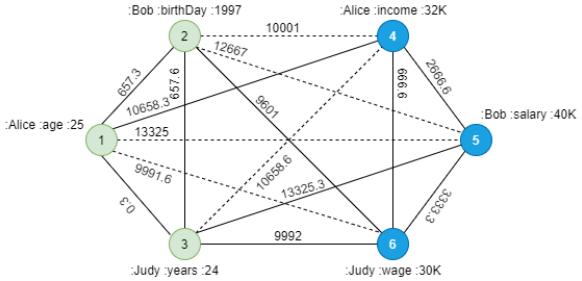

Figure 1: Semantic Community Detection

metric can be considered as the "hyperbolic" analog of the traditional Edge Betweenness Centrality (EBC) metric. The HEBC quantifies for each edge, the amount of shortest paths in the graph that pass through it. In its computations, it uses the hyperbolic distance between nodes. Even though the HEBC values do not always match the nominal EBC values, the ordering of the edges is very similar. The edges of the graph are ranked in descending order and in that order edges are removed in batches, until one of the following occurs. Either the graph becomes disconnected, meaning a new community is discovered, or the maximum number of edges to remove in a single round is reached. Then, if a pre-specified number of communities is reached the algorithm terminates, otherwise the previous process is repeated for the current largest connected component of the graph. The details of the basic version of the hyperbolic-embedding can be found in our previous work[9]. In [13], we have enhanced this work for efficient and scalable processing of very large graphs.

Example. In Figure 1 we present how the proposed algorithm detects the communities on a small RDF dataset. The figure shows six nodes that relate to the ages and salaries of three people, as well as the complete graph created from the first and second step of the algorithm. The distances for each node pair are calculated as follows: the Euclidean distance is used for the numerical properties and the distance based on the semantic metric is used for the rest. This dataset is split into three semantic groups, $G_1 = (Alice, Bob, Judy)$, $G_2 = (age, years, birthDay)$, $G_3 = (salary, income, wage)$, with scores $(0.167, 0.25, 0.25)$. For example, the distance between the triplets (:Alice, :age, :25) and (:Judy, :years, :24) is computed as follows. First, the distance $d('Alice','Judy') = 0$, then $d('age','years') = 0$ and finally the distance $d(25, 24) = 1$. From that it follows that the distance between these two triplets is $\frac{0+0+1}{3} = 0.3$. In the third step, the algorithm performs the DMST, removing edges shown as dashed lines, and then we proceed to the core part of community detection, which results in two communities denoted by the two different node colors. The algorithm discovers a community that corresponds to the ages of the people and a community that contains information about their salaries. It is important to note that a different averaging technique could result in a different clustering, as more importance could be given to one attribute over another. As an example, if we wanted the analysis to focus more on people's names, and thus possibly discover clusters containing information about the attributes of a person, we could use an average with larger weight on the distance between names.

Experimental results. For the Facebook network provided by SNAP library [2], our community detection algorithm obtains a modularity score of 0.59. The Girvan-Newman method resulted in a score of 0.7 but it required almost double the time necessary for our approach. As discussed in detail in [9], this result proves the benefits of our algorithm when applied to real life networks and its the ability of finding high quality clusters.

Algorithm 1: Semantic Community Detection

Input: RDF dataset, distance metric, # communities cm, # spanning trees to join k, embedding parameters $params$, maximum # edges to remove per round $batch$
Output: clusters stored as communities in a Graph DB

```
D ← distances between RDF triplets
D is stored in DB
G ← DMST(D, k)
comm_found ← 0
while comm_found < cm do
    coords ← embed(G, params)
    top_edges ← HEBC(G, coords)
    i ← 0
    while i < batch & isconnected(G) do
        G remove top_edge[i]
    if not isconnected(G) then
        comm_found ← comm_found + 1
        store newly found community in Graph DB
```

3 ARCHITECTURE & FUNCTIONALITIES

As shown in Figure 2, we have developed a server-client architecture that takes as input a RDF dataset, process it using the proposed *Semantic Community Detection Algorithm*, stores the communities to a *Graph Database* and further processes the information through two dedicated modules, the *Inner Community Visualization* and the *Intra Community Visualization*. The processed information is presented to the user through the *Visualization Interface* that it is based on a novel visualization technique [10].

Graph Database Due to the structure of the RDF model and the needs of the *Visualization Interface*, the Neo4j[14] graph database is used for the storage of the dataset. For the *Semantic Community Detection Algorithm* the custom 3D graph is stored in the graph database, but now the dataset has been restored to its initial structure and stored here. Each entity has as properties its unique identifier, the community id it belongs to, the ground truth community, if available, and its coordinates within the 2D graphical representation of the community.

Intra-Community Visualization The *Intra-Community Visualization* uses the Scalable Force Directed Placement algorithm [8] for the graphical representation of the entities and their relationships that belong in each community in the 2D space. This algorithm was selected as it allows the parameterization of many attributes of the output graph including the overlapping percentage and the size of the nodes.

Inter-Community Visualization The *Inter-Community Visualization* creates an overview of the dataset by connecting the communities in a super-graph. Two communities, i and j are super-nodes that are connected with super-edges that contain information about the real edges that connect pairs of nodes (n_i, n_j), where n_i belongs to i and n_j to j. In addition, each super-node contains information about the content of the respective community. It includes the three most prevalent nodes and edges as they are calculated based on a popularity score, calculated as follows: first, the frequency of appearances of each label in the super-node is calculated; the counter is then converted to the percentage of the label appearances in the overall dataset. As an example, if the label 'happiness' appears in the dataset 1000 times and the label 'hate' just once, then for a super-node that includes the label 'hate' once and the label 'happiness' 700 times, the score for the first would be 1 and for the second 0.7.

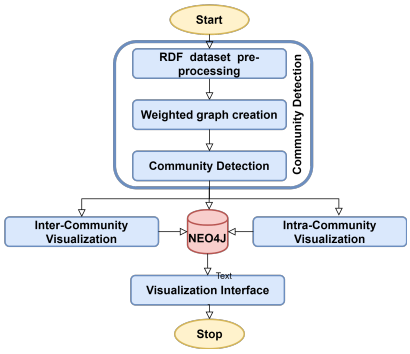

Figure 2: System Overview

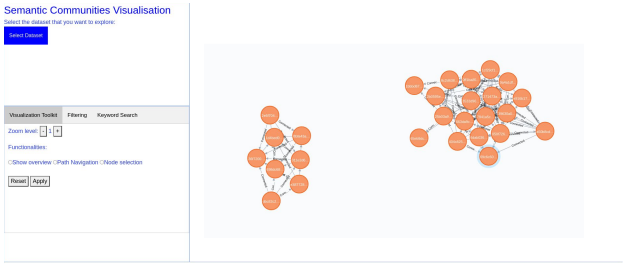

Figure 3: User interface

This way, nodes and edges that are important to the communities but also contribute to their differentiation from the others, are included. Similarly, each super-edge has a list with the three most important edges connecting the two super-nodes.

Visualization Interface The information is presented visually through a user interface in two abstraction levels. The high level presents the interconnections between communities, as well as a brief summary regarding the context of each community. The second layer presents as a graph all the triplets of information of the community and as meta-information connections with other communities. In order to allow the user to explore the input dataset and the detected communities a series of functionalities are provided through the interface. Employing the indexing and querying capabilities of the graph database, the functionalities are provided in real-time. One key functionality is the zoom support, where the user can use the mouse wheel or the buttons at the functionality panel to visualize the graph in different levels of detail. In addition, dedicated controls allow the user to navigate between the two abstraction layers, focusing either on the overview of the communities or visualizing its information. Also, the visualization panels for the two abstraction layers are interactive and responsive to user requests. The user can browse information within them using panning and scrolling actions.

The system, also, enables the user to view the information within the communities using multiple filtering and aggregation criteria either independently or at the same time. For example the user can isolate a specific edge type as well as nodes with a given node degree. To further support the exploration of the dataset, the user can locate information through keyword search. A term is matched against node and edge labels, and the result is presented as an interactive list. Figure 3 shows the interface of the system, the dataset selector, the functionality toolkit and the graph visualization. The graph panel depicts two of the communities based on the user selections.

4 DEMONSTRATION

We will demonstrate the prototype system that implements the proposed technique using three datasets. The audience will be able to explore the datasets through the visualization of semantic communities and compare our results with the ground truth.

Scenarios A: Semantic communities overview. The first scenario will be based on the Wikidata[3], which is a free and open knowledge base, mainly the central storage for the structured data of Wikipedia. The dataset contains a plethora of diverse information that can be used for multiple analysis based on topic, source or category. Initially, the users will be able to access multiple visualized overviews of semantic communities focusing on different attributes of the RDF entities. The overviews will focus on popular topics such as persons and places. Next, the users will delve into the fine exploration of a community graph. Given the volume and diversity of the information available in this dataset, examples of the customizable filtering and aggregation functions will allow the users to focus on specific information. The users may discover for example, based on their interests collaborations between scientists, artists or countries. Finally, the users will be encouraged to follow paths between entities aiming to identify patterns and information exchanges.

Scenarios B: Ground truth comparison. For the third scenario the DBLP and Amazon datasets will be used, which are offered with ground-truth by SNAP [4]. The DBLP dataset provides open bibliographic information on major computer science journals and proceedings. The RDF dataset extracted includes information about co-authors and the semantic communities are formed based on the journal or conference that a paper was published. The Amazon dataset contains a product co-purchasing network. The users will be also provided with statistics about the comparison of the detected communities with the ground truth.

Acknowledgement. The research work of Mr. Tsitseklis was co-financed by Greece and the European Union (European Social Fund-ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning", in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research" (MIS-5000432), implemented by the Greek State Scholarships Foundation.

REFERENCES

[1] 2020. https://lod-cloud.net/.
[2] 2020. https://snap.stanford.edu/data/ego-Facebook.html.
[3] 2020. https://www.wikidata.org/wiki/Wikidata:Main_Page.
[4] 2020. https://snap.stanford.edu/data/#communities.
[5] Miguel Carreira-Perpiñán and Richard Zemel. 2004. Proximity Graphs for Clustering and Manifold Learning. *Adv. Neural Inf. Process. Syst.*
[6] Šejla Čebirić, François Goasdoué, Haridimos Kondylakis, Dimitris Kotzinos, Ioana Manolescu, Georgia Troullinou, and Mussab Zneika. 2019. Summarizing semantic graphs: a survey. *The VLDB Journal* (2019).
[7] S. Harenberg, G. Bello, La Gjeltema, S. Ranshous, J. Harlalka, R. Seay, K. Padmanabhan, and N. Samatova. 2014. Community detection in large-scale networks: a survey and empirical evaluation. *Computational Statistics.*
[8] Tomihisa Kamada, Satoru Kawai, et al. 1989. An algorithm for drawing general undirected graphs. *Information processing letters* (1989).
[9] Vassilis Karyotis, Konstantinos Tsitseklis, Konstantinos Sotiropoulos, and Symeon Papavassiliou. 2018. Big Data Clustering via Community Detection and Hyperbolic Network Embedding in IoT Applications. In *Sensors.*
[10] Maria Krommyda, Verena Kantere, and Yannis Vassiliou. 2019. IVLG: Interactive Visualization of Large Graphs. In *IEEE ICDE.*
[11] Dekang Lin et al. 1998. An information-theoretic definition of similarity.. In *Icml*, Vol. 98. 296–304.
[12] Yuanyuan Tian, Richard A Hankins, and Jignesh M Patel. 2008. Efficient aggregation for graph summarization. In *ACM SIGMOD.*
[13] K. Tsitseklis, M. Krommyda, V. Karyotis, V. Kantere, and S. Papavassiliou. 2020. Scalable Community Detection for Complex Data Graphs via Hyperbolic Network Embedding and Graph Databases. *IEEE TNSE* (2020).
[14] Jim Webber. 2012. A programmatic introduction to neo4j. In *Conference on Systems, programming, and applications.*
[15] Xiaohan Zhao, Alessandra Sala, Haitao Zheng, and Ben Y. Zhao. 2011. Efficient shortest paths on massive social graphs. In *IEEE Collaborative Computing.*