iTopic: Influential Topic Discovery from Information Networks via Keyword Query

Jianxin Li
University of Western Australia
Perth, Australia
jianxin.li@uwa.edu.au

Chengfei Liu
Swinburne University of Technology
Melbourne, Australia
cliu@swin.edu.au

Lu Chen
Swinburne University of Technology
Melbourne, Australia
luchen@swin.edu.au

Zhenying He
Fudan University
Shanghai, China
zhenying@fudan.edu.cn

Amitava Datta
University of Western Australia
Perth, Australia
amitava.datta@uwa.edu.au

Feng Xia
Dalian University of Technology
Dalian, China
f.xia@ieee.org

ABSTRACT

The rapid growth of information networks provides a significant opportunity for people to learn the world and find useful information for decision making. To find influential topics in a given context, instead of searching widely over the whole information network, normally it is wise to find the related communities first and then identify the influential topics in those communities. In this demonstration, we present a novel framework to compute the correlated sub-networks from a large information network such as CiteSeerX based on a user's keyword query, and to extract the *influential topics* from each correlated network. To help users understand the influential topics as a whole, we utilize a word cloud to represent the discovered topics for each correlated network. As such, multiple word clouds can be generated for different correlated networks, by which users can easily pick up their interested ones by reading the visualized topic descriptions over word clouds. To determine the sizes of different terms in a word cloud, we introduce a scoring scheme for assessing the influence of these terms in the corresponding networks. We demonstrate the functionality of our influential *topic* system, called *iTopic*, using the CiteSeerX information network data.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems

General Terms

Algorithms

Keywords

Information Network, Topic Discovery, Keyword Query

WWW'17 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054719

1. INTRODUCTION

In recent years we have witnessed a rapid growth of information networks in a wide spectrum of application domains. Making sense of the big data in these information networks finds tremendous applications and could bring huge benefits to the society, enterprises, and individuals. Indeed, data collected from Twitter, Facebook, CiteSeerX, LinkedIn etc. can be used to identify and track influential topics and help make important decisions. These networks are typically modelled as large graphs with nodes representing entities and edges depicting relationship between entities [1]. One way to retrieve interesting information from large graphs is via keyword queries, known as *keyword search over graph data*. Most of existing works return all or top-k minimally matched subgraphs as individual results to a user [13]. This will either overwhelm the user as too many results may be returned or provide insufficient or incomplete information as the granularity of the results is too fine.

Often a user prefers to explore useful information from a bigger community consisting of multiple individual results with high correlations among them. In fact, persons' opinions towards a topic in a large and closely connected community are more believable than those individual ones or those in a loosely connected community. This is because a more general picture can be obtained from a large and closely connected community thus more conclusive information can be drawn. For example, a PhD student Anna wants to find interesting research topics in the area of *XML Database*. She may issue a keyword query: q{xml, database} in CiteSeerX, which normally returns a list of individual publications containing the keywords. Then, she may spend significant time on going through the list of the publications and finding out influential topics by applying additional analysis, such as clustering the closely linked publications into groups (denoted as communities). It would be ideal to Anna if the result returned are topics extracted from those communities related to the area of *XML Database*, and each extracted topic is ranked higher in terms of its impact within a community or across all communities. This motivates our work to discover the related communities first and then extract those highly impacted topics (denoted as influential topics).

Existing works on discovering dense subgraphs [4, 2] may be applicable to computing keyword query related communities. However, all these works concentrate their research on the efficiency of discovering the densest subgraph, or the top-k dense subgraphs from a graph. There is no existing work to allow users to find their interested dense subgraphs with a search request (e.g., a keyword

query). In addition, Li et al. in [11] investigate the personalized influential topic search in social networks. But it only focuses on the influence of the users in the social networks. Fakas et al. in [6, 7, 8, 9] study the size-l object summaries to answer users' keyword queries in relational databases. But these works didn't consider the structure information in evaluating the importance of keyword search results.

This demonstration presents *iTopic*, a system to discover influential topics from information networks when a user issues a keyword query representing the area of interest. Several prototype systems have been proposed to detect topics in Information Retrieval. But all of these works have to depend on the external knowledge, e.g., query log [3], #hashtag as a training dataset [12], and word knowledge [5].

What distinguishes *iTopic* from other topic detection systems is that *iTopic* does not depend on any domain knowledge, and excludes the noise information by using the concept of community. That is to say, closely connected communities normally hold high probabilities to have their own topic areas. Given a query, we first compute the closely connected communities and then detect the corresponding topics. Further, the topics are selected based on the influence in their communities. To make it easy to compare different topics in different communities, we utilize word clouds to represent the influential topics in different communities, i.e., each computed community will produce a word cloud. In these word clouds, the font size of the terms depends on the influential score of the terms.

The remainder of the paper is structured as follows. In Section 2, we detail our methods to first discover communities from keyword queries and then identify influential topics from the returned communities. Afterwards, we briefly discuss the system architecture in Section 3 before outlining the demonstration scenarios in Section 4.

2. INFLUENTIAL TOPIC DISCOVERY

iTopic performs influential topic discovery in two main steps. First, it discovers influential communities for a given keyword query [10]. Second, it selects influential topics based on a scoring function and presents the topics as a word cloud.

2.1 Discovering Correlated Networks CNs

To find influential communities of a keyword query, we first generate all individual keyword query results and then compute the correlated networks of these individual results.

Generating γ-bounded Keyword Matched Subgraphs: Given a keyword query representing a context, we define a query result as a γ-bounded keyword matched subgraph consisting of a set of keyword matched nodes, the corresponding connection nodes, and connection edges. It satisfies three conditions:

(1) There is at least one occurrence of each given keyword matched node in the subgraph;

(2) It keeps all the shortest paths of any two keyword matched nodes in the subgraph;

(3) the distance of each shortest path of any two keyword matched nodes in the subgraph is no more than user-specified hop number γ.

To reduce the high computational cost of generating γ-bounded keyword matched subgraphs for a keyword query against a graph G, we devise a novel tree data structure. The key idea is to record the shortest paths of graph nodes up to the given maximal number Γ of hops in the tree T. To transform graph G to its tree structure T, firstly, we copy the directly connected-edges from G to T, which guarantees the 1-Hop Correctness. And then, we compare the connected edges with 2 hops between G and T. If some edges do not appear in T, then we need to copy them in T such that all the edges that are connected within 2 hops in G should also appear in T, which guarantees the 2-Hop Correctness. Similarly, we can check the connected edges up to Γ hops, which can guarantee the Γ-Hop Correctness. The transformation does not depend on any query, which can be done offline. As such, we have that given a graph G, its pre-built tree T can guarantee to correctly answer any keyword queries with the given hop number $\gamma \leq \Gamma$ where γ is a user specified hop number value while Γ is the maximal hop number bound set by system administrators.

With the help of transformed tree data structure, it becomes easy to compute the γ-bounded keyword matched subgraphs. The basic idea is to find a set of subtrees where each subtree should have a maximal covering of keyword nodes bounded by γ. Here, maximal covering means to include keyword nodes as many as possible, but the maximal distance of any two keyword nodes is bounded to γ. To efficiently find the set of subtrees, we can read the tree nodes in a top-down manner and incrementally generate the subtree candidates where each candidate is represented by its node-set. Although it may still produce a few duplicate candidates due to the existence of copied nodes, the number of duplicate candidates is much less than that of the straightforward method that runs the breadth-f rst traversal algorithm for each keyword node up to the hops. Here, the straightforward method suffers from a large number of repeated scans on the graph for each keyword node, a large number of unqualified candidates produced, and unnecessary cost for identifying the shortest path of any two nodes in every candidate.

Generating Correlated Networks: Instead of finding individual results of a keyword query, we are more interested in finding the communities that have high impact on the context denoted by the keyword query. We define such a community as a correlated network with multiple γ-bounded keyword matched subgraphs as components. The network is a correlated one if and only if all its components are correlated to each other. The correlation of any two components G_1 and G_2 can be measured by

$$\frac{\sum\{weight(v, G_1) * weight(v, G_2) | v \in G_1' \cap G_2'\}}{|G_1' \cup G_2'|} \quad (1)$$

where G_1' contains the nodes in G_1 and the neighbour nodes of G_1, and G_2' consists of the nodes in G_2 and the neighbour nodes of G_2. A correlation ratio can be set and adjusted to express the tightness of the community.

In our def nition of the correlated network, we consider both the nodes in components and their close neighbour nodes with different weights. That is to say, an overlapped node appearing in both components should contribute more to the correlation value of the two components than an overlapped node appearing in only one of the components or outside both components. Therefore, in query evaluation, we can assign 1 as the weight of each node v in G_1 because v is an inner node with regard to G_1. Based on the diffusion weighted model, each outlinked node v' in $G_1' \setminus G_1$, in regard to G_1, can be weighted by

$$weight(v', G_1) = 2^{-minDist(v', G_1)} \quad (2)$$

where $minDist(v', G_1)$ is the minimal distance of the shortest paths from the outlinked node v' to any node in G_1.

We employ the Shingling technique to efficiently approximate the measure of correlations among the γ-bounded keyword matched results where the results can be represented by their corresponding node-sets [10]. Each node-set can be rewritten into a constant-size fingerprint. As such, these results can be compared by simply com-

paring their fingerprints. As we weight the nodes in the components and their close neighbour nodes differently, we adapt the Shingling technique and propose a weighted Shingling algorithm that classifies nodes into different subsets based on their weight values. We then develop a merge-sort-based approach to find the correlated networks based on the generated shingles and their weights [10].

2.2 Influential Topic Selection and Representation

The topics in a correlated network can be represented by a set of terms extracted from the network. In this section, we will address three problems:

(1) How to formally define a scoring function to select the influential terms from a correlated network?

(2) How to represent the influential topics in a friendly visualized way?

(3) How to rank the correlated networks based on their influential topics?

Given a correlated network, the influence score of a term depends on the term frequency of the term in the network, and the correlated degree of the nodes with the term to other nodes in the network. As such, the influence score of a term t_i in a correlated network c can be defined as:

$$iScore(t_i, c) = \sum_{t_i \in v \in c} tf(t_i, v) \times rDegree(v, c) \qquad (3)$$

where $rDegree(v, c)$ is the number of neighbors that node v has in c, and $tf(t_i, v)$ represents the frequency of t_i in v. Equation 3 satisfies the observation for discovering influential topics, i.e., the more a term appears in a network and the higher the correlations of the nodes are in the network, the more influence the term can make to the network.

To address the second problem, we adopt the *word cloud* as a visual representation to show the selected terms as influential topics in a correlated network. The different influences of words are shown with different sizes, which can be calculated based on the iScore values of the term in Equation 3. For each correlated network, we generate one word cloud. Users can quickly go through the word clouds and check which word clouds contain their interested information. We provide functions allowing users to get statistical information about influential topics and details about word clouds.

Regarding the third problem, we need to provide users the discovered correlated networks in the descending order based on their influential topics. That is to say, the word cloud with more important influential topics will be ranked at the higher position, by which it is more likely for users to see the top word clouds and choose the corresponding correlated networks. To do this, the score of a correlated network (or the score of its word cloud) is defined as:

$$score(c_j) = \sum_{t_i \in c_j} iScore(t_i, c_j) \times log \frac{|C|}{|\{c_x \in C : t_i \in c_x\}|} \qquad (4)$$

where C is the set of all correlated networks found. Equation 4 satisfies the observation that if a correlated network contains too many common terms, then these terms in the correlated network have less probability of producing highly influential topics. As such, the position of the correlated network would be degraded in the order of the influential list.

3. SYSTEM OVERVIEW

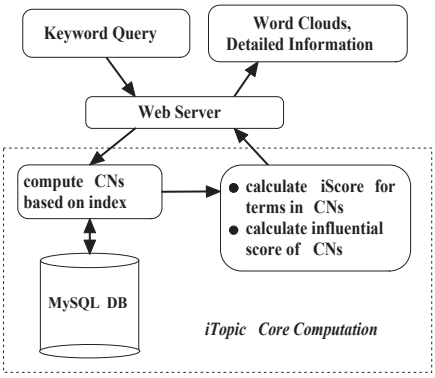


Figure 1: System Overview of iTopic

Figure 1 shows the system overview of *iTopic*. It is implemented in *Java SE* using *Tomcat Server* and *MySQL* database server and consists of the following components:

(1) The Component of Data Preparation and Storage: We extract the title, authors and published venue of each publication as a node, and take the citations as edges of the node to the other nodes, from CiteSeerX dataset. The extracted graph data are stored in MySQL database. The node inverted list index and graph2tree index are built offline.

(2) The Core Computational Component: Most of computational cost is spent on discovering the correlated networks via the given keywords, calculating influential scores of terms in the correlated networks, computing the influential scores of correlated networks based on the scores of terms. In the process of generating word cloud, we will make tokenization, word cleaning based on stop word list, and noise word removal. Since the correlated networks discovered definitely contain the given keywords, we do not include the given keyword in the generated word clouds.

(3) The Component of Web Server: It is used to handle multiple queries and the same query for multiple users. For a query, its related correlated network data information will be temporarily maintained at the web server in the format of JSON, by which users can easily see the nodes and their relationships in a correlated network by simply clicking his interested word cloud.

(4) The Component of Visualizer. At the web browser, users only need to type keyword queries for searching the query-related topics from CiteSeerX data. After the query is processed, a set of word clouds will be presented to the users. Users are allowed to learn more statistical information about topics and details about the correlated networks by playing with the word clouds on the web browser page.

4. DEMONSTRATION

In this section, we show the ability of *iTopic* to discover query-based influential topics from the bibliographic dataset. We download the CiteSeerX dataset and transform it into undirected graph data, which will be taken as an example application of finding query-related research topics in this demonstration. *iTopic* can be

applied to other graph datasets such as Twitter and Facebook for influential topic discovery.

Figure 2 shows the main interface of *iTopic*, which is divided into three parts. The top part includes the search box that allows users to input their keyword queries and start the search by clicking the search button. Two adjustable parameters are provided to allow users to change the grouping criteria. The first parameter *keyword hop number* specifies the γ -bound value, which can be used to control the tightness of the areas of interest represented by the set of keywords. The second parameter *correlation ratio* can be used to control the tightness of the correlated networks. The middle part is used to show the generated word clouds for different correlated networks. The bottom part is reserved to display the publication information.

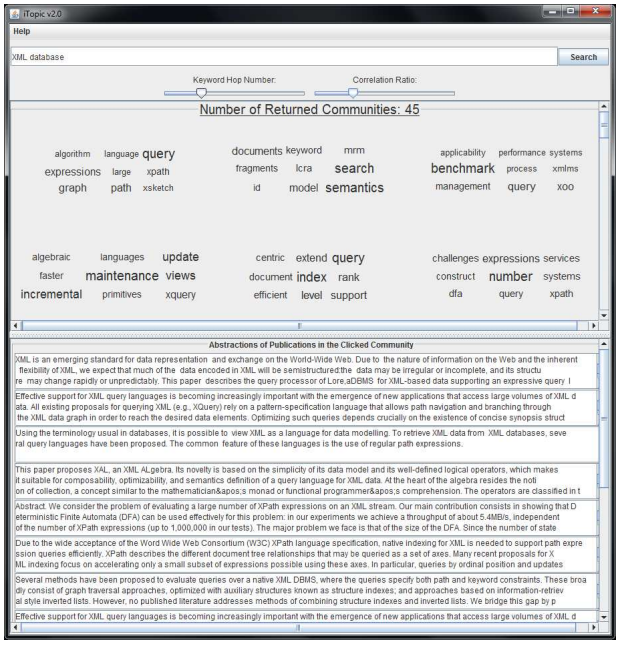
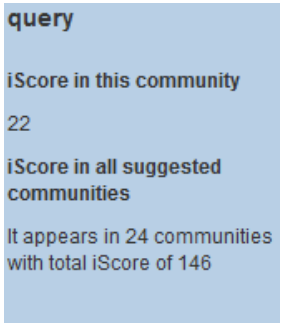
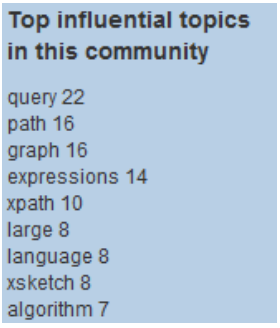


Figure 2: Interface of iTopic

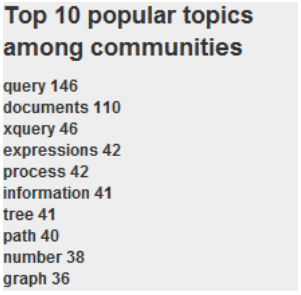
To demonstrate how *iTopic* helps users discover influential topics, let us come back to the query example discussed in Section 1. When Anna inputs keywords "XML database", uses the default settings for the parameters and starts the search in the top part of the interface, 45 communities will be returned in the form of word clouds. As shown in Figure 2, 6 word clouds representing the top 6 out of 45 communities are displayed in the sliding window in the middle part of the interface. In each word cloud, some top ranked topics are displayed with different font sizes, where a bigger font size shows the bigger influential score of the topic. Anna may be interested in *querying* XML databases so she would like to know more about this topic. In *iTopic*, she may move the mouse over to "query" in the first cloud, then information about this topic will be shown in Figure 3(a), which shows the influential score of the topic "query" in this cloud as well as in all the clouds. Anna may also want to know the publication and statistical details of the first cloud. In *iTopic*, she can simply click any place in the first cloud to find the descriptions of the list of publications contained in this cloud as shown in the sliding window of the bottom part of the interface. She can also move the mouse over the area of the first cloud (not over a particular word) to find the top ranked influential topics in this cloud as shown in Figure 3(b). Anna would also like to check the other influential topics related to the area of XML database. In *iTopic*, she may simply click the link at the top of the middle part, i.e., "Number of Returned Communities", then the statistical information about top ranked popular topics across all communities and top ranked hot topics in a single community will be shown in Figure 3(c) and Figure 3(d), respectively. The former shows the popularity of a topic in all communities about the area of interest while the latter shows the degree of intensive discussion of a topic in a single community. From the above explorations, Anna may end up with an interesting topic for her PhD study.



(a) Mouse over a topic



(b) Mouse over a community



(c) Inter-community statistics

(d) Intra-community statistics

Figure 3: Screenshots of statistics for the search

However, Anna may explore other alternatives. To help with Anna with further exploration, *iTopic* allows her to adjust the parameters to attempt different ways of grouping areas and communities, possibly leading to different influential topics. In the current version of *iTopic*, the range of keyword hop number parameter is set from 0 to 7, with 2 as the default. The smaller this parameter is set, the tighter the area. The range of correlation ratio parameter is set from 0 to 1, with 0.25 as the default. The smaller this parameter is set, the looser the community.

5. ACKNOWLEDGMENTS

This work was supported by the ARC Discovery Projects under Grant Nos. DP160102114, DP160102412, and DP170104747 . The work is also partially supported by the NSFC under Grant No.61370080 and No.61170007, and the Shanghai Innovation Action Project under Grant No.16DZ1100200.

6. REFERENCES

[1] C. C. Aggarwal and H. Wang. Graph data management and mining: A survey of algorithms and applications. In *Managing and Mining Graph Data*, pages 13–68. 2010.

[2] B. Bahmani, R. Kumar, and S. Vassilvitskii. Densest subgraph in streaming and mapreduce. *PVLDB*, 5(5):454–465, 2012.

[3] Z. A. Bawab, G. H. Mills, and J. Crespo. Finding trending local topics in search queries for personalization of a recommendation system. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 397–405, 2012.

[4] J. Chen and Y. Saad. Dense subgraph extraction with application to community detection. *IEEE Trans. Knowl. Data Eng.*, 24(7):1216–1230, 2012.

[5] Z. Chen, A. Mukherjee, B. Liu, M. Hsu, M. Castellanos, and R. Ghosh. Discovering coherent topics using general knowledge. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 209–218, 2013.

[6] G. J. Fakas, Z. Cai, and N. Mamoulis. Size-l object summaries for relational keyword search. *PVLDB*, 5(3):229–240, 2011.

[7] G. J. Fakas, Z. Cai, and N. Mamoulis. Versatile size-l object summaries for relational keyword search. *IEEE Trans. Knowl. Data Eng.*, 26(4):1026–1038, 2014.

[8] G. J. Fakas, Z. Cai, and N. Mamoulis. Diverse and proportional size-l object summaries for keyword search. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pages 363–375, 2015.

[9] G. J. Fakas, Z. Cai, and N. Mamoulis. Diverse and proportional size-l object summaries using pairwise relevance. *VLDB J.*, 25(6):791–816, 2016.

[10] J. Li, C. Liu, and M. S. Islam. Keyword-based correlated network computation over large social media. In *ICDE*, 2014.

[11] J. Li, C. Liu, J. X. Yu, Y. Chen, T. K. Sellis, and J. S. Culpepper. Personalized influential topic search via social network summarization. *IEEE Trans. Knowl. Data Eng.*, 28(7):1820–1834, 2016.

[12] X. Meng, F. Wei, X. Liu, M. Zhou, S. Li, and H. Wang. Entity-centric topic-oriented opinion summarization in twitter. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 379–387, 2012.

[13] J. X. Yu, L. Qin, and L. Chang. Keyword search in relational databases: A survey. *IEEE Data Eng. Bull.*, 33(1):67–78, 2010.