Semantic query expansion method based on pay-as-you-go fashion for graph model

Nan Wu and Ying Pan*
School of computer and information engineering, Nanning Normal University, Nanning, Guangxi, 530100, People’s Republic of China
*Corresponding author’s e-mail: panying@nnnu.edu.cn

Abstract. The search requirements of users are usually vague and unclear, and the traditional keyword query method is difficult to obtain satisfactory query results. Therefore, the semantic query expansion technology uses semantic information to modify and extend the initial search requirements of users, to obtain more comprehensive and accurate query results. However, the construction cost of the extended query graph is high, and the existing research is difficult to adjust and optimize the user's query demand in time, resulting in a poor query effect. To solve the above problems, this paper proposes a semantic expansion method based on pay-as-you-go fashion for graph model: Firstly, the initial query graph is constructed according to the user's search requirements, and the semantic similarities between the search requirements and knowledge bases are calculated; Then, the extended sets of attribute-values and edges are generate by sorting the similarity value in descending order; Finally, all or part of the elements in the extended set are combined to generate the extended query graph. This method can dynamically adjust and optimize the initial query based on the relevant semantic information in time, and improve the query efficiency and effect.

1. Introduction
Traditional information searches technology uses keyword queries to obtain the information containing keywords. Due to the lack of semantic support from the search engines, it is difficult to obtain implicit information that does not contain keywords but is semantically related to keywords [1]. In addition, due to the lack of cognition and expression, users' search requirements are often vague and unclear. In order to obtain appropriate query results, it is necessary to analyse and modify the initial search requirements from the relevant semantic of background knowledge and search content.

To solve these problems, semantic query expansion technology is usually used [2]. Semantic query expansion refers to the synonym, homonym, and hierarchical relationship of the user's initial query words as extended phrases to synthesize the new query, and then, the search engine processes the new query and return more accurate results.

At present, the research of semantic query expansion has achieved a lot of results. To avoid repeated calculation, Venetis et al. [3] proposed to extend the previously calculated initial queries method by decomposing the query about atoms first and then incrementally processing each atom. To help develop the KBC (Knowledge Base Construction) system, Shin et al. [4] proposed an incremental reasoning method based on sampling and variational techniques. Kang et al. [5] proposed an incremental optimization method guided by semantic rules for the periodic query scenario of the data warehouse. By extending the query syntax, users can describe the repetition period and increment table of the query, and the optimized incremental query plan can be executed on major distributed computing frameworks.
(such as Map-Reduce). In addition, Yuan et al. [6] proposed dynamically to expand the query scope of feedback information and puts forward the retail commodity search and recommendation method that embeds weighted TF-IDF (Term Frequency-Inverse Document Frequency) in keywords. However, the existing semantic query expansion methods still have a large room for improvement [7,8]. Most of the research lacks support for dynamic data search, and it is difficult to adjust and optimize users’ searches requirements in time, resulting in poor query results.

Aiming at the above problems, this paper proposes a semantic query expansion method based on pay-as-you-go fashion [9], which is a data management style for mass, heterogeneous and dynamic data. Our method can dynamically optimize and extend the search demand of graph data according to semantic information, thus improving the efficiency and effect of the semantic query.

2. Semantic query expansion based on pay-as-you-go fashion

2.1. Generation of Multi-dimensional Semantic Expansion Set

We define the general graph model \( G \) as follows:

**Definition 1.** Graph model \( G \) consists of nodes and edges, denoted as \( G = (N, E) \), where \( N \) is the node set \( \{N_1, \ldots, N_k\} \), \( N_i \) is composed of attribute-value pairs, denoted as \( N_i = \{(a_{i1}^j, v_{i1}^j), \ldots, (a_{in}^j, v_{in}^j)\} \). Let \( A \) be the set of attributes, and \( V \) be the set of values, then there is a property-value pair \( (a_k^j, v_k^j) \in A \times V \), \( 1 \leq k \leq n \). \( E \) is the set of labelled edges, and denoted as \( (N_i, N_j, L) \), \( N_i, N_j \in N \) and \( i \neq j \). \( L \) represents the label of an edge.

The generation steps of the multi-dimensional semantic expansion set are shown in Figure 1:

1) Based on the concept/relation of user search requirements, construct an initial query graph on a graph model \( G \).

2) Compare the concept/relation of the search requirement with the existing concept/relation of semantic information such as related ontology, knowledge graph, Wikipedia, etc. Then calculate the coarse-grained semantic similarity functions \( T_{\text{sim}}(N_i, N_j) \) of the node vocabulary of the two.

The node vocabulary mainly includes the node name and the vocabulary composed of attribute-value pairs. \( T_{\text{sim}}(N_i, N_j) \) compares node names and attribute-value pairs as a whole, which is a coarse-grained comparison of node vocabulary similarity. It is calculated based on the cosine similarity formula as follows:

\[
T_{\text{sim}}(N_i, N_j) = \frac{\Sigma_{k=1}^{h} T_{ik} T_{jk}}{\sqrt{\Sigma_{k=1}^{n} T_{ik}^2} \sqrt{\Sigma_{k=1}^{n} T_{jk}^2}}
\]  

(1)

Where, nodes \( N_i \) and \( N_j \) respectively represent concept nodes related to search requirements and concept nodes in ontology, knowledge graph or Wikipedia, etc. \( 0 < i \leq n \), \( 0 < j \leq m \), \( n \) and \( m \) respectively represent the number of nodes that need to be compared in the search requirements and the number of nodes to be compared in ontology, knowledge graph or Wikipedia. \( w_l \) is the weight, \( w_l \in [0,1] \), \( \Sigma_{l=1}^{2} w_l = 1 \), \( T_i \) and \( T_j \) respectively represent the word frequency of the words corresponding to node \( N_i \) and \( N_j \), \( h \) is the number of total words.
3) If the coarse-grained semantic similarity functions \( T_{sim}(N_i, N_j) \geq t1 \) (threshold), then calculate the fine-grained semantic similarity function \( T'_{sim}(N_i, N_j) \). Otherwise, go to step 2 to compare the similarity with the next node.

In order to judge the similarity more accurately, we define the fine-grained node vocabulary similarity function \( T'_{sim}(N_i, N_j) \) as follows:

\[
T'_{sim}(N_i, N_j) = s_1 \times M_{sim}(N_i, N_j) + s_2 \times A_{sim}(A_i, A_j) + s_3 \times V_{sim}(V_i, V_j) + s_4 \times AV_{sim}(a_i, v_i, a_j, v_j)
\]  

(2)

Where, \( s_i \) is the weight, \( s_i \in [0,1], \sum_{i=1}^{4} s_i = 1 \), \( M_{sim}(N_i, N_j) \) is the node name similarity function, \( N_i, N_j \) are the names of nodes \( N_i \) and \( N_j \) respectively. \( A_{sim}(A_i, A_j) \) is the attribute similarity function, \( A_i \) and \( A_j \) are the attribute set of nodes \( N_i \) and \( N_j \) respectively, such as \( \{a_{i1}, a_{i2}, \ldots\}, \{a_{j1}, a_{j2}, \ldots\} \). \( V_{sim}(V_i, V_j) \) is the value of the similarity function, \( V_i, V_j \) are the values of \( A_i \) and \( A_j \) respectively, such as \( \{v_{i1}, v_{i2}, \ldots\}, \{v_{j1}, v_{j2}, \ldots\} \). \( AV_{sim}(a, v_i, a, v_j) \) is the attribute-value pair similarity function, \( (a, v_i) \) and \( (a, v_j) \) are the attribute-value pairs of \( N_i, N_j \) respectively, such as \( \{(a_{i1}, v_{i1}), \ldots\}, \{(a_{j1}, v_{j1}), \ldots\} \).

In order to improve the calculation efficiency of similarity, the following method is proposed: first calculate the coarse-grained similarity \( T_{sim}(N_i, N_j) \), If the threshold \( t1 \) is exceeded, then calculate the fine-grained similarity \( T'_{sim}(N_i, N_j) \).
4) If $M_{\text{sim}}(N_{i, \text{name}}, N_{j, \text{name}}) \geq t_2$, the similarity value is sorted in descending order, and $N_{j, \text{name}}$ is stored in the node name expansion set SET1 of $N_i$; If $A_{\text{sim}}(A_i, A_j) \geq t_3$, the similarity value is sorted in descending order, and $A_j$ is stored in the attribute expansion set SET2 of $A_i$; If $V_{\text{sim}}(V_i, V_j) \geq t_4$, the similarity value is sorted in descending order and $V_j$ is stored in the attribute expansion set SET3 of $V_i$; If $A_{V_{\text{sim}}}(a, v), (a, v) \geq t_5$, the similarity value is sorted in descending order, and $(a, v)$ is stored in the attribute expansion set SET4 of $(a, v)_i$.

5) Calculate the edge similarity function $L_{\text{sim}}(N_i, N_j)$. If $L_{\text{sim}}(N_i, N_j) \geq t_6$, store $L_j$ in the edge expansion set SET5 of $L_i$. The similarity function $L_{\text{sim}}(N_i, N_j)$ of node edge is calculated as follows:

$$L_{\text{sim}}(N_i, N_j) = \beta_1 \frac{L_{N_i \cap N_j}}{L_{N_i}} + (1 - \beta_1) \frac{L_{N_i \cap L_{N_j}}}{L_{N_i}}$$

(3)

Where, $\beta_1$ is the weight, $L_{N_i}$ and $L_{N_j}$ respectively represent the number of edge connections of $N_i$ and $N_j$, $L_{N_i \cap L_{N_j}}$ represents the number of common edges of $N_i$ and $N_j$, and $L_{N_i \cap L_{N_j}}$ represents the number of common neighbor nodes of $N_i$ and $N_j$.

6) If $i \geq n$ and $j \geq m$, that is, all nodes have been compared, then return SET1-SET5; otherwise, go to step 2 and continue to compare the similarity of the next node.

2.2. Expansion graph construction and query processing

On the basis of the expansion set, the semantic query expansion graph is constructed. The basic steps are shown in Figure 2:

1) If $\bigcup_{k=1}^{p} \text{SET}_k \neq \emptyset$, that is, if there is a non-empty expansion SET, calculate the total semantic similarity functions $N_{\text{sim}}(N_i, N_j)$, otherwise, perform query processing on the initial query graph and return the query results. $N_{\text{sim}}(N_i, N_j)$ takes into account the lexical similarity of the concept node and the similarity of the node’s connection edge (association), and it is calculated as follows:

$$N_{\text{sim}}(N_i, N_j) = w_1 \times T^{\text{sim}}(N_i, N_j) + w_2 \times L_{\text{sim}}(N_i, N_j)$$

(4)

Where, $w_c$ is the weight, $w_c \in [0, 1]$, $\sum_{c=1}^{2} w_c = 1$.

2) If $N_{\text{sim}}(N_i, N_j) \geq t_7$, combine all elements of non-empty SETk to generate an expanded query graph on the initial queries graph, otherwise, select some elements of non-empty SETk for combination, and generate an extended query graph on the initial query graph. Each extended query graph describes one of the user’s possible query requirements.

For example, suppose that the sets SET1, SET2, and SET3 are non-empty, and SET1= {S11, S12}, SET2= {S21, S22}, SET3= {S31}. If $N_{\text{sim}}(N_i, N_j) \geq t_8$, all elements are selected to construct the combination as follows: {S11, S21, S31}, {S11, S22, S31}, {S12, S21, S31}, {S12, S22, S31}. If $N_{\text{sim}}(N_i, N_j) < t_8$, select some elements in turn to construct the combination {S11, S21, S31}, {S11, S22, S31}. Here, the first element of SET1, the first two element of SET2 and the first element of SET3 are combined. In practical applications, the number of selected elements can be determined according to the importance of non-empty SETk. The greater the importance of SETk, the greater the number of selected elements.
Figure 2 Construction and query processing of semantic expansion graphs.

The semantic query expansion graph is constructed as follows: The elements from the attribute-value pair expansion set SET4 (form \((a, v)\)) in the combination are added to the attribute-value pair list of the node \(N_i\); The elements from SET1, SET2, and SET3 in the combination are converted into attribute-value pairs in the form of \((\text{Node name}, N_j.name), (A_i, A_i.value), (V_{\text{attribute}}, V_j)\), where \text{Node name} represents the attribute “node name”, \(A_i.value\) represents the value of the attribute \(A_i\), and \(V_{\text{attribute}}\) represents the attribute corresponding to the value \(V_i\), and then these attribute-value pairs are added to the attribute-value pairs list of the node \(N_i\). The elements from SET5 in the combination are added as the edges of node \(N_i\) and its neighbor nodes.

3) Conduct query processing on the extended query graph and return the query result.

In particular, the above steps involve the selection of the weights \(w_i\), \(s_i\), \(\beta_i\) and the threshold \(t_i\), which can be determined by the previous practical experience or other analysis methods, such as principal component analysis, factor analysis, regression analysis, and average analysis.

3. Conclusion

This paper proposes a query expansion method of graph models, which has dynamic and semantic characteristics. The method can dynamically optimize and extend the search requirements according to the semantic information, so as to improve the efficiency and effectiveness of semantic query. Moreover, the method is not only suitable for our defined graph model, but also suitable for RDF (Resource Description Framework), knowledge graph and other similar graph models.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant No. 61862010; Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing.

References

[1] Ghanbarpour, A., Niknafs, K., Naderi, H. (2020) Efficient keyword search over graph-structured data based on minimal covered r-cliques. Frontiers of Information Technology & Electronic Engineering, 21(3):448-464.

[2] Liu, G.J., Fang, X., Duan, J.Y. (2020) Query extension based on deep semantic information. Computer application, 363(11):98-103.
[3] Venetis, T., Stoilos, G., Stamou, G. (2014) Query extensions and incremental query rewriting for OWL 2 QL ontologies. Journal on Data Semantics, 3(1):1-23.

[4] Shin, J., Wu, S., Wang, F., Sa, C.D., Zhang, C., C Ré. (2015) Incremental knowledge base construction using deepdive, Vldb Journal, 8(11):1-25.

[5] Kang, Y.L., Feng, L.I., Wang, L. (2017) Incremental Optimization Method for Periodic Query in Data Warehouse. Journal of Software, 28(8):2126-2147.

[6] Yuan, T.T., Zhang, Z. (2018) Merchandise recommendation for retail events with word embedding weighted TF-IDF and dynamic query expansion. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. pp.1347-1348.

[7] Azad, H.K., Deepak, A. (2019) Query expansion techniques for information retrieval: A survey. Information Processing & Management, 56(5):1698-1735.

[8] Elghandour, I., Kara, A., Dan, O., Vansummeren, S. (2018) Incremental techniques for large-scale dynamic query processing. In: The 27th ACM International Conference on Information and Knowledge Management. Torino Italy. pp. 2297-2298.

[9] Salles, M., Dittrich, J.P., Karakashian, S.K., Girard, O.R., Blunschi, L. (2007) iTails: pay-as-you-go information integration in dataspaces. In: Proceedings of the 33rd International Conference on Very Large Data Bases. University of Vienna, Austria. pp. 663-674.