



Advanced Preferred Search Engine

Aishwarya Tirthgirikar, Aashish Tamsya, Shyam Deshmukh, Sumit Devkar
Computer Engineering, UOP,
India

Abstract—An advanced preferred search engine (APSE) captures the users' preferences in the form of concepts by mining their click through data. Due to the importance of location information in mobile search, APSE classifies these concepts into content concepts and location concepts. In addition, users' locations (positioned by GPS) are used to supplement the location concepts in APSE. Based on the client-server model, we present a detailed architecture and design for implementation of APSE. In our design, the client collects and stores locally the click through data to protect privacy, whereas heavy tasks such as training and re-ranking are performed at the APSE server. We prototype APSE on the Google Android platform and experimental results show that APSE significantly improves the precision comparing to the general search results.

Keywords— APSE: Advanced Preferred Search Engine, Google Android Platform, Search Engine, Click-through data, Location and Content Concepts.

I. INTRODUCTION

A major problem in mobile search is that the interactions between the users and search engines are limited, as a result, mobile users tend to submit shorter and hence, more ambiguous queries. In order to return highly relevant results to the users, mobile search engines must be able to profile the users' interests and personalize the search results according to the users' profiles. A practical approach to capturing a user's interests for personalization is to analyse the user's click-through data. To add more relevance to search results, location concept, tracked by GPS can also be applied. For example, a user who wants to visit a hotel may issue a simple query "hotel" and in return APSE will show results relevant to its location and past clicking history. Users' click will help APSE to train itself using RSVM algorithm to show even more relevant result, next time. Thus, more the number of searches, more relevant results can be obtained in future.

II. RELATED WORK

Kenneth Wai-Ting Leung developed a search engine personalization method based on users' concept preferences and showed that it is more effective than methods that are based on page preferences. However, most of the previous work assumed that all concepts are of the same type. Observing the need for different types of concepts, we present in this paper an Advanced Preferred Search Engine (APSE) which represents different types of concepts in different ontologies. In particular, recognizing the importance of location information in mobile search, we separate concepts into location concepts and content concepts. APSE also handles privacy issue effectively, which wasn't handled in most of previous work.

III. PROPOSED SYSTEM

In this project, we propose a realistic design for APSE by adopting the meta-search approach which relies on Google to perform an actual search. The client is responsible for receiving the user's requests, submitting the requests to the APSE server, displaying the returned results, and collecting his/her click-through in order to derive his/her personal preferences. The APSE server, on the other hand, is responsible for handling heavy tasks such as forwarding the requests to backend search engine (Google), as well as training and re-ranking of search results before they are returned to the client. The user profiles for specific users are stored on the APSE clients, thus preserving privacy to the users. APSE has been prototyped with APSE clients on the Google Android platform and the APSE server on a PC server to validate the proposed ideas.

IV. SYSTEM MODULES

A. APSE Client

Simple tasks, such as updating click-throughs and ontologies, creating feature vectors, and displaying re-ranked search results are handled by the APSE clients, implemented on Android Mobile Device. APSE clients stores user click-throughs and the ontologies so that privacy preserved. APSE client submits a query together with the feature vectors to the APSE server, and the server automatically return a set of re-ranked search results according to the preferences stated in the feature vectors. The data transmission cost is minimized, because only the essential data (i.e., query, feature vectors, ontologies and search results) are transmitted between client and server during the personalization process.

B. APSE Server

Heavy tasks, such as RSVM training and re-ranking of search results, are handled by the APSE server. User interest profiling into Content ontology and Location ontology is also handled by APSE server. APSE server is implemented on PC.

C. Re-Ranking Search Results

When a user submits a query on the APSE client, the query together with the feature vectors containing the user's content and location preferences are forwarded to the APSE server, which in turn obtains the search results from the Google. The content and location concepts are extracted from the search results and organized into ontologies. The feature vectors from the client are then used in RSVM training to obtain a content weight vector and a location weight vector, representing the user interests based on the user's content and location preferences for the re-ranking. The search results are then re-ranked according to the weight vectors obtained from the RSVM training. Finally, the re-ranked results and the extracted ontologies for the personalization of future queries are returned to the client.

D. Ontology update and Click-through collection

When the user clicks on a search result, the click-through data together with the associated content and location concepts (organized into ontologies) are stored in the click-through database on the client.

V. DESIGN AND IMPLEMENTATION

A. Design and Working

1) APSE's client-server design

APSE client is responsible for storing the user click-throughs and therefore the ontologies derived from the APSE server. Easy tasks such as collecting click-throughs and ontologies, creating feature vectors and displaying re-ranked search results are done at client. On the opposite hand, significant tasks, like RSVM training and re-ranking of search results is handled by the APSE server. Moreover, so as to reduce the information transmission between user and server, the APSE client will submit a query along with the feature vectors to the APSE server, and therefore the server would automatically come back a group of re-ranked search results according to the preferences expressed within the feature vectors. Issues like limited power on mobile devices and reduction in knowledge transmission are self handled by APSE.

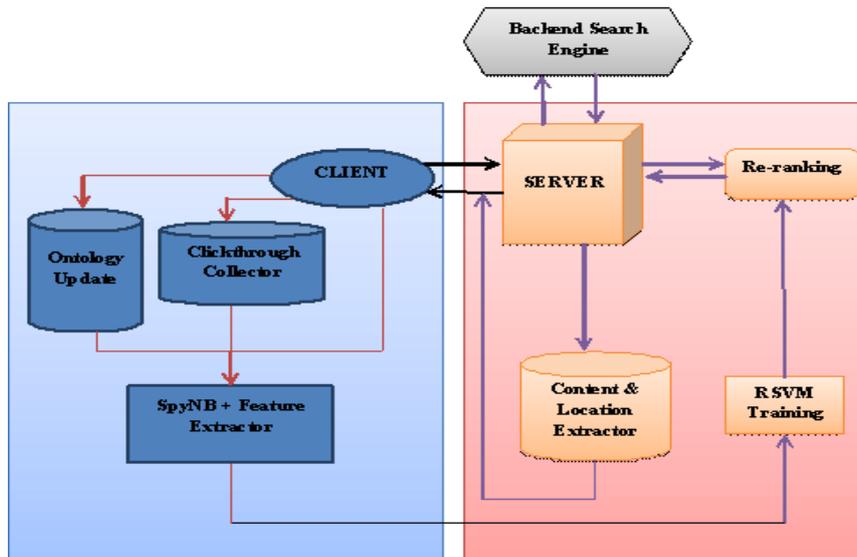


Fig. 1. System Design

2) Working of APSE App

Users interact with APSE system through APSE client which is implemented using Android App on mobile device. User submits the query to the APSE client, client in return forwards the query along with feature vector i.e., combination of location vector and content vector to the APSE server. Server searches for the query on backend search engine i.e., Google. The APSE server performs re-ranking of obtained search results using RSVM algorithm based on location and content vectors. The APSE client receives re-ranked search results, click-through data, and ontologies of location and content concepts from server. The click-through database is updated on basis of the user clicks and ontologies are updated for that particular query on client's side. Ontologies play important role in re-ranking the searched results as they are responsible for setting the preferences of the user and thus, giving more relevant result with repeated use.

B. Algorithms

1) Spy NB Method (User Preference & Privacy)

In this section, we propose a new preference mining algorithm, called Spy Naïve Bayes (SpyNB). It consists of two main components: a spying technique to obtain more accurate negative samples and a voting procedure to consider the

opinions of all spies. According to our clickthrough interpretation, we need to categorize unlabeled data in order to discover the predicted negative links. Naive Bayes [Mitchell 1997] is a simple and efficient text categorization method. However, conventional Naïve Bayes requires both positive and negative examples as training data, while we only have positive examples. To address this problem, we employ a spying technique to train Naive Bayes by incorporating unlabeled training examples. Moreover, in order to obtain more accurate predicted negatives, we further introduce a voting procedure to make full use of all potential spies.

Algorithm: The Spy Naive Bayes (SpyNB) Algorithm

Input: P { a set of positive examples; U { a set of unlabeled examples; Tv { a voting threshold;

Output: PN { the set of predicted negative examples

Procedure:

- 1: PN1 = PN2 = = PN|P| = {} and PN = {};
- 2: for each example pi ∈ P do
- 3: Ps = P - {pi};
- 4: Us = U U { pi };
- 5: Assign each example in Ps the class label 1;
- 6: Assign each example in Us the class label -1;
- 7: Train a Naive Bayes on Ps and Us using Naïve Bayes Training algorithm;
- 8: Predict each example in Us using trained Naive Bayes;
- 9: Spy threshold Ts = Pr(+|pi);
- 10: for each ui ∈ U do
- 11: if Pr(+|uj) < Ts then
- 12: PN_i = PN_i ∪ {uj};
- 13: end if
- 14: end for
- 15: end for
- 16: for each uj ∈ U do
- 17: V otes = the number of PN_i such that uj ∈ PN_i
- 18: if V otes > Tv . |P| then
- 19: PN = PN ∪ {uj};
- 20: end if
- 21: end for

2) RSVM training Algorithm:

RSVM algorithm is used to extract the feature vectors for a document and to combine the content and location weight vectors into one integrated weight vector.

Two feature vectors, namely, content feature vector $\phi_C(q, d_k)$ and location feature vector $\phi_L(q, d_k)$ to represent the content and location information associated with documents.

Document d_k embodies - content concept c_i and location concept l_i .

The following relationships of the concepts in the extracted concept ontologies are also incorporated in the training

1. Similarity,
2. Ancestor,
3. Descendant, and
4. Sibling

If content concepts c_i is in a web-snippet s_k , their values are incremented in the content feature vector,

$$\forall c_i \in s_k, \phi_C(q, d_k)[c_i] = \phi_C(q, d_k)[c_i] + 1.$$

For other content concepts c_j that are related to the content concept c_i (either they are similar or c_j is the ancestor/descendant/sibling of c_i) in the content ontology, they are incremented in the content feature vector,

$$\begin{aligned} \forall c_i \in s_k, \phi_C(q, d_k)[c_j] = & \phi_C(q, d_k)[c_j] \\ & + sim_R(c_i, c_j) + ancestor(c_i, c_j) \\ & + descendant(c_i, c_j) + sibling(c_i, c_j). \end{aligned}$$

Similarly, if location concepts l_i is in a web-snippet s_k , their values are incremented in the location feature vector,

$$\forall l_i \in d_k, \phi_L(q, d_k)[l_i] = \phi_L(q, d_k)[l_i] + 1.$$

And, for other location concepts l_j that are related to the location concept l_i (either they are similar or l_j is the ancestor/descendant/sibling of l_i) in the location ontology, they are incremented in the location feature vector,

$$\forall l_i \in d_i, \phi_L(q, d_k)[l_j] = \phi_L(q, d_k)[c_j] + \text{ancestor}(l_i, l_j) + \text{descendant}(l_i, l_j) + \text{sibling}(l_i, l_j).$$

C. Complexity

Our project uses spyNB algorithm (Naïve Bayes Classifier) for mining user’s preferences and privacy. The NP-hardness of this algorithm has already been proved by Chlebus and Nguyen. Thus, our problem is NP-HARD.

But, as we know 3-SAT problem is NP-COMPLETE and problems those reduce to any NP-COMPLETE problem are NP-COMPLETE themselves.

Our problem can be reduced to 3-SAT problem, as followed

Consider 3 literals a,b,c
 A – denotes prediction of user preference
 B – denotes query has been entered
 C – denotes reranking is done
 $(a \vee b \vee c) \wedge (a \vee b \vee \neg c) \wedge (\neg a \vee b \vee \neg c)$
 $(1 \vee 1 \vee 1) \wedge (1 \vee 1 \vee 0) \wedge (0 \vee 1 \vee 0)$
 $(1) \wedge (1) \wedge (1)$
 1

Hence, Our System reduces to 3-SAT problem.

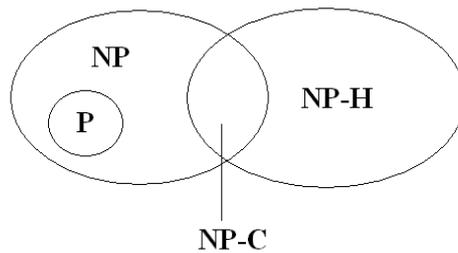


Fig. 2. Venn diagram for feasibility assessment

As shown in Fig. 2, NP-hard problems are included in NP-COMPLETE; therefore our problem is **NP-complete**.

VI. RESULTS, DATA TABLES AND GRAPHS

A. Screenshots of actual working system

1) Login to enter into user’s profile

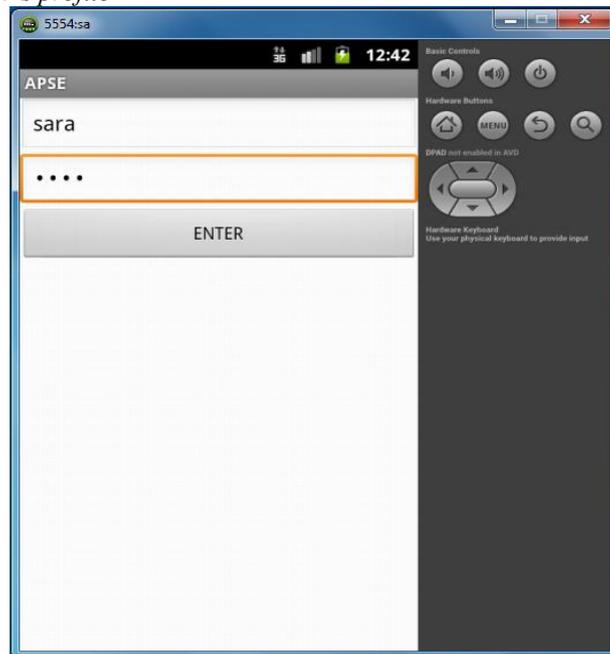


Fig. 3. Login page

2) User enters the query and hits S for searching query on Google



Fig. 4. Entering query

3) Search results



Fig. 5. Query results

4) Finding the rank and weight of the query

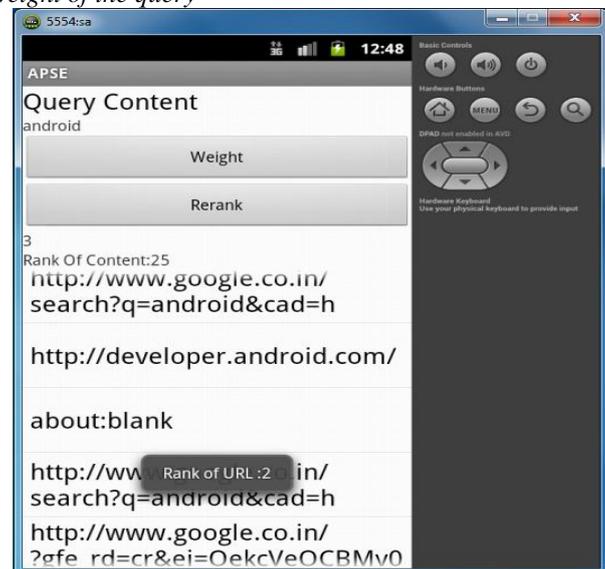


Fig. 6. Weight and rank of query

B. Result's Comparison (Google vs. APSE)

Parameter – Relevant search results

Here as the no. of searches increases APSE gets more trained and thus provides more relevant or personalized results for that particular user. Vertical axis denote the no. of results displayed by the search engines. Initially both APSE and Google search show same results, but after more than 30 times a query is searched, APSE shows significant increase in relevance of search.

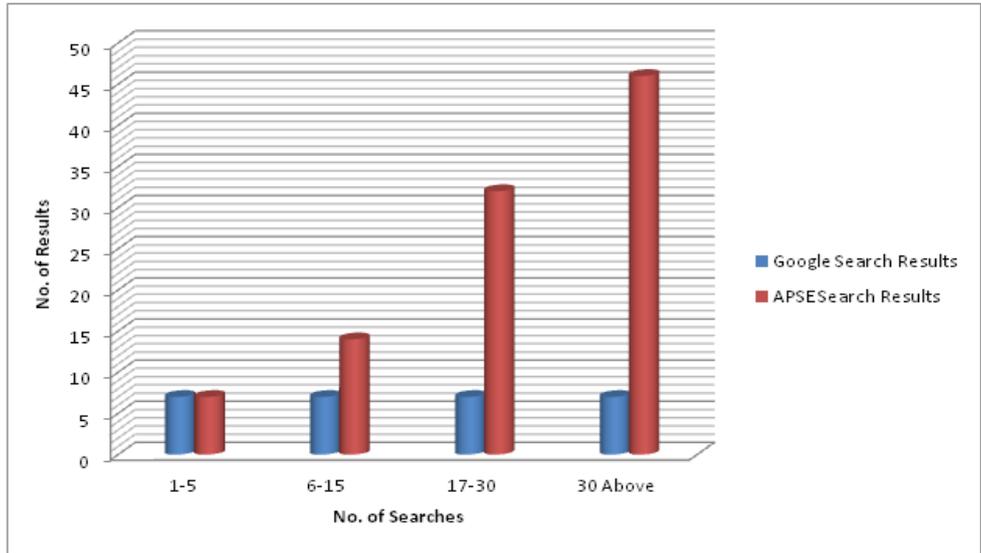


Fig. 7. Relevant Search Graph

Parameter – Response time

Here as the no. of searches increases APSE gets more trained and thus provides results more quickly. This is due to JSON parsing which occurs at APSE server, due to which APSE doesn't have to fetch the results every time they get stored on client machine itself and user can get the results faster than Google, if used more than 30 times. Vertical axis denotes the no. of results displayed by the search engines. Initially Google search shows results faster than APSE, but after more than 30 times a query is searched, APSE shows significant decrease in response time.

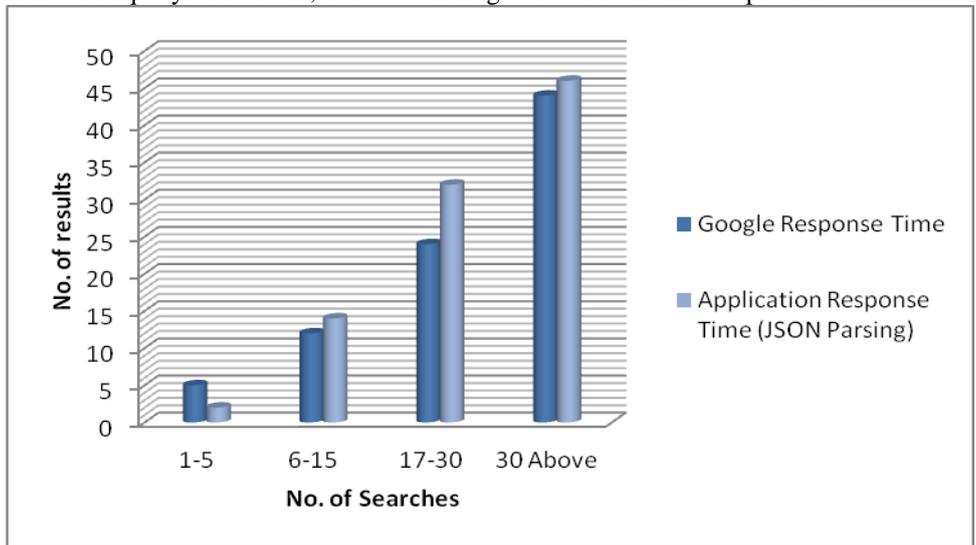


Fig. 8. Response Time Graph

VII.CONCLUSION

Since users are given with predefined queries and topical interests, they have to synthesize their information needs from the given queries and topical interests and conduct their searches correspondingly. Thus, their search behaviors in the experiments may be quite different from what they might have exhibited when they attempt to resolve real-life information needs. Ideally, a large-scale user study should be conducted in which APSE is subjected to real-life use, users' behaviors are monitored transparently and satisfaction of the users is analyzed and compared with other systems, but a large-scale, in-the-wild study is beyond the scope of this paper.

ACKNOWLEDGMENT

The authors would like to express their sincere thanks to the editors and reviewers for giving very insightful and encouraging comments.

REFERENCES

- [1] Kenneth Wai-Ting Leung, Dik Lun Lee, and Wang-Chien Lee, "PMSE: A Personalized Mobile Search Engine", IEEE Transactions On Knowledge And Data Engineering, Vol. 25, No. 4, April 2013
- [2] B. Liu, W.S. Lee, P.S. Yu, and X. Li, "Partially Supervised Classification of Text Documents," Proc. Int'l Conf. Machine Learning (ICML), 2002.
- [3] W. Ng, L. Deng, and D.L. Lee, "Mining User Preference Using Spy Voting for Search Engine Personalization," ACM Trans. Internet Technology, vol. 7, no. 4, article 19, 2007.
- [4] J.Y.-H. Pong, R.C.-W. Kwok, R.Y.-K. Lau, J.-X. Hao, and P.C.-C. Wong, "A Comparative Study of Two Automatic Document Classification Methods in a Library Setting," J. Information Science, vol. 34, no. 2, pp. 213-230, 2008.
- [5] C.E. Shannon, "Prediction and Entropy of Printed English," Bell Systems Technical J., vol. 30, pp. 50-64, 1951.
- [6] Q. Tan, X. Chai, W. Ng, and D. Lee, "Applying Co-Training to Click-through Data for Search Engine Adaptation," Proc. Int'l Conf. Database Systems for Advanced Applications (DASFAA), 2004.
- [7] J. Teevan, M.R. Morris, and S. Bush, "Discovering and Using Groups to Improve Personalized Search," Proc. ACM Int'l Conf. Web Search and Data Mining (WSDM), 2009.
- [8] Voorhees and D. Harman, TREC Experiment and Evaluation in Information Retrieval. MIT Press, 2005.
- [9] Y. Xu, K. Wang, B. Zhang, and Z. Chen, "Privacy-Enhancing Personalized Web Search," Proc. Int'l Conf. World Wide Web (WWW), 2007.
- [10] S. Yokoji, "Kokono Search: A Location Based Search Engine," Proc. Int'l Conf. World Wide Web (WWW), 2001.