JCSSE

2012

2012 Ninth International Joint Conference on Computer Science and Software Engineering (JCSSE)

30 May - 01 June 2012

IEEE

School of Engineering

University of the Thai Chamber of Commerce, Bangkok, Thailand

Editor : Adisorn Leelasantitham

IEEE Catalog Number: CFP1232P-ART

JCSEE 2012

The 9th International Joint Conference
on Computer Science and Software Engineering

30 May - 01 Jun 2012

University of the Thai Chamber of Commerce

Bangkok, Thailand

Editor:

Adisorn Leelasantitham

Copyright and Reprint Permission: Abstracting is permitted with credit to the source. Libraries are permitted to photocopy beyond the limit of U.S. copyright law for private use of patrons those articles in this volume that carry a code at the bottom of the first page, provided the per-copy fee indicated in the code is paid through Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923. For other copying, reprint or republication permission, write to IEEE Copyrights Manager, IEEE Operations Center, 445 Hoes Lane, Piscataway, NJ 08854.

All rights reserved. Copyright © 2012 by IEEE.

IEEE Catalog Number :	CFP1232P-ART
ISBN :	978-1-4673-1921-8
IEEE Catalog Number (CD-ROM) :	CFP1232P-CDR
ISBN :	978-1-4673-1919-5

Ranking Application Software Retrieval Results Using Application Attributes and Context

Anugoon Leelaphattarakij
Department of Computer Engineering,
Faculty of Engineering
Chulalongkorn University, Bangkok, Thailand
Email: Anugoon.L@student.chula.ac.th

Nakornthip Prompoon
Department of Computer Engineering,
Faculty of Engineering
Chulalongkorn University, Bangkok, Thailand
Email: Nakornthip.S@chula.ac.th

Abstract—**Nowadays, various search engines provide an easy way to search an application that meets the user requirements. However, there are an increasing number of similar feature applications or a large number of applications that are intentionally implanted popular keywords in their information. Thus, search results may have the tendency not to satisfy the user needs. In this paper, we present a new method to rank application search results focusing on two criteria: application attributes and application context. Using application attributes and its context, we propose three ranking scores; rating score, content relevant score and context score. From our experiment, applying all three scores for application retrieval, significantly gives a better search quality results than the baseline and the selected combined scores using r-precision and normalized discounted cumulative gain score metric.**

Keywords-component; information storage and retrieval; mobile application, application software; ranking; popularity; pagerank;

I. Introduction

Nowadays, the number of application software are growing rapidly to serve the different requirements of the market [3][4]. Application distribution portals (application market) such as Google's Android Market and Apple's App Store are providing application acquiring process for acquirer or user and supplier or developer in a convenient way. However, it is becoming increasingly difficult for users to find out exactly what they are looking for. One of the main reasons for this problem is because the number of mobile applications on each market becomes very large. There are roughly 450,000 and 550,000 applications on Google's Android Market and Apple's App Store, respectively [3][4]. Flood of low quality applications is also an important reason that leads to poor search results quality. In October 2011, Google removed 37% of all applications in the Android Market and Apple removed 24% of all applications in the App Store in order to clean up an inappropriate or outdated content [6]. The impact of search engine optimization, usually by employing highly competitive keywords, is another root cause of this problem. Most mobile application search engines that rely solely on keywords often returns too many irrelevant applications in the search results because some applications may implant a list of popular keywords into the application.

In this paper, we proposed a new ranking method to find applications that meet user needs by focusing on two criteria: application attributes and application context. The application attributes used in this work are application rating score, which represents the popularity of the application among users, and the similarity between search query and application content. Both are used to produce a rating score, a content relevant score respectively. Application content composed of two elements, application title and description.

The application context used in this research is derived from the relationship between applications such as a list of other applications that are viewed by users who also viewed one particular application. The relationship information is used to produce context score using PageRank algorithm. Traditionally, PageRank algorithm is used for computing a ranking for web documents using hyperlinks or citations between web documents to construct webgraph. In this research, we used application context information, which can be viewed as links between applications to construct the webgraph instead. We believe that these links are better for the representation of application popularity rather than rating score because rating score may vary according to user perception. For example, some users may rate an application score out of five for a good application, while others may rate score out of four for a good application as well. Another reason is because these links do not depend on the number of users who rate each application, like the rating score. Since this number is cumulative, which means that older applications are usually rated by more users rather than newer application, this can impact on the creditability of rating score.

II. Related Works

In order to help users find quality applications, a variety of methods have been proposed. There is a large body of research that focuses on using contextual information such as user location [7] and usage behavior [8]. Other research focuses on improving the quality of application search results by providing a more fine grained control of search parameters [5].

AppBrain [5] is a search engine for Android applications that filter low quality applications from search results using statistical information such as rating, and a number of applications that each developer had published on the market. It also provides a filtering mechanism on application categories and popularity. Different from AppBrain, our work focuses on improving application search results by using both application content and context.

AppAware [7] is a mobile application that allows users to be aware and possibly discover quality applications by providing information about installation, updates, and removal of applications from other users in his or her proximity, together with rating and comments given by other users in a real-time manner. However, this method does not serve the major users need which largely concerns getting the applications that have the features fulfill their requirements.

AppJoy [8] is a recommendation system for Android applications that suggests new applications based on user behavior (e.g. how long does the user spend his or her time on a particular application) using Collaborative Filtering algorithm. It can collect information automatically without user intervention. However, collecting context of user behavior may suffer from the violation user privacy.

III. Page Rank Algorithm

PageRank is a popularity driven ranking algorithm, originally designed for measuring the relative importance of the web pages and ranking web documents [10]. Using content within web document is not sufficient to effectively rank web documents because document collection is very large and usually relevant to search query. To avoid less relevant document appearing at the top search results, PageRank use a link analysis method to measure the importance of a document. In web document, if page B has a hyperlink to page A then it means that page A has a backlink from page B which serve as a citation to page A.

PageRank algorithm calculation is based on a random surfer model, which randomly select a page and then follow a hyperlink in that page randomly without returning to previous page [9]. It is defined as

$$PR(A) = \frac{(1-d)}{N} + d\left(\frac{PR(T_1)}{C(T_1)} + \ldots + \frac{PR(T_n)}{C(T_n)}\right) \qquad (1)$$

where $PR(A)$ is a PageRank score of page A, N is the number of documents in collection, T_n is the n-th web page that page A has backlink from, $C(T_n)$ is the number of outgoing links of T_n, and d is a damping factor, usually set to 0.85 [14], which is the probability of random surfer will get bored and restart at another random page.

IV. Application Ranking

In this section, we explain how we calculate and combine rating score together with context score and content relevant score to create ranking function for calculating ranking score to sort application search results.

A. Rating Score

Rating score is one of the popularity scores that users give to an application. In order to use a rating score to rank an application, a rating count (the number of users who rate application) is also important. It refers that rating score which rates by more users are more credible than rating score that rates by less users. Since the rating counts are widely different between applications, we need to normalize the rating counts. For example "Angry Birds" rating count is more than one million while the overall average rating count for all applications is 556. Thus, we reduce this variant by converting a rating count to log scale before we calculate rating score as shown in

$$RatingScore(j) = \overline{R(j)} \times \log_{10} C(j) \qquad (2)$$

where j is an application, $RatingScore(j)$ is an application rating score, $\overline{R(j)}$ is average rating score of application j, and $C(j)$ rating count of application j.

B. Context Score

Context score is one of the ranking score we derived from the relationship between applications. The score was calculated by using PageRank algorithm. Unlike web documents, applications do not have backlink or citation required to calculate PageRank score. In this case, we use lists of related applications (RAL) and lists of also installed applications (IAL), often appear in the application's detail page of each application as backlink instead. These backlinks were considered as citation information which came from users who viewed or already installed related applications. These lists of applications reflected the user interest based on installation and relationship between applications which are harder to manipulate than rating score or number of users who installed the application. We can calculate the application context score using the following equation

$$ContextScore(j) = \frac{(1-d)}{N} + d\left(\frac{PR(T_1)}{C(T_1)} + \ldots + \frac{PR(T_n)}{C(T_n)}\right) \qquad (3)$$

where j is an application, $ContextScore(j)$ is an application context score, $PR(T_n)$ is PageRank score of n-th application that j has backlinks from, d is damping factor and N is number of applications in the collection.

C. Content Relevant Score

Content relevant score is used for measuring relevance between text in documents, which are title and description of software application, and search query. In this research, we use TF-IDF algorithm [12] to calculate content relevant score which can be defined as

$$ContentScore(i, j) = tf(i, j) \cdot idf(i) \qquad (4)$$

Given that j is a document, $ContentScore(i,j)$ is a document content relevant score, $tf(i,j)$ is normalized term frequency of term i in document j, $idf(i)$ is the inverse document frequency of term i.

D. Ranking Function

In order to rank applications, we defined our ranking function, aiming at capturing user preference and ranking applications on how well they match the query intent. By combining the context score, the rating score, and the content relevant score, we propose our ranking function as

$$RankingScore(j) = \omega_1 \cdot z_1 + \omega_2 \cdot z_2 + \omega_3 \cdot z_3 \qquad (5)$$

As a first step, we transform the context score, the rating score, and the content relevant score into Z-scores, and then we combined them into the final ranking score using linear combination as shown in Eq. (5)

where ω_1, ω_2 and ω_3 are predefined weight constants. z_1, z_2 and z_3 are Z-score of rating score, context score, and content relevant score respectively.

V. Experiments

To evaluate the effectiveness of the proposed method, we setup an information retrieval (IR) system, create search queries, and define ranking strategies.

A. Information Retrieval System

Our information retrieval system has three modules as shown in Figure 1. The first module is data crawler module. This module works as web spiders, responsible for crawling, extracting application information from application market and storing it in our local database for offline processing. Details of information application extraction will be described in section VI.

The second module is indexing and ranking score calculation module. This module is used for calculating each application rating score and context score using Eq.2 and Eq.3 respectively. Then, we create inverted file to store ranking score and applications content used for retrieval module.

The final module is retrieval module. This module will be used by users for searching an application. The process will begin when user enters a search query into the system. This module will process the search query and retrieve application search result from the index file. The content relevant score is calculated in this module using Eq.4. Then, we rank search results according to ranking strategy which will be explained later in the ranking strategies part in this section. The ranked search result will be presented to the user.

B. Search Queries

In our experiment, we manually created two different sets of the search queries for searching application. The first one uses generic terms and the other uses proper names. All search queries are monograms, which is a sequence of characters that contains no white space between characters, and each set of search queries have different evaluation criteria which will be describing in section VII.

1) Generic Queries (Q_n): All queries in this group are terms that used to describe the functionality of applications. For example, “news”, “video”, and “camera” are used to describe applications that provide functionality related to news feeds, video, camera, respectively. There are fifty query terms in this set.

2) Unique Queries (Q_u): Unlike the previous query set, all queries in this set are the names of well-known applications in the market such as “foursquare”, “adobe”, “google”, or “facebook”. We use thirty query terms for this set.

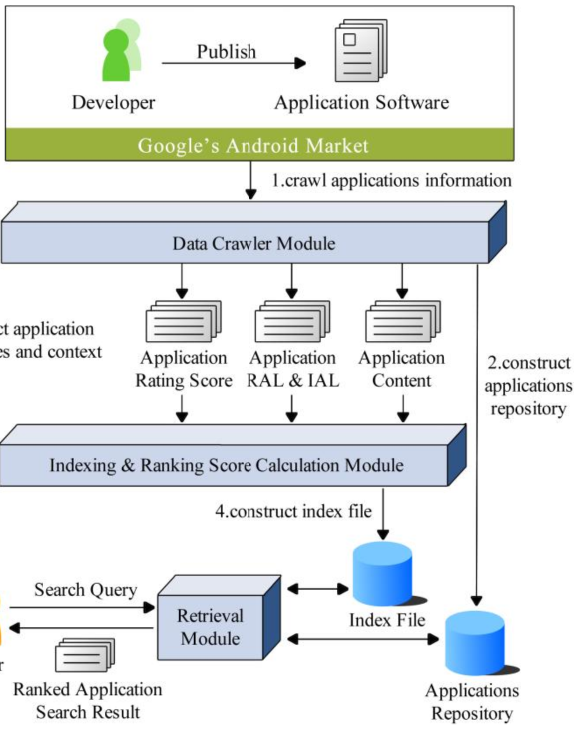

Figure 1. Architechture of the information retrieval system

C. Ranking Strategies

As we wanted to monitor the effect of each application ranking score, the weight set [$\omega_1, \omega_2, \omega_3$] in Eq. (5) would be varied while other constants would be fixed. We defined four weight sets to use in experiment [1,0,0] as S_r, [0,1,0] as S_p, [1,1,0] as S_{rp}, and [1,1,1] as S_{rpc}. The weight set [1,0,0] which use only the rating score for ranking is selected as baseline.

VI. Data Collection

As we mentioned in the above section, we performed experiments on real world data collected from Google's Android Application Market [1]. It is an online application market website for Android devices that provides detailed information about Android applications. Although Google's Android Market has provided an API for extracting application information, it is somewhat limited because the API does not provide necessary information for us to derive application context. Therefore, we created a script that crawl and extract application information from the Google's Android Market website directly without using the API.

A. Application Attributes

We collected application attributes for each Android application as described in the following list.

1) Package Name: The package name. It serves as a unique identifier for the application.

2) Title: It is an application's name.

3) Description: The application's description which are limited to 4,000 characters [2].

4) Category: The application's category such as "Productivity", "Medical", or "Lifestyle".

5) Rating: The average star rating from application users ranging from zero to five stars.

6) Rating Count: The number of users who rate the application.

B. Application Context

We derived application context information from the relationship between applications as described in the following list.

1) Related Application Links (RAL): List of other applications that are viewed by users who also viewed this application.

2) Installed Application Links (IAL): List of other applications which users who installed application also installed.

In total 160,850 applications and 1,241,233 application links are acquired for our experiment. From the collected application links we divide them into two groups. The first group consists of 627,006 RAL from all applications. Another group consists of 614,227 IAL from all applications. TABLE I shows the statistics of application links related to a single application.

TABLE I. SINGLE APPLICATION'S BACKLINK STATISTICS

Backlink Type	Min	Max	Average	SD
Related Application Links	0	722	3.898	16.591
Also Installed Application Links	0	199	3.818	6.711

VII. EVALUATION

To evaluate the retrieval effectiveness, two metrics, R-Precision ($P@r$) [11] and normalized discounted cumulative gain ($nDCG_p$) [13] are selected. The R-Precision assesses how many relevant documents for the query can be retrieved at R-th position in retrieved documents. It is defined as

$$P@r = \frac{R_a}{r} \qquad (6)$$

where R_a is the number of applications that are classified as relevant and r is the ranking position of an application.

On the other hand, the normalized discounted cumulative gain assesses how effective of retrieval is by taking the ranking of applications in the search result into account. If there are less relevant applications at the top ranking position, the search result should be penalized as graded relevance value is reduced logarithmically proportional to the ranking position [13]. It is defined as

$$nDCG_p = \frac{DCG_p}{IDCG_p} \qquad (7)$$

where DCG_p is discounted cumulative gain of the search result and $IDCG_p$ is an ideal result, perfect ranking, of DCG_p. Discounted cumulative gain is defined as

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(1+i)} \qquad (8)$$

given that p is document ranking cutoff, i is document ranking and rel_i is relevance grade of retrieved result determined by using predefined criteria.

In our experiment, the retrieval effectiveness is judged with these two metrics with the document position cutoff for $r = 24$ and document ranking cutoff for $p = 24$, which is equal to the number of applications shown per page in Google's Android Market. The evaluation of each retrieved result is performed by experts using criteria as described in TABLE II and TABLE III to assign relevance grade for DCG for each search queries set. For R-Precision value, application that has relevance grade above zero will be considered as relevant to the search queries.

TABLE II. RELEVANCE GRADE CRITERIA FOR GENERIC QUERY

Relevance Grade	Criteria
3	The query term matches with the main functionality of the application.
2	The query term matches with the minor functionality of the application
1	The query term matches with the minor functionality of the application, but it is not obviously apparent or visible.
0	There is no function matches with the query in the application

TABLE III. RELEVANCE GRADE CRITERIA FOR UNIQUE QUERY

Relevance Grade	Criteria
3	The query term exactly or partially matches with the application title
2	The functionality of the application, which its title matches with the query term, and the functionality of the applications in the search result are the same or can be used for the same purpose.
1	Application is in the same category
0	The query term does not relate to the application in anyway.

VIII. RESULTS AND DISCUSSION

TABLE IV, V, and VI shows the results of search experiment of four different ranking strategies using generic queries, unique queries, and all queries, respectively. It can be seen that, the combination of context score, rating score, and content relevant score (S_{rpc}) yielded higher R-precision and nDCG than other settings significantly. This indicates that the content relevant score plays an important role in the quality of the application search results because it aims to capture on how relevant the application match with the query intent. TABLE VI shows that the context score (S_p) yielded higher R-precision and nDCG than using the rating score (S_r) alone. If we look closer at TABLE IV and V, the context score alone also yielded higher R-precision than the rating score alone when searching with generic queries, but the rating score achieves better nDCG when searching with unique queries. This is because most unique queries are names of well-known applications that usually have higher rating score when compared to other applications that are less well known or new to the market, but their context scores are often comparable making S_p suitable for discovering new applications. We also calculated percentage of improvement for each ranking strategy when compared to baseline ranking strategy. The results in TABLE VI show the ranking strategy, which combines all three ranking scores, yields the highest percentage of improvement.

TABLE IV. RETRIEVED RESULT USING GENERIC QUERIES(Q_N)

Ranking Strategy	Evaluate Result		Improvement Compare To Baseline	
	P@24	*nDCG@24*	*P@24*	*nDCG@24*
S_r	0.5042	0.6313	-	-
S_p	0.5225	0.6910	3.64%	9.46%
S_{rp}	0.5283	0.6817	4.79%	7.98%
S_{rpc}	0.6283	0.7537	24.63%	19.39%

TABLE V. RETRIEVED RESULT USING UNIQUE QUERIES (Q_U)

Ranking Strategy	Evaluate Result		Improvement Compare To Baseline	
	P@24	*nDCG@24*	*P@24*	*nDCG@24*
S_r	0.3486	0.6926	-	-
S_p	0.3847	0.6394	10.36%	-7.68%
S_{rp}	0.3861	0.6740	10.76%	-2.68%
S_{rpc}	0.5278	0.8468	51.39%	22.27%

TABLE VI. RETRIEVED RESULT USING BOTH GENERIC AND UNIQUE QUERIES

Ranking Strategy	Evaluate Result		Improvement Compare To Baseline	
	P@24	*nDCG@24*	*P@24*	*nDCG@24*
S_r	0.4458	0.6543	-	-
S_p	0.4708	0.6717	5.61%	2.66%
S_{rp}	0.4750	0.6788	6.54%	3.75%
S_{rpc}	0.5906	0.7886	32.48%	20.53%

IX. CONCLUSIONS AND FUTURE WORKS

In this paper, we introduced a new method to rank application search results using application attributes and its context. By utilizing application attributes and its context, we proposed three ranking scores; rating score, context relevance score, and context score. Applying all three scores for application retrieval results significantly gives a better search quality compared to the baseline. It yields 32.48 percent improvement evaluated by R-precision metric and 20.53 percent improvement evaluated by normalized discounted cumulative gain score metric.

Although our ranking method performed reasonably well, its performance was still far from perfect. Therefore, there is still room for improvement. Incorporating additional information reflecting user preferences such as location-based and temporal-based information should be explored. The method for classifying search queries into either broad terms or narrow terms could also be used to explore the contribution of each ranking score in order to weigh their importance differently and dynamically.

REFERENCES

[1] Google. "Android Market." Internet: market.android.com, Feb. 20, 2012 [Feb. 20, 2012].

[2] Google. "Android Developers." Internet: developer.android.com, Feb. 20, 2012 [Feb. 20, 2012].

[3] Google Mobile Team. "Google Mobile Blog, News and notes from the Google Mobile team." Internet: googlemobile.blogspot.com, Feb. 27, 2012 [Feb. 27, 2012].

[4] Apple. "Apple Press Info." Internet: www.apple.com/pr/, Mar. 5, 2011 [Mar. 5, 2012].

[5] AppBrain. "AppBrain Blog" Internet: blog.appbrain.com, Sep. 29, 2011 [Feb. 20, 2012].

[6] TechCrunch. "TechCrunch" Internet: techcrunch.com, Oct. 21, 2011 [Feb. 20, 2012].

[7] A. Girardello and F. Michahelles, "AppAware: which mobile applications are hot?," in *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, 2010, pp. 431–434.

[8] B. Yan and G. Chen, "AppJoy: personalized mobile application discovery," in *Proceedings of the 9th international conference on Mobile systems, applications, and services*, 2011, pp. 113–126.

[9] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web.," *World Wide Web Internet And Web Information Systems*, pp. 1-17, 1999.

[10] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan, "Searching the web," *ACM Transactions on Internet Technology (TOIT)*, vol. 1, no. 1, pp. 2–43, 2001.

[11] R. Baeza-Yates and B. Ribeiro-Neto, "Modern Information Retrieval," ACM Press, Addison Wesley, 1999.

[12] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill Education, 1983.

[13] K. J¨arvelin and J. Kek¨al¨ainen. "Cumulated gain-based evaluation of IR techniques." *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[14] L. Becchetti and C. Castillo, "The distribution of PageRank follows a power-law only for particular values of the damping factor," in *Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 941–942.