sustainability

MDPI

Article

# Evaluating Retrieval Effectiveness by Sustainable Rank List

**Tenvir Ali [1], Zeeshan Jhandir [1], Ingyu Lee [2], Byung-Won On [3,*] and Gyu Sang Choi [1,*]**

[1] Department of Information and Communication Engineering, Yeungnam University, Gyeongsan, Gyeongbuk 38541, Korea; tenvirali@ynu.ac.kr (T.A.); zeeshanjhandir@ynu.ac.kr (Z.J.)
[2] Sorrell College of Business, Troy University, Troy, AL 36082, USA; inlee@troy.edu
[3] Department of Software Convergence Engineering, Kunsan National University, Gunsan-si, Jeollabuk-do 54150, Korea
* Correspondence bwon@kunsan.ac.kr (B.-W.O.); castchoi@ynu.ac.kr (G.S.C.)

Received: 29 April 2017; Accepted: 5 July 2017; Published: 8 July 2017

**Abstract:** The Internet of Things (IoT) and Big Data are among the most popular emerging fields of computer science today. IoT devices are creating an enormous amount of data daily on a different scale; hence, search engines must meet the requirements of rapid ingestion and processing followed by accurate and fast extraction. Researchers and students from the field of computer science query the search engines on these topics to reveal a wealth of IoT-related information. In this study, we evaluate the relative performance of two search engines: Bing and Yandex. This work proposes an automatic scheme that populates a sustainable optimal rank list of search results with higher precision for IoT-related topics. The proposed scheme rewrites the seed query with the help of attribute terms extracted from the page corpus. Additionally, we use newness and geo-sensitivity-based boosting and dampening of web pages for the re-ranking process. To evaluate the proposed scheme, we use an evaluation matrix based on discounted cumulative gain (DCG), normalized DCG (nDCG), and mean average precision ($MAP_n$). The experimental results show that the proposed scheme achieves scores of MAP@5 = 0.60, $DCG_5$ = 4.43, and $nDCG_5$ = 0.95 for general queries; $DCG_5$ = 4.14 and $nDCG_5$ = 0.93 for time-stamp queries; and $DCG_5$ = 4.15 and $nDCG_5$ = 0.96 for geographical location-based queries. These outcomes validate the usefulness of the suggested system in helping a user to access IoT-related information.

**Keywords:** Internet of Things; information retrieval; pseudo relevance feedback; search engines; bipartite graph; random walk

## 1. Introduction

Data functions as a fuel that helps the Internet run. However, the profusion of data on the World Wide Web (WWW) is creating considerable problems for Internet users. In the past few years, the fields of Internet of Things (IoT) and Big Data have evolved rapidly and the combination of these two has enormously increased the growth of data. According to the predictions in recent studies, by 2020 there will be tens of billions of devices and sensors [1] that will generate 40 Zettabytes (40 trillion GB) of data [2]. These interconnected devices are placed in real-time environments and play their role in assembling, communicating, and distributing data. For example, modern cars produced by different companies are equipped with multiple sensors. The use of these sensor-based cars helps in collecting facts such as sale by area, fuel efficiency, route finding, tracking of lost vehicles, or driver assessment. On the other hand, devices such as wearables (e.g., fitness bands) are collecting data for health monitoring, goal tracking, and user location. All the data produced by these devices are stored on clouds. The most difficult tasks are the use of data when it is on the move and extracting valuable information from them. This change in the field of computer science is changing many technologies.

For example, to handle this burst of data, existing storage mechanisms are gradually replaced by new emerging technologies such as Platform-as-a-Service (PaaS) or cloud Spark.

On a daily basis, new devices are added to the Internet world and they share a huge amount of data. This addition of devices is making the data analysis to derive information increasingly difficult. Data analysis is a necessary operation to find the hidden patterns and trends in data. Nowadays, almost every Internet user seems to be interested in topics related to IoT. The question that arises here is whether present search engines can find this information or not. The IoT produces a different scale of data, so the search engines should be able to meet the requirements of rapid ingestion and processing followed by accurate and fast extraction. In large-scale IoT systems, search engine techniques are crucial for the efficient retrieval of desired data.

Nowadays, half of the world population uses the Internet and most of them use search engines [3]. The information related to the latest breakthroughs of IoT is also searched with the help of search engines. The principal challenge for search engines is to achieve an *optimal ranking* of returned web pages based on users' information need regarding IoT. It has become very important to observe the quality of search results in dynamic environments of IoT and Big Data. The performance-related statistics found in such studies can be used by search engine industries to offer improved and up-to-date services. Measuring the effectiveness for a larger database of search engines by using human judgment is becoming impossible owing to cost, time, and regular scheduling. With these points in mind, relevance assessments should be transformed toward automatic schemes.

This study examined the ranking performance of search engines against IoT-related queries. Additionally, we proposed a method of relative performance evaluation of search engines using an automatically prepared sustainable optimal rank list for IoT-related searches. Mostly automatic relevance assessments have performed using two approaches: a black box approach and a clear box approach. The black box approach gave search engines the freedom to use different methods. To evaluate the performance of a web search engine, we need a set of web pages, a set of queries, and relevance-judgment-related information about each web page. Usually, it is difficult to obtain relevance responses for all indexed pages. Therefore, pooling (fraction of top similar documents regarding queries and relevance assessment) is used by the Text Retrieval Conference (TREC) to solve the problem of size [4]. Zobel studied the reliability of pooling while evaluating the performance of Information Retrieval (IR) for a large set of web pages [5]. Harter and Voorhees [6,7] found that assessors usually disagree while assessing relevance but this disagreement does not matter for the assessment of the relative effectiveness of IR.

Modern search engines consume diverse kinds of signals to categorize web pages, to find locations, to read behavioral aspects, etc. Search engine rank completes the task of ranking with the help of various signals while searching.

Scoring: Modern search engines manipulate hundreds of signals. A variety of algorithms used these signals as input in different combinations for calculating scores for the ranking.
Boosting: These search engines consider multiple factors in the calculation of the final score. They give some factors more weight than others, and if a page holds such factors than it will be promoted to the rank. For example, fast mobile sites are boosted while user searching from the Google app.
Dampening: These factors can affect the higher rank of a web page. It is taken as a penalty; this factor stops a page from getting a top rank if it does not satisfy the condition.

The scheme described in this paper is not the first work related to the topic of automatic search engine performance evaluation. Thus far, researchers [8–11] have presented automatic evaluation systems for web search based on rank aggregation, data fusion, AWSEEM, and wisdom of crowds, among others. As we discussed above, IoT is the main emerging technology. Here, we proposed an automatic performance evaluation scheme for search engines based on Pseudo Relevance Feedback (PRF) for IoT-related user queries. We prepared information needs related to the top 10 most Googled IoT topics. We followed the topics given in study [12], where the rank presents the sum for number

of searches at Google, Twitter, and LinkedIn. We then prepared 10 queries related to each of these topics with the help of students. After performing some processing on these 100 information needs, we selected 60 of them. We translated each of these information needs using an offline-made page corpus. After translation, the scheme executed the top-K translated queries on search engines and prepared a pool of documents. Afterward, the scheme calculated the score of text similarity among queries and web pages. Using this PRF-based scheme, we prepared a sustainable optimal rank list to measure the relative performance of the search engines. Lastly, the proposed PRF scheme performed boosting based on location and freshness of web pages to achieve a sustainable optimal rank. The proposed approach automatically figures out the relevance of web pages. To the best of our knowledge, this is the first study that examines the ranking performance of search engines for IoT-related queries. This study verifies the effectiveness of the proposed scheme for sustainable optimality with the help of evolution metrics based on discounted cumulative gain (DCG), normalized DCG (nDCG), and Mean Average Precision ($MAP_n$). In this work, we compare the proposed scheme results with two different search engines (Bing and Yandex). The experimental results showed that proposed scheme achieves satisfactory results for all the metrics used in this work. Table 1 below shows all the terms and abbreviations used in this work.

**Table 1.** Symbols and abbreviations used in the paper.

| Term | Description | Term | Description |
|---|---|---|---|
| PRF | Pseudo-relevance feedback | $q_i$ | One of the rewritten query |
| DCV | Document cut off values | $d_{ui}$ | Degree of URL i |
| $u_i$ | URL retrieved in a search result | $T_{wp}$ | Text of web page |
| $PR_s$ | Pseudo-relevance score | $SS_t$ | Similarity score with text |
| $PR_{st}$ | Pseudo-relevance score for time sensitive pages | $SS_u$ | Similarity score with URL |
| $PR_{sg}$ | Pseudo-relevance score for GEO-sensitive pages | $C_t$ | Time sensitive query class |
| $\alpha$ | Page score boosting factor | $C_{nt}$ | Non-time sensitive query class |
| B | Page score dampening factor | $S_{new}$ | Score for newness of web page |
| $C_g$ | Geo-sensitive query class | SSE | Selected search engine |
| $C_{ng}$ | Non-Geo-sensitive query class | $Q_c$ | Confidence score of query |
| $F_u$ | Frequency factor for each URL | $V_t$ | Vocabulary related to time words |
| $V_g$ | Vocabulary related to geographic words | | |

The remaining paper is organized as follows. Section 2 explains earlier work related to the evaluation of IR systems. Section 3 describes in detail the proposed a scheme for automatic evaluation of search engines. Section 4 describes the dataset and evaluation matrix used in the proposed work. Section 5 describes the experimental results. Finally, we conclude the paper and discuss future work in Section 6.

## 2. Related Work

Relevance has a vital role in information sciences and plays a key part in various IR models [13]. Human evaluation for checking web page relevance with given queries is a multifaceted procedure that needs harmonization of numerous intellectual jobs [14], which is difficult to manage for large systems such as a search engine. The relevance feedback from users proved very helpful in evaluating the relevance of returned web pages. Relevance feedback is a technique in which we first input a query and use their results for future queries; there are three kinds of relevance feedback [15].

Explicit Feedback (EF): This kind of feedback is usually started by assessors against a group of documents retrieved for a query (Relevance Judgment). It can be graded using numbers, characters, or binary (relevant or irrelevant) relevance systems. This kind of information is used in the query and documents. Usually, nDCG is used for this kind of feedback.

Implicit Feedback (IF): This is based on user behaviors (view, copy, paste, etc.); in this kind of feedback, the user directly assesses the relevance of web pages or documents. However, the user is not aware of this process. For example, dwell time shows the time spent by users on the web page.
Blind Feedback (BF)/Pseudo Relevance Feedback (PRF): This is an automatic way of judging the relevance between query and documents. Using this method, we can find top-K relevant document among a set of results. This scheme consists of following steps:

(1). Take the results returned by initial queries (top-K);
(2). Select the top 20 or 30 terms from these documents using some scheme;
(3). Execute query rewriting and then match with returned documents to find the top relevant results.

The results of an earlier study [16] show that query expansion based on BF/PRF can find all the results that are missed by the initial query. This method relies on the choice of 300 to 530 expansion terms. The main problem in this scheme is that it can suffer from query drift because of too many terms, although their proposed scheme is able to achieve 7–25% effectiveness. Soboroff et al. [17] proposed a random-based mechanism to label the document as relevant or irrelevant. They randomly select documents against a query and then evaluate these. Using TREC-4 and -6, they showed that there was a low correlation among different relevance judgments for a document. They found 0.938 correlation score for selected ranking systems for TREC-4. Random selection was unable to achieve a high ranking on search engines, where it is difficult to find relevance judgments. In [18], the authors consider a page as an answer to the query if page title has query terms in it. Their approach is mostly successful for navigational queries and their scheme is based on query logs, which are not available nowadays. They perform their experiments using Open Directory Project (ODP), and used a Mean Reciprocal Rank (MRR)-based evaluation to compare the performances of search engines. The problem with their work is that ODP does not treat all languages equally. Mahmoudi et al. [19] perform same work for the Persian language. The study in [20] is based on finding the commonness or overlapping between a query and the results given by a search engine. The authors first select a group of queries and then execute these queries on the SSEs one by one to select a certain DCV. Then, they compare these results lists of a search engine with those of other search engines. The sum of the top results in common will be used as the score. This score is used for the ranking of results (reference counting). The evaluation matrix they used is based on R-precision and precision. Joachims et al. in [21] proposed a semiautomatic approach. With the help of workforce, they collected clicks data. However, this human source does offer relevance judgment. The hired workforce is used to send queries and clicking on results for any kind of information need. They used two search engines (Google and MSN search) and shows mixed results of both queries for a search initiated by them. In AWSEEM 2004 [10] authors gathered some 25 information needs from users and then prepared queries to represent the information needs. Once they had prepared the queries, they executed all of them on SSEs (All the Web, AltaVista, HotBot, Infoseek, Lycos, Msn, Netscape, and Yahoo). After executing all the queries, they prepared a pool of 200 results and re-ranked based on relevancy. They applied DCV on the re-ranked results and considered them relative to the need. Top-t = Top-s. Then, they calculated the correlation between Top-t and Top-s also between the manual and automatic scheme. The main drawback of their approach is the only used textual content to build Pseudo Relevance. In [22], the authors construct a training dataset based on user behavior. They used this for learning functions for IR. The implicit feedback from users proved very helpful in evaluating the relevance of returned web pages. This implicit feedback is then compared with explicit feedback. The authors used click-through information as a preference sign for results; while checking the relevance in their study, they found that those implicit relevance signals are less consistent. A study in [23] used user behaviors as a sign of relevance check. The authors considered distinct levels for a relevance model in their work. They consider saving and scrolling. Based on these actions of users, they decide the relevancy. This observation makes this approach powerful. However, it still needs human behavior, which is sometimes difficult to

arrange. This approach is also very difficult to implement. In [11], the authors used data fusion based on voting such as Borda Count and Condorcet for ranking of results. They prepared a Pseudo Relevant set with the help of data fusion. They mixed results of the search engine using different methods and consider the top-t as relevant ones. Based on these top-t documents, they evaluate the effectiveness of IR systems. They also consider overlapping among search engine results. The study in [24] is about the classification of queries. They assume mostly for a single query that if users click on a single result, then this is a navigational query and otherwise, it belongs to other classes. The study in [25] measures search engine performances with the help of navigational queries and user behavioral data. For each navigational query, there is one right answer. They calculate the MRR measure. The problems with their study are that they used only navigational queries and they compare very few search engines. In [26], the authors combined different rankings for documents to find a new rank. They use a learning mechanism for finding ranking rules. They find results for queries using four selected methods. Then they cross-check these results for the learned ranking rules. The work in [27] uses the scheme of AWSEEM and combined it with PageRank and AlexaRank to improve the scores. The studies in [28,29] represent an effort to find trends based on an association between authors and institutions for IoT-related topics. The article [30] is a study regarding adaptive smart homes using user-created feedback and [31] discusses the evaluation of self-monitoring devices for clinical purposes. Singh and Dwivedi in [32] study the reason (e.g., only one document in the corpus or if a number of documents containing query terms and total documents are same) behind the failing of cosine similarity. After studying the impacts of these factors, they proposed enhanced from of vector space model to improve efficiency. In [33], Lewandowski et al. take randomly selected 1000 informational and 1000 navigational queries for a major German SE and compare their performance with Google and Bing. According to their results, Google is able to establish the precise responses in 95.3% of cases, while Bing only yields the right response 76.6% of the time for navigational queries.

## 3. Sustainable Optimal Rank Preparation for Relative Assessment of IoT-Related Searches

Most users, whether they are from academia or the general public, are interested in the knowledge of IoT-related aspects. However, because of their limited domain knowledge, they face difficulties while making queries. Therefore, they seek help from search engines to get the desired result. Modern search engines take seed queries as input and fetch related web pages based on some scoring criteria, ranking them accordingly. These rank scores are known as relevancy scores, which are calculated using different methods. While evaluating search engine performance, we evaluate the returned rank lists using measures such as MRR, DCG, nDCG, and MAP@n.

As we described in the previous section, the PRF is an automatic way to get the relevance between a query and documents. This section describes the proposed scheme for checking relevance, which is based on a PRF-related approach. The proposed scheme is shown in Figure 1.

In the proposed scheme, we use 60 seed queries against 10 selected topics. These seed queries are prepared with the help of a group of students. First, we need to build a page corpus for query rewriting. Therefore, the proposed PRF scheme executes these seed queries on two search engines, Bing and Yandex, and prepares a page corpus. For this purpose, we manually selected 10 pages for each seed query and stored them in the word vector form. Afterward, we re-write these seed queries with the help of the page corpus, select top-2 queries, and execute them on the two search engines. During our experiment, we found that this second execution provides more relevant pages than the seed query. Afterward, the proposed scheme stored the web pages using different document cutoff values (DCV) {5, 10, and 20} for each executed top-2 query. These stored pages are used to calculate the relevancy score based on text similarity and boosting parameters of each page. Using this calculated pseudo relevance score, we re-ranked the retrieved web pages and prepared the sustainable optimal rank list for PRF. Afterward, a comparison between the sustainable optimal rank list and the search engine results lists is performed. This is how the relative performance evaluation of search engine is conducted. We evaluate the performance with the help of DCG, nDCG, and MAP@n.

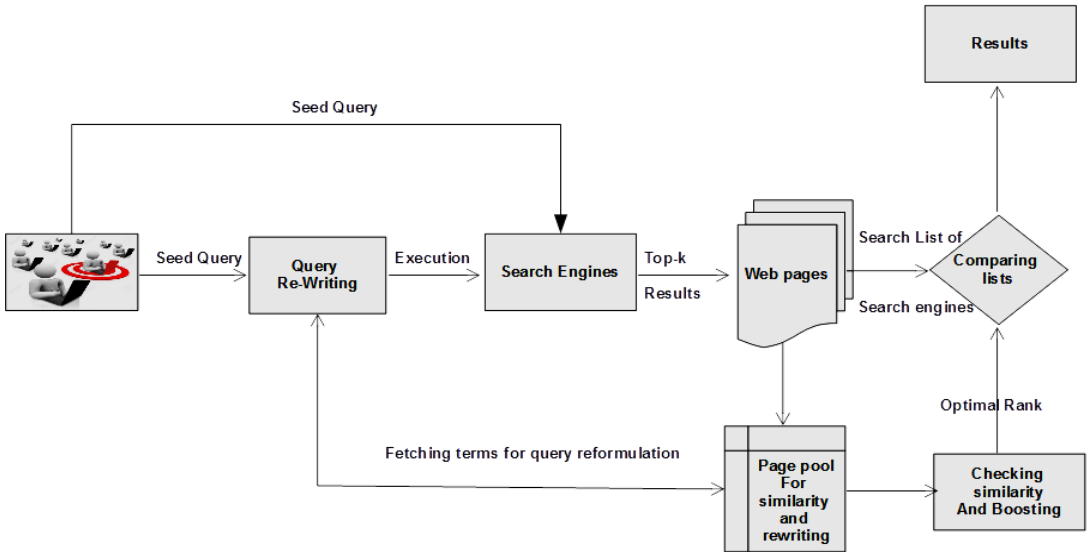



**Figure 1.** Automatic evaluation of the relative performance of search engines for IoT-related queries.

### *3.1. Formulation of Information Need*

In the past few years, the fields of the IoT and Big Data have evolved very rapidly, and their combination has increased the growth of data. Every other day, some new research or breakthrough appears on the web and a new company announces a new IoT product. Therefore, the topicality of IoT domains keeps changing every day. Thus, it is difficult to cover every query and topic in this research.

For this work, we selected the top 10 topics of IoT shown in the study by IoT ANALYTICS [9]. These topics are listed in Table 2, according to the number of searches made on Google. We select 10 students and teach them about these topics for two months. Afterward, we ask them to prepare at least one query about each topic as an information need for submission on the search engine. Hence, in this way, these students initially prepare 100 information needs. After removing similar or converging queries, we selected 60 seed queries for the experiments.

**Table 2.** Topics used for the construction of information needs. Here, the topic ranking is given according to the number of searches on Google.

| No. # | Topic | # of Searches |
|---|---|---|
| 1 | Smart home | 61 k |
| 2 | Smart City | 41 k |
| 3 | Smart grid | 41 k |
| 4 | Wearable | 33 k |
| 5 | Industrial Internet | 10 k |
| 6 | Connected Car | 5 k |
| 7 | Connected Health | 2 k |
| 8 | Smart retail | 1 k |
| 9 | Smart farming | 1 k |
| 10 | Smart Supply Chain | 0 k |

### *3.2. Query Rewriting*

The user queries given from inexperienced users are usually short and ineffective [13]. After obtaining the seed queries, we expanded them in an offline manner using PRF. PRF usually used the terms for retrieval that are most frequent (term distribution) in the top-ranked relevant documents [12]. For example, one student is interested in buying a smart light for his smart home. He or she will write a query about "smart light" on Bing or Yandex to search the Internet, as shown in

Figure 2. According to Figure 2, for the Bing search engine, the top-3 results are not relevant to the information need and for Yandex top result is not relevant. Consequently, it can be assumed that the rank of results for the information need is not correct. Hence, the query should be rephrased to achieve better results. The proposed PRF scheme rewrites the seed queries using the page corpus; we predict that by doing so, better results can be fetched on top of the sustainable optimal rank list. Suitably, the PRF extends the seed query "smart light" with attributes such as "Company = TP-Link, Detail: Smart LED Light Bulb, Network type = Wi-Fi, Color = Dimmable White, Voltage = 50 W Equivalent, Price: $19.99." For the experimental work, we prepared an attribute corpus for all such information needs. The corpus stored each attribute for an IoT-related entity in a comma separated value text file. The number of attributes in each definition can be different. This work extends the queries by adding terms one after the other and making different candidate queries.

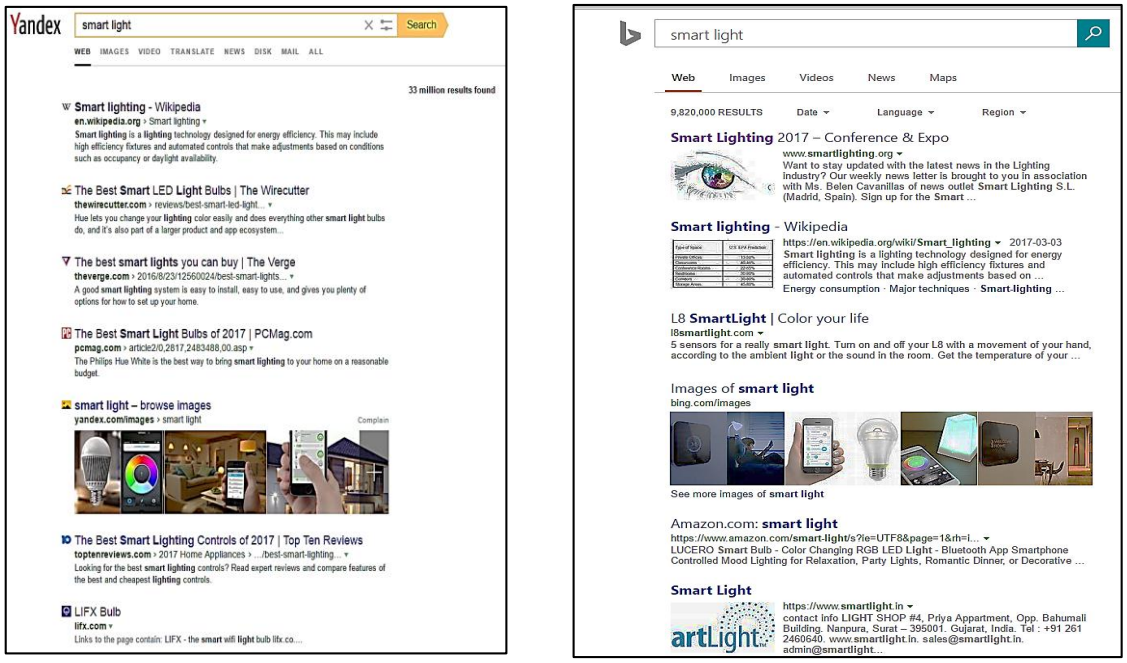



**Figure 2.** Search results for seed query "Smart Light" for (**a**) Yandex and (**b**) Bing search engines.

### *3.3. Execution of Queries*

Once we formed this candidate query set by rewriting the seed queries, the execution of these rewritten queries on Bing and Yandex fetched the results list for different DCV values {5, 10, and 20}. This execution of queries gave us the pool of web pages used in selecting the top-K queries and top-URLS, and checking for similarity with the query. After fetching the results, every list about the executing query is stored in the database. For example, if the scheme executed a rewritten query for smart light that has the keyword "TP-Link" + "Smart LED Light Bulb" on Bing, it returns list as shown in Table 3. On $DCV_5$, the shown five URLs are returned by Bing.

**Table 3.** Results for query "TP-Link" + "Smart LED Light Bulb" on Bing.

| Label | URL |
|---|---|
| $u_1$ | http://www.tp-link.com/us/products/details/cat-5609_LB100.html |
| $u_2$ | https://www.amazon.com/TP-Link-Dimmable-Equivalent-Assistant-LB100/dp/B01HXM8XF6 |
| $u_3$ | http://uk.tp-link.com/products/details/cat-5609_LB110.html |
| $u_4$ | http://uk.tp-link.com/products/details/cat-5609_LB120.html |
| $u_5$ | http://www.homedepot.com/p/TP-LINK-60-Watt-Smart-Wi-Fi-LED-Bulb-with-Energy-Monitoring-LB110/207104829 |

In Table 3, the label shows the rank returned by the search engine to each URL. Among the returned resulting URLs, $u_1$, $u_3$ and $u_4$ show the same link. In an analogous manner, we store the results returned for each rewritten query.

### *3.4. Selecting Top-K Queries*

After executing all the queries, we prepared a database of all the information needs. Then, PRF extracts the number of unique URLs and how many queries return that URL using the database. Subsequently, we re-ranked the URLs based on their frequency count. In this work, we selected the top-K rewritten queries using this frequency score, with the help of random walk on a query–URL bipartite graph. Bipartite graphs represent a mapping between the queries set Q and the URL set U. The scheme models the web search as a query–URL bipartite graph G = (Q, U, E), where nodes are divided into two separate sets: Q, the set of queries, and U, the set of URLs. Every edge in set E links a query and a URL. There is an edge between a query and a URL if the query fetches the web page. Using this many-to-many relationship among queries and URLs, PRF created a bipartite graph. In the proposed scoring scheme, each record has a query and URLs with their proper degrees. In a bipartite graph, the edges in E represent the confidence values of the query and pages in the search results. Figure 3 represents the proposed example of a bipartite graph. In this probabilistic retrieval model, the scheme fulfills an information need with a query set Q. Given an undirected graph G = (Q, U, E) with $l = |Q|$ queries, $m = |U|$ URLs, and $n = |E|$ edges, the natural random walk starts with one of these random queries, $q_i$, and then selects one of the fetched URLs using the $n \times n$ transition matrix M, as shown in Table 4.

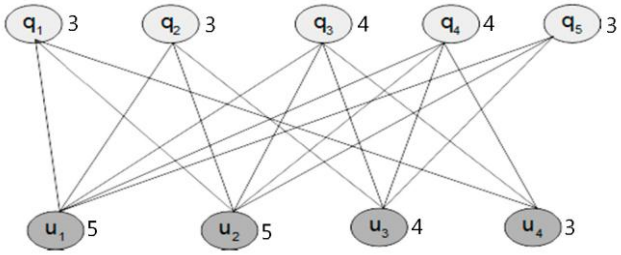



**Figure 3.** The bipartite graph for the sample table shown in Table 4, which has four URLs and five queries.

**Table 4.** Transition matrix M shows the random walk probabilities for the query–URL bipartite graph shown in Figure 3.

| URL | $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_5$ |
|---|---|---|---|---|---|
| $u_1$ | 1/5 | 1/5 | 1/5 | 1/5 | 1/5 |
| $u_2$ | 1/5 | 1/5 | 1/5 | 1/5 | 1/5 |
| $u_3$ | 0 | $\frac{1}{4}$ | 1/4 | 1/4 | 1/4 |
| $u_4$ | 1/3 | 0 | 1/3 | 1/3 | 0 |

According to the example, the random walk starts by choosing random query $q_i$ from set Q, and then it selects a random URL $u_i$ connected to $q_i$; from this URL $u_i$, it then selects another connected query, and so on. This process of the query–URL transition is repeated, or it stops at a query or URL node, based on the value of n. Moreover, the scheme limits the number of transitions and keeps them according to the information need. It is also a simplifying assumption that the proposed random walk only visits an edge once. That means if an edge connects two nodes, once it traversed them, the scheme will choose a new random edge the next time. The random walk follows Algorithm 1 and executes in the following manner, where the notations used are heavily influenced by the study from Jaakkola and Szummer [34]. Let U be the set of URLs, and let Q be the set of queries; the scheme

constructs a bipartite graph G. In G, the number of edges E, assigns weights to the queries and URLs, given by incoming edge counts (degree $d_{qi}$; $d_{ui}$) of each node. Scheme defines transition probabilities $P_{t+1|t}(u_i \mid q_i)$ from query $q_i$ to URL $u_i$ and vice versa, so $P_{t+1|t}(u_i \mid q_i) = 1/d_{qi}$, where i ranges over all connected URL nodes with a given query. The notation $P_{t2|t1}(q_i \mid u_i)$ will denote the transition probability from node $u_i$ at time $t_1$ to node $q_i$ at time $t_2$, while transition probabilities $P_{t+1|t}(u_i \mid q_i)$ are not the same because the number of degrees varies across nodes. Let the transition probability be:

$$P_{t+1|t}(q_i \mid u_i) = \begin{cases} \frac{1}{d_{u_i}}, \text{ if } uq \in E \\ 0, \text{ otherwise} \end{cases} \quad (1)$$

We organize the transition probabilities as a matrix M whose $u_i$:$q_i$ entry is $P_{t+1|t}(u_i \mid q_i)$. The matrix M is row-stochastic, such that the rows sum to 1. Consequently, we perform the random walk and calculate the probability of transitioning and degrees from node $q_i$ to node $u_i$ in time t, denoted by $P_{t|t-1}(u_i \mid q_i)$, which is equal to $P_{t|t-1}(u_i \mid q_i) = Mt\ (q_i u_i)$. Algorithm 1 is the pseudo code of the random walk process on a bipartite graph. According to Algorithm 1, a single node is visited once. Matrix R will store the degrees of URLs in reference $q_i$:$u_i$ transitions. Both these data structures help in the confidence score calculation for each query and URL. The confidence of a query or URL is the ratio of the co-occurrence of the query/URL pair in the collection of results. The ratio value is always $\leq 1$, which makes it easier to process than showing the actual high co-occurrence values. The confidence scores are calculated as follows:

$$Q_c = \frac{\sum_i (d_{q_i})}{\sum_u (d_u)} \quad (2)$$

Equation (2) shows the confidence value of query $q_i$. Here, the numerator is the sum of the degrees of all URLs fetched by qi and the denominator is the sum of degrees for all URLs fetched by all queries.

**Algorithm 1: Random Walk on a Bipartite Graph**

**Input: $q_0$ = seed query, run size n = |E|.**

**Output: Sample queries and their degrees; Sample Pages with Degree.**

```
1: Matrix R and list Us are empty;      //Matrix R and list Us will store degrees of URLs because of
Random Walk.
2: i = 1;
3: while i ≤ n do
4:    u_i = random (URL);            //Assign one random URL which is fetched by the query.
5:    if U_s(cell)! = Null then
6:       U_s(cell) = d_ui;           //Store degree of URL both in Us and R matrix against q_i u_i.
7:    else
8:       u_i = random (URL)
9:       U_s(cell) = d_ui;
10:      Q_i+1 = random (q_i);       //Assign one random Query which is fetching URL.
11:   end if
12: end while = 0
```

During each iteration, the random walk visits one of the nodes from the query or the URLs and updates the matrix R. Here, the scheme fills the cells using source and destination node name $q_i u_i$, and enter the degrees of the URL in the proper cell. Using these degrees shown by R, we calculate the confidence in the query rank queries according to their confidence scores. Table 5 shows that the top-2 queries in the random walk are $q_3$ and $q_4$ with scores 1.

**Table 5.** Top-2 query suggestions using random walk on bipartite based scheme with the help of R matrix.

| Q # | $d_{u1}$ | $d_{u2}$ | $d_{u3}$ | $d_{u4}$ | $\sum(d_{q_i})$ | $Q_c = \frac{\sum(d_{q_i})}{\sum(d_u)}$ |
|---|---|---|---|---|---|---|
| $q_3$ | 5 | 5 | 4 | 3 | 17 | 1.00 |
| $q_4$ | 5 | 5 | 4 | 3 | 17 | 1.00 |
| $q_2$ | 5 | 5 | 4 | | 14 | 0.82 |
| $q_5$ | 5 | 5 | 4 | | 14 | 0.82 |
| $q_1$ | 5 | 5 | | 3 | 13 | 0.76 |

Note: Here, $q_3$ and $q_4$ are suggested for seed query "smart light" shown by right-most columns.

### 3.5. Text Similarity Score

The proposed scheme uses the vector space model for calculating text similarity score between query and text extracted from web pages and URLs of the links. The proposed scheme starts from seed query processing, as shown in Figure 4. First, it recognizes the language of the query and then tokenizes the seed query. Afterward, it performs some preprocessing such as spelling correction and segmentation and expands the seed query using the rewriting technique explained above. Once it has created the set of candidate queries, the scheme deploys the top-query selection technique. Afterward, we select top-2 queries among rewritten queries. Henceforth, we only use these two queries for checking text similarity with the page text and fetched URLs. Subsequently, the scheme classifies the query and finds location and newness related words in them and, finally, finds the entity the user is searching for.

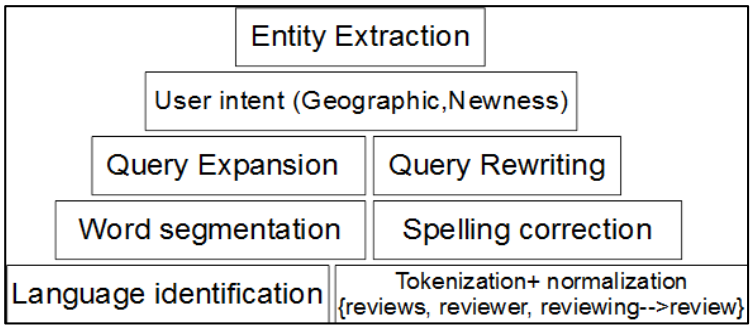



**Figure 4.** Query model used for text similarity in the proposed scheme.

The scheme prepares a term matrix TF-IDF for both the query and text of web pages. TF-IDF is one of the oldest and most well-known approaches that represent each query and document/web page as a vector. Using this vector representation, the PRF scheme calculated the cosine similarity score between the information need, which is described by an attribute-oriented definition and text documents made after extracting text from web pages and URLs of the links. We removed stop words from each web page text and do not perform stemming because it can change the intention or meaning. The scheme makes use only pages fetched by all queries for the all three selected DCV values {5, 10 and 20} for similarity check. From each URL in the list, the scheme extracted the text using the Alchemy API (http://www.alchemyapi.com/). Each obtainable web page and URL text are saved separately to build corpuses, for each IoT-related information need. We assume dead links to be irrelevant during this process. The same page URL retrieved by multiple search queries is considered once for text similarity check, as we have already saved its frequency. The extracted text from all unique URLs links and pages is stored as $u_i$ and $T_{wp}$. After text retrieval from URLs, the scheme performs some basic text processing on each $T_{wp}$ and URL $u_i$. Nowadays, search engines return results in different sections, such as related video, news, and blogs, among others. Thus, we filter retrieved URLs and consider only unique web page URLs returned for the query. Afterward, we perform normalization on

the retrieved URLs. URL normalization is a process to convert URLs into a standard and consistent format. Uniform resource locators (URLs) comprise a lot of information. For example, the URL "*https://www.amazon.com/TP-Link-Dimmable-Equivalent-Assistant-LB100/dp/B01HXM8XF6*" suggests a home page of "Amazon" having the company name "TP-Link", light type "Dimmable", etc. After normalization of URLs, a segment of the URL provides "www.amazon.com" which becomes "www", "Amazon" and "com", or lengths of tokens, orthographic features, sequential n-grams, and sequential bi-grams. We tokenize URLs with the help of nonalphanumeric characters as boundaries and remove English stop words, including web-specific stop words, as well as file and domain extensions, etc. Then, we generate n-grams of length 2 and 3 for URLs and web page text. Punctuation marks (",", "." etc.) are removed. In this work, the extracted n-grams from URLs are used to check similarity with the query. n-grams that consist of stop words only or that do not contain at least one alphanumeric character (e.g., n-grams such as "at the" or "#,@") are removed. We drop sparsity in this work by constructing a vector for only query terms, not considering the other terms from a web document. The bag of words approaches faces problems if the terms have any spelling mistakes (e.g., with a unigram). Hence, we used n-grams rather than a collection of unigrams, where occurrences of pairs of consecutive words are counted. We removed stop words from the each $T_{wp}$ and gave query–URL and query–wp vector forms. The computation of cosine similarity between a query and URL and query and $T_{wp}$ are named as $SS_u$ (similarity score of URL) and $SS_t$ (similarity score of text), respectively. The score of similarity for every fetched URL is calculated in the following fashion:

$$SS_u = \text{cosine}(Q, u_u) \quad (3)$$

$$SS_t = \text{cosine}(Q, u_t) \quad (4)$$

Equations (3) and (4) show the cosine similarity scores between query $q_i$, $URL_i$, and $T_{wp}$. We use three factors for calculating relevance score are web query–wp text similarity, query–URL text similarity and normalized frequency of URL ($F_u$). The normalized $F_u$ calculation is shown in Equation (5). We calculated it by dividing degree of $u_i$ by sum of all URLs degrees. For example, the degree of URL $u_1$ in Table 5 is 5; we normalized it by dividing with 17, which is the sum of all URLs degrees. Equation (6) shows the Pseudo Relevance score ($PR_s$) for the pages fetched for the IoT-related information need.

$$F_u = \frac{d_{u_i}}{\sum_u (d_u)} \quad (5)$$

$$PR_s = SS_t + SS_u + F_u \quad (6)$$

Using the $PR_s$, we will create an initial rank of all the fetched web pages and compare it with the rank returned by the search engines

### *3.6. Boosting-Based Page Re-Rank*

The initial rank created on the basis of $PR_s$ considers all the web pages fetched using a query set while ranking. Afterward, the proposed PRF scheme selected the top-n results from the rank list prepared by $PR_s$ and applied boosting criteria based on newness and locality of each web page. We do so based on some concrete motives, as initially the number of pages can be very high and applying boosting criteria on each fetched web page can be time-consuming. That is why we are applying boosting based re-ranking on the top-n results. While using different DCVs, we found that, in most cases, the number of irrelevant URLs increases drastically after the 20th URL. Therefore, we re-ranked only the top 20 URLs after calculating the boosting scores.

#### 3.6.1. Newness-Based Re-Rank

Newness-based re-rank is based on the time feature. Therefore, the URL that is most recent should be presented before the other pages while ranking. Here, we first explore whether the provided query

is time-sensitive or not. Users often trying to explore the things or document based on their time. For example, the user is searching for "latest Smart Lights" and enters the same keywords as a query. In the given query, the word "latest" shows that the query is time-sensitive and the resulting rank of this query should boost the latest pages. If the search engine ranks an older page or irrelevant page higher, it creates a bad image of the search engine in user's mind. We used query classification method to divide the query into time-sensitive and non-time-sensitive classes. Text classifiers regularly use a document denoted as a BOW. This is a simple illustration: consider a query $q_i$, whose class is given by $C_t$. In the case of queries, there are two classes $C_t$ = Time-sensitive and $C_{nt}$ = non-time-sensitive. We classify $q_i$ as the class which has the highest posterior probability $P(c_i \mid q_i)$, which can be re-expressed using Bayes Theorem:

$$P(c_i|q_i) \;=\; P(q_i|c_i)\,P(c_i)\,P(q_i) \;\propto\; P(q_i|c_i)\,P(c_i) \tag{7}$$

If we have a vocabulary V, having $|V_t|$ word types, then the feature vector dimension $d = |V_t|$. The Bernoulli query model is represented by a feature vector with binary elements taking the value 1 if the corresponding time-sensitive word exists in the query $q_i$ and 0 if not. Consider the vocabulary example:

$V_t$ = {latest, up-to-date, state-of-the-art, advanced, hot, modern, new, newest, etc.}

Now, consider the query "latest Smart Lights", then its Bernoulli model is $q^b$ = (1,0,0). Using this information, we can classify it with the help of Equation (7). This classification helps us in deciding whether we should add the time stamp based factor or not while ranking. Equation (8) shows this time-sensitive relevancy score calculation. Here, $PR_{st}$ shows pseudo relevance score calculation for URL $u_i$, while $PR_s$ shows initial relevancy score and $s_{new}$ = ($\alpha$ or $\beta$) shows score based on time stamp for $u_i$. Here, $\alpha$ is a boosting factor and its value for newness is equal to 1, while $\beta$ is dampening factor and its value is $-1$. In Equation (8), $C_t$ shows the class related to time-sensitive queries.

$$PR_{st} \;=\; \begin{cases} PRs(u_i) + s_{new}(u_i), \text{ if } u_i \in C_t \\ PRx(u_i), \text{ otherwise} \end{cases} \tag{8}$$

Hence, here we proposed a Page Rank Boosting (PRB) scheme for time-sensitive queries. In PRB, first, we check the time-sensitive nature of the query and consequently apply PRB on the top-20 pages from the initial rank.

Table 6 shows the effects of boosting on and the promotion or demotion for different labeled documents from the initial rank list. According to the scheme, if the time-sensitive detail is not attached to a web page, then the dampening factor damps it, whereas, if a page is labeled as somewhat useful and has a time stamp label new, then PRB boosts its score to useful, and, if the time stamp is old, then PRB demotes it to not useful. In the PRB scheme, we consider that if the label is not useful that shows it belongs to irrelevant class so PRB neither promotes it nor demotes it. The PRB helps a lot in improving the DCG of the rank list towards the ideal DCG. We consider the number of relevancy labels (3) and time-sensitive stamps (3) and they are equal. To check the newness of web pages, PRB used the in-house made scraper to find out the dates in the web page HTML code. This work uses the latest date in the code as the freshness of page rather than its inception date. If a page code does not return a date within the last one year, we consider it as old.

**Table 6.** Boosting scores for queries having a timestamp to calculate initial scores given by the PRB scheme.

| | **New** | **Old** | **Not Given** |
|---|---|---|---|
| Useful | Useful (no change) | Somewhat useful (damped using $\beta$) | Somewhat useful (damped using $\beta$) |
| Somewhat useful | Useful (boosted using $\alpha$) | Not useful (damped using $\beta$) | Not useful (damped using $\beta$) |
| Not useful | Not useful (no change) | Not useful (no change) | Not useful (no change) |

3.6.2. Geo-Sensitive-Based Re-Rank

Text similarity and time-sensitive feature-based ranking of results are helpful for the user. However, when the users are looking for retail stores or services around some particular locations, then these are not sufficient. Hence, here we proposed a PRB scheme for geo-sensitive queries. In this scheme, PRB first checks the geo-sensitive nature of the query and if it is geo-sensitive, then PRB is applied to the top-20 pages from the initial rank. For example, the user is interested in buying or availability of some IoT-related item near his geographic location but while searching he gets the results which are not around him. The PRB names these queries as geo-sensitive queries. These results will create an adverse impact on the search engine on the user. Hence, here we discuss the location-sensitive boosting method for pages. Like the time-sensitive queries, the PRB only boosts those queries which have some location-determining word in them. For example, if query $q_i$ "smart lights available in Daegu" contains a word such as a city name or a phrase such as "nearest shop of smart lights", the PRB classifies these queries as GEO queries. Like the newness-based boosting, let us consider a query $q_i$, whose class is given by $C_g$. In the case of queries, there are two classes $C_g$ = GEO-sensitive and $C_{ng}$ = non-GEO-sensitive. The PRB classify $q_i$ as the GEO class if it has highest posterior probability $P(c_g \mid q_i)$, which can be re-expressed using Bayes' Theorem:

$$P(c_g|q_i) \;=\; P(q_i|c_g)\,P(c_g)\;P(q_i) \;\propto\; P(q_i|c_g)\,P(c_g) \tag{9}$$

If the PRB has a vocabulary $V_g$, having $|V_g|$ word types, then the feature vector dimension $d = |V_g|$. The Bernoulli query model is represented by a feature vector with binary elements taking value 1 if the corresponding time-sensitive word exists in the query $q_i$ and 0 if not. Consider the vocabulary example:

$V_g$ = {adjacent location, local, nearest, nearby, besides, site, locality, Daegu, Seoul etc.}

Now consider the query "Smart Lights available in Daegu" then its Bernoulli model is $q^b$ = (0,0,0,0,1). Using this model, the PRB can classify the query with the help of Equation (9). This classification helps us in deciding whether PRB should add the location-based factor or not while ranking.

$$PR_{sg} \;=\; \begin{cases} PRs(u_i) + s_{geo}(u_i), \text{ if } u_i \in C_g \\ PRs(u_i), \text{ otherwise} \end{cases} \tag{10}$$

Equation (10) shows this geo-sensitive relevancy score calculation. Here, $PR_{sg}$ indicates a pseudo-relevance score calculation for URL $u_i$, while $PR_s$ indicates an initial relevancy score and $s_{geo}$ = ($\alpha$ or $\beta$) shows score based on time stamp for $u_i$. Here, $\alpha$ a boosting factor and its values for location are equal to the $PR_s$ label while $\beta$ is dampening factor and its value is $-PR_s$. In Equation (10), $C_g$ shows class related to geo-sensitive queries.

To improve the ranking for geo-sensitive queries, the PRB extracts the locations from the query and URLs. If the query holds a location-oriented label such as "Daegu", then the PRB extracts the location-related information of URLs from the contact us section within the website text, such as the company address, whereas, if the query does not have any city name, then the PRB extracts the IP locations for matching. The PRB stores the location information in a separate vector. Once we have the locations for both the queries and URLs, the PRB can easily compute the similarity cosine ($q_i$, $u_i$). When the cosine between these are 0, it means there is no location match, and the page should be demoted, while if it is other than 0, then it should be promoted to the rank according to the Table 7. The proposed boosting of page rank list for geo-sensitive queries takes the initial ranking of the page for boosting. The PRB performed a list-wise comparison of URLs while boosting; for example, $p_1$ is top rank page according to initial rank and labeled as useful. There are two possibilities that $p_1$ has no location and should be penalized, so the PRB uses a dampening factor, which is $\beta = PR_s$, and for

useful it is (−2). However, if $p_1$ shows a location match, then there will be no change as it is already labeled useful. The same kind of dampening is performed for pages of all kind labels.

**Table 7.** Boosting of the score for location-sensitive queries we have applied these scores on the initial scores given by the PRB scheme.

| | **Similar** | **Different** | **Not Given** |
|---|---|---|---|
| Useful | Useful (no change) | Not useful (damped using β) | Not useful (damped using β) |
| Somewhat useful | Useful (boosted using α) | Not useful (damped using β) | Not useful (damped using β) |
| Not useful | Not useful (no change) | Not useful (no change) | Not useful (no change) |

## 4. Dataset and Evaluation Mechanism

IoT is one of the emerging fields of the modern era, and, to evaluate the searches related to it, we need search query log from search engines. Nowadays, however, because of privacy and some commercial factors, it is problematic to acquire real query logs from a search engine such as Microsoft, Yahoo, Google, Bing, and so on. Therefore, for this purpose, we prepared in-house data set. We select 10 students and brief them about these 10 selected topics, as mentioned in Section 3.1. Afterward, we ask them to prepare at least one query about each topic as an information need for submission on the search engine. Accordingly, we prepared 100 information needs initially by all these students. After removing the similar queries, we have selected 60 seed queries for the experiments. We then rewrite these queries and execute them on Bing and Yandex to form a pool of web pages. We executed these queries on all SSE and extracted unique URLs depending on DCV, against each query in common. Consequently, the pool consists of 60 × 100 = 6000 web pages. For the initial ranking, we use the top-2 queries and pages and find text similarities for them.

To check the relative performance of the search engines, we compared it with the sustainable optimal rank list prepared by us. In this work, we tried to achieve optimal rank list for testing the relevance. We check the sustainability of the optimal rank list with the help of relevance judgment prepared by students for all the web pages of the pool. We gave the returned web pages to the student according to their information need and requested them to the label. To evaluate the optimal rank list accuracy, we used label web pages, which annotated by students as a base. The student's label pages as highly useful (2), somewhat useful (1), or not useful (0). The search system that achieves higher performances relative to the sustainable optimal rank list with high accuracies gets a higher score in the evaluation and we hope that it gets more attraction of users. We use DCG, nDCG, and $MAP_n$ for evaluation purposes to check the initial performance of search engine relative to the sustainable optimal rank list at each step in this work. DCG and nDCG rates higher the list that can fetch the highest number of relevant pages and rank them following their relevance score. In the results, DCG and nDCG show the averages for all 60 information needs for each set of experiments. We have obtained absolute relevance judgments from a group of students for all query–URL pairs of the dataset. We report the DCG and nDCG scores for the SSE and rank the list prepared by the methods for the same DCV value {5, 10, and 20}. We also calculated the MAP@n scores for all three schemes with the selected DCV values. In this work, we calculated the precision scores for each of the queries and then calculated its mean average.

## 5. Results and Discussion

The proposed scheme uses query rewriting as the preliminary theme and, after executing the rewritten queries, we prepared a pool using DCV values {5, 10, and 20}. This selection variation helps in proving the efficiency of a search engine for different DCVs. We take these lists of results and used student evaluation initially to check the relevancy of query–URL pairs for each search engine result for each DCV. We used student judgments as the gold standard to check the relevancy of fetched results list against the queries.

Table 8 represent the average results for both the search engines and Optimal Rank performance in the form of DCG, iDCG, and nDCG using the student assessment. These results are calculated for the 60 information needs provided by the students and then averaged. Table 8 shows the DCG values for Bing and Yandex in comparison to the Optimal Rank. The results clearly show that Yandex performs better than Bing for the given information need. The results depict that Yandex fetched more relevant documents than Bing, and gave these fetched documents more appropriate rank for all three DCV values. When we compare Yandex performance with Optimal Rank, we found that the use of rewritten queries fetches more relevant results than both the search engines using seed query. However, the rank is still not ideal as we can see the ideal DCG values in Table 8. An IR system ideal DCG shows the rank achieved after sorting it on their evolution labels in descending form. Consequently, none of the lists can achieve this ideal rank, but the Optimal Rank performance is better than others. Table 8 shows the normalized values for DCG achieved by the all three of them. Here, we use DCV values {5, 10, and 20}, and the results in Table 8 clearly show that as we increase the DCV values the number of relevant fetched documents decreases, which is shown by the lower values in $nDCG_{20}$ column.

**Table 8.** DCG, iDCG, and nDCG scores for the search engines against the seed queries provided by the students and optimal rank using rewriting.

| **SSE** | **$DCG_5$** | **$DCG_{10}$** | **$DCG_{20}$** |
|---|---|---|---|
| Bing | 2.19 | 3.46 | 4.73 |
| Yandex | 3.13 | 6.27 | 7.30 |
| OR | 4.43 | 7.16 | 8.98 |
| SSE | **$iDCG_5$** | **$iDCG_{10}$** | **$iDCG_{20}$** |
| Bing | 3.63 | 6.19 | 8.08 |
| Yandex | 3.63 | 7.89 | 9.25 |
| SSE | **$nDCG_5$** | **$nDCG_{10}$** | **$nDCG_{20}$** |
| Bing | 0.60 | 0.56 | 0.50 |
| Yandex | 0.86 | 0.79 | 0.79 |

Once we have calculated the results for the 60 queries with the help of DCG and nDCG, we filter the queries which contain the time stamp. Out of the 60 queries, there are only seven different queries made by the students that have time-related words in them. We extracted these queries using the classification mentioned in the Section 3.6.1. After separating these queries, we applied same rewriting scheme on these queries and executed them on the search engines to prepare the pool of web pages. From these pages using the technique mentioned in Section 3.6.1, we prepare the time-related information from the queries and web pages and check the threshold gap of one year for boosting and dampening. Once we have found this information we, re-ranked the pool of pages and prepared a list.

Table 9 show the results calculated after the re-rank based on the newness of web pages for time-sensitive queries. For time-sensitive queries, the performances of both the SSE decrease. The DCG values in Table 9 show this effect clearly for Optimal Rank, Bing, and Yandex for all DCV values. In terms of the overall performance, the Optimal Rank + New, which are the results achieved by re-ranking the web pages, shows better performance. During the analysis of the results list, we found that the list produced by re-ranking shows the latest page at the top with 89% accuracy. In the above results, Table 9 show the results for the ideal DCG and nDCG achieved by all of the information retrieval methods. In these results, Yandex performs relatively better than Bing.

Table 10 represent the results for the queries based on the location-related information. We classify the queries using the location vocabulary shown in Section 3.6.2 for the location-sensitive queries. Out of 60 queries, there are only five queries that contain the location-related keyword(s). Using these five seed queries, we rewrite them and generate more related queries to form a pool of web pages. After forming the pool, we calculated the scores shown in Table 10.

**Table 9.** DCG, iDCG, and nDCG scores for the search engines against the seed queries given by the students and optimal rank + timestamp-based re-ranking of results.

| **SSE** | **$DCG_5$** | **$DCG_{10}$** | **$DCG_{20}$** |
|---|---|---|---|
| Bing | 2.00 | 3.14 | 3.53 |
| Yandex | 3.13 | 6.27 | 7.30 |
| OR | 4.10 | 7.00 | 8.32 |
| OR + New | 4.47 | 7.01 | 9.00 |
| SSE | **$iDCG_5$** | **$iDCG_{10}$** | **$iDCG_{20}$** |
| Bing | 3.43 | 6.00 | 7.58 |
| Yandex | 3.50 | 7.46 | 9.00 |
| OR | 4.41 | 7.32 | 9.80 |
| OR + New | 4.41 | 7.32 | 9.80 |
| SSE | **$nDCG_5$** | **$nDCG_{10}$** | **$nDCG_{20}$** |
| Bing | 0.58 | 0.52 | 0.46 |
| Yandex | 0.89 | 0.84 | 0.81 |
| OR | 0.92 | 0.95 | 0.84 |
| OR + New | 0.93 | 0.95 | 0.91 |

**Table 10.** DCG, iDCG, and nDCG scores for the search engines against the seed queries provided by the students and optimal rank + geographic location based re-ranking of results.

| **SSE** | **$DCG_5$** | **$DCG_{10}$** | **$DCG_{20}$** |
|---|---|---|---|
| Bing | 2.10 | 3.12 | 3.48 |
| Yandex | 3.06 | 6.16 | 7.20 |
| OR | 4.00 | 7.01 | 8.20 |
| OR + GEO | 4.15 | 7.04 | 9.01 |
| SSE | **$iDCG_5$** | **$iDCG_{10}$** | **$iDCG_{20}$** |
| Bing | 3.41 | 5.90 | 7.42 |
| Yandex | 3.41 | 7.40 | 8.90 |
| OR | 4.29 | 7.18 | 9.60 |
| OR + GEO | 4.29 | 7.18 | 9.60 |
| SSE | **$nDCG_5$** | **$nDCG_{10}$** | **$nDCG_{20}$** |
| Bing | 0.61 | 0.52 | 0.46 |
| Yandex | 0.89 | 0.83 | 0.80 |
| OR | 0.93 | 0.97 | 0.85 |
| OR + GEO | 0.96 | 0.98 | 0.93 |

Table 10 shows that there is a smaller decrease in the DCG value for the location-based queries than time-sensitive queries. Here, in these tables, the Optimal Rank + GEO shows the impact of location-based re-ranking of results. Similar to the above results, this re-ranking shows better performance than the three other methods, including the basic optimal rank method. The same phenomenon is shown for all three evaluation values, i.e., as we increase the DCV, the overall performance of the lists decreases.

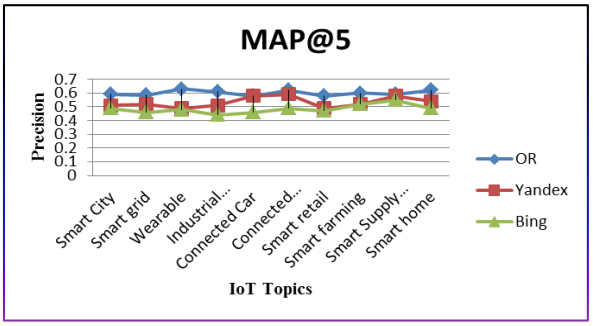
The next evaluation matrix we used is MAP@n for checking the number of retrieved results. This evaluation matrix represents the average precision obtained by different IR systems while searching for all topics against the selected DCVs. Table 11 shows the results for MAP@n achieved using the three systems. The results clearly show that the optimal rank achieves the highest MAP@n. Here, n is equal to the DCV values, at $DCV_5$, the proposed scheme achieved MAP@5 = 0.60, which shows clear improvement over the other system results.

**Table 11.** Mean average precision scores at different DCV values for the all three search lists.

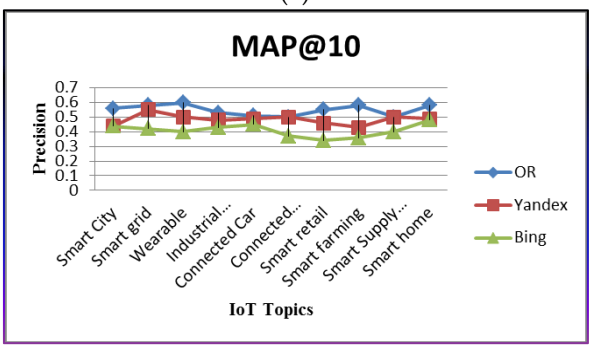
| | MAP@5 | MAP@10 | MAP@20 |
|---|---|---|---|
| OR | 0.60 | 0.54 | 0.44 |
| Yandex | 0.53 | 0.48 | 0.42 |
| Bing | 0.48 | 0.40 | 0.37 |

Figure 5 shows the same MAP@n scores achieved for each of the topics separately. For all topics, the proposed scheme proves its worth. The highest MAP score is 0.62, which is achieved for the topic related to smart homes. This is because of the topic's popularity among Internet users and most number of indexed pages. While the lowest MAP scores are for smart retail related queries: 0.57. Figure 5c shows the MAP scores for $DCV_{20}$; the proposed scheme underperforms the Yandex for only two topics: smart retail and smart farming. The proposed scheme performance degrades as it is unable to find right terms for query rewriting, because of the poor standard page corpus for these topics.

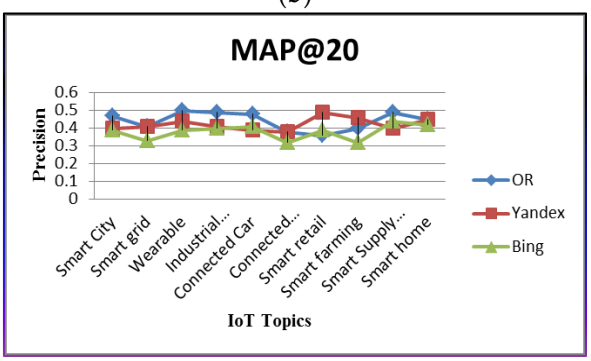
The obtained statistics related to search engine performance evaluation can be helpful for people related to a search engine, and it allows them to still be informed and helps for rational decisions.




(**a**)




(**b**)




(**c**)

**Figure 5.** Mean average precision scores wise performance of all retrieval lists for DCV values {5, 10 and 20}. (**a**) MAP@5; (**b**) MAP@10; (**c**) MAP@20.

## 6. Conclusions and Future Work

In this work, we proposed a scheme for obtaining the sustainable optimal rank to fetch search results with higher precision for IoT-related topics. The proposed scheme rewrites the seed query with the help of attribute terms extracted from the page corpus. We also used newness- and geo-sensitivity-based boosting and dampening of web pages for re-ranking. We verified the proposed scheme for sustainable optimal ranking with the help of an evolution metric based on DCG, nDCG, and MAP@n. In this work, we compared the proposed scheme results with two different search engines. The experimental results showed that the scheme achieves scores of MAP@5 = 0.60, $DCG_5$ = 4.43, and nDCG = 0.95 for general queries, $DCG_5$ = 4.14 and nDCG = 0.93 for time-stamp queries, and $DCG_5$ = 4.15 and nDCG = 0.96 for geographical location-based queries. The outcomes validate the usefulness of the suggested system in helping the user to access IoT-relevant results. In the future, we plan to expand this work for automatic performance evaluation of SSEs with the help of text similarity, semantic features, and clustering of results.

**Acknowledgments:** This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. NRF-2016R1A2B1014843) for the fourth author (Byung-Won On), and supported by the Ministry of Trade, Industry & Energy (MOTIE, Korea) under Industrial Technology Innovation Program. No.10063130, by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2016R1A2B4007498), and the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2017-2016-0-00313) supervised by the IITP(Institute for Information & communications Technology Promotion) for the fifth author (Gyu Sang Choi).

**Author Contributions:** All authors contributed extensively to the work presented in this paper. G.S.C, B.-W.O. and I.L designed the experiment while T.A conceived and designed the experimental setup and collected the data. T.A also analyzed the data and derived results with the help of Z.J. G.C.S and B.-W.O administered the experiment and helps the T.A in writing the manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. The Four Vs of Big Data. Available online: http://www.ibmbigdatahub.com/infographic/four-vs-big-data (accessed on 10 March 2017).
2. KDnuggets: Analytics, Big Data, Data Mining and Data Science Feed. Available online: http://www.kdnuggets.com/2016/09/big-data-iot-match-made-heaven.html (accessed on 10 March 2017).
3. Number of Internet Users (2016)—Internet Live Stats. Available online: http://www.internetlivestats.com/internet-users/ (accessed on 10 March 2017).
4. Voorhees, E.M.; Harman, D. Overview of TREC 2001. Available online: http://trec.nist.gov/pubs/trec10/papers/overview_10.pdf (accessed on 6 July 2017).
5. Justin, Z. How reliable are the results of large-scale information retrieval experiments? In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, 24–28 August 1998; ACM: New York, NY, USA, 1998.
6. Harter, S.P. Variations in relevance assessments and the measurement of retrieval effectiveness. *JASIS* **1996**, *47*, 37–49. [CrossRef]
7. Voorhees, E.M. Variations in relevance judgments and the measurement of retrieval effectiveness. *Inf. Process. Manag.* **2000**, *36*, 697–716. [CrossRef]
8. Dwork, C.; Kumar, R.; Naor, M.; Sivakumar, D. Rank aggregation methods for the web. In Proceedings of the 10th International Conference on World Wide Web, Hong Kong, China, 1–5 May 2001; ACM: New York, NY, USA, 2001.
9. Meng, W.; Yu, C.; Liu, K.L. Building efficient and effective metasearch engines. *ACM Comput. Surv.* **2002**, *34*, 48–89. [CrossRef]
10. Fazli, C.; Nuray, R.; Sevdik, A.B. Automatic performance evaluation of Web search engines. *Inf. Process. Manag.* **2004**, *40*, 495–514.
11. Nuray, R.; Fazli, C. Automatic ranking of information retrieval systems using data fusion. *Inf. Process. Manag.* **2006**, *42*, 595–614. [CrossRef]

12. IoT Analytics. The 10 Most Popular Internet of Things Applications Right Now. Available online: https://iot-analytics.com/10-internet-of-things-applications/ (accessed on 3 October 2017).
13. Tefko, S. Relevance: A review of the literature and a framework for thinking on the notion in information science. Part III: Behavior and effects of relevance. *J. Am. Soc. Inf. Sci. Technol.* **2007**, *58*, 2126–2144.
14. Du, J.T.; Spink, A. Toward a web search model: Integrating multitasking, cognitive coordination, and cognitive shifts. *J. Am. Soc. Inf. Sci. Technol.* **2011**, *62*, 1446–1472. [CrossRef]
15. Lv, Y.; Zhai, C.X. Adaptive relevance feedback in information retrieval. In Proceedings of the 18th ACM Conference on Information and Knowledge Management, Hong Kong, China, 2–6 November 2009; ACM: New York, NY, USA, 2009.
16. Buckley, C.; Salton, G.; Allan, J.; Singhal, A. Automatic Query Expansion Using SMART: TREC 3. Available online: https://pdfs.semanticscholar.org/7859/071375af210096a2003f355df17817297173.pdf (accessed on 6 July 2017).
17. Soboroff, I.; Nicholas, C.; Cahan, P. Ranking retrieval systems without relevance judgments. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, LA, USA, 9–13 September 2001; ACM: New York, NY, USA, 2001.
18. Chowdhury, A.; Soboroff, I. Automatic evaluation of worldwide web search services. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland, 11–15 August 2002; ACM: New York, NY, USA, 2002.
19. Mahmoudi, M.; Badie, R.; Zahedi, M.S. Evaluating the retrieval effectiveness of search engines using Persian navigational queries. In Proceedings of the 2014 7th International Symposium on Telecommunications (IST), Tehran, Iran, 9–11 September 2014; IEEE: Piscataway, NJ, USA, 2014.
20. Wu, S.; Crestani, F. Methods for ranking information retrieval systems without relevance judgments. In Proceedings of the 2003 ACM Symposium on Applied Computing, Melbourne, FL, USA, 9–12 March 2003; ACM: New York, NY, USA, 2003.
21. Joachims, T. Evaluating Retrieval Performance Using Clickthrough Data. Available online: http://www.cs.cornell.edu/People/tj/publications/joachims_02b.pdf (accessed on 7 July 2017).
22. Thorsten, J.; Granka, L.; Pan, B.; Hembrooke, H.; Gay, G. Accurately Interpreting Clickthrough Data as Implicit. In Proceedings of the 28th Annual International ACM SIGIR Conference, Salvador, Brazil, 15–19 August 2005; ACM: New York, NY, USA, 2005.
23. Sharma, H.; Jansen, B.J. Automated evaluation of search engine performance via implicit user feedback. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, 15–19 August 2005; ACM: New York, NY, USA, 2005.
24. Liu, Y.; Zhang, M.; Ru, L.; Ma, S. Automatic query type identification based on click-through information. In Proceedings of the Third Asia Information Retrieval Symposium, AIRS 2006, Singapore, 16–18 October 2006; Springer: Berlin, Heidelberg, Germany, 2006.
25. Liu, Y.; Fu, Y.; Zhang, M.; Ma, S.; Ru, L. Automatic search engine performance evaluation with click-through data analysis. In Proceedings of the 16th International Conference on World Wide Web, Banff, AL, Canada, 8–12 May 2007; ACM: New York, NY, USA, 2007.
26. Ali, R.; Beg, M.M.S. Automatic performance evaluation of web search systems using rough set based rank aggregation. In Proceedings of the First International Conference on Intelligent Human Computer Interaction, Allahabad, India, 20–23 January 2009; Springer: Allahabad, India, 2009.
27. Badie, R.; Azimzadeh, M.; Zahedi, M.S. Automatic evaluation of search engines: Using web pages' content, web graph link structure and websites' popularity. In Proceedings of the 2014 7th International Symposium on Telecommunications (IST), Tehran, Iran, 9–11 September 2014; IEEE: Piscataway, NJ, USA, 2014.
28. Mehmood, A.; Choi, G.S.; von Feigenblatt, O.F.; Park, H.W. Proving ground for social network analysis in the emerging research area "Internet of Things" (IoT). *Scientometrics* **2016**, *109*, 185–201. [CrossRef]
29. Mehmood, A.; On, B.-W.; Lee, I.; Choi, G.S. Prognosis Essay Scoring and Article Relevancy Using Multi-Text Features and Machine Learning. *Symmetry* **2017**, *9*, 11. [CrossRef]
30. Karami, A.B.; Fleury, A.; Boonaert, J.; Lecoeuche, S. User in the Loop: Adaptive Smart Homes Exploiting User Feedback—State of the Art and Future Directions. *Information* **2016**, *7*, 35. [CrossRef]
31. Leth, S.; Hansen, J.; Nielsen, O.W.; Dinesen, B. Evaluation of Commercial Self-Monitoring Devices for Clinical Purposes: Results from the Future Patient Trial, Phase I. *Sensors* **2017**, *17*, 211. [CrossRef] [PubMed]

32. Singh, J.N.; Dwivedi, S.K. Performance Evaluation of SE Using Enhanced Vector Space Model. *J. Comput. Sci.* **2015**, *11*, 692–698. [CrossRef]
33. Lewandowski, D. Evaluating the retrieval effectiveness of Web SE using a representative query sample. *J. Assoc. Inf. Sci. Technol.* **2015**, *66*, 1763–1775. [CrossRef]
34. Szummer, M.; Jaakkola, T. Partially labeled classification with Markov random walks. *NIPS* **2001**, *14*, 945–952.