INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

Instant fuzzy search using probabilistic-correlation based ranking

J Ujwala Rekha[1]*

1 Associate Professor of CSE, JNTUH College of Engineering Hyderabad, Telangana, India

OPEN ACCESS

Received: 28-02-2020

Accepted: 05-04-2020

Published: 02-05-2020

Editor: Dr. Natarajan Gajendran

Citation: Rekha JU (2020) Instant fuzzy search using probabilistic-correlation based ranking. Indian Journal of Science and Technology 13(11): 1270-1275. https://doi.org/10.17485/IJST/v13i11.2020-32

***Corresponding author**.
J Ujwala Rekha

Associate Professor of CSE, JNTUH College of Engineering Hyderabad, Telangana, India
ujwala_rekha@jntuh.ac.in

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

Abstract

Background: Instant search recommends completions of the query `on the fly', and instantly displays the results with every keystroke. It is desirable that these query results be robust against typographical errors that appear not only in the query but also in the documents. Additionally, instant search requires instant response time and ranking of the results to focus on the most important answers. **Method:** In this study, simple and efficient methods for instant fuzzy single keyword and multi-keyword search that are resilient to typographical errors and that employ no more than inverted and forward indices are studied. While computing search results incrementally using the cached results, the answers are ranked based on their relevance to the query using probabilistic correlation based ranking. **Findings:** Experiments are conducted on data sets DBLP and Medline and the execution time for obtaining answers to instant fuzzy single keyword search is recorded for different prefix lengths. Similarly, the execution time for obtaining answers to instant fuzzy multi-keyword search is recorded for sub-queries of two keywords and three keywords for various prefix lengths on the same data set. Furthermore, in order to measure the usefulness of the proposed correlation based ranking, precision is calculated for the search results. Experimental evaluation demonstrates the efficacy of the instant fuzzy search algorithms and the probabilistic correlation based ranking. **Applications:** The proposed instant fuzzy keyword search for single and multiple keywords not only improves the efficiency but also the quality of the search results.

Keywords: Keyword Search; Multi-keyword Search; Fuzzy Search; Probabilistic Correlation

1 Introduction

In instant search, for each keyword or for the keyword that is presently being typed, the search engine must return results not only for the similar words but also for the words whose prefix is similar to the keyword.

One of the earliest examples of instant search is the Unix Shell, which presents the list of all file names starting with the letter that has been typed on the command line. While the purpose of the instant search is finding information, it is also employed in text editors for predicting user input [1–3]. In the context of information retrieval systems,

Bast et al.[4–6] proposed methods for indexing keywords and then querying for instant search. However, these techniques require large processing times and a lot of space. There are also studies on fuzzy instant search that are typo-tolerant and word-order independent[7–10]. More recently, it is integrated into a number of search engines, for example, Google Instant for searching the web, Facebook that searches for the relevant people, Internet Movie Database that recommends movies, YouTube Instant that suggests the videos, interactive query suggestions within an e-mail system[11] and so on. However, these search engines make use of massive logs of previous queries, and fail to generate appropriate answers if the query posed is not in the log.

An additional challenge in search engines is dealing with massive amounts of documents in the data repository. One of the solutions in dealing with massive amounts of data is ranking query results; because ranking directs attention towards only the most relevant answers. Ranking algorithms are extensively studied in databases and information retrieval. The Ranking models can be classified as vector space models[12], probabilistic models[13], statistical language models[14] and hybrid models[15].

In this article, simple and efficient methods for fuzzy instant search are studied that answer single keyword and multi-keyword queries. The methods make use of no more than inverted and forward indices. Moreover, the methods compute answers incrementally using the cached results, and rank the answers based on their relevance to the query using probabilistic correlation based ranking.

The rest of the article is organized as follows: Section 2 presents preliminaries of the study. While Section 3 presents instant fuzzy keyword search, Section 4 describes instant fuzzy multi-keyword search, and Section 5 defines probabilistic-correlation based relevance ranking. Finally, Section 6 demonstrates the experiments conducted and Section 7 concludes the paper.

2 Preliminaries

Data Set: Let $R = \{r_1, r_2, ...\}$ be a set of records and $D = \{w_1, w_2, ...\}$ be a dictionary that contains the set of all distinct keywords of R. While in Table 1 an example of a set of records is presented, in Table 2 an inverted index of the set of records is given.

Similarity Measurement: In this study, edit-distance is employed to measure the dissimilarity between two keywords, which is defined as the minimum number of edit operations such as insertion, deletion, and substitution of single characters required to convert one keyword to the other. Let us denote the edit-distance between two keywords S_1 and S_2 as $edist\ (s_1, s_2)$ then the two keywords are similar if $edist\ (s_1, s_2) \leq \tau$ where τ is the edit distance threshold.

Table 1. Example of data set

Record ID	Title
r_1	Information retrieval: data structures and algorithms
r_2	Information storage and retrieval
r_3	Using linear algebra for intelligent information retrieval
r_4	Approaches to intelligent information retrieval
r_5	Freenet: a distributed anonymous information storage and retrieval system
r_6	Survivable information storage systems

3 Instant fuzzy keyword search

Instant fuzzy keyword search consists of searching keywords that have a prefix close to the query string. More formally, given a query string $q = c_1c_2\ldots$, instant fuzzy keyword search generates a ranked list of pairs (w_i, r_j) such that prefix v of w_i is similar to q and $w_i \in D$ is a keyword of $r_j \in R$. In this section, a simple and efficient algorithm for instant fuzzy keyword search is presented where the information system accepts a sequence of queries from the user who is keying in character by character, and computes answers to the query from the answers generated in the previous query in the sequence. The answers generated are ranked based on their relevancy which will be described in Section 5.

3.1. Algorithm Description

Let $q = c_1c_2\ldots$ be the query being typed character by character, and τ be an edit-distance threshold. For the sub queries $q_1 = c_1, q_2 = c_1c_2, \ldots, q_\tau = c_1c_2\ldots c_\tau$ the answer is set of all pairs of (w_i, r_j), such that $w_i \in D$ is a keyword of record $r_j \in R$.

For the sub-query $q_{\tau+1} = c_1c_2\ldots c_{\tau+1}$ the entire inverted index is scanned in order to generate the answer. Let $\phi_{\tau+1}$ be the answer for query $q_{\tau+1}$, and $v_{i,\tau+1}$ be the prefix of w_i whose length is equal to $\tau + 1$, then for each $w_i \in D$ the prefix $V_{i,\tau+1}$ is

Table 2. Inverted Index

1.	algebra	r_3
2.	algorithms	r_1
3.	approaches	r_4
4.	anonymous	r_5
5.	data	r_1
6.	distributed	r_5
7.	Freenet	r_5
8.	information	r_1 r_2 r_3 r_4 r_5 r_6
9.	intelligent	r_3 r_4
10.	linear	r_3
11.	retrieval	r_1 r_2 r_3 r_4 r_5
12.	storage	r_2 r_5 r_6
13.	structures	r_1
14.	survivable	r_6
15.	system	r_5 r_6
16.	using	r_3

compared against the query $q_{\tau+1}$, and the pair (w_i, r_j) is included in the answer set of query $q_{\tau+1}$ if and only if $v_{i,\tau+1}$ is similar to $q_{\tau+1}$. In other words, the answer to query $q_{\tau+1}$ is a set of pairs defined as follows:

$$\phi_{t+1} = \left\{(w_i, r_j) \,|\, edist\,(v_{i,\tau+1}, q_{\tau+1}) \leq \tau\right\} \tag{1}$$

For the subsequent queries $q_x = c_1 c_2 \ldots c_x$ such that $x > \tau + 1$, the answer ϕ_x is computed from ϕ_{x-1} as follows. Instead of scanning the entire inverted index, only the keywords of the pairs included in ϕ_{x-1} are considered. At first, ϕ_x is initialized to be empty, then for each $(w_i, r_j) \in \phi_{x-1}$, the edit-distance of prefix $v_{i,x}$ of w_i whose length is equal to x is computed against the query q_x, and the pair (w_i, r_j) is included in ϕ_x if and only if edist $(v_{i,x}, q_x) \leq \tau$. Formally, the answer to query q_x is a set of pairs defined as follows:

$$\phi_x = \left\{(w_i, r_j) \,|\, (w_i, r_j) \in \phi_{x-1} \wedge \text{ edist } (v_{i,x}, q_x) \leq \tau\right\} \tag{2}$$

The proposed algorithm is typo-tolerant, simple and efficient. The algorithm employs an inverted index that is not scanned in entirety for all the sub-queries. Furthermore, while calculating the edit-distance between the prefix of a keyword and the given query, only the prefix whose length is equal to the length of the query is considered.

4 Instant fuzzy multi-keyword search

A multi-keyword query q_l consists of a sequence of keywords $(w_1, w_2, \ldots, w_l)$. In an instant search, a query is generated for each character typed in by the user, and instant fuzzy multi-keyword search constitutes of searching records $r_j \in R$ such that the following two conditions hold:

- The record r_j has a keyword similar to w_i for $1 \leq i \leq l-1$.
- The record r_j has a keyword with prefix similar to w_i.

4.1. Algorithm description

Initially, as the user types the query character by character, the answer for the first keyword w_1 is computed using an inverted index as described in Section 3.1. Upon completion of the first keyword the records that match the first keyword are cached. Subsequently, when the user enters a sequence of characters generating multi-keyword query, a sequence of sub-queries are generated for each character being typed and the results of the previous query are cached. Let w_l be the keyword being typed where $l > 1$, then the prefixes similar to the keyword w_l are searched in the records of the cached results of the previous query rather than the complete set of keywords.

Let R_{i-1} be the set of records cached for the sub-query q_{i-1}, then the set of records that contain the keywords similar to $w_1, w_2, \ldots, w_{l-1}$ and the keyword with prefix similar to w_l is computed for q_i as follows. For each $r_i \in R_{i-1}$, forward index is

used to determine if there exists a keyword with prefix similar to w_l in r_i. If a keyword with prefix similar to w_l exists in r_i, then r_i is included in R_i.

The algorithm described employs an inverted and a forward index. Furthermore, the algorithm is both typo-tolerant and word-order independent. The algorithm is efficient since it does not scan all records for every sub-query.

5 Probabilistic-correlation based relevance Ranking

Keywords in a record must be associated with weights based on how well it distinguishes a particular record from other records. In general, a keyword that occurs more often in various records is a bad discriminator and must be assigned smaller weight when compared to a keyword that occurs less often in various records.

Correlation between the keywords is a measure of inter-dependence and can be calculated based on conditional probability as follows:

$$\begin{aligned} \mathrm{cor}(w_i, w_j) &= \frac{P(w_i|w_j)\,P(w_j|w_i)}{P(w_i)\,P(w_j)} \\ &= \frac{P(w_i \wedge w_j)^2}{P(w_i)\,P(w_j)} \end{aligned} \tag{3}$$

It can be noticed that $\mathrm{cor}(w_i, w_j) = \mathrm{cor}(w_j, w_i)$. Moreover, $\mathrm{cor}(w_i, w_j) = 1$ implies that the keywords w_i and w_j are interdependent on each other and always appear together in the records; and conversely, $\mathrm{cor}(w_i, w_j) = 0$ implies that the keywords w_i and w_j are independent of each other and never appear together.

More generally correlation of a keyword w_i with one or more keywords simultaneously can be calculated as follows:

$$\mathrm{cor}(w_i, W) = \frac{P(w_i|W)\,P(W|w_i)}{P(w_i)\,P(W)} \frac{P\left(w_i \bigwedge\limits_{w_j \in W} w_j\right)^2}{P(w_i)\,P\left(\bigwedge\limits_{w_j \in W} w_j\right)} \tag{4}$$

Let $R_l \in 2^R$ be a set of cached results and $W_l \in 2^D$ be a set of all distinct keywords in cached results R_l. If keyword $w_i \in \mathrm{W}_l$ then the correlation of w_i given a set of keywords $W \subseteq W_l$ over cached results R_l is denoted as $cor_l(w_i, W)$, and is computed from the cached results instead of the entire data repository R.

5.1. Relevance ranking

Let $r \in R_l$ be a record in the cached results R_l consisting of keywords $r = \{k_1, k_2, \ldots, k_n\}$ where $k_i \in W_l$ for $1 \leq i \leq n$; then the relevance or importance of keyword k_i in record r can be defined as follows:

$$\mathrm{rel}(k_i, r) = \frac{\sum_{W \in 2^r \wedge k_i \notin W} freq(W)^* cor_l(k_i, W)}{2^{|r|-1} - 1} \tag{5}$$

It can be noted that $0 \leq \mathrm{rel}(k, r) \leq 1$; and $\mathrm{rel}(k, r) = 1$ implies that the keyword k has higher relevance in r while $\mathrm{rel}(k, r) = 0$ implies no relevance of keyword k in r.

5.1.1. ***Relevance Ranking of Single Keyword Query Results***

Let ϕ_x be set of answers to the query q_x defined according to (2), then for all pairs $(w_i, r_j) \in \phi_x$ the relevance of keyword w_i in relation r_j denoted as $\mathrm{rel}(w_i, r_j)$ is the rank of the corresponding pair.

5.1.2. ***Relevance Ranking of Multi-Keyword Query Results***

Let $q_l = (w_1, w_2, \ldots, w_l)$ be a multi-keyword query, and $R_l \in 2^R$ the corresponding set of cached results. Then the relevance of answer $r \in R_l$ given multi-keyword query q_l can be calculated as follows:

$$\mathrm{rel}'(r, q_l) = \frac{\sum\limits_{1 \leq i \leq l} rel(w_i, r)}{l} \tag{6}$$

6 Experiments

The performance of the proposed algorithms is evaluated on two real data sets namely, DBLP and Medline. For this study, 1000 DBLP log queries and 1000 Medline log queries are extracted and the experiments are conducted on Intel i3 CPU @ 2.6 GHz and 2GB of memory. The average time for generating answers to an instant fuzzy keyword search described in section 3.1 is recorded for various prefix lengths and presented in Figure 1A. Similarly, in order to evaluate instant fuzzy multi-keyword search, the average running time for generating answers to sub-queries of two keywords and three keywords are recorded for different prefix lengths in Figure 1B and Figure 1C respectively. It can be noted from Figure 1A, Figure 1B, and Figure 1C that the proposed algorithm constantly performed well for different prefix lengths.

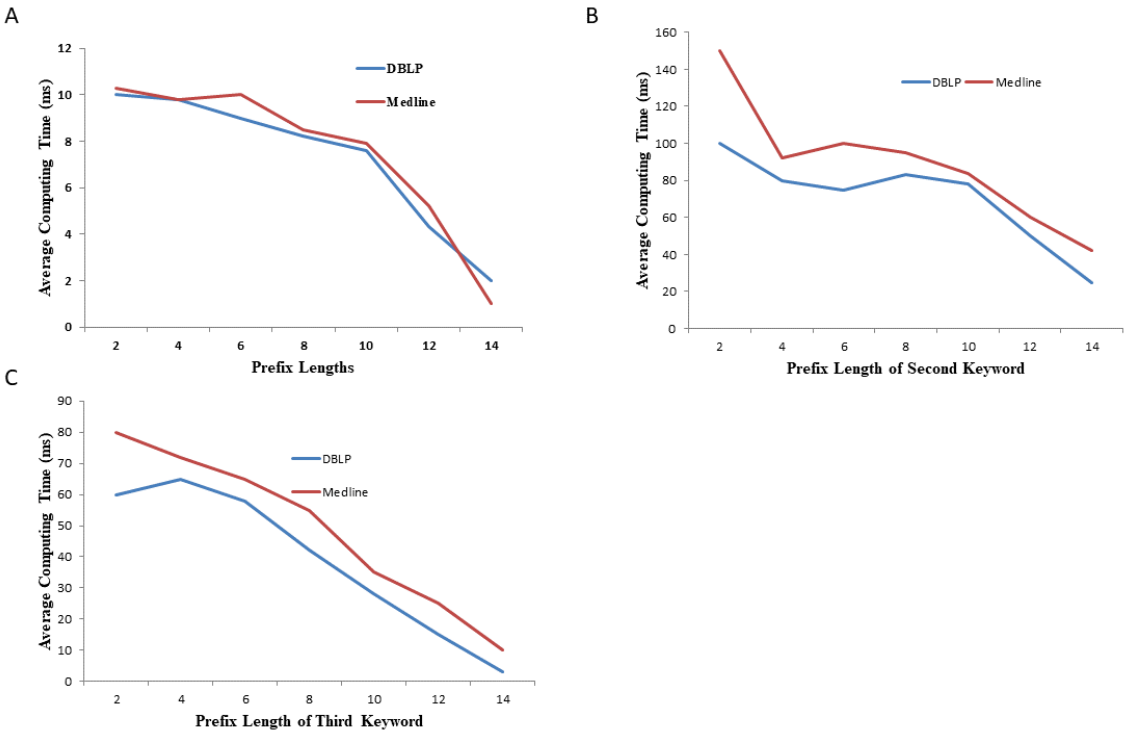

Fig 1. A) Average computing times of instant fuzzy keyword search for different prefix lengths B) Computingtimes of instant fuzzy keyword search C) Computing times of instant fuzzy multi-keyword search

To gauge the quality of the answers generated by the proposed algorithms and the ranking method, precision is measured by determining the percentage of the expected results generated by these approaches. Based on the investigation, the precision of instant fuzzy keyword search is 85% and that of instant multi-keyword search is 90%.

7 Conclusions

In this paper, fuzzy instant search algorithms for answering single keyword and multi-keyword queries are proposed. The algorithms compute the answers incrementally by caching the results of the previous query. Furthermore, a probabilistic-correlation based ranking is proposed to determine the relevance of an answer generated. Experiments are conducted on the real data which validate the efficiency of the proposed algorithms.

References

1) Bickel S, Haider P, Scheffer T. Learning to complete sentences. In: Machine Learning, Vol. 3. Springer. 2005;p. 497–504. doi:doi.org/10.1007/11564096_47.
2) Grabski K, Scheffer T. Sentence completion. In: and others, editor. Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. ACM. 2004;p. 433–439.
3) Nandi A, Jagadish HV. Effective phrase prediction. In: and others, editor. Proceedings of the 33rd international conference on Very large data bases. 2007;p. 219–230.

4) Bast H, Mortensen CW, Weber I. Output-sensitive autocompletion search. *Information Retrieval*. 2008;11(4):269–286. Available from: https://dx.doi.org/10.1007/s10791-008-9048-x; doi:10.1007/s10791-008-9048-x.
5) Bast H, Weber I. Type less, find more: fast autocompletion search with a succinct index. *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 2006;p. 364–371.
6) Bast H, Weber I. The Complete Search engine: Interactive, efficient, and towards IR & DB integration. *Third Biennial Conference on Innovative Data Systems*. 2007;p. 88–95.
7) Cetindil I, Esmaelnezhad J, Kim T, Li C. Efficient instant-fuzzy search with proximity ranking. *IEEE 30th International Conference on*. 2014;p. 328–339.
8) S J. 2011. Available from: https://eric.ed.gov/?id=ED524397.
9) Qin J, Xiao C, Hu S. Efficient query autocompletion with edit distance-based error tolerance. *The VLDB Journal*. 2019.
10) Zhou X, Qin J, Xiao C, Wang W, Lin X, Ishikawa Y. BEVA: An efficient query processing algorithm for error-tolerant autocompletion. *ACM Trans Database Syst*. 2016;41(1):44–44.
11) Lewin-Eytan L, Raviv CD, Libov A, Maarek A, Monaco Y, Media PIV, et al.. 2009. Available from: https://patents.google.com/patent/US20200012686A1/en.
12) Lee DL, Chuang H, Seamons K. Document ranking and the vector-space model. *IEEE Software*. 1997;14(2):67–75. Available from: https://dx.doi.org/10.1109/52.582976; doi:10.1109/52.582976.
13) Robertson S. On event spaces and probabilistic models in information retrieval. *Information Retrieval*. 2005;8(2):319–329.
14) Croft B, Lafferty J. Springer Science & Business Media. 2003. Available from: https://link.springer.com/book/10.1007/978-94-017-0171-6.
15) Devi B, Shankar VG, Srivastava S, Srivastava DK. AnaBus: A Proposed Sampling Retrieval Model for Business and Historical Data Analytics. InData Management, Analytics and Innovation. Singapore. Springer. 2020. Available from: https://doi.org/10.1007/978-981-13-9364-8_14.