Neurocomputing 124 (2014) 43–56

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Optimized weights of document keywords for auto-reply accuracy

Jinn-Tsong Tsai *

Department of Computer Science, National Pingtung University of Education, 4-18 Min-Sheng Road, Pingtung 900, Taiwan, ROC

ARTICLE INFO

Article history:
Received 1 June 2013
Received in revised form
9 July 2013
Accepted 22 July 2013
Communicated by Cheng-Wu Chen
Available online 24 August 2013

Keywords:
Keyword weight
Auto-reply accuracy
e-learning
Differential evolution
Taguchi method

ABSTRACT

A Taguchi-crossover differential evolution (TCDE) algorithm is proposed to optimize weights of document keywords for auto-reply accuracy. The proposed TCDE algorithm combines the use of differential evolution for exploring the optimal feasible region in macro-space with the use of the Taguchi method for exploiting the optimal solution in micro-space. For learning purpose, an answer needs to be exactly given for a specific query. Notably, teachers give a problem answer to elementary students who need to have the clear and accurate solution for learning according to their queries. This study shows the TCDE which integrates a cosine similarity measure and an evaluation function to successfully find the best weights of document keywords for auto-reply accuracy. Performance comparisons confirm that the TCDE algorithm outperforms existing methods presented in the literature in finding the best weights of document keywords and obtaining accurate answers.

© 2013 Elsevier B.V. All rights reserved.

1. Introduction

Online learning and testing systems, in recent years, have been fast developed for students to implement their learning and get knowledge they need, owing to the rapid progress of computer and network technologies [7,25]. Students can proceed with their learning at any time and everywhere, and they need a clear answer when encountering problems during the learning process. If an e-learning system cannot automatically provide students with the exact answer, students do not obtain learning purpose. Particularly, elementary students, in the learning process, need the clear relationship between a query and an answer in a specific course. Therefore, an accurate auto-reply mechanism is needed to develop and help students for their learning. The key point is how to find the best term weighting for information retrieval.

The main function of term weighting was employed to enhance information retrieval effectiveness. Term-frequency weights have been used for many years in automatic indexing environments. A term frequency (TF) was used as part of the term-weighting system measuring the frequency of occurrence of the terms in the document or query ([4,22,23,35]). Term frequency alone cannot ensure acceptable retrieval performance, when the high frequency terms are not concentrated in a few particular documents. The inverse document frequency (IDF) factor, therefore, was proposed to solve the problem. The IDF varies inversely with the number of documents n to which a term is assigned in a collection of N documents. A typical IDF may be computed as $\log(N/n)$ [21]. However, the best terms should have high frequencies but low overall collection frequencies. A reasonable measure of term importance may be obtained by using the product of the TF and the IDF [22]. The “key word spotting rule” approach was proposed to classify e-mails and has been applied to the development of e-mail management system [3]. An intelligent network-based customer service system was proposed. The system assigns a weighted keyword set to each frequently asked questions (FAQ), and compares the keywords requested by a customer with those in each FAQ to find the feasible answer [14]. A customer service system was proposed to automatically handle customer requests by analyzing the contents of the requests and finding the most feasible answer from the FAQ database [33]. There has been a great work on keyword proximity search on databases. These works follow various techniques to overcome the problem, to which the keyword proximity search problems can be reduced [1,10,12]. Additionally, a genetic algorithm was used to adapt the weights of document keywords for query optimization in information retrieval [9,11]. Hwang et al. [11] proposed an enhanced genetic approach to optimize the weights of document keywords for each candidate answer according to the feedbacks provided by the students, hence more accurate answers can be provided. According to aforementioned research works, finding the best weights of document keywords is lastingly discussed. In some specific needs (e.g., one-to-one relationship between queries and documents), the traditional methods to determine the weights of document keywords in information retrieval did not work well. Therefore, a highly efficient evolutionary algorithm is needed to solve information retrieval problems.

* Tel.: +886 8 7226141; fax: +886 8 7215034.
E-mail addresses: jttsai@mail.npue.edu.tw, jttsainpue@gmail.com

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.07.045

The differential evolution (DE) algorithm proposed by Storn and Price [26,27] has attracted growing interest for use in solving global design problems. Its advantages include its simplicity, powerful search capability, compact structure, minimal control parameters, and high convergence characteristics [8,20]. Like other evolutionary algorithms, the DE is highly reliable for population-based and stochastic global optimization in nonlinear and multi-modal environments. The DE has been used in many engineering applications, such as digital filter design [13], controller design [36], power system design [34], and parameter design [2,8,18]. However, its use of stochastic recombination is not a systematic approach to breeding individual trial vectors. Many studies have been conducted to enhance the DE performance by adding local search method. Fan and Lampinen [5] presented a trigonometric mutation operation for local search in order to obtain a better tradeoff between robustness and convergence speed. Noman and Iba [16] proposed a local search technique by adaptively adjusting the length of the search, using a hill-climbing heuristic. This study proposes a robust Taguchi-crossover operation for local search to enhance DE and to find improved offspring [29–32]. The Taguchi-crossover differential evolution (TCDE) algorithm integrates a cosine similarity measure and an evaluation function to successfully find the best weights of document keywords for auto-reply accuracy in an e-learning system.

This paper is organized as follows. Section 2 describes relevant works and methods. The TCDE is presented in Section 3. Section 4 describes the use of the TCDE to find the best weights of document keywords and compare the results with those obtained by the methods presented in the literature. Finally, Section 5 concludes the study.

2. Relevant works and methods

The main relevant works and methods, used in this study, are described below.

2.1. Term-weighting method

Salton and Buckley [22] presented the term-weighting method to combine the TF and the IDF measure. The term weighting w_{qj} is given below and named TWA-1.

$$w_{qj} = \mathrm{TF} \times \mathrm{IDF} = \left(0.5 + 0.5\frac{n_{qj}}{n_{q,\,max}}\right) \times \mathrm{IDF}_j, \quad \mathrm{IDF}_j = \log\left(\frac{N}{n_j}\right), \tag{2.1}$$

where n_{qj} is the number of occurrences of the considered term (or keyword) j in the document q, $n_{q,\,max}$ is the maximum number of occurrences of any considered term in the document q, N is the number of all considered documents, and n_j is the total number of documents containing the term j. The TF was normalized to lie between 0.5 and 1.

According to the concept of Salton and Buckley [22], the TF was transformed to be another type, $\mathrm{TF} = n_{qj}/\sum_{k=1}^{m} n_{kq}$ [9]. The TF × IDF is redefined below and named TWA-2.

$$w_{qj} = \mathrm{TF} \times \mathrm{IDF} = \frac{n_{qj}}{\sum_{k=1}^{m} n_{kq}} \times \mathrm{IDF}_j, \quad \mathrm{IDF}_j = \log\left(\frac{N}{n_j}\right), \tag{2.2}$$

where n_{qj} is the number of occurrences of the considered term j in the document q, m is considered number of terms in the document q, and $\sum_{k=1}^{m} n_{kq}$ (the denominator) is the sum of number of occurrences of all considered terms in the document q.

A query-document similarity measure was obtained by comparing the corresponding the query and document vectors. The well-known cosine vector similarity formula has been used in many retrieval systems ([22,23]). The cosine similarity measure method is shown below.

$$S(Q, D) = \frac{\sum_{k=1}^{m} (w_{qk} \times w_{dk})}{\sqrt{\sum_{k=1}^{m} (w_{qk})^2}\sqrt{\sum_{k=1}^{m} (w_{dk})^2}}, \tag{2.3}$$

where w_{dk} (or w_{qk}) represents the weighting of the considered term in the document D (or query Q), m is considered number of terms in the document q.

2.2. Automatic customer service system

Tseng and Hwang [33] simplified the TF × IDF when the average number of words is apparently small. As TF × IDF aims at taking both the keyword frequency in a specific document and its frequency in all the considered documents into consideration, for a set of documents $D = \{D_1, D_2, \ldots, D_q \ldots, D_r\}$, a character vector (CV) used to represent document D_q is as follows:

$$\mathrm{CV}(D_q) = ((K_1, w_{q1}), (K_2, w_{q2}), \ldots, (K_j, w_{qj}), \ldots, (K_m, w_{qm})) \tag{2.4}$$

where K_j represents jth keyword and m is the number of keywords in the document q. The w_{qj} is given below and named TH-2007.

$$w_{qj} = \frac{n_{qj}}{\sum_{k=1}^{r} n_{kj}}, \tag{2.5}$$

where n_{qj} is the number of occurrences of the keyword j in the document q, r is the number of all considered documents (e.g., FAQ database), and $\sum_{k=1}^{r} n_{kj}$ (the denominator) is the sum of number of occurrences of keyword j in all considered documents. It is worthy to note that the definition of $\sum_{k=1}^{r} n_{kj}$ is different with the definition of $\sum_{k=1}^{m} n_{kq}$ in Eq. (2.2). In addition, the CV of query Q was also defined and given as follows:

$$\mathrm{CV}(Q) = ((K_1, w_1), (K_2, w_2), \ldots, (K_j, w_j), \ldots, (K_m, w_m)) \tag{2.6}$$

and

$$w_j = \frac{n_j}{\sum_{k=1}^{r} n_{kj}} \tag{2.7}$$

where n_j is the number of occurrences of the keyword j in the query.

Tseng and Hwang [33] employed the inner product and Euclidean distance methods to compute the degree of similarity. They are shown below.

$$S(K(Q),\ K(D_q)) = \sum_{k=1}^{m} (w_{qk} \times w_{dk}), \tag{2.8}$$

and

$$S(K(Q),\ K(D_q)) = \sqrt{\sum_{k=1}^{m} (w_{qk} - w_{dk})^2}, \tag{2.9}$$

where the degree of similarity for the inner product method is the larger the better, but for Euclidean distance method is the smaller the better (the best value is 0).

2.3. Auto-reply accuracy optimization

Hwang et al. [11] used the term-weighting approach for document keyword weights that is the same as Eq. (2.5), named HW-2008, to determine document keyword weights as an initial population of chromosomes. They used an enhanced genetic approach to find the weights of document keywords, $w(K_j(D_q))$. However, for auto-reply accuracy, a query-document similarity measure was redefined and it is different with the aforementioned similarity measure method. Assume that someone submits a query Q comprised of m keywords, $K_i(Q)$, $i = 1, \ldots, m$, to an auto-reply system which is connected to a database of r documents, $D_1, D_2, \ldots, D_q \ldots, D_r$. Each documents D_q contains m keywords, $K_j(D_q)$, $j = 1, \ldots, m$. Let $w(K_j(D_q))$ be the weighting of the jth keyword in the