Sustainable Computing: Informatics and Systems 20 (2018) 174–191

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

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

User behavior analysis-based smart energy management for webpage ranking: Learning automata-based solution

Aaisha Makkar*, Neeraj Kumar

Computer Science Engineering Department, Thapar University, Patiala, India

ARTICLE INFO

Article history:
Received 16 August 2017
Received in revised form
20 November 2017
Accepted 13 February 2018
Available online 17 February 2018

Keywords:
Learning automata
Webpage ranking
Social analytics
PageRank
User behavior analysis

ABSTRACT

Search engines are widely used for surfing the Internet. Different search engines vary with respect to their accuracy and time to fetch the information from the distributed/centralized database repository across the globe. However, it has been found in the literature that webpage ranking helps in saving the user's surfing time which in turn saves considerable energy consumption during computation and transmission across the network. Most of the earlier solutions reported in the literature uses the hyperlink structure of graph which consume a lot of energy during the computation. It may lead to the link leakage problem with the occurrence of spam pages more often. Nowadays, hyperlink structure alone is inadequate for predicting webpage importance keeping in view of the energy consumption of various smart devices. User browsing behavior depicts its real importance. It is essential to demote the spam pages to increase the search engine accuracy and speed. Hence, user behavior analysis along with demotion of spam pages can improve Search Engine Result Pages (SERP) which in turn saves the energy consumption. In the proposed approach, web page importance score is computed by analyzing user surfing behavior attributes, dwell time, and click count. After computing the webpage importance score, the ranks are revised by implementing it in Learning Automata (LA) environment. Learning automaton is the stochastic system which learns from the environment and responds either with a reward or a penalty. With every response from the environment, the probability of visiting the webpage is updated. Probability computation is done using Normal and Gamma distribution functions. In the proposal, we have considered only the dangling pages for experiments. Inactive webpages are punished and degraded from the system. We have validated proposed approach with Microsoft Learning to Rank dataset. It has been found in the experiments performed that 3403 dangling pages out of 12211 dangling pages have been degraded using the proposed scheme. The objective of the proposed scheme is achieved by saving web energy and computational cost. It takes 100 iterations to convergence which executed in 21.88 ms. However, the user behavior analysis helped in improving PageRank score of the webpages.

1. Introduction

Today, everyone is aware of the term social networks. Even the rural and most remote areas have heard about facebook and twitter. Social networking has become a pervasive part of our daily life. Globally connected society is not only required but essential for carrying out day to day activities. It started from newsgroups, tiny chat rooms, then mailing systems and social networking sites.

The first social networking site named 'Six Degrees' became popular in 1997. It lasted till 2011. It was named after the theory of 'six degrees of freedom'. A user can register in it and then can start conversation by connecting with other users. Its popularity was limited because of less connective features. It gave birth to the platform of instant messaging. In the year 2000, around 100 million people got aware about Internet, by using various chat rooms and blogs for chatting. The MySpace was the first social media website, launched in early 2000s. It was the platform for making profile and connecting with friends through it. It worked as a platform for the websites like Facebook. MySpace was mostly used by musicians like Colbie Caillat for promoting their tracks. Then came the next boom for social networks, LinkedIn. LinkedIn was basically meant for professionals and businessman and is popular today also. For connecting people irrespective of their profession, Mark Zuckerberg came with the Facebook in 2005. In 2006, it inspired Jack Dorsey, Biz Stone, Noah Glass and Evan Williams to create Twitter. More than 500 people got connected through twitter. Around 2010,

* Corresponding author.
E-mail address: aaisha.makkar@thapar.edu (A. Makkar).

https://doi.org/10.1016/j.suscom.2018.02.003
2210-5379/

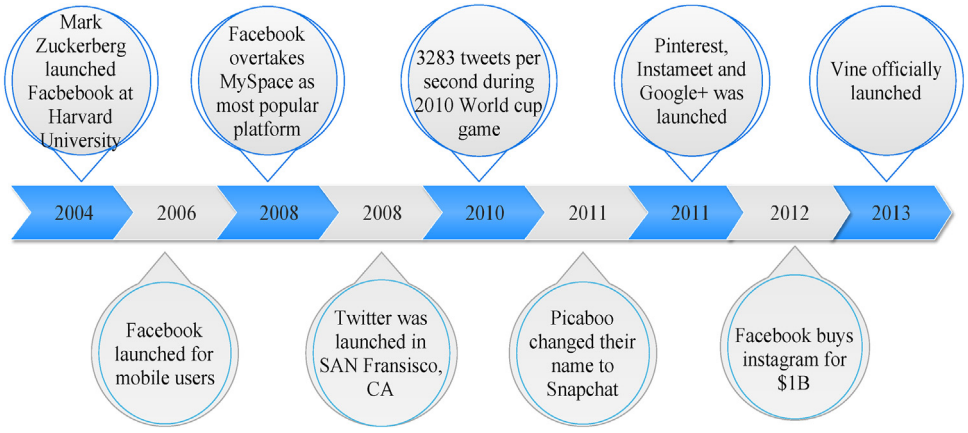

Fig. 1. Social media evolution.

many social networking websites like flickr, photobucket, instagram, tumblr and foursquare were launched. These websites not only lead to connecting people but also for promoting businesses. They resulted in professional benefits and are still very popular. The social media evolution is summarized in Fig. 1. It demonstrates the yearly adaptations in social network zone.

User participation in the Internet is increasing at a rapid pace. It is reported from Internetlivestats [1] (an Internet survey company), that there are currently 3.4 billion Internet users in the world. These users access Internet by different ways. Most frequent is through search engines. According to Netmarketshare's latest statistics [2], the largest market share of search engine usage is from Google (75.2%) followed by Bing, Baidu and Yahoo. According to a survey conducted by an Internet service company Netcraft [3], it has been observed that there are currently 1,003,887,790 web sites in World Wide Web (WWW). All the social networking websites do exist in WWW. The rapid increase in Internet users and social networking sites have made the world more connective. The maximum number of active users is at Facebook as surveyed in January, 2017 which is shown in Fig. 2(a). It can be judged by the survey as shown in Fig. 2(b) that social media users are increasing rapidly.

However, user satisfaction is one of the important aspects behind the success of search engine. Ranking methodology should fairly assign rank to each webpage. The webpage importance is difficult to evaluate and it is directly proportional to user's engagement. If the user spends time on a webpage, it may be useful but it is also contradictory for dangling pages.

There are techniques [4–9] by which the person known as spammer try to boost the rank of its website. These spam websites are really very difficult to detect. Sometimes, searching for authoritative sites consumes a lot of cost and time. This is because the spam websites are boosted by gaining high ranks. Users indulge in Internet surfing activities always expect valuable SERP. Since 1997, Google adopted PageRank methodology for ranking websites and, it is updated periodically. PageRank is highly dependent on number of in-links.

1.1. Motivation

Dangling pages are the webpages which do not have hyperlinks. The ratio of dangling pages is increasing due the documents such as – pdf, technical reports from research communities. Spam pages are mostly dangling pages [10]. Spammers create artificial inlinks to boost the rank of webpages but the outgoing links are not focused. Thus, it results in dangling pages. Evaluating the importance of dangling page can greatly help in refining the SERP's. The system in green computing always targets at energy reduction [11].

Although a lot of work has been done for handling dangling pages [12–14] and improving the ranking algorithm [4,10]. But, none of these have handled dangling pages with respect to user behavior analysis. User surfing activities can only predict the true picture of a webpage. Like in BrowseRank [15], the experiments have successfully ranked all the websites. But, today it is essential to degrade the spam dangling pages as well.

Learning automaton is the system which learn from the environment. Analyzing user behavior also requires learning methodology. LA environment have been used in wide range of applications [16–19]. As this technique proved to be beneficial, so it has been adopted in our system for extracting real life user behavior and learning through it.

1.2. Contribution

This Section presents our contributions for revising the SERP's. Rank of the webpages is revised by analyzing the user's behavior. In this, firstly the proposed approach tries to detect spam pages by considering only dangling pages for experiments. Then the pages with low rank score are punished and PageRank is recomputed.

- The PageRank score is recomputed by the considering click count and dwell time as major factors.
- LA have been used which accept response as input from the environment and perform action as output to the environment.
- The scheme has been validated with Microsoft Learning to Rank dataset.

1.3. Organization

In this paper, Section 2 describes the work already done with respect to dangling pages and social networks. The work flow in designed model is described in Section 3. The proposed scheme is elaborated in Section 4. Section 5 provides results and discussion. Finally, Section 6 concludes the article.

2. Related work

Search engines play the vital role for accessing the authentic information. Its success is directly dependent upon the ranking methodology. Google uses PageRank algorithm as its ranking methodology. Being the largest market share holder, still it lacks in user satisfaction. There are many reasons behind inaccurate SERP's.

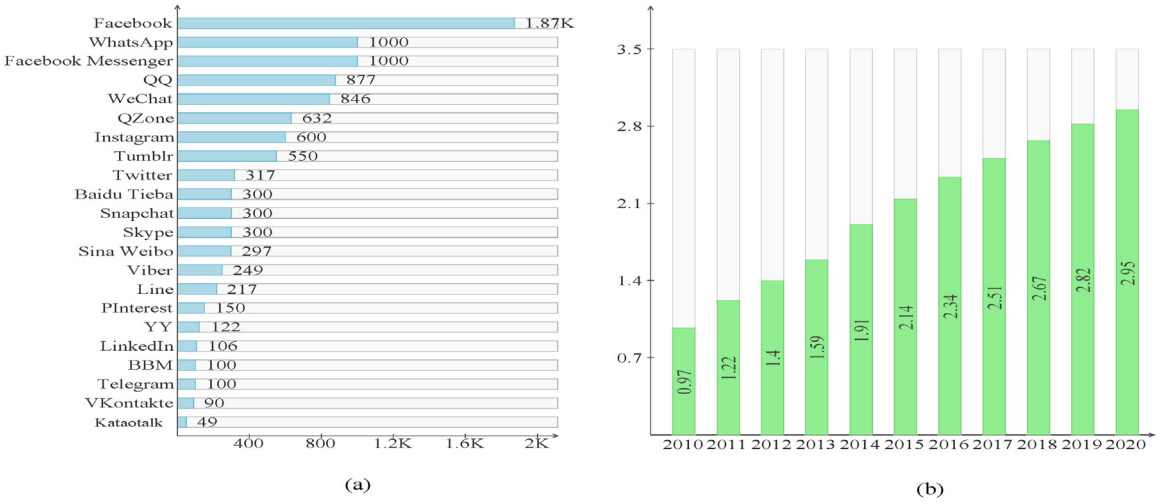

Fig. 2. (a) Social network sites worldwide ranked by number of active users (in millions, as of January, 2017). (b) Number of social media users worldwide from 2010 to 2020 (in billions).

Spam is one of its major causes. A lot of work has been done for improving PageRank methodology by detection of spam sites.

$$PR_p = d \sum_{q \in pa[p]} \frac{x_q}{h_q} + (1 - d) \tag{1}$$

PageRank algorithm is proposed by Larry Page and Sergey Brin at Stanford university [20] in 1996. Power method is used for the computation of PageRank which considers the structure of web. The web surfer selects random links to find the efficient search results. PageRank measures this factor of switching from one page to another page. This probability to jump between webpages is known as damping factor (d) [21]. Eq. (1) computes PageRank score of page p, where d is damping factor, x is an array of all the web pages pointing to page p and h is an array of PageRank score of each webpage pointing to page p.

By decreasing the ratio of contribution of influential nodes in the PageRank score, *Robust PageRank* was introduced to fight against link spam [5]. Link spamming is the result of the major leakage issue called as zero-one gap problem. The huge gap in the ranks are observed between the PageRank of the page with no out-link and the PageRank of the page with one out-link. This gave rise to the new concept by updating the PageRank algorithm. The updated algorithm is known as *Dirichlet algorithm*. It is more resistant against link spamming. This algorithm works by visiting the out-links of the present node first rather than following any graph streamed algorithm such as Breadth First Scheme [10].

TrustRank is the another algorithm introduced with the help of inverse PageRank which follows the out-links rather than in-links. The trust among the seed sets is computed in this algorithm. The seed sets are selected with the application of inverse PageRank. It basically computes the status of the web pages residing in the each seed set by oracle function. Approximate isolation tree is generated with the status computed for webpages, which helps in evaluating the other connected web pages [4]. To reduce the complexity of web pages, seed selection is done on the basis of various topics. The trustrank is revised with this change and renamed as *topical trustrank*. In this, trust score is computed in the same manner but there can be multiple scores for a single page (a page may lie in more than one topic). These scores are thus combined by quality biasness or simple summation to generate the topical trustRank score [22].

Dangling nodes contribute largely in the PageRank computation, thus its appropriate handling is required. Many techniques are introduced with different scenarios for PageRank computation inferring dangling nodes. There are many lumping algorithms in the literature such as – *Lumping Dangling Nodes Algorithm* [12]. These algorithms focuses on the structure of web matrix with respect to dangling nodes as summarized in Fig. 3 and are explained below.

A two-stage algorithm defines that the non-dangling and dangling pages are treated as different blocks. This formulation clearly defines the dangling pages contribution in the PageRank computation [13]. This approach concluded that the PageRank calculation of dangling pages strongly depends on the PageRank values of non-dangling pages but the vice versa is not true [24]. The calculations were reduced by further dividing the non-dangling pages in two blocks, weak non-dangling and strong non-dangling pages [14]. Evaluating the prestige of web pages helps to differ them from untruly and truly sources. The algorithm introduced to compute the prestige score of a page, named as *Proportionate Prestige Score (PPS)* [25]. The experiments proved that the PPS is more appropriate than PageRank score to depict webpage importance.

GPS facility is nowadays popular for locating remote areas. It works with the help of protocols like PPMAC [26]. Gowalla and Foursquare are the well known Location-based Social Networks (LBSN). These help seekers to locate and exchange the geographic locations. The cloud server collects and deals with the location information [27]. Commenting over these networks are also prone to spam. This type of spam is called as *Tip Spam*. The detection system for the same considers various attributes: user post location attribute, content-based features, user behavior features, relationship among these users. Experiments have been conducted on Apontador and Brazilian LBSN system. With these experiments, tip spam has been detected successfully [6].

SOcial network Aided Personalized (SOAP) algorithm is proposed for the detection of *Email spam* in social networks. Initially the emails are manually checked that whether they are spam or legitimate. The Bayesian spam classifier is trained with spam emails for the detection of spam keywords. Email spam has been detected with the probabilities computed for various spam keywords [7].

Sometimes, online advertisements also lead to unreliable content. A spam detection approach has been proposed for the detection of *online spam advertisements*. Content features and domain features, both are extracted for the experiments. Using these features, judges evaluate the advertisements and are then processed by spam classification framework. The overall framework is described in Fig. 4 [8].

Lumping Algorithms Depending upon number of Nodes (Dangling or Non-Dangling)							
The Jacobi Iteration (1.4) without Lumping Strategy	The Lumping algorithm with respect to 2 type nodes (Lee et al. 2007)	The Lumping algorithm with respect to 3 type nodes (Lin et al. 2009)	Recursively reordering algorithm for PageRank (Langville and Meyer 2006a)	The Lumping algorithm with respect to 5 type nodes	Boldi et al.'s strategy for computing the first order derivative (Boldi et al. 2009)	Gleich et al.'s strategy for computing the first order derivative (Gleich et al. 2009)	Using Lumping 5 strategy for computing the first order derivative

Fig. 3. Summary of lumping algorithms [23].

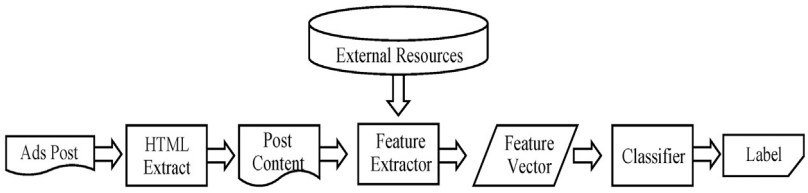

Fig. 4. An overall framework for spam detection in online advertisements [8].

In the world of online marketing, reviews about the products play the important role. Personal benefits encourage defaulters to publish spam reviews. Quality assessment and reference assessment contribute largely in *review spam* detection system. The features considered in this system are: information about the product, meta-data of review and the content of review. The different methods used for its detection are: finding duplicates, content based methods and other methods [9].

As spammers try to harm the social networking sites, so the spam detection techniques have been proposed differently for social networks.

(a) *Social Video Networks* are prone to spam nowadays. The classifier that has been designed to detect the spam, also classifies the different types of users namely, promoter, spammer, legitimate. The dataset required for experiments has been collected by the crawler with three different sets. First, collection of all the videos along with their responses and responded videos. Second, the user's behavior attributes. Third, the relationship between the videos being downloaded and uploaded. All the experiments are done with SVM followed by 5-fold Cross-Validation approach [28].
(b) Another framework is designed for *Spam Detection, Demotion and Prevention* of social websites. In this, strategies adopted for spam detection are: Identification-Based (Detection), Rank-Based (Demotion) and Interface or Limit-Based (Prevention) as shown in Fig. 5. Identification based spam is detected by analyzing the text, source and link. Rank-Based demotion of spam is done by TrustRank algorithm and considered geographical location of web pages as well. Interface spam prevention is done by introducing CAPTCHA's [29].
(c) *Online Social and Business Websites* also uses various spam detection techniques. Categorization of these techniques has been done namely: Adaptive Fusion for Spam Detection (AFSD), SMS filtering (SMSF), MailRank. These all are traditional techniques being followed, also works as the basic method for the development of another models like Spam Behavior Regression Model (SBRM). The result of all the spamming techniques such as fake reviews detection techniques, Social spam detection techniques, link farm detection are presented in Fig. 6 [30].

The above discussed PageRank updation techniques are summarized in Table 1. The above discussed web spam detection techniques are summarized in Table 2.

2.1. User behavior analysis

As today the web documents are increasing, so the people are getting highly involved in retrieving information from it. WWW is the collection of large number of documents from different domains. The world is also interested in variety of information and the people also belong to various communities, domains and regions. Analysis of user web engagement is one of the crucial criteria for search engines and web developers. Here, we examine the following issues where user behavior has been analyzed with experiments.

Depth-level dwell time prediction. Online advertisement benefits can be achieved once the most frequently web pages are known where it can be displayed. It has been achieved by dwell time prediction. Depth-level dwell time prediction seems to be most prominent than page-level dwell time. Pixels of the web page are recorded for capturing which view area the user focuses on. Real time dataset has been evaluated in factorization machines model [34].

Web search results prediction. The web search results are predicted using post-search user behavior analysis. As there can be unreliable internet users as well, so probability estimate functions have been used. The user when searches for a particular query, considers various features for visiting a webpage [35]. All these features are considered in the model RankNet along with the different strategies used which are summarized in Table 5.

Re-ranking web search results. Re-ranking of web search results has been done using dwell time. In this, the document or the webpage are considered at the same level. Top 300 search results by Google search engine are used for experiments comprising of 300 webpages. The results have been compared with Google, yahoo, Bing and AT08 [36].

Web spambot detection. Web robots performs human-user tasks such as registering user accounts, performing vulnerability assessment of tasks, navigation through websites, line checking, searching/submitting tasks, page indexing. Spam robot detection has been done by considering two features action time (time spend on doing a particular action) and action frequency (frequency of

Table 1
PageRank algorithm updations.

Author	Method	Dataset	Evaluation parameters	*D*	System requirements	Execution time	Advantages	Future_scope
Wang et al., 2008 [10]	DirichletRank	1.GOV dataset of the TREC crawled in 2002, 1,247,753 webpages each with average 8.94 out-links 2.UK Dataset crawled in 2006, 77,741,046 webpages each with average 38.14 out-links	ranktext and ranklink	0.85	–	Depends upon the number of iterations	Solves the zero-one gap problem of PageRank, more stable and more resistant against link spam	
Lee et al., 2003 [13]	Markov chain	451,237 webpages by Stanford WebBase Project in 2001	Damping factor(d)	0.85, 0.95	2.4 GHz dual-Xeon workstation with 4GB and a 70GB * 4 RAID-0 hard disk system	Reduced 20% execution time as compare to original PageRank	Dangling nodes are not included at the last stage of computation	
Lin et al., 2009 [14]	Power Method	6012 Web Pages and 23875 hyperlinks	–	0.85	–	–	Reduces computational operations of PageRank vector	
Prabha and Vasantha, 2014 [25]	Proportionate Prestige Score (PPS)	Social Circle: Facebook, Wikipedia vote network, Enron email network	Dangling Pages	0.5, 0.85	Intel 2nd Generation Core i3 Processor, 8GB RAM	0.203 ms	Improved quality of Search Engine Results	Can be used with other algorithms and Fuzzy Based Information System
Gyongyi et al., 2004 [4]	TrustRank algorithm	AltaVista crawled and indexed pages in August 2003	Convergence rate, Oracle Function	0.85	–	Polynomial Execution time	Can be used with PageRank to detect the spam sites	Selection of seed set and oracle function
Wu et al., 2006 [22]	Topical trustrank	Standford's WebBase Project and country specific web crawl courtesy of search.ch	number of spam sites	0.85	–	–	Topic selection resulted in better detection of spam sites.	Topic selection
Eiron et al., 2004 [31]	Naive Method using simple power iteration	37 Billion Links, 4.75 Billion URLs	–	–	–	–	Introduced algorithm hostrank and DirRank methods which is much faster and efficient than PageRank and handle Dangling Pages while crawling	effective crawling methods for spam detection.

Table 2
Web spam detection.

Author	Algorithm	Description	Dataset	Results	Methods used for experiments	Advantages	Future_scope
Heymann et al., 2007 [29]	Spam detection, demotion and prevention for social websites	Detection: Link analysis, Demotion: TrustRank algorithm, Prevention: CAPTCHA	Cloud Social websites	–	–	Testing done by deploying the strategies with spam models and metrics.	Can be collaborated with new ranking method for better performance
Liu et al., 2008 [32]	User Behavior-oriented Web Spam Detection Framework	Algorithm based on Bayesian Learning Method using extracted user behavior features: Search Engine Oriented Visiting Rate, Source Page Rate and Short-time Navigation Rate.	1564 web sites	345 “Spam Sites”, 1060 “Non Spam” sites, 159 “cannot tell” sites	Bayesian Learning Method	Able to detect the type of spam with Web user Behavior	More content and link features can be deployed in the same framework
Benevenuto et al., 2009 [28]	Detection of video spammers and promoters	User Test collection by crawling YouTube, Analyzing User Behavior and then Detecting Spammers and Promoters by building SVM Classifier	Youtube: 264,460 users, 381,616 responded videos, 701,950 video responses	96% promoters, 57% spammers, 95% Legitimate	SVM Classifier	able to provide good results even with small dataset collected.	Different classification Methods can be deployed
Tran et al., 2011 [8]	Spam detection framework for online Spam Advertisements	Content and Domain specific Features are extracted for classification of spam	Craigslist Advertisements	*F*-measure: 0.786	Decision Tree	Efficiently detected spam in Online Advertisements	Collect more data for dataset and Active Learning Techniques can be used instead of Classification Techniques
Liu et al., 2012 [33]	User Behavior-oriented Web Spam Detection Framework	Algorithm based on Bayesian Learning Method using extracted three new user behavior features: Query Diversity (QD), Spam Query Number (SQN) and User-oriented TrustRank	1997 web sites	345 Spam Sites	Bayesian Learning Method	Able to detect the type of spam with Web user Behavior	More content and hyperlink features can be deployed in the same framework

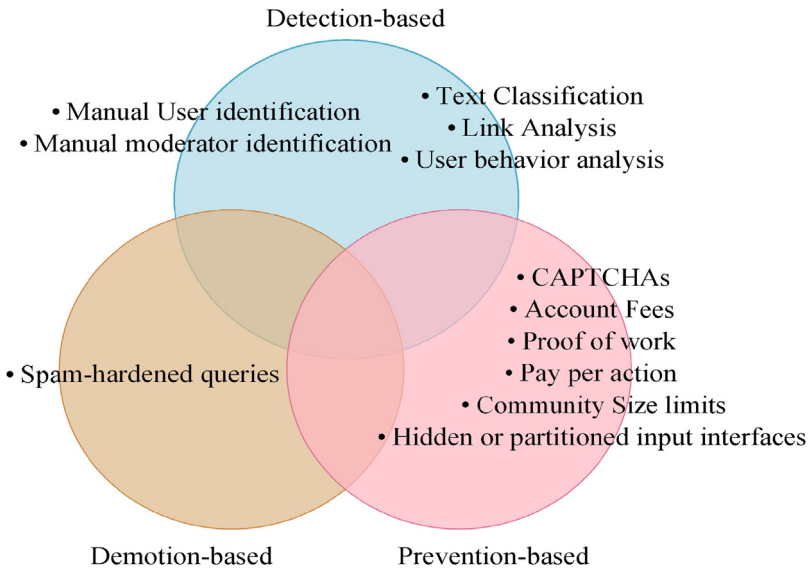

Fig. 5. Three anti-spam strategies: detection, demotion and prevention [29].

Type of Spam	Algorithm/Technique	Dataset	Results
Traditional Spam	AFSD	NetEase	AUC- 0.991
	MailRank	NetEase	Accuracy- 92%
	SMSF	5 million senders	AUC- 0.883
Fake Reviews	LBP	Amazon(Text Features)	Accuracy- 0.78
	SBM	Amazon(Attribute Features)	Accuracy- 0.63
	GSRank	Amazon(Group Features)	Accuracy- 0.85
Social Spam	Decorate	MySpace(1.5million profiles)	Accuracy- 0.992
		Twitter (210,000 profiles)	Accuracy- 0.889
	SybilRank	Tuenti (200000 accounts)	Accuracy- 0.856
	URLSpam	MySpace	Accuracy- 0.765
	SSDM	Twitter	Accuracy- 0.987
Link Farming	CatchSync	Twitter (Zombie followers)	Accuracy- 0.751
		Tencent Weibo	Accuracy- 0.694

Fig. 6. Web spam detection techniques for social networks.

doing one certain action). The scheme has been validated in SVM [37].

Web spam detection. Web spam is detected by analyzing user behavior. It not only detects the spam pages but also categorizes them into different types of spam. The algorithm uses Bayesian learning method along with the features extracted from the behavior of user, thus leading to better understanding for detection of spam [32]. Same work has been extended by extracting three new features namely Query Diversity (QD), Spam Query Number (SQN) and User-oriented TrustRank [33]. Click spam in the Ad networks have also been addressed [38].

Identifying relevant websites. As the search consists of different trails, so finding the relevant authoritative websites is a challenging job. The framework has been developed which analyses the user's searching behavior and laid the solution for appropriate searches. Heuristic retrieval model and Probabilistic retrieval model are used for searching the trails. This approach concluded that the trails of domain ranking is better than interactive ranking [39].

Recommendation by analyzing dwell time. Dwell time has been used for personalized recommendation system. Before this system development, a new method for retrieving dwell time at client side and server side is introduced [40]. Dwell time in this system assumes that the user is interested in a particular webpage, only when returns after visiting other webpages. This absence at webpage is measured for retrieving the content interesting facts [41].

Evaluating webpage importance. The importance of a web page may depend upon the web user's real behavior as well. Browsing behaviors has been modeled by the stochastic approach with the combination of different parameters from real data. The user behavior attributes are computed to form a rank which is known as BrowseRank. Web service application collected the user's behavior. As the collected data was large in size, so only the useful information was extracted and user browsing graph was build [15].

The user behavior analysis for different domains are summarized in Table 3.

The above discussed techniques used were for refining and improving efficiency of search results. None of the method has been used for handling dangling pages by user behavior analysis. Dangling pages are the pages with no outlinks. Means the existence of such pages is the result of either the artificially created pages to form the link farm or the documents such as pdf. It is very important to handle them for fair page rank assignment to other webpages. We found that user behavior analysis has never been done for handling dangling pages. Why user behavior attributes are used? Because the time user's spends on the webpage and number of times the page is opened clearly depicts its importance. The two attribute used in the proposed approach is Dwell time and Click count.

Table 3
Summarized user behavior analysis.

Author	Algorithm	Description	Dataset	Features considered for experiments	Results
Wang et al., 2016 [34]	Depth-level dwell time prediction	The algorithm evaluates the depth level dwell time prediction for online advertisement benefits	Web publisher (Forbes Media) collects 2 million page views, 150K+ for training and 20K+ for testing	1. User id 2. Page URL 3. State-level user geo location 4. User agents 5. Browsing events i.e. open/left/read the page	presented in Table 4
Agichtein et al., 2006 [35]	Web search result preferences	User behavior analyzed for predicting web search preferences	3500 queries, 120,000+ searches in 3 weeks	Query-text features (TitleOverlap, SummaryOverlap), Click through rate (ClickFrequency (IsClickBelow, IsClickAbove)), Browsing Features (Dwell Time (TimeOnPage, TimeOnDomain))	Precison: 0.7+
Hayati et al., 2010 [37]	Web Spambot detection	Spam robots have been detected by analyzing user behavior	Number of human records: 5555, Number of Spambot records: 11,039, Number of total sessions: 4227, Number of actions: 34	Action Time and Action Frequency	Accuracy: 94.70% (ActionTime: 93.18, ActionFrequency: 96.23%)
Liu et al., 2008 [32]	Web spam detection	Web spam Detection by analyzing user behavior	Web access log from July, 2007 to August 26, 2007 comprising of 2.74 billion user clicks in 800 million web pages, 22.1 million user sessions during 57 days	Search engine oriented visiting rate, source page rate and short-time navigation rate	Total websites: 1564, Spam websites: 345, Non-spam websites: 1060, Cannot tell: 159 websites
Bilenko et al., 2008 [39]	Improving search results	Identifying relevant websites by analyzing user's searching behavior	140 million search trails over the year 2006 with two sets of queries, HumanRanking (33,150), UsageRanking (10,000)	Visiting count and dwell time	0.317 at Normalized Discounted Cumulative Gain (NDCG) @ 10
Liu et al., 2010 [15]	BrowseRank	User behavior attributes collected for ranking of webpages	Three billion records, One Billion Unique URLs, 7500 queries	Dwell time and Click count	Outperforms PageRank
Proposed work	User behavior oriented ranking methodology	Webpage ranking by analyzing user behavior analysis for dangling pages	Microsoft Learning to rank dataset	Dwell time and Click count	Outperforms PageRank

Table 4
Depth dwell time prediction comparison [34].

Approaches	RMSD
GlobalAverage	13.8346
ChannelAverage	13.8219
Regress_bc	14.1009
Regress_view+dp	13.8301
FM(viewport; $k = 20$)	11.0309

Table 5
Strategies used for web search results prediction [35].

Approaches	Description
SA	"Skip Above" ClickThrough strategy
SA+N	"Skip Above" and current search engine strategy
D	Refinement of SA+N for selecting trusting clicks
CDiff	computing probability for CD for generalization
CD + CDiff	Simplified union of CD and CDiff
UserBehavior	Higher rate for higher confident webpage

2.2. Energy management for web page ranking

Web is the collection of different communities (set of web pages) consisting of various topics. A community is connected to other pages in the web with the help of outgoing links. There are pages which do not connect outside the community and not even to other pages. Such pages are known as dangling pages. These pages lead to loss of energy. Energy efficiency is crucial in IoT (group of different communities) environment [42,43]. The community's energy is dependent upon four components [44]:

$$E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp} \tag{2}$$

In the above equation, $|I|$ refers to the total number of pages in the community. E_I^{in} refers to the energy coming from the other communities. E_I^{out} refers to the energy going to the other communities. The presence of out (I) states the energy flowing outside the community. E_I^{dp} states the energy lost due to dangling pages. The equation clearly proves the loss of energy due to dangling pages. Energies can be computed by these equations:

$$E_I^{in} = \frac{d}{1-d} \sum_{i \in in(I)} f_i x_i^* \tag{3}$$

$$E_I^{out} = \frac{d}{1-d} \sum_{i \in out(I)} (1 - f_i) x_i^* \tag{4}$$

$$E_I^{dp} = \frac{d}{1-d} \sum_{i \in dp(I)} x_i^* \tag{5}$$

where f_i is the fraction of hyperlinks of page i to the pages in I, with respect to total outgoing links from page i. Number of pages going from the community and number of dangling pages, both

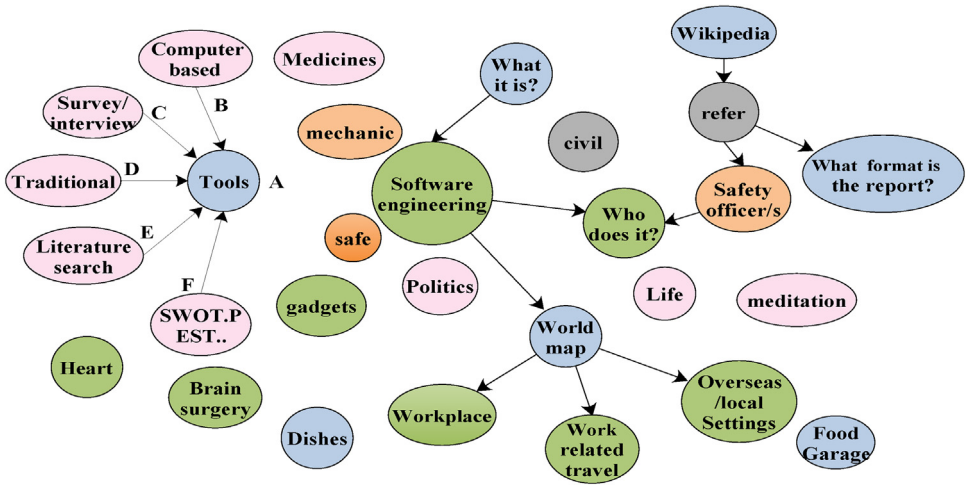

Fig. 7. Web structure.

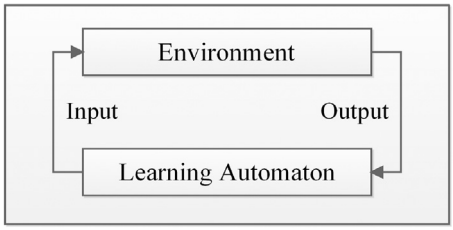

Fig. 8. The working environment of learning automata.

leads to the loss of energy. Dangling pages should be handled appropriately. Many authors [12,13,24,14] have tried to handle them by lumping algorithms which states the treatment of Dangling and non dangling pages separately for PageRank score computation. In this proposal, energy is saved by handling pages by analyzing user's behavior as discussed in Section 4 by the proposed scheme algorithm.

Also, the reduced computational cost by efficient ranking of web pages saves energy. It can be easily understood by Fig. 7, a sample of dataset, there can be dangling pages existing in individual. As depicted, the dangling node A has been promoted by the nodes dedicatedly created for promotion, So, such type of dangling nodes should be degraded.

3. System model

The approach used for our proposed system is discussed in this section. It covers three terminologies as shown in Fig. 8, LA, environment and input/output (action probability vector). In this, LA system learns from the environment and accordingly updates the action probability vector. These concepts are discussed below.

3.1. Learning automaton

The LA is a system having the potential to learn from the environment for taking an action. This learning ability enhances the chances for updating the vector with appropriate action. The action may be a reward or a penalty, it depends upon the environment. This system results with an optimal solution with minimum number of penalties [16]. Structure of learning automaton is drawn by β and α, where β is set of output from environment and *alpha* is set of actions. In β, 0 represents a reward and 1 represents a penalty. With every learning from environment, the action is taken. The action taken is random which is based on probability distribution. The action is then served to the environment as an input. The environment then takes the decision by generating a reinforcement signal. Based upon the reinforcement signal being taken by the environment, the action probability vector gets updated. The main aim of this dedicated system is to decrease the ratio of penalty received from the environment. The system is thus developed with the capability to converge after repeating same action of course. The set of iterations are used to update the action probability vector with defined formulas (as discussed below). In this proposal, we are updating the PageRank score vector with either considering the previous score as a reward or a penalty depending upon its value.

3.2. Environment

The environment is the region from where the LA learn and react. The reaction may be positive or negative it depends upon the output from the environment. Environment helps the system to behave optimally. LA perform each action after receiving the output from the environment. The updated value is served as an input to the environment for the LA next iteration. This process is continued till the optimal solution is achieved. The environment may be stochastic or probabilistic. In our proposed system, it is probabilistic.

3.3. Action probability update

According to the response generated by the environment, action is taken by the LA in the form of reinforcement signal. There are various reward/penalty schemes exist: Linear Reward Penalty, Linear Reward Inaction, and Linear Inaction Penalty [45,46]. In this paper, we have considered Linear Reward Penalty, in which each response whether the action results in reward/penalty, the action probability vector is updated [47,48,19]. The formulas for updating the probability are:

$$p_j(n+1) = (1-a)p_j(n), \qquad j \neq i, \quad Y = 0 \tag{6}$$

$$p_j(n+1) = ap_j(n), \qquad j = i, \quad Y = 0 \tag{7}$$

$$p_j(n+1) = p_j(n), \qquad Y = 0 \tag{8}$$

where a is the learning parameter.

4. Proposed scheme

In this work, the PageRank value is revised by evaluating the webpage importance. Webpage importance is computed by measuring the time user spends on a webpage and total number of clicks of that webpage. The value thus computed is feed into the automata environment. The action probability vector is updated through learning automaton according to the desirable or undesirable response from the environment. The proposed methodology is shown in Fig. 9.

Algorithm 1. Proposed Scheme

Input: PageRank vector of Dangling pages, Dwell time (Computed with Gamma Distribution), Click count (Evaluated using Empirical probability).
Output: Revised PageRank vector.

1: **procedure** FUNCTION(PageRank)
2: $E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp}$ ▷ Energy flow within web
3: **for** $i = 1$ *to* n **do**
4: Compute $PR(n)$ ▷ vector of PageRank values by using Eq. (1)
5: Set $i \longleftarrow i + 1$
6: **end for**
7: **for** $PR(n) = 1$ *to* n **do**
8: **if** Outlink != 0 **then**
9: Discard Node
10: **end if**
11: **end for**
12: **for** $i = 1$ *to* n **do**
13: Compute Revised PageRank $PR(n) = PR(n).D_t + C_c$ ▷ Using Dwell time and Click count
14: Set $i \longleftarrow i + 1$
15: **end for**
16: ▷ Entering LA scheme
17: Input: $PR(n)$
18: Output: 0 (reward), 1 (penalty)
19: $\forall P_i(n) = 0$ ▷ Initializing the population
20: **if** 0 **then**
21: Desirable response
22: $P_i(n) = P_i(n) + P_\sigma$
23: $\forall m m \neq i P_i(n) = (1 - P_i(n)) + P_\sigma$
24: **end if**
25: **if** 1 **then**
26: Undesirable response
27: $P_i(n) = P_i(n) - P_\sigma$
28: $\forall m m \neq i P_i(n) = (1 - P_i(n)) - P_\sigma$
29: **end if**
30: $E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp}$ ▷ Energy flow after handling dangling pages
31: **end procedure**

Algorithm 1 describes the proposed scheme, computes energy efficiency at the beginning and at the end of algorithm. This results in handling dangling pages. Following are the steps performed:

1. The energy flow within the web is computed in the step 2. $|I| = 241, 527$, $E_I^{in} = 0$, $E_I^{out} = 0$, (Because we have considered the web pages as one community, flowing in and out from it is not considered.) $E_I^{dp} = (d/1 - d) * 12, 211$.

$$E_I = 241, 527 + (d/1 - d) * 12, 211.$$

2. The steps 3 to 6 compute the PageRank by using Eq. (1).
3. The steps 7 to 11 discard the pages other than dangling pages by the constraint of outlinks, the dangling pages have zero outlinks.
4. The steps 12 to 15, revise the PageRank score by the considering two evaluation parameters, dwell time (d_t) and click count (C_c).
5. In the step 16, Revised PageRank vector enters the LA environment.
6. Depending upon the output from the environment as a reward or a penalty, the action probability vector is updated by the steps from 21 to 28.
7. 3403 pages out of 12211 dangling pages are punished after the convergence of the system.
8. After performing all the steps, the energy is recomputed:

$$E_I = 241, 527 + (d/d - 1) * (12, 211 - 3403).$$

4.1. Initialize the population

As LA have been used as a learning environment, so the values get changed after getting the response from the environment. The response may be desirable or undesirable. It depends upon user behavior of web surfing, that is the time user stays at a particular webpage and the number of clicks of that webpage. The values get periodically updated. Initially all the values are set to zero.

4.2. PageRank

Previously computed PageRank only depends upon the number of incoming links and the PageRank of those incoming links. The PageRank is revised by measuring the importance of webpage. The importance of webpage is computed by considering three factors: Previous PageRank, Dwell time, Click count. After calculating webpage importance, new PageRank score is computed. Required parameters are evaluated as discussed below:

PageRank: Existing PageRank which is computed using Power method is taken. Then, normalizing process is done for simplifying the values. The range considered is from 0 to 1.

Dwell time: Calculation of time that the user spends on a webpage is done by computing its probability. It is calculated using Gamma Distribution.

Click count: Calculation for the number of clicks on a webpage is normalized by computing its probability. The empirical probability method is used for the same.

After evaluating the above parameters, new PageRank is measured. Now the score is entered into the LA environment. Depending upon the response from the environment, the action probability vector is updated.

4.3. Action probability update

Variable structure learning automata involves a learning algorithm. The output received from the environment is processed in the LA system with the help of learning algorithm. Environment generates an reinforcement signal in which either a reward is granted or a penalty is forced. According the corresponding action, the probability vector is updated.

(a) Desirable response

$$P_i(n) = P_i(n) + P_\sigma \tag{9}$$

$$\forall m \quad m \neq i \quad P_i(n) = (1 - P_i(n)) + P_\sigma \tag{10}$$

In Eq. (9), ranking probability vector of query-url pair is updated. Since it works in case of reward signal from the environment, the probability vector is incremented corresponding to the empirical probability computed for its inlinks. For which url pair Eq. (9) is executed, is selected randomly. Simultaneously, the rest of the query-url pair are updated with Eq. (10). In this equation, the ranking probability vector is updated by the fact, that the sum of all the probability is one. Say, if $P(i)$ is updated by Eq. (9), $(1 - p(i))$ is updated by Eq. (10).

(a) Undesirable response

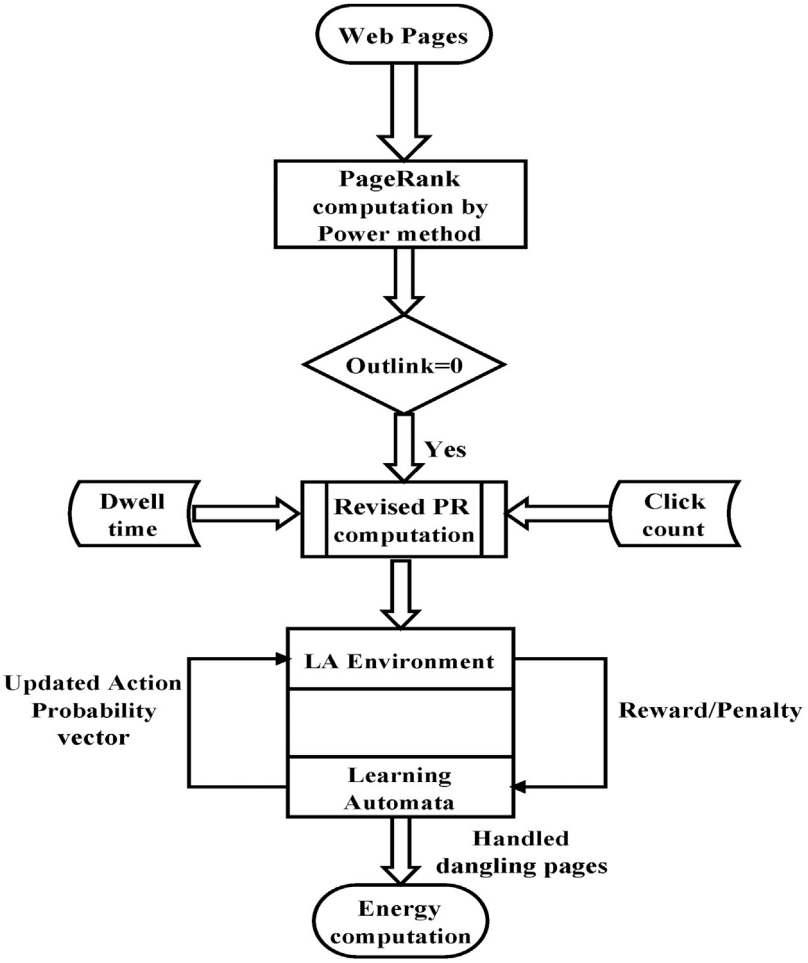

Fig. 9. Proposed scheme.

$$P_i(n) = P_i(n) - P_\sigma \tag{11}$$

$$\forall m \quad m \neq i \quad P_i(n) = (1 - P_i(n)) - P_\sigma \tag{12}$$

Penalty signal being received from the environment, Eqs. (11) and (12) are executed. In equation, the ranking probability vector is punished and reduced by the empirical probability of its inlink. The query url pair being selected is at random and rest of the pairs are punished by Eq. (12).

Accordingly, the ranking probability vector which represents the PageRank score, is updated. Again this vector acts as an input to the environment and the signal is generated which updates the vector. This cycle continues until the optimal solution is received.

The working of action probability update vector has been illustrated in Fig. 10. In this, we have taken the PageRank score of first 10 url pairs. Random selection of target url pair is done by the environment, headed by reward or penalty. According the value is updated along with updating rest of the vector. Finally, system convergence is achieved after multiple iterations which leads to optimal solution.

5. Results and discussion

To validate our proposed scheme, we have conducted experiments on the renowned benchmark dataset. We have performed the experiments in Matlab 2016a. The PageRank value has been revised by considering user behavior activities as discussed in Section 4. The variation between the old PageRank and the revised PageRank is shown in Fig. 11. It can be analyzed that our method accelerated the PageRank score. The relation between the previously computed PageRank and the new is computed by executing it in neural network model. Levenberg Marquardt algorithm has been used as training algorithm.

5.1. Data collection

The web page ranking has been revised in the proposed work, by updating the PageRank score. The experiments are done on the publicly available dataset 'Microsoft Learning to Rank dataset' [49] (MSLR-WEB10K). This dataset has been released on June 2010 and is the random sampling of 10,000 queries. Microsoft Bing search engine provided the labeling set for each query-url pairs varying from 0 (irrelevant) to 4 (perfectly relevant). In data file, each row represents query-url pair. A query-url pair is represented by a 136-dimensional feature vector. The first column represents the relevance label of the pair, the second column is used by query id and the other columns are used for features. The dataset has been partitioned with respect to performing five fold cross validation scheme. Each fold has been divided into three parts training, validation, testing in the ratio of 70:15:15. As the proposed scheme targeted at improving PageRank score by considering the link-based features, so other content based features are ignored. The parameters used for evaluation are described in Table 6. As, we have

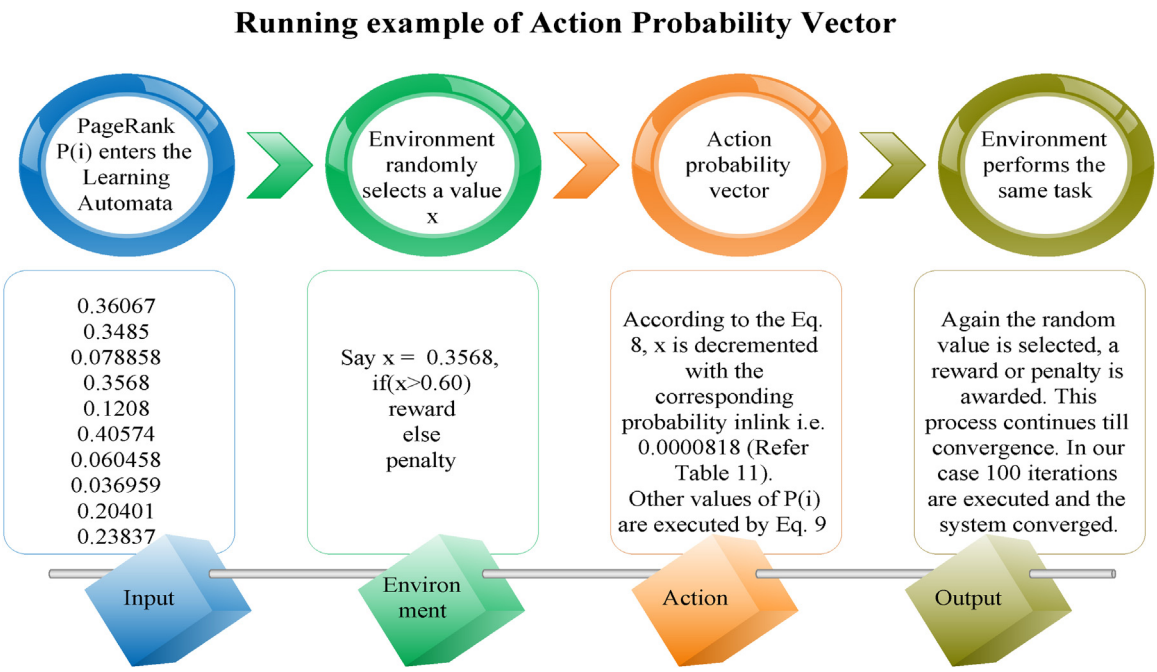

Fig. 10. Working of action probability update vector.

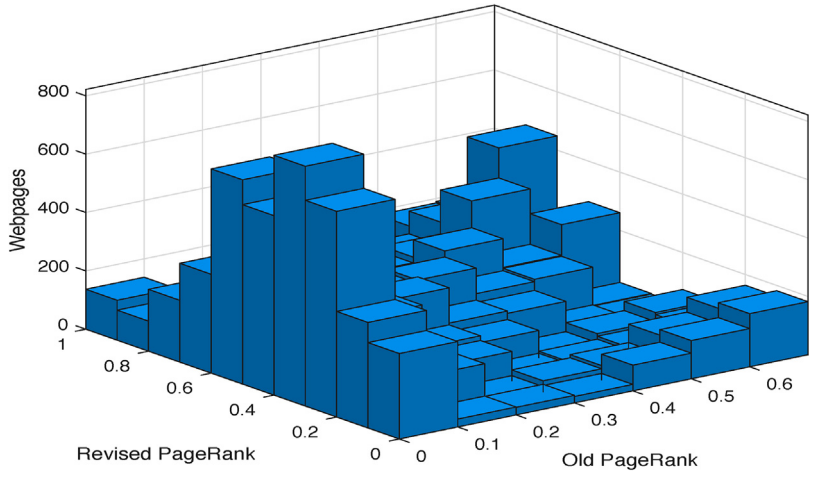

Fig. 11. Comparison of old PageRank with revised PageRank.

Table 6
Parameters of dataset used in proposed scheme.

Feature number	Parameter	Description
128	Inlink number	Empirical probability vector of Inlink is computed for updating the Action Probability vector
129	Outlink number	The dangling pages are used for experiments, if the outlink is zero, then, only it is considered
130	PageRank	It is the already computed PageRank score
135	url click count	The click count of a url is measured for computing the importance of web page
136	url dwell time	The time user spends on the url measured for computing the importance of web page

considered only the dangling pages for the experiments, so, out of 241,527 query-url pair, only 12,211 query-url pairs are considered.

To evaluate the performance of the proposed scheme, the following parameters are considered:

- Learning rate: It is defined as the rate at which the action probability vector is updated. Proper selection of learning rate leads to the success of the algorithm.
- Damping factor (d): It is defined at the rate by which the user switches from one web page to another web page. It is introduced by Larry and Page, and its default value is 0.85.
- Energy consistency: It is defined at the rate by which the energy within the web is maintained. It depends upon the presence of different types of web pages.
- Error rate: It is defined by the difference between the target value and the output value.

5.2. Impact of learning rate on the proposed scheme

The performance of the proposed scheme strongly depends upon the learning rate α. The trade-off between the computa-

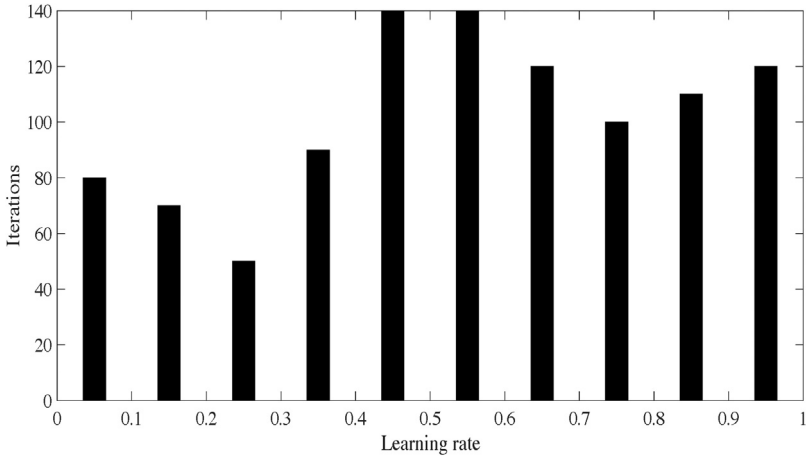

Fig. 12. Learning rate.

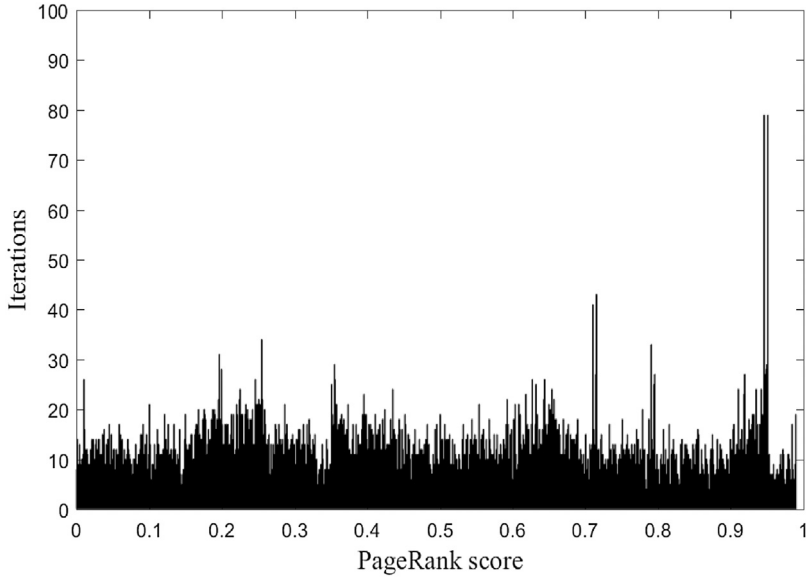

Fig. 13. System convergence.

Table 7
Impact of learning rate on iterations.

Learning rate	Iterations
0.05	80
0.15	70
0.25	50
0.35	90
0.45	140
0.55	140
0.65	120
0.75	100
0.85	110
0.96	120

tional cost and convergence rate can be done optimally with the help of proper selection of learning rate. The computational cost is inversely propositional to the efficient ranking which is target of the proposed scheme. In the experiments, learning rate changes from 0.05 to 0.95 as shown in Table 7. However, the implementation resulted with the iteration cycle with the change in learning rate which is clearly shown in Fig. 12. It can be concluded that the lower learning rate resulted in less number of iterations to converge. Numerical results confirm the best trade-off between the computational cost and convergence rate when the learning rate is set to 0.75. It took 100 iterations for our system to converge as shown in Fig. 13.

5.3. Impact of energy on proposed scheme

The proposed scheme is designed to manage web energy both in terms of energy loss due to dangling pages and the energy loss in computational cost. Dangling pages can be designed by artificially created web pages known as link farm. Detection of such bias dangling pages is essential. In the experiments, energy is management by demoting 3403 dangling pages out of 12,211 dangling pages.

$$E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp} \tag{13}$$

E_I^{out} and E_I^{in} are zero, because the link are not flowing outside the side and none of the webpage outside the web is pointing inside the web. Energy loss due to dangling pages is managed by demoting 3403 web pages. Total energy computed is:

$$E_I = 241,527 + (d/d-1) * (12,211 - 3403) \tag{14}$$

Table 8
Impact of *d* on iterations.

Learning rate	Iterations
0.05	97
0.15	88
0.25	70
0.35	65
0.45	85
0.55	45
0.65	60
0.75	38
0.85	22
0.96	18

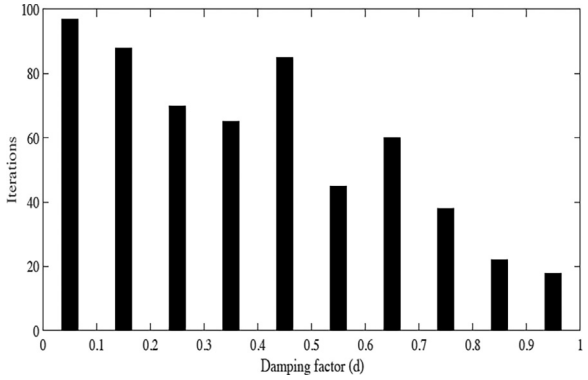

Fig. 14. Damping factor.

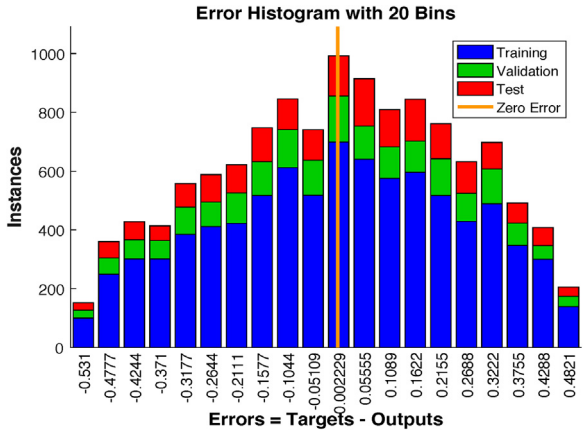

Fig. 15. Error between target and output.

5.4. Impact of damping factor (d) on computational cost

The computational cost of the algorithm is highly dependent upon the value of damping factor (d) used for computation of PageRank. Computational cost is proportional to the number of iterations required for the computation. The trade-off between computational cost and convergence rate is set by fixing the value of d which is 0.85. The iterations for computation is comparatively less when d is 0.95, but the accuracy of the computed values is more accurate in the case when d is 0.85. The experiments conducted for the value of d ranging from 0.05 to 0.95 as shown in Table 8 and results are presented in Fig. 14.

5.5. Impact of error on output

The difference between the target and the output, states the occurrence of error and is depicted in Fig. 15. The histogram is drawn by plotting the training, validating and testing values distributed within 20 bins. The value of bin is 20 by default and can be changed. Just for the simplicity among the fluctuating values, it has not been changed. By considering the maximum number of instances, the error almost diminishes.

Table 9
Correlation: input, output, target and error.

Module	Target value	MSE
Training	8547	6.46256e−2
Validation	1832	6.61828e−2
Testing	1832	6.38464e−2

Table 10
Performance of models.

Model	RMSE	Complexity	Parameters
Bayesian Regression	0.47	12	Conjugacy = −6.194e+02
Stochastic Gradient Descent	0.59	20	Eta() = 0.000049
Closed Form Solution	0.55	14	Lambda() = 0.01

The error corresponding to the values of training, validation and testing (70:15:15), is depicted in Table 9. The correlation between input and error with target and output is presented in Fig. 16.

5.6. Machine learning models

Three machine learning models have been used for scheme validation.

- Bayesian Linear Regression (BLR): It helps to predict the target values with multiple equations having two common parameters, alpha and beta. The parameters are chosen randomly and are used in iterations of equations till they converged. Lamda is computed in between the iterations and is the eigen value vector of Beta.
- Stochastic Gradient Descent (SGD): This method works with the mechanism of computing one row as a weight vector. This vector is then iterated over the other rows for error calculation. As the comparisons are done among all rows, so small number of rows are considered for training.
- Closed Form Solution (CFS): It is one of the simplest regression model which considers the average number of all vectors. Design matrix is in the form of $N^* M$, where N is the number of training samples and M is the number of basis functions.

Results of all the models are shown in Fig. 17.

The performance of each model is depicted in Table 10.

As seen in the above table, Bayesian Regression performed best in all. As the value of alpha and beta are chosen randomly in this model, so the convergence did not actually took place. It is done randomly. By hit and trial method, best suitable values were selected. Thus, resulted in low RMSE. Normal distribution in this model proved to be helpful.

Model performance is shown in Fig. 18. As the training values were large in number, so the colored pixels. The total values are distributed among training, validating and testing pairs of the target and the output. The tested target values and the tested output values lies mainly between 0.4 and 0.6. The other two are scattered between 0 and 1. It can be observed that the error remains constant throughout 12211 values (roughly taken as 12,000).

Regression R values are computed which depicts relation between the target and the output. This relationship is presented in Fig. 19. The more the value is close to 1, the more is the correlation. At all the three phases, training, validating and testing, the R value remains close to 0.1. This means, it is a random relation and is generated randomly.

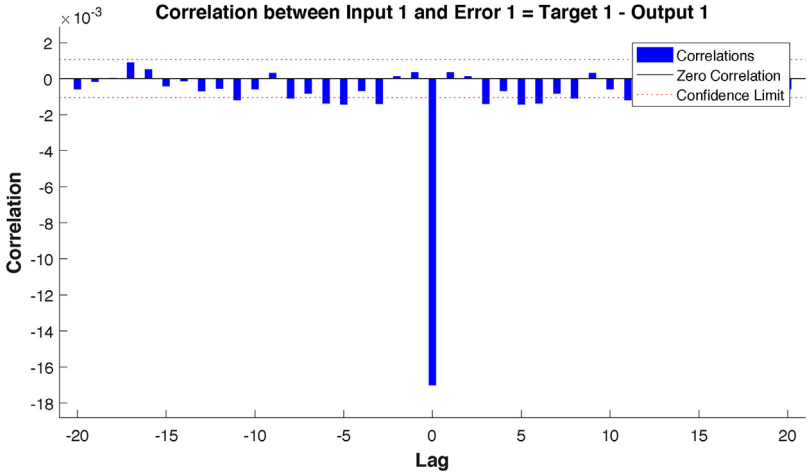

Fig. 16. Input, output, target, error correlation.

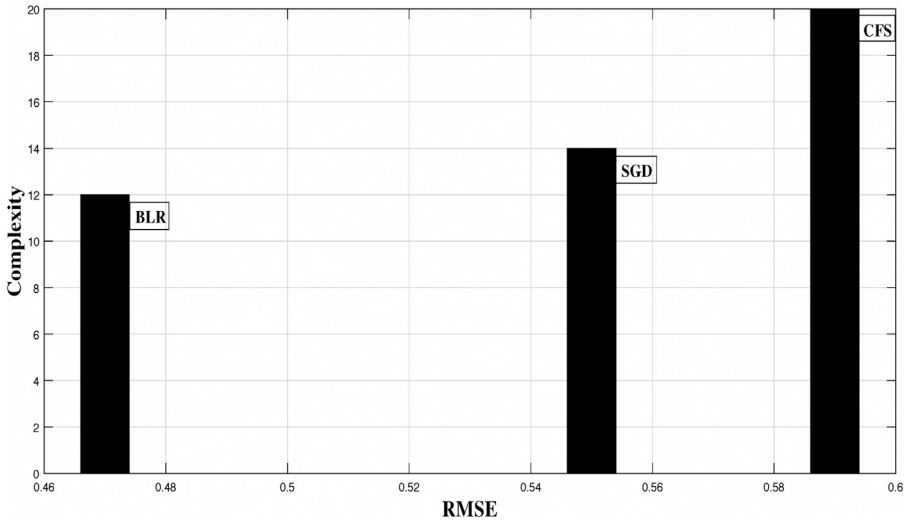

Fig. 17. Performance of models.

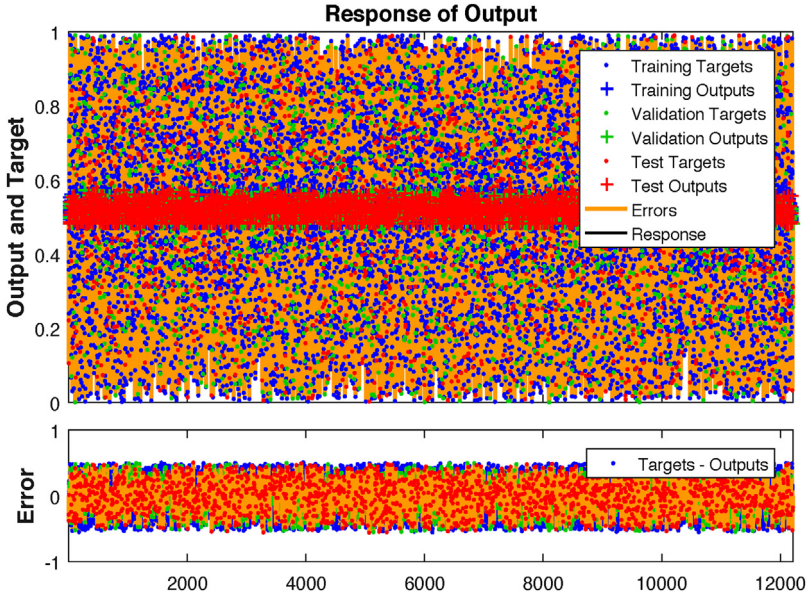

Fig. 18. Overall performance of algorithm.

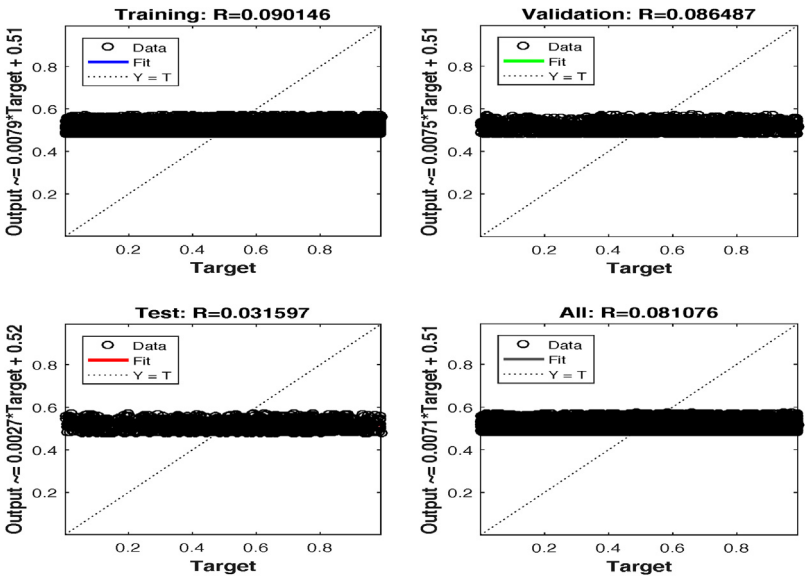

Fig. 19. Relation between output and target.

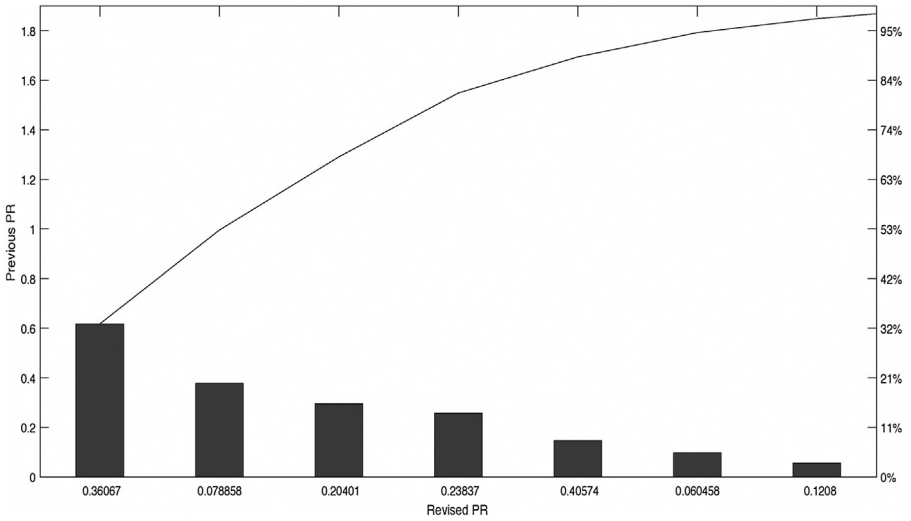

Fig. 20. Comparison between previous PageRank with revised PageRank.

5.7. Analysis

The proposed scheme updates the PageRank vector which is more efficient in terms of computational cost, convergence rate, energy efficient, handling irrelevant dangling pages. The updated PageRank is compared with existing PageRank and few query url pair is represented in Fig. 20.

Table 11 represents the snapshot of the values generated in our experiments.

1st column: It represents the query-url pair number. Total number of pairs considered for experiments are 12,211.
2nd column: It represents the outgoing links of each pair. As, we have considered dangling pages for experiments, so, the value of outgoing links is always zero.
3rd column: It represents the PageRank score computed with the Power method.
4th column: It represents the time user spends on web page. It is calculated using Gamma Distribution.
5th column: It represents the number of clicks on the web page. It is calculated using empirical probability method.
6th column: It represents the modified PageRank computed after analyzing user behavior attributes (dwell time, click count).
7th column: It represents the number of inlinks of each pair. It is normalized using empirical probability.
8th column: It represents the PageRank computed by the proposed scheme in LA environment.

5.7.1. Findings/significance of comparison between old PageRank and revised PageRank

- 3403 dangling web pages out of 241,527 web pages, have been degraded by the proposed approach.
- Computational cost gets automatically decreased by the demotion of dangling pages, which in turns saves energy.
- Energy flow within web becomes more efficient.

5.7.2. Complexity analysis

Time Complexity: In this algorithm, Step 2 is the linear computation of energy which takes O(n) time. The loop in Steps 3–6 and steps 7–11 takes O(n) time in worst case. Steps 20 to 29 are

Table 11
Snapshot of values in experiments.

Query-url pair number	Number of Outgoing Links	PageRank	Dwell Time	Click count	Revised PageRank	Inlink	Final PageRank
1	0	0.61775	5.8	1	0.36067	0.000327	0.36591
2	0	0.00621	11.6	1	0.3485	0.41397	0.9515
3	0	0.37786	14.4	7	0.078858	0.41397	0.62114
4	0	0.0384	3.4	1	0.3568	0.0000818	0.35811
5	0	0.05635	5.2	3	0.1208	0.41397	0.7792
6	0	0.14588	2.6	1	0.40574	0.41397	0.00574
7	0	0.0979	7.2	9	0.060458	0.41397	0.63954
8	0	0.0062	16.2	6	0.036959	0.41397	0.66304
9	0	0.29533	14.7	7	0.20401	0.41397	0.89599
10	0	0.25744	8.1	4	0.23837	0.41397	0.86163
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
12,202	0	0.06982	15.2	1	0.41787	0.049955	0.23245
12,203	0	0.00266	6.6	2	0.18594	0.41397	0.71406
12,204	0	0.37256	3.8	3	0.47575	0.41397	0.07575
12,205	0	0.25436	5.8	2	0.43764	0.000163	0.44026
12,206	0	0.02942	13.3	5	0.074543	0.049955	0.87382
12,207	0	0.14851	5.6	1	0.49656	0.009991	0.65642
12,208	0	0.14809	7.8	2	0.33137	0.023995	0.71529
12,209	0	0.01336	7.9	2	0.19664	0.0073704	0.31457
12,210	0	0.00598	4.5	2	0.18926	0.41397	0.71074
12,211	0	0.0325	5.1	4	0.098916	0.013103	0.30856

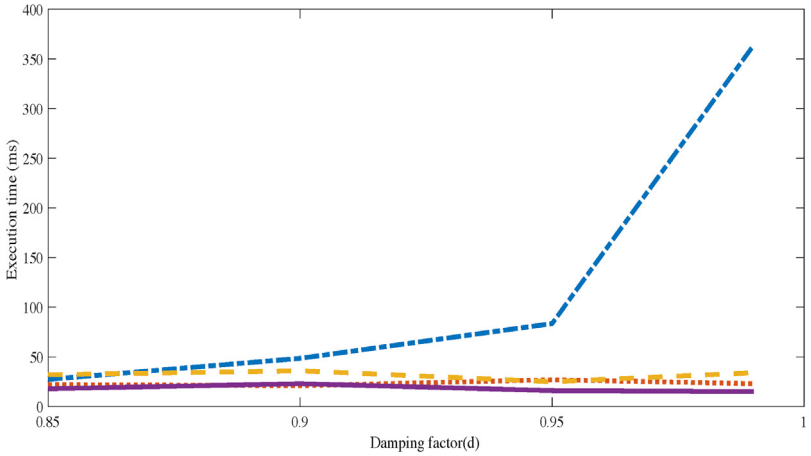

Fig. 21. Execution time analysis of various algorithms.

conditional operations in LA system which takes O(1) time. Time complexity (TC) is computed as below:

$$\Rightarrow \quad TC = O(n) + O(n) + O(1)$$

$$\Rightarrow \quad TC = O(n)$$

Space Complexity: In this algorithm, input is fixed which cannot exceed n, thus taking $O(n)$ space. The loops also takes $O(n)$ space. The arithmetic operations take O(1) space. Space complexity (SC) is computed as below:

$$\Rightarrow \quad SC = O(n) + O(n) + O(1)$$

$$\Rightarrow \quad SC = O(n)$$

5.8. Comparisons with existing methods

The proposed scheme is compared with existing benchmark algorithms. PageRank as the base algorithm and the evaluating parameter is the value of d which is used for computing the PageRank. The proposed scheme for the PageRank computation is more beneficial as it is cost effective. The executive time of PageRank, MIRAN [50], revised PR (by effective user behavior measures) with the proposed method computed in LA environment. The experimental values are depicted in Table 12.

Table 12
Comparison of proposed algorithm with other algorithms.

d	Power method	MIRAN	Revised PR	Proposed method
0.85	27.29	22	32	21.88
0.90	48.53	21	36	23
0.95	83.56	27	25	26
0.99	363.94	23	34	24

Fig. 21 presents the analysis of four algorithms. The proposed scheme takes less than 50 ms to execute for all the values of d.

6. Conclusion

In this article, we have proposed the framework for ranking of webpages. We paid attention towards user browsing behavior, not only on the hyperlink structure of graph. The Markov process of inlinks is followed in our model. Firstly, the dangling

pages are distinguished and their importance is computed. Webpage importance is evaluated using two parameters, dwell time and click count. Secondly, the PageRank score is revised using webpage importance score. Then, the revised score is updated in the LA environment depending upon the response. We have used PageRank score as the benchmark. Our scheme has been validated using three machine learning models namely, Bayesian Regression, Stochastic Gradient Descent, Closed Form Solution. In all three, Bayesian Regression performed best.

In future, we are planning to work upon the following issues:

1. One of the important user behavior for Internet surfing is switching between the webpages. Visiting patterns are not easy to analyze. We are planning to investigate the principled method for recording and evaluating the webpage importance through it.
2. Spam pages are difficult to detect. We target to introduce the framework which can demote the spam pages while entering the search engine ranking algorithm.

References

[1] W.R. Algorithm, Internet Users, 2016 (accessed 10.11.16) http://www.internetlivestats.com/internet-users/.
[2] N. Applications, Desktop Search Engine Market Share, 2016 (accessed 10.11.16) https://www.netmarketshare.com/search-engine-market-share.aspx?qprid=4&qpcustomd=0.
[3] Netcraft, March 2016 Web Server Survey, 2016 (accessed 10.11.16) https://news.netcraft.com/archives/2016/03/18/march-2016-web-server-survey.html.
[4] Z. Gyöngyi, H. Garcia-Molina, J. Pedersen, Combating web spam with trustrank, Proceedings of the Thirtieth International Conference on Very Large Data Bases, vol. 30, VLDB Endowment (2004).
[5] R. Andersen, C. Borgs, J. Chayes, J. Hopcroft, K. Jain, V. Mirrokni, S. Teng, Robust PageRank and locally computable spam detection features, in: Proceedings of the 4th International workshop on Adversarial Information Retrieval on the Web, ACM, 2008.
[6] H. Costa, F. Benevenuto, L.H. Merschmann, Detecting tip spam in location-based social networks, in: Proceedings of the 28th Annual ACM Symposium on Applied Computing, ACM, 2013.
[7] H. Shen, Z. Li, Leveraging social networks for effective spam filtering, IEEE Trans. Comput. 63 (11) (2014).
[8] H. Tran, T. Hornbeck, V. Ha-Thuc, J. Cremer, P. Srinivasan, Spam detection in online classified advertisements, in: Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality, ACM, 2011.
[9] A. Heydari, M. ali Tavakoli, N. Salim, Z. Heydari, Detection of review spam: a survey, Expert Syst. Appl. 42 (7) (2015).
[10] X. Wang, T. Tao, J.-T. Sun, A. Shakery, C. Zhai, Dirichletrank: solving the zero-one gap problem of PageRank, ACM Trans. Inf. Syst. (TOIS) 26 (2) (2008).
[11] M. Elhoseny, A. Hosny, A.E. Hassanien, K. Muhammad, A.K. Sangaiah, Secure automated forensic investigation for sustainable critical infrastructures compliant with green computing requirements, IEEE Trans. Sustain. Comput. (2017).
[12] L. Li, X. Chen, Y. Song, The PageRank model of minimal irreducible adjustment and its lumping method, J. Appl. Math. Comput. 42 (12) (2013).
[13] C.P.-C. Lee, G.H. Golub, S.A. Zenios, A fast two-stage algorithm for computing PageRank and its extensions, Sci. Comput. Comput. Math. 1 (1) (2003).
[14] Y. Lin, X. Shi, Y. Wei, On computing PageRank via lumping the Google matrix, J. Comput. Appl. Math. 224 (2) (2009).
[15] Y. Liu, T.-Y. Liu, B. Gao, Z. Ma, H. Li, A framework to compute page importance based on user behaviors, Inf. Retr. 13 (1) (2010).
[16] N. Kumar, S. Tyagi, D.-J. Deng, LA-EEHSC: learning automata-based energy efficient heterogeneous selective clustering for wireless sensor networks, J. Netw. Comput. Appl. 46 (2014) 264–279.
[17] Z. Anari, M.R. Meybodi, B. Anari, Web page ranking based on fuzzy and learning automata, in: Proceedings of the International Conference on Management of Emergent Digital EcoSystems, vol. 24, ACM, 2009.
[18] J.A. Torkestani, An adaptive learning automata-based ranking function discovery algorithm, J. Intell. Inf. Syst. 39 (2) (2012) 441–459.
[19] N. Kumar, K. Kaur, A. Jindal, J.J. Rodrigues, Providing healthcare services on-the-fly using multi-player cooperation game theory in Internet of Vehicles (IoV) environment, Digit. Commun. Netw. 1 (3) (2015) 191–203.
[20] L. Page, S. Brin, R. Motwani, T. Winograd, The PageRank Citation Ranking: Bringing Order to the Web, 1999.
[21] S.J. Kim, S.H. Lee, An improved computation of the PageRank algorithm, in: European Conference on Information Retrieval, Springer, 2002, pp. 73–85.
[22] B. Wu, V. Goel, B.D. Davison, Topical trustrank: using topicality to combat web spam, in: Proceedings of the 15th International Conference on World Wide Web, ACM, 2006.
[23] Q. Yu, Z. Miao, G. Wu, Y. Wei, Lumping algorithms for computing Google's PageRank and its derivative, with attention to unreferenced nodes, Inf. Retr. 15 (6) (2012).
[24] I.C. Ipsen, T.M. Selee, PageRank computation, with special attention to dangling nodes, SIAM J. Matrix Anal. Appl. 29 (4) (2007).
[25] V.L. Praba, T. Vasantha, Efficient hyperlink analysis using robust Proportionate Prestige Score in PageRank algorithm, Appl. Soft Comput. 24 (2014).
[26] Z. Zheng, A.K. Sangaiah, T. Wang, Adaptive communication protocols in flying ad hoc network, IEEE Commun. Mag. 56 (1) (2018) 136–142.
[27] T. Qiu, X. Liu, K. Li, Q. Hu, A.K. Sangaiah, N. Chen, Community-aware data propagation with small world feature for internet of vehicles, IEEE Commun. Mag. 56 (1) (2018) 86–91.
[28] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, M. Gonçalves, Detecting spammers and content promoters in online video social networks, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2009.
[29] P. Heymann, G. Koutrika, H. Garcia-Molina, Fighting spam on social web sites: a survey of approaches and future challenges, IEEE Internet Comput. 11 (6) (2007).
[30] M. Chakraborty, S. Pal, R. Pramanik, C.R. Chowdary, Recent developments in social spam detection and combating techniques: a survey, Inf. Process. Manag. (2016).
[31] N. Eiron, K.S. McCurley, J.A. Tomlin, Ranking the web frontier, in: Proceedings of the 13th International Conference on World Wide Web, ACM, 2004.
[32] Y. Liu, R. Cen, M. Zhang, S. Ma, L. Ru, Identifying web spam with user behavior analysis, in: Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web, ACM, 2008.
[33] Y. Liu, F. Chen, W. Kong, H. Yu, M. Zhang, S. Ma, L. Ru, Identifying web spam with the wisdom of the crowds, ACM Trans. Web (TWEB) 6 (1) (2012).
[34] C. Wang, A. Kalra, C. Borcea, Y. Chen, Webpage depth-level dwell time prediction, in: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, ACM, 2016, pp. 1937–1940.
[35] E. Agichtein, E. Brill, S. Dumais, R. Ragno, Learning user interaction models for predicting web search result preferences, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2006, pp. 3–10.
[36] S. Xu, H. Jiang, F.C.-M. Lau, Mining user dwell time for personalized web search re-ranking, in: International Joint Conference on Artificial Intelligence (IJCAI 2011), AAAI Press/International Joint Conferences on Artificial Intelligence, 2011.
[37] P. Hayati, K. Chai, V. Potdar, A. Talevski, Behaviour-based web spambot detection by utilising action time and action frequency, Comput. Sci. Appl. ICCSA 2010 (2010) 351–360.
[38] V. Dave, S. Guha, Y. Zhang, Measuring and fingerprinting click-spam in ad networks, ACM SIGCOMM Comput. Commun. Rev. 42 (4) (2012) 175–186.
[39] M. Bilenko, R.W. White, Mining the search trails of surfing crowds: identifying relevant websites from user activity, in: Proceedings of the 17th International Conference on World Wide Web, ACM, 2008, pp. 51–60.
[40] X. Yi, L. Hong, E. Zhong, N.N. Liu, S. Rajan, Beyond clicks: dwell time for personalization, in: Proceedings of the 8th ACM Conference on Recommender systems, ACM, 2014, pp. 113–120.
[41] G. Dupret, M. Lalmas, Absence time and user engagement: evaluating ranking functions, in: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, ACM, 2013, pp. 173–182.
[42] T. Qiu, Y. Zhang, D. Qiao, X. Zhang, M.L. Wymore, A.K. Sangaiah, A robust time synchronization scheme for industrial internet of things, IEEE Trans. Ind. Inf. (2017).
[43] T. Qiu, R. Qiao, M. Han, A.K. Sangaiah, I. Lee, A lifetime-enhanced data collecting scheme for the internet of things, IEEE Commun. Mag. 55 (11) (2017) 132–137.
[44] M. Bianchini, M. Gori, F. Scarselli, Inside PageRank, ACM Trans. Internet Technol. (TOIT) 5 (1) (2005) 92–128.
[45] W. Saad, Z. Han, A. Hjorungnes, D. Niyato, E. Hossain, Coalition formation games for distributed cooperation among roadside units in vehicular networks, IEEE J. Sel. Areas Commun. 29 (1) (2011) 48–60.
[46] S. Misra, B.J. Oommen, S. Yanamandra, M.S. Obaidat, Random early detection for congestion avoidance in wired networks: a discretized pursuit learning-automata-like solution, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 40 (1) (2010) 66–76.
[47] N. Kumar, S. Misra, M.S. Obaidat, Collaborative learning automata-based routing for rescue operations in dense urban regions using vehicular sensor networks, IEEE Syst. J. 9 (3) (2015) 1081–1090.
[48] N. Kumar, J.-H. Lee, J.J. Rodrigues, Intelligent mobile video surveillance system as a Bayesian coalition game in vehicular sensor networks: learning automata approach, IEEE Trans. Intell. Transp. Syst. 16 (3) (2015) 1148–1161.
[49] T. Qin, T. Liu, Introducing LETOR 4.0 Datasets, CoRR abs/1306.2597, http://arxiv.org/abs/1306.2597.
[50] Z. Liu, N. Emad, S.B. Amor, M. Lamure, PageRank computation using a multiple implicitly restarted Arnoldi method for modeling epidemic spread, Int. J. Parallel Program. 43 (6) (2015) 1028–1053.