AN EFFICIENT SPAMMER CLASSIFICATION FOR RANKING OF WEB PAGES

A Thesis submitted in fulfillment of the requirement for the award of the degree of

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Aaisha Makkar

(Registration No. : 901503025)

Under the guidance of:

Dr. Neeraj Kumar
Associate Professor, CSED

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY,
PATIALA – 147004

JUNE 2020

CERTIFICATE

I, Aaisha Makkar, Regn. No. 901503025, hereby declare that the thesis entitled **"An Efficient Spammer Classification for Ranking of Web Pages"** submitted to the Computer Science and Engineering Department at Thapar Institute of Engineering & Technology, Patiala, Punjab, India is an authenticated record of my own work for the award of the degree of "Doctor of Philosophy" under the supervision of Dr. Neeraj Kumar. This report has not been submitted to any other institution for award of any other degree.

Aaisha Makkar
Regn. No. 901503025

Place: Patiala
Date: 06/07/2020

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Verified by:

Dr. Neeraj Kumar
Professor
CSED
Thapar Institute of Engg. & Tech.

ABSTRACT

Inaccurate search engine result page (SERP) is one of the significant drawbacks of the search engine ranking algorithm. Web spam is one of its primary cause. Although there are many techniques which have detected web spam by analyzing the content features and link features of a web page. These spam detection techniques primarily focused on revising the rank score of a web page after being included in SERP. But, none of these techniques targets at preventing the web spam before assigning a rank by the ranking algorithm.

For the successful SERPs, the web pages should be completely spammed free before entering into the ranking module. For this purpose, web spam pages should be demoted by the ranking algorithm itself to reduce their rank score. This mechanism should be implemented in such a way that the authoritative web sites get promoted. The complete analysis and study of Google ranking methodology, i.e., PageRank, is done. The various measures affecting the rank score computation in PageRank are investigated. As a result, the primary cause of injection of spam web pages on the web is due to the presence of dangling web pages.

Dangling pages are the webpages which do not have hyperlinks. The ratio of dangling pages is increasing due to the documents such as pdf, technical reports from research communities. Spammers create artificial in-links to boost the rank of webpages, but the outgoing links are not focused. Thus, it results in dangling pages. Although a lot of work has been done for handling dangling pages and improving the ranking algorithm. But, none of these has handled dangling pages concerning user behavior analysis. User surfing activities can only predict the real picture of a webpage. Evaluating the importance of dangling page can significantly help in refining the SERPs. This task has been accomplished in this research work with two different approaches.

The first approach detects the spam dangling web pages by considering the user behavior attributes, i.e., dwell time and click count. Web page importance score is computed by analyzing user surfing behavior attributes, dwell time, and click count. After calculating 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 val-

idated the 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 system 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.

The second approach presents an intelligent cognitive spammer framework, Cognitive spammer, which eliminates the spam pages during the web page rank score calculation by search engines. The framework updates Googles ranking algorithm, PageRank in such a way that it automatically prevents link spam. It considers the link structure of the web for rank score computation. The updated PageRank algorithm provided a better ranking of web pages. The proposed framework is validated with the WEBSPAM-UK2007 dataset. Before processing, the dataset is preprocessed with a new technique, called Split by Over-sampling and Train by Under-fitting to remove the trade off between imbalanced instances of the target class. After data cleaning, we applied machine learning techniques (Bagged model, Boosted linear model, etc.) with the web page features to make accurate predictions. The detection classifiers only consider the link features of the web page irrespective of the page content. Out of the fifteen classifiers, the best three are ensemble, which results in better performance with overall accuracy improvement. Ten-fold cross-validation has also been applied with the resulted ensemble model, which results in getting the accuracy of 99.6% in the proposed scheme.

ACKNOWLEDGMENTS

Before discussing my journey of Ph.D., I would like to thank the almighty God who gave me strength and courage to overcome all the obstacles and complete this endeavour. The aim of my life, to be called by the salutation of a 'Doctor', seems to become a reality, when I got admission in Doctorate of philosophy in Thapar University. Research initiated with this startup in my life. Without acknowledging the people who supported me throughout this journey, this task would be incomplete. I know words are never enough to express the gratitude; I am just delivering the phrase for the acceptance of regards.

Firstly, I would like to express my sincerest thanks to my parents. With their consent, support and motivation, I thought to accept this biggest challenge in my life. They have been the true source of real inspiration for me. Secondly, I would never forget the day when Dr. Deepak Garg gave me the opportunity to take admission after conducting my interview. I am really grateful to him. Then, I would like to thank my supervisor, Dr. Neeraj Kumar, who have supported me throughout my Ph.D. work with his patience and knowledge; while providing me with the room to work in my own way. Apart from providing me with excellent supervision, active cooperation and constant encouragement throughout this journey, he also shared their invaluable experiences with me to succeed in life. I will always remain indebted to him.

My research is incomplete without the technology of machine learning, which is taught and guided by Dr. Prashant Singh Rana. I am really thankful to him. He led me to the correct direction at every stage of this research work. I am also grateful to the head of the department, Prof. Maninder Singh, Prof. Inderveer Chana, Ph.D. Coordinator, and members of my doctoral committee, Prof. Seema Bawa, Dr. Shalini Batra and Prof. Kulbir Singh, for their constructive suggestions and ensuring the correct pace of my work. I am also obliged to the Director, Prof. Prakash Gopalan, Dean (RSP), Prof. O. P. Pandey and the management of Thapar Institute of Engineering and Technology, who provided me with all the necessary resources and facilities to complete my work.

The chain of my gratitude will definitely be incomplete if I forget to thank my complete family, my husband Sr. Komal Pal Singh, my father Sr. Harbhajan Singh, my mother Mrs. Balwant Kaur, my younger brother Pavneet Singh and my daughter Prabhleen Kaur, for their unconditional love, support and encouragement in every phase of my life. It was due to my

father's and husband's confidence and vision that motivated me to overcome every obstacle during the research. Since then, the journey of Ph.D. has been a sweet and bitter ride at times which leads to a special mention for my mother who stood by me through thick and thin and gave me courage at the times when I felt really low. Her constant motivation showed me the silver lining in the dark clouds.

I would also like to express heartfelt thanks to my elder brother, Maninder Singh, who always believed in me and whose blessings have truly played the role of game-changer in my life. It would also be unworthy of me if I forget to mention about the love, blessings, and good luck constantly showered on me by my lovely daughter, Prabhleen Kaur and my younger brother, Pavneet Singh. I would also like to pay my sincere regards to all my relatives and cousins for their constant motivation and support. They made this journey more comfortable with words of encouragement which helped me in finishing my work.

I would also like to thank my friends and colleagues with whom I have traveled this journey of research. A special thanks to my research group, Dr. Kujeet Kaur, Dr. Anish Jindal, Dr. Gagangeet Singh and budding doctors, Ms. Ankita Wadhawan, Mr. Rajat Chaudhary, Mr. Ishan Budhiraja, Ms. Shubhani Aggarwal, Ms. Arzoo Miglani, and Mr. Himanshu Sharma. These people have made my research journey all the more memorable and pleasant. As one cannot mention the names of all well-wishers, friends and beloved ones, I would like to pay my regards to one and all who supported me during this journey of knowledge.

(**Aaisha Makkar**)

List of Publications

Journal Publications (SCI/SCIE):

1. **Aaisha Makkar**, Neeraj Kumar, *User behavior analysis for Webpage Ranking: Learning Automata based Solution,* **Sustainable computing, informatics and systems, Vol No.: 20, pp. 174–191, 2018**, SCI indexed, (IF:1.800), DOI: doi.org/10.1016/j.suscom.2018.02.003

2. **Aaisha Makkar**, Neeraj Kumar, *Cognitive Spammer: A Framework for PageRank analysis with Split by Over-sampling and Train by Under-fitting*, **Future Generation Computer Systems, Vol No.: 90, pp. 381–404, 2019**, SCI indexed, (IF: 5.768), DOI: doi.org/10.1016/j.future.2018.07.046

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Today the world is highly dependent upon the Internet. Different classes of users access plenty of knowledge from the world wide web (WWW). According to a survey conducted by an Internet service company Netcraft [40], it has been observed that there are currently 1,003,887,790 web sites on the WWW. Moreover, according to the recent report from Internetlivestats [41] (an Internet survey company), states that there are currently 3.4 billion Internet users in the world.

Nowadays, web users are not only looking for relevant information but also for authoritative sources of that information, i.e., trusted sources of correct and authentic information [42]. For example, they are getting the data directly from the home page of a company. Therefore, in current web searches, there is a paradigm shift towards authoritativeness from relevance. So, the primary job of the search engine ranking algorithms is shifted in finding and ranking the more authoritative web documents [43].

According to Netmarketshare's latest statistics [44], the largest market share of search engine usage is from Google (75.2%) followed by Bing, Baidu and Yahoo. Google holds the maximum market shares of search engines, as it uses more than 200 features in its ranking algorithm. Few of the features such as - PageRank, Host, M_DES, Path are used by Google search engine, which in turn gives 90% accuracy in the search results [45]. Google crawling methodology has experimented the user queries to show the actual working concerning 60 keywords of four categories. It downloads the top 100 web pages for each keyword [45].

The ranking is the most important module which is responsible for the success of search engines. Google uses the PageRank algorithm for ranking the search results. According to the ranking algorithm [46], PageRank is highly dependent on the number of in-links and the PageRank score of those in-links. PageRank works the same as the citation indexes in the literature of science for depicting the publications importance [47]. After its successful implementation, its methodology is adopted in various domains such as networking, multimedia, web spam detection, security, medical and web graphs. It has been surveyed that 23% of web pages are changed daily using poisson process model [1].

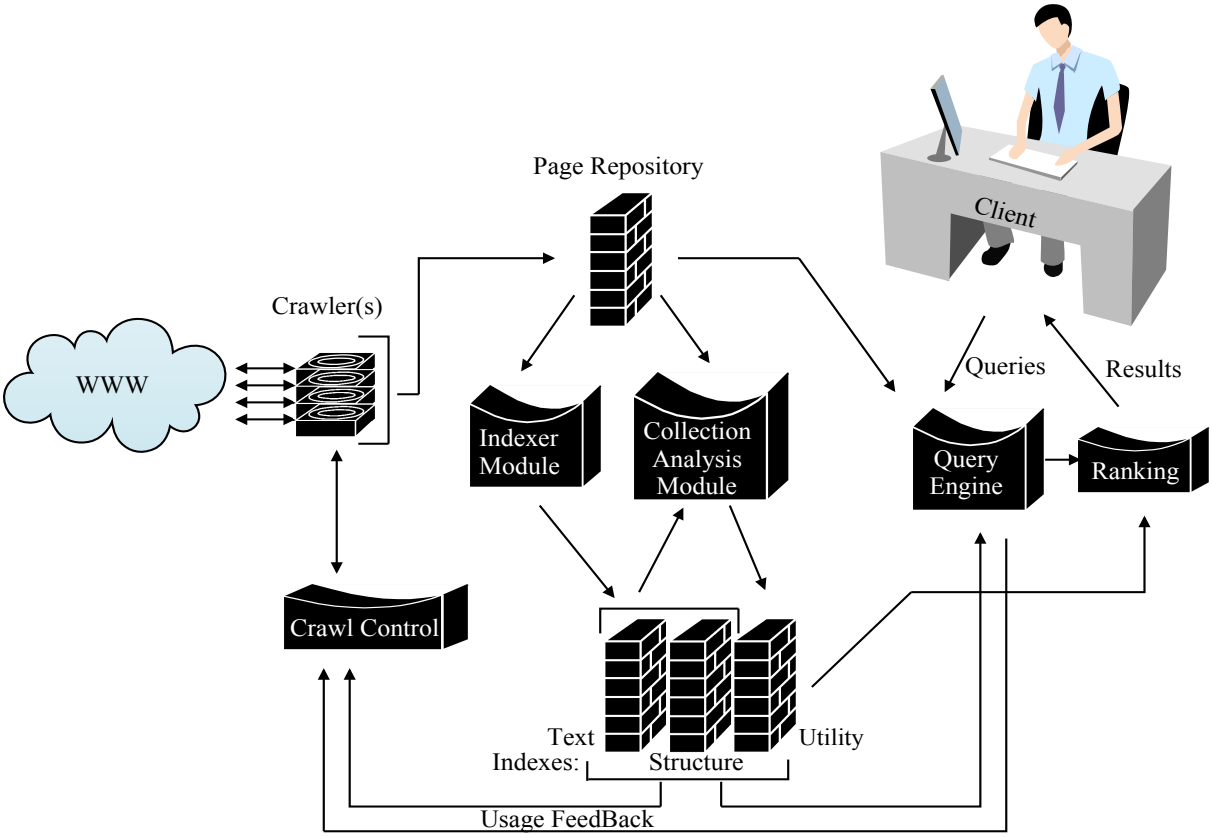


Figure 1.1: General search engine architecture [1]

1.1 Architecture and components of Search Engine

The most common search engine architecture is as shown in Fig. 1.1. There are four important modules of a search engine, namely crawler, storage, indexer, and ranking. These modules are as follows.

1.1.1 Crawler module

Crawler browses the web by traversing through it. The indexer examines the pages retrieved by the crawler. The crawler uses the steps are discussed below:

1. Initial set of URLs is provided to the crawler using which it starts its functioning.
2. By placing all the set of URLs in the queue, prioritization starts.
3. The order of the queue is then changed, and the pages are downloaded in the new sequence.
4. The downloaded pages may contain more URLs which are placed in queue again.
5. Steps 2 to 4 continues until all the out-links of downloaded pages are crawled.

However, after getting the results, page selection is an important decision [1], which follows the following steps:

Selection

The pages to be selected and downloaded may depend upon a number of parameters such as - user query, popularity, and location. [1].

Updation

The crawler needs to refresh the pages periodically as a search engine should be updated with the new web pages. As a large number of pages are added to the web every day, crawler should have the capability to check the newly added pages to be downloaded.

Crawling is a challenging task as each HTTP request takes one second to respond. So, only 86,400 pages can be fetched per day. The crawling of 20 million web pages, the estimated time period is 634 years [48]. Hence, crawling is done using hundreds of machines. A hashing function is defined to perform the role of each device with a fixed number of URLs. But, if the machine encounters an unexpected URL, then it is generally passed to another machine. A server may have multiple URLs at the same time. But, crawling algorithms use a single request at a time. The crawler needs to take care of many issues, and few of these are- duplicate content, excluded content, politeness, and spam rejection [48].

Refreshing the web pages needs to be performed periodically by the crawler. Different policies exist in the literature for this purpose. Firstly, probabilistic models, along with sampling and classification-based approaches, have been designed to monitor the updated history of the web and predictions for the future while training the system [49]. After drawing the evaluation matrices, the frequency of updating the web pages is drawn [49]. But, the major limitation of such schemes is that they need historical information of all the web pages for future prediction. To address this issue, a new clustering-based algorithm [50] was developed which did not require any factual information.

This algorithm was based on clustering for downloading the pages efficiently as it took the web page features into consideration [50]. Along with static features like the number of images, file size in bytes, and several tables, dynamic features were also added to know which new pages are added. Moreover, dynamic content features were also used namely change in the file size, change in the number of images, and change in the number of tables. Furthermore, dynamic linkage features were used, for instance- change in PageRank, change in the number of outgoing links, and change in the number of incoming links. For the construction of hierarchical clusters, repeated bisection clustering (RBC) algorithm was followed [50]. A focused crawler has been used for downloading the web pages of a particular topic which implemented a best-first search strategy. This crawler was rule-based, which has used the linkage information among the different topics [51].

1.1.2 Page repository

The crawler passes the downloaded pages to the module responsible for storage known as-page repository, as shown in Fig. 1.1. The indexer and the collection analysis module uses the data stored by this module. It is similar to the database systems or file systems where many operations such as - insertion, deletion, and updation, are performed on the data. However, storage module of search engines has many challenges [52] as described below:

1. Scalability: With the large size of the web, it should be scalable enough to distribute the data among various devices. Uniform and hash distribution techniques like Locality-Sensitive Hashing (LSH) [53] are widely used for distributing the pages.

2. Type of Access: As shown in Fig. 1.1, indexer and query engine use the storage module. But, both the demand for different types of access. Query engine expects any single or multiple webpages in response to a single query. These web pages are located at random positions in the page repository. Indexer requires the complete set or subset of web pages which are collected by the crawler but maintained by the storage module.

3. Updations: New web pages are added on a daily basis, so the storage module is responsible for updating its collection with the new web pages. The space occupied by the previous pages should also be reclaimed.

4. Garbage collection: As some web pages occupy space and are no more alive. So, these web pages should be either removed or shifted to external storage for efficient storage of other web pages. An Internet service company Factshunt [54] states that the total number of websites on WWW is 759 Million, among which only 510 Million are live websites. So, dead web pages/sites should be removed from the repository.

5. Page Identification: A page is stored with a unique identification number, which is similar to the roll number of a student. But, sequential order of assigning the page identifier may not work due to the large size of the web. Each page has an associated URL with it. This URL is encrypted using the technique, checksum, which generates an identifier.

6. Refresh schemes: The two most important functions performed by the storage manager is updation and garbage collection. A page may be updated with time. But, if other pages refer the older version of the page, then it should not be removed. In such a case, two versions of a particular page should be maintained. Many update strategies like batch update or incremental update can be adopted for this purpose. Spam pages may also become obsolete with time and have to be removed from the repository.

1.1.3 Indexing module

As shown in Fig. 1.1, indexer prepares indexes along with collection analysis modules. Specifically, the indexer module builds two major indexes, text and link. Index preparation is similar to of book index, but the difference is concerning their size. As web index may contain millions of entries referring to trillions of web pages. So, collection analysis modules build various useful indexes by analyzing two indexes(text and link) prepared by the indexer and the pages collected in the page repository. It also checks the requirement of indexes by query engine and ranking module.

These modules are inter-dependent as it has been observed that the crawler saves the page content while saving the URLs of the downloaded pages in the queue. Indexing is based on the index terms, which are extracted from the inverted index created by the indexer in the following two phases. The first phase is the scanning in which the text of the document is scanned for encountering the index term. In this, each term is numbered along with the document number and saved in a temporary file. The second phase is an inversion in which the temporary file is sorted with the term number as primary key and document number as the secondary key. The document created by the second phase is then summarized into dictionary [48]. Following are the different types of web indexes prepared by an indexer.

1. Link-Based Index: Web Structure is treated as a graph where each node represents the web page, and each edge represents the relationship between two web pages. As seen in Fig. 1.1, this index is accessed by the query engine and ranking module. The link index maintains the link structure of the complete graph. The web page content is not considered while building this index.

2. Content-Based Index: It builds an alphabetical index with the help of keywords, which are presented in web pages returned by the inverted index. Query fired by the user may be keyword based. The query processor returns the URLs of the web pages matching with the keywords present in the user query.

3. Other Indexes: These utility indexes are build by collection analysis module after analyzing the requirement of the query engine and ranking module. For example, the Google search engine uses PageRank algorithm for ranking the web pages. Though the ranking module assigns the rank and the collection analysis module maintains it in the form of a rank index.

1.1.4 Ranking module

A ranking module consists of ranking algorithm which starts with fetching of relevant and appropriate information from the collection of linked documents by finding the most authoritative pages. Scarcity and abundance are the two major problems while returning search results

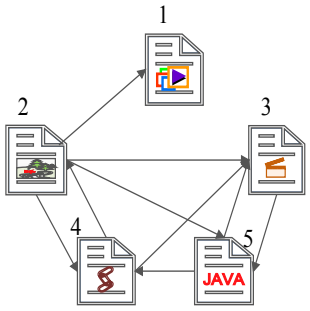


Figure 1.2: Web structure of five nodes

to the user. Scarcity problem arises when the query is particular in which case it is difficult to find the web pages of a specific topic only. Abundance problem occurs when the question is broad, and it is difficult to detect which are the relevant web pages to be returned. Pages which are returned for a particular query not only depend upon the in-degree. It is also dependent on the overlapping of documents. Such pages are known as official pages which are used to depict the relationship between hubs and the authoritative pages [55]. Few ranking algorithms are briefly discussed below:-

- Based on the concept of authorities and hubs, an algorithm called HITS has been proposed [55] which is used to find the relationship between authorities and hubs and works as follows:

  1. Firstly, it finds the topmost web pages returned by the search engines such as - Altavista for a particular query. Then around 200 pages form a root set which is then expanded by covering the neighbouring nodes to form a base set T. The size of T depends upon the out-links of those 200 pages.
  2. Each page is assigned two types of weight, one is authority weight, and another is hub weight. All the weights are initialized to 1 in the initial step.
  3. The scores are updated till the first ten pages for each category are retrieved to build a community. This process is continued until the stable set of hubs and authorities is retrieved.

- An algorithm called as- a stochastic approach for link-structure analysis (SALSA) [56] has also been proposed. It works with the authorities and hubs but removes the law of Kleinbergs mutual reinforcement. SALSA is a stochastic approach where hubs and authorities are lightly coupled. Markov chain is used as the base of the methodology used for the random walk by SALSA. SALSA proved to be more efficient than the mutual reinforcement approach. Mutual reinforcement approach has the drawback of Tightly Knit Community (TKC) effect. It prevents the identification of meaningful authorities [56].

- DistanceRank algorithm is also based on the concept of reinforcement approach, but it takes the total number of links between the two web pages as a punishment. As in Fig. 1.2, 5 web pages are connected. The distance between the web page 5 to web page 1 is 2 as there are two links has to be used. Higher the distance, lower the distance rank and vice-versa. But, for calculating the distance rank, it requires high computation, as backlinks are also involved. The target of this algorithm is to find the page having less distance for assigning high rank to it. Power method can be used for the computation of distance rank [57].

Chapter 2

Literature Review

2.1 PageRank algorithm applications

According to the survey performed, the PageRank algorithm is applied in various domains. Few of them are listed below:

2.1.1 Network

Urban areas also use the PageRank methodology by assuming street intersections as nodes and paths as edges. It is not only affected by considering the internal factors as in PageRank, but also by external factors. The proposed algorithm is used for ranking the restaurants, shopping centres etc. by converting the public network into graph theory language. Structural issues by civil engineers can also be handled with this concept. A real example of the city of Spain has experimented [58], [59]. PageRank algorithm methodology is used at server-side for computing the global score of web pages. An algorithm Juxtaposed approximate PageRank (JXP), computes the PageRank score of web pages in a peer-to-peer network by computing locally at the average cost. Lumping technique of markov chain is used for its computation. With the help of JXP, a trust model is generated which avoids the effect of malicious peers by anomaly detection. This concept is called as TrustJXP [60].

2.1.2 Web graphs

PageRank is widely used in ranking the graphs. Web Graphs distributed over the clusters can be processed in parallel by using pregel for computation. It uses the MapReduce algorithm for the conversion of PageRank vectors to sweep the orderings using Hadoop [61]. This clustering has been proposed as an algorithm and is known as ParallelNibble algorithm [62]. Structural sense ranking of a tree data is also done with the help of PageRank approach. It can be implemented in two alternatives. First is the relational structure of the tree of a single type where each node contains the instances of the PageRank. Second with relational structure

	Algorithm	Accuracy
1	RTL-GC	89.24
2	QoC	86.39
3	Mass Estim.	83.22
4	QoL	82.73
5	TrustDistrust	82.37
6	TrustRank	78.92
7	AntiTrustRank	70.58
8	BadRank	66.62
9	Cautious Surf	63.55
10	PageRank	58.70
11	ParentPenalty	51.98

Figure 2.1: Comparison of RTL with other graph methods [2]

of the tree of multiple type by biasing the PageRank implemented with Monte Carlo algorithm. This ranking method is known as Multi-Structure Semantic PageRank (MSSPR) [63]. The computational cost issue of Personal PageRank Vector (PPV) is removed by the algorithm FastPPV which handles the accuracy and efficiency constraints online. This has been achieved by managing random walks layer by layer which simultaneously increases the accuracy. It is based upon inverse P-distance concept already existing in literature [64].

PolarityRank is another PageRank based algorithm which is used to rank nodes in the graph with two measures one positive and one negative. It can also be used for web spam detection once the proper ranking of nodes is achieved [65]. Social Networks are the most significant source of marketing, connectivity and business. But the spammers try to harm such networks for personal benefits. A ranking approach is developed for ranking the users after analyzing the positive or negative opinion by the users. A tendency to check the negative side is significant because it is necessary to know the malicious sellers and marketers. Whereas in the search engine's ranking, only the preference of the node is measured. This is how the trustworthiness of the user is measured, and it is known as PolarityTrust algorithm. It uses the basic concept of PolarityRank. Non-Negative Propagation and ActionReaction Propagation both are measured for computing the PolarityTrust [66].

2.1.3 Medical

Spam Detection is a major issue in the case of medical websites as well. It has been surveyed that 90% of medical websites are fraud and can lead to major health issues due to the selling of harmful products and drugs. An adaptive learning algorithm, known as Recursive Trust Labelling (RTL), is proposed for the detection of fake medical websites. This algorithm uses the same methodology of PageRank that the good pages point to the good pages only. Performance of RTL is compared with 10 Graph-Based Methods, as shown in Fig. 2.1 and RTL obtained highest accuracy [2]. Another work done is ranking the user influence in health care media by two approaches: weighted in-degree and UserRank. Both approaches are based on the methodology of PageRank [67]. Bio-medical literature has been ranked by the PageR-

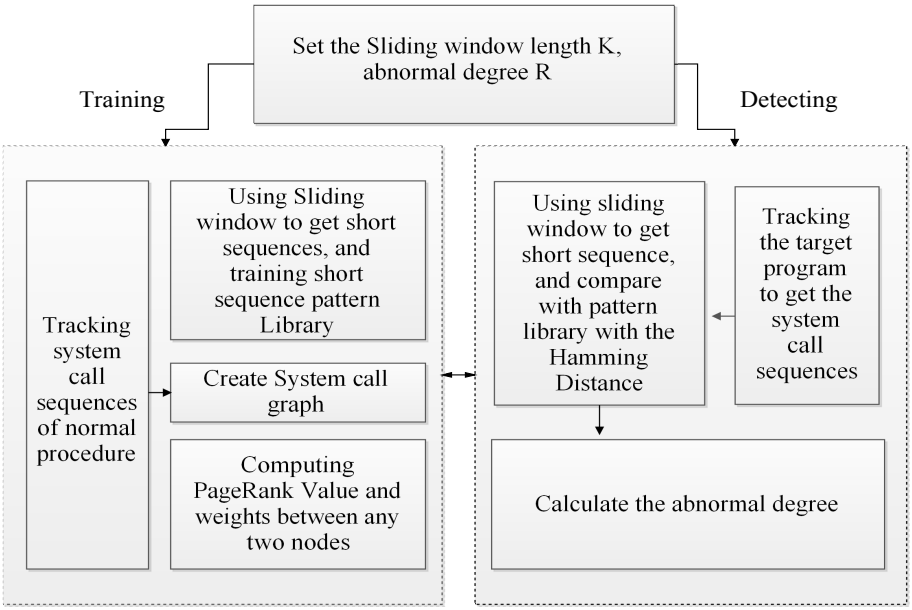


Figure 2.2: Model of PageRank algorithm based anomaly detection [3]

ank algorithm, as the citation count should involve the importance of the reference as well as [68]. Another medical issue solved by the PageRank methodology depends upon the molecular interactions within the cell. It is very crucial to predict the protein level, done by Modified PageRank methodology, known as ProteinRank. In this algorithm, proteins are the nodes of the graph and annotations are the links between the nodes [69].

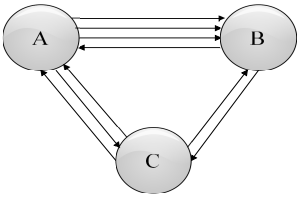
2.1.4 Security

PageRank is also helpful in anomaly detection using random walks on proximity graphs. Firstly a graph is created using dataset, which requires high computational cost and secondly a random walk over the graph using the methodology of PageRank. This two-step framework is called Anomaly Detection using Proximity Graph and PageRank (ADPP). The idea behind this method is that the closely packed nodes are assigned with higher weights, which proportionally gets the higher probability. Such nodes get higher chances of together visit during the random walk, and the nodes with lower weights are declared as anomalies [70]. Sequence Time Delay Embedding (STIDE) is used for anomaly detection in system calls. This algorithm lack behind because of the abnormal behavior of hamming distance. New framework removed this problem by weighted Hamming System based on the PageRank algorithm, as shown in Fig. 2.2. It works in two modules. The first module consists of three steps.

1. Creation of sliding window of fixed length, which is used for creating a sequence pattern library.

2. A system call graph is constructed using pattern library.

3. Finally assigns weights to the nodes with the help of PageRank algorithm.

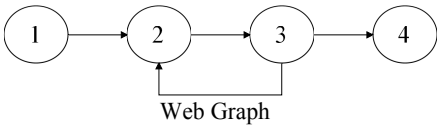


Sample of CEN Network

	A	B	C
A	0	3/4	1/4
B	1/2	0	1/2
C	2/3	1/3	0

Weights of Sample CEN

Figure 2.3: Modified PageRank for CEN [4]



Web Graph

	1	2	3	4
1	0	0	0	0
2	1	0	1/2	0
3	0	1	0	0
4	0	0	1/2	0

PageRank Matrix

	1	2	3	4
1	0	1/2	0	0
2	0	0	1	0
3	0	1/2	0	1
4	0	0	0	0

Inverse PageRank Matrix

Figure 2.4: Matrix representation [5]

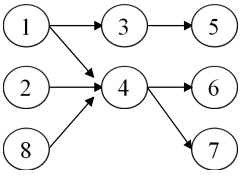
In the second module, the hamming distance is computed between system calls and the anomalies are detected [3]. We can judge the PageRank methodology provides security as well.

2.1.5 Multimedia

PageRank is also used for searching the multimedia objects in web collection. The algorithm proposed for this purpose is named as Multimedia PageRank algorithm. In this, the same methodology of PageRank is used, but instead of nodes, there are objects in graph construction like images or videos [71]. PageRank is also used for finding an expert in the online communities by social network analysis. It is used for computing the score of each user in the community expertise network(CEN) by asking questions and answers. Modified PageRank for this score calculation states that there can be multiple links between the nodes at a time [4] as illustrated in Fig. 2.3.

2.1.6 Detecting web spam

PageRank methodology is applied for detecting webspam as well. An algorithm is introduced with the help of inverse PageRank by following the out-links rather than in-links. This algorithm is known as TrustRank; it is used for computing the trust among the seed sets. TrustRank algorithm does not completely differentiate between the spam and non-spam pages.

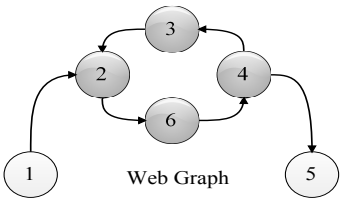


Web Graph

Node	t1	t2	t1+t2	t
1	0.408	0.000	0.408	0.140
2	0.000	0.214	0.214	0.140
3	0.173	0.000	0.173	0.060
4	0.173	0.364	0.538	0.299
5	0.147	0.000	0.147	0.051
6	0.049	0.103	0.152	0.085
7	0.049	0.103	0.152	0.085
8	0.000	0.214	0.214	0.140

Simple summation of trust score obtained in different seed sets

Figure 2.5: Simple summation of different topic scores [6]



Web Graph

	1	2	3	4	5	6
1	0	0	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1/2	0	0
4	0	0	0	0	0	1
5	0	0	0	1/2	0	0
6	0	1	0	0	0	0

Link Farm Matrix

	1	2	3	4	5	6
1	0	0	0	0	0	0
2	1	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0

Non Link Farm Matrix

Figure 2.6: Link farm matrix and non link farm matrix [7]

Algorithm	Parameter settings	Seed
PageRank	Damping factor: 0.85	No need
TrustRank	Damping factor: 0.85	Need
DiffusionRank	Damping factor: 0.85, Thermal conductively: 1	Need
AIR	Conductance:0.5	Need
DRank	Damping factor: 0.85, k-nearest neighbor radius:2	Need

Figure 2.7: The parameter settings of different algorithms [8]

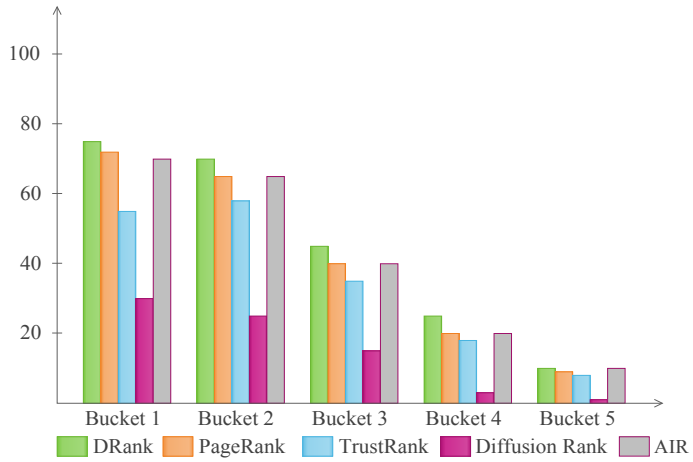


Figure 2.8: The performance of spam suppressing of respective algorithms [8]

Firstly the small seed set is selected by the algorithm which is handed over to the team of experts in the form of matrices as discussed in Fig. 2.4. They particularly determine the status of all the pages existing in the set by oracle function. Then depending upon the output of this process, other pages are evaluated by applying the same decision tree generated, which is called as relative isolation. The seed selection is made with the help of inverse PageRank [5]. The issue addressed in topical TrustRank is very obvious that the web consists of a large number of topics. The seed set is partitioned into various topics. For each web page, a trust score is evaluated by the TrustRank algorithm. There can be multiple scores for a single page which are combined by simple summation, as shown in Fig. 2.5 or by quality bias to generate the topical TrustRank score [6].

Removing the artificial link farm's effect from the PageRank computation can also help in web spam detection. In this approach, firstly the PageRank score is computed with its aggregation with a link farm. Then this score is distributed among the other nodes in the graph [7]. For measuring the link farm score, two matrices are generated, the link-farm matrix and non-link farm matrix, as discussed in Fig. 2.6. DRank algorithm is proposed with the measure of the diversity of in-links so that the link farm weight is easily computed, which helps in the detection of spam pages. The parameter settings, as shown in Fig. 2.7, DRank is compared with other algorithms such as - PageRank, TrustRank, DiffusionRank, AIR with similar evaluation parameters, as shown in Fig. 2.8. DRank is resulted to be the best one for spam detection [8].

The PageRank is used in various applications, as summarized in table 2.1.

Table 2.1: Applications of PageRank Algorithm

Author	Method	Dataset	Evaluation Parameters	D	Advantages	Future_scope
Gyongyi *et al.*, 2004 [5]	TrustRank algorithm	AltaVista crawled and indexed pages in August 2003	Convergence rate, Oracle Function	0.85	Can be used with PageRank to detect the spam sites	Selection of seed set and oracle function.
Wu *et al.*, 2006 [6]	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.
Rungsawang *et al.*, 2007 [7]	Link Farm Effect Detection	YahooAPI	PageRank score of spam sites	0.85	Demotion of spam pages	more link farm configurations with larger web graph can be explored.
Yang *et al.*, 2015 [8]	DRank algorithm	WEBSPAM-UK2006	MAE, RMSE for diversity and Precision , running time for algorithm comparisons	0.85	Better detection of spam pages	Anti-Spam ranking techniques development .
Parreira *et al.*, 2008 [60]	JXP, TrustJXP	Amazon.com	JXP: Linear score measure= JXP score - PR score, TrustJXP: Histogram and rank divergence	0.85	Low computational cost	Handling churn phenomenon.
Perozzi *et al.*, 2014 [62]	ParallelNibble Algorithm	Various Web Graphs like web-Stanford, web-google	Louvain Fast Unfolding	0.85	Parallel Clustering saves time	Large Graph Processing for the same can also be explored.
Interdonato and Tagarelli, 2015 [63]	Multi Structure Semantic PageRank (MSSPR)	INEX 2009	normalized discounted cumulative gain(nDCG), Binary preference function(Bpref), Fagin intersection metric (F)	0.85	Structural sense ranking improved by Monte Carlo with PageRank approach	Structural sense ranking under a rank aggregation framework or a cross-view random walk paradigm can be explored.
Zhu *et al.*, 2015 [64]	Fast Personalized PageRank Vector(FastPPV)	DBLP and LiveJournal	FastPPV, HubRankP, MC1, MC2	-	FastPPV is scalable than HubRankP, MC1, MC2	automatic configuration, tackling dynamic graphs, generalizing to other graph algorithms are the open research issues.
Cruz *et al.*, 2012 [65]	PolarityRank	Product Reviews from Epinions website	Kendall distance	-	-	Explore balanced graphs.
Cruz *et al.*, 2012 [65]	Weighted Instance Typicality Search Algorithm (WITS)	Australian Credit Approval(ACA), Breast Cancer Wisconsin(BCW), Echocardiogram(ECH), Hepatitis(HEP), Horse Colic(HOR), Ionosphere(ION), Liver Disorders(LIV),Pima Indians Diabetes (PIM), Molecular Biology(PRO),Voting Records(VOT)	Euclidean Distance	-	can be Implemented with both PageRank and PolarityRank	can be used for web spam detection.
Huang *et al.*	Robust Personalized PageRank (RPR)	IMDB, MovieLens	Relevance, Robustness	0.85	PageRank(RPR) techniques are efficient and removed biasness from the seed sets.	
Wu *et al.*, 2013	ProteinRank, accelerated Arnoldi algorithms	Wb-cs-stanford, Ubc-cs-2006, Ubc-2006, Stanford-Berkeley, Flickr, Wikipedia-20051105, Wikipedia-20061104, indochina-2004, Wb-edu, uk-2002	Running Time	0.50, 0.70, 0.85, 0.99	Arnoldi algorithms can also be applied to the protein function prediction problem.	
Yao *et al.*, 2012	Anomaly Detection using Proximity graphs and the PageRank algorithm (ADPP)	KDDCUP 99	-	-	PCA, Kernel PCA (KPCA), Kernel recursive least square Online Anomaly Detection(KOAD)	Make the framework online.

2.2 Web spam detection

Spam is injecting into the web rapidly, so it is essential to detect the webspam. Many systems have been developed for this purpose, and it is an open research issue as the spammers use new techniques for web spamming periodically. Spam can be detected in numerous ways, by modifying the existing classifier, by extracting new content or link based features, by exploring the relationship between the URLs and user's query, by language model analysis and many more as discussed below:

a) Two strategies have improved an existing classifier for web spam detection with both content-based and link-based features: *Relabeling and Secondary Classifier* [9]. It is implemented with decision tree C4.5 and support vector machine in which decision tree proved to be better, as shown in Fig. 2.9.

b) Proposed a classifier for web spam detection consisting of both link-based and content-based features. The basic idea behind the development was the Labels of the host is that they

Results		Precision	Recall	F-measure
Decision Tree C4.5	**Spam**	0.897	0.812	0.852
	Non-spam	0.882	0.925	0.903
SVM	**Spam**	0.879	0.812	0.844
	Non-spam	0.863	0.913	0.887

Figure 2.9: Results of improved classifier [9]

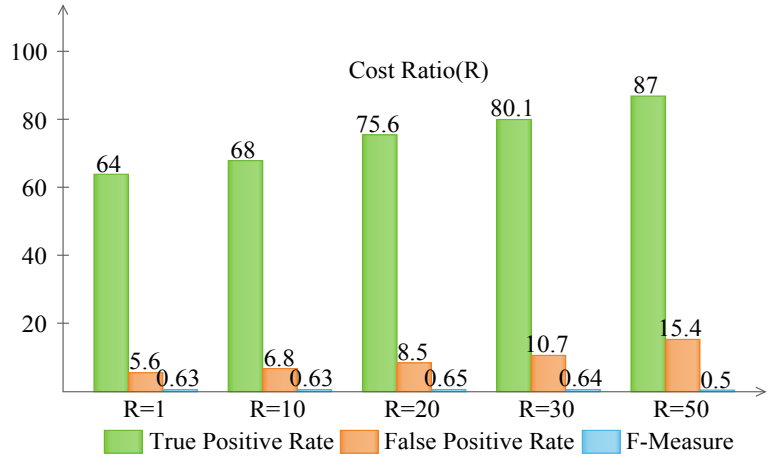


Figure 2.10: Cost-sensitive decision tree [10]

are linked to each other. The labels belong to the same class, either spam or non-spam. For detection of linked hosts, *topology of web graph* was used, which then renames the labels after evaluating the structure. The new classifier was formed with the same method of decision tree but with the new structure. F-measure achieved was highest when R=20. Ten-Fold cross-validation was done for improvement [10]. The improved results for the cost ratio are shown in Fig. 2.10.

c) Web spam detection system has been further improved by *re-extracting the features* of the same dataset after preliminary detection. The smoothing is done with new features, and the results are compared with the previous detection. Five-Fold cross-validation is done six times [11]. The experiment results are shown in Fig. 2.11.

d) Proposed a webspam detection framework by analyzing the *user's searching behavior*.

Features	TP	FP	Precision	Recall	F-measure	AUC
Original Features	0.832	0.0626	0.845	0.832	0.838	0.958
Stack Graph Learning	0.885 **(6.41%)**	0.0645 **(-2.98%)**	0.848 **(0.47%)**	0.885 **(6.41%)**	0.867 **(3.38%)**	0.967 **(0.088%)**
Combined Features	0.900 **(8.12%)**	0.0606 **(3.08%)**	0.859 **(1.63%)**	0.900 **(8.12%)**	0.879 **(4.80%)**	0.971 **(1.26%)**

Figure 2.11: Web spam detection performance with different strategies [11]

Page Type	Percentage
Non-spam pages	6.00%
Web spam pages (Term spamming)	21.67%
Web spam pages (Link spamming)	23.33%
Web spam pages (Other spamming)	10.67%
Pages that cannot be accessed	38.33%

Figure 2.12: Page types of 300 possible spam pages identified [12]

Page Type	Percentage
Non-spam pages	5.33%
Web spam pages (Term spamming)	31.67%
Web spam pages (Link spamming)	25.33%
Web spam pages (Term + Link spamming)	11.33%
Web spam pages (Other spamming)	27.00%
Pages that cannot be accessed	9.33%

Figure 2.13: Page types of 300 possible spam pages identified [13]

It not only detects the spam pages but also categorizes them into different types of spam, as shown in Fig. 2.12. Three features that are extracted from the user behavior are: Search engine oriented visiting rate, source page rate and short-time navigation rate. The algorithm uses the Bayesian learning method along with the features extracted from the behavior of the user, thus leading to a better understanding of the detection of spam [12]. Same work has been extended, and three new features are extracted, namely Query Diversity (QD), Spam Query Number (SQN) and User-oriented TrustRank [13]. Like previous work, different types of spam pages are identified, as shown in Fig. 2.13.

e) *Language Model analysis* has also proved to be a right approach for web spam detection. Kullback-Leibler(KL) divergence is used for the formulation of the language model, which clearly depicts the relationship between two web sites. KL divergence along with the content and link-based features is used for experiments by the algorithm, (cost-sensitive decision tree with bagging) [14]. Results by combining the elements are presented in Fig. 2.14. A similar approach with qualified-link (QL) features has been used to build a robust classifier. In this methodology, four types of features have been combined, as shown in Fig. 2.15 and implemented with the two different datasets [15].

f) Current information on web pages is used to detect the webspam. However the *Historical Information* can also be used for this purpose, which is retrieved by the Internet Archive's Wayback Machine. It is implemented with the new approach along with the current information on web pages. With the help of such historical information, as shown in Fig. 5.4, the performance of the new classifier is improved by 30% concerning base classifier [16].

g) *Graph Regularization* methods are also used in web spam detection. Webspam Identification Through Content and Hyperlinks(WITCH) is an example which uses SVM with Graph Regularization. It is the first technique which uses three different tools while training i.e.,

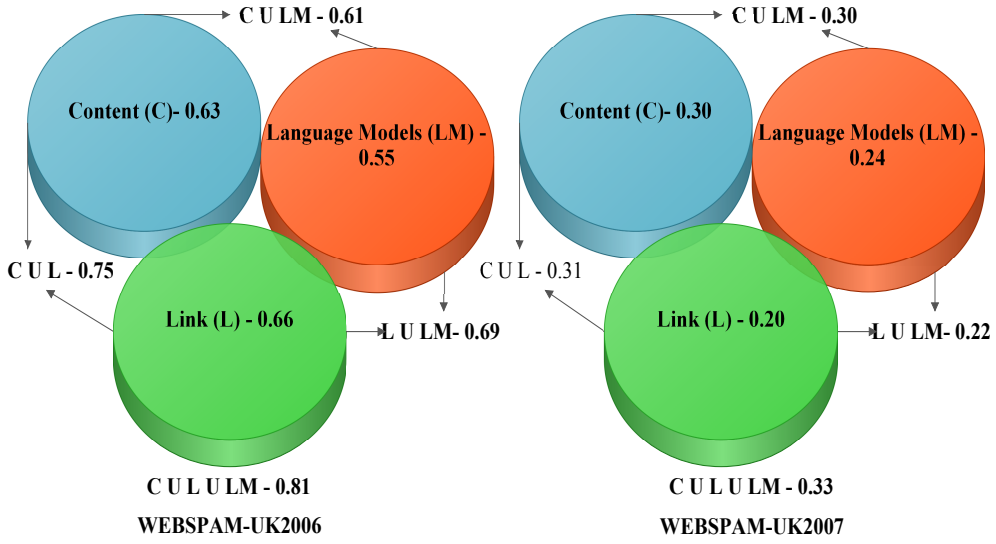


Figure 2.14: F-Measure after experiments by language model analysis approach for web spam detection [14]

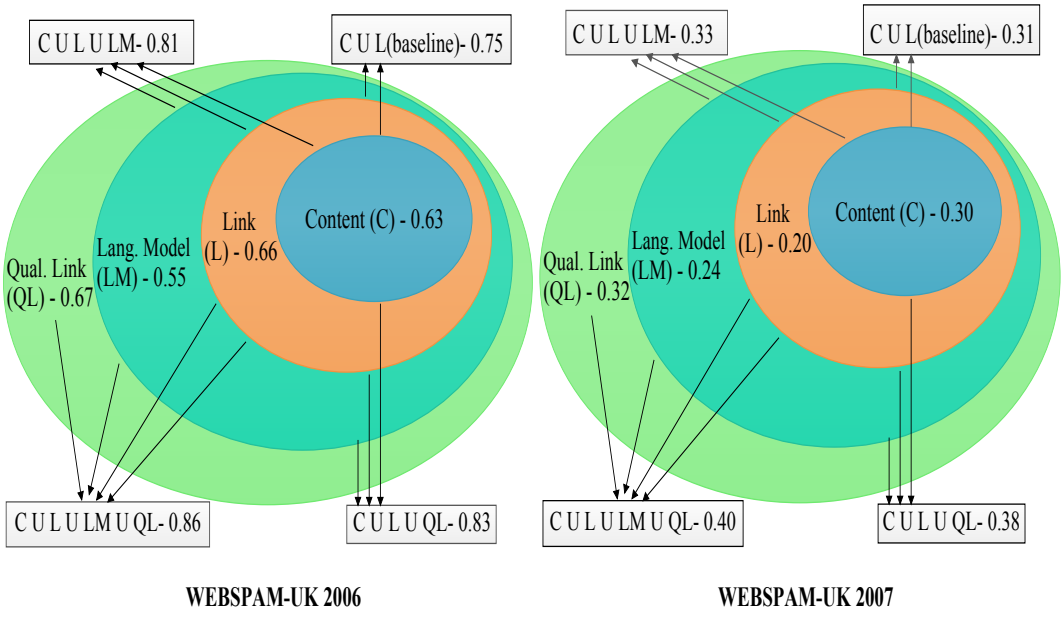


Figure 2.15: F-Measure after experiments by qualified link analysis and language model analysis approach for web spam detection [15]

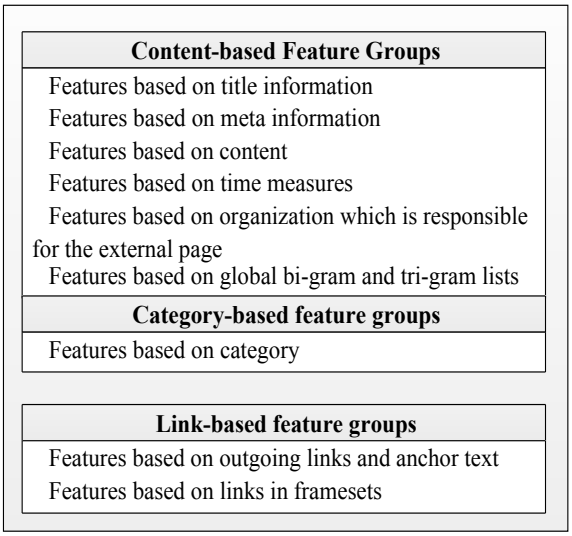


Figure 2.16: The feature groups (Organization Historical features) used in classifying whether site ownership has changed within the past four years [16]

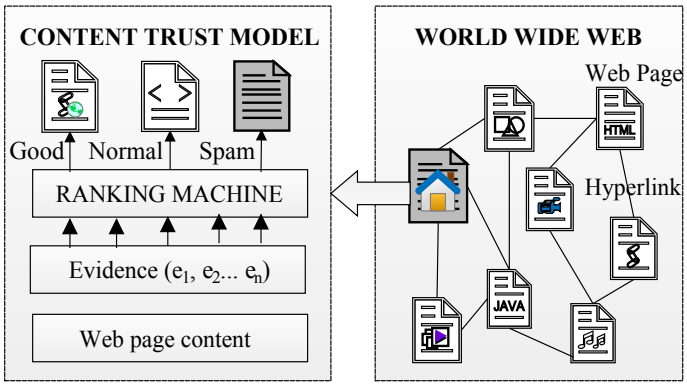


Figure 2.17: Overview of the Content Trust Model [17]

Host features, Regularization along with the hyperlink graph (link-based features) and slack variables. The experiments proved that WITCH outperforms the stacked graphical learning and transductive link spam detection as well, due to the entire graph utilization at the training time [72].

h) Quality of the content can also be a major consideration for web spam detection. It has been addressed in detecting spam information based on the *Content Trust Model*. It uses information quality and test features for spam classification, which is based upon evidence, and it has been deployed with the ranking algorithm, as shown in Fig. 2.17. Here, the spam detection problem is viewed as a ranking problem and can be easily solved with any machine learning technique [73]. Once the evidence is building, as shown in Fig. 2.18. SVM method used is compared with Decision-tree based ranking techniques (R-DT) and Naive Bayesian-based ranker (R-NB) as shown in Fig. 2.19 [17].

i) *Selection of best combination of features and suitable machine learning model* for web spam detection is also a significant challenge. The work to accomplish this task investigates var-

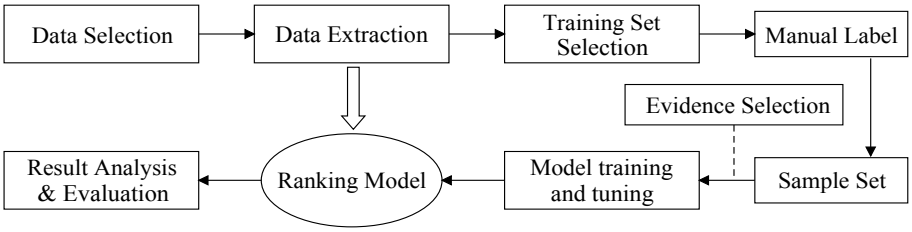


Figure 2.18: Process of ranking model building in content trust model [17]

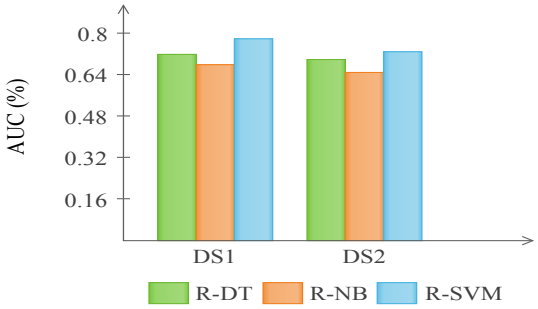


Figure 2.19: Comparison of various ranking algorithm with content trust model algorithm [17]

ious web spam features when combined with different machine learning models using ensembling approach has increased the accuracy [74]. WEB-SPAM-UK2007 and the ECML/PKDD Discovery Challenge data set DC2010 are used for the experiments. Various models such as - Bagged and Boosted Decision trees, Logistic Regression, Naive Bayes and Random Forests models are used [75].

j) There is quite an interesting relation between the keywords present in the *User's Query and URL's of the Spam Sites*. This is being focused, and thus, a spam detection framework has been proposed by analyzing the click-through data at site-level and page-level. The algorithm is known as Label Propagation. The experiments concluded concerning PageRank and TrustRank for traversing over the crawled web link graph. This algorithm outperforms when compared to other algorithms, as shown in Fig. 2.20 [18].

k) *Web Page content features* depict its liability. Spam Analyzer And Detector (SAAD), the

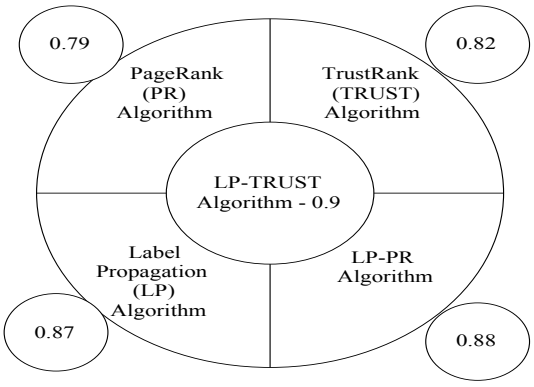


Figure 2.20: AUC value for different spam detection strategies [18]

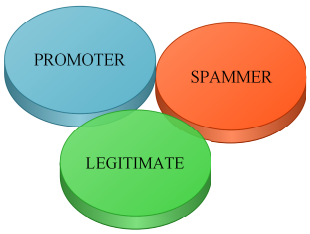


Figure 2.21: Flat classification of you-tube users [19]

content-based classifier has been proposed which uses a decision tree as a classifier followed by bagging and boosting techniques for spam detection. SAAD can be applied at different domains like crawling, due to different functionality of the modules. In experiments, various heuristics have been analyzed for some time and used with different classifiers in which decision tree proved to be the best one [76].

l) As the score computed by PageRank for every webpage depicts only the importance of the webpage, but it is crucial to calculate the distrusted of the webpage to label it as spam. The framework is developed which assigns two ranks to each webpage, a GoodRank and a BadRank, with the help of spam detection algorithm called *GoodBadRank(GBR)* algorithm. When Compared with other anti-spam algorithms on WEBSPAM-UK2007 dataset, GBR outperforms [77].

m) There are multiple algorithms which can be used for spam detection, by integrating a few algorithms (Naive Bayes, Support Vector Machines, and C5.0). The new framework is designed known as *Web Spam Filtering Framework (WSF2)*. The operation mode of this approach is globally implemented by the sequential execution of four main stages:
(i) Retrieve the most appropriate filtering rules
(ii) Reuse of these rules for assessing an initial solution
(iii) Revise the confidence of the generated output and
(iv) Retain the system configuration parameters to maintain the filtering performance over time. The experiments are conducted with two classifiers SVM and decision tree c4.5 [78].

As society is developing, so the spammers. Even they try to harm the social networking sites. Spam detection techniques have also been proposed separately for social networks.

a) Spam promoters are not only injecting into the web but also in *social video networks*. A classifier has been designed to detect and classify the different types of users, as shown in Fig. 2.21. The dataset required for experiments was not available anywhere, so the crawler collected it with three sets of attributes. First was the collection of all the videos along with video responses and responded videos. The second was the user's behavior attributes. The third was the relationship between the videos while uploading and downloading videos with responses. All the experiments after collection of the dataset are carried out with SVM followed by 5-fold Cross-Validation approach [19].

b) A framework for *Spam Detection, Demotion and Prevention* for social websites is de-

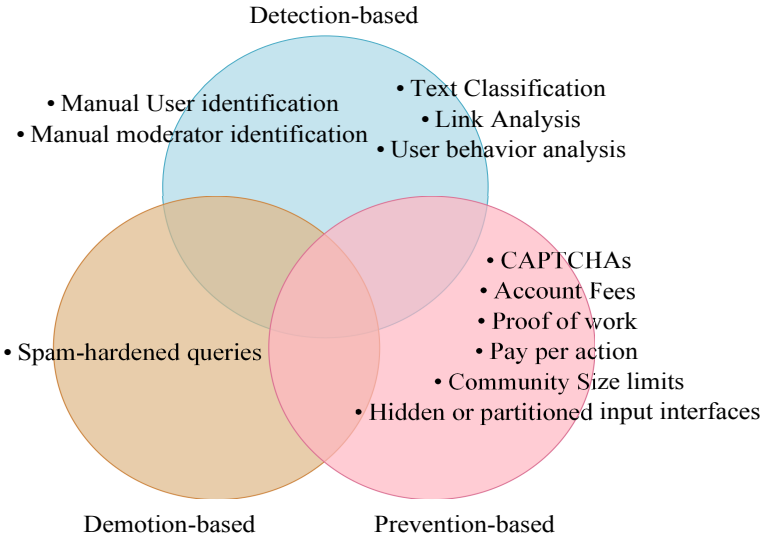


Figure 2.22: Three anti-spam strategies: Detection, Demotion and Prevention [20]

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

Figure 2.23: Web spam detection techniques for social networks

signed. In this, three main anti-spam strategies are focused: Identification-Based (Detection), Rank-Based (Demotion) and Interface or Limit-Based (Prevention) as shown in Fig. 2.22. Detection is done by analyzing the text, source and link. For Demotion, TrustRank algorithm is used, but the ranking is done Geographically, and prevention is done by embedding the CAPTCHAs [20].

c) Several web spam detection techniques are used based on various scenarios in todays *Online Social and Business Websites*. Three techniques, namely Adaptive Fusion for Spam Detection (AFSD), SMS filtering (SMSF), MailRank, have been revised. These techniques are now used as traditional grounds for new techniques. Experiments showed that the content features alone could achieve 0.63 on AUC. Thus the Spam Behavior Regression Model (SBRM) is introduced, and the results are shown in Fig. 2.23 [79].

The webspam detection techniques that exist in literature are summarized in Table 2.2.

Table 2.2: Web Spam Detection Techniques

Author	Algorithm	Description	Dataset	Results	Methods used for Experiments	Advantages	Future_scope
Heymann *et al.*, 2007 [20]	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.
Gan and Suel, 2007 [9]	Improved Classifier for web spam detection	Basic classifier consists of more than twenty features, including both content-based and link-based ones, additional two features, Relabeling and Secondary classifier.	web sites in the Swiss ch top-level domain crawled in 2005 using the PolyBot web crawler.	Check figure 23	decision tree C4.5, Support Vector Machine	Large farms are detected easily by the secondary classifier with Support of Decision tree	
Castillo *et al.*,2007 [10]	Web Spam Detection System	Classifier with Content and link based features using the topology of web graph	WEBSPAM UK-2006	classifier detects 88.4% of the spam hosts with 6.3% false positives(check figure 24)	Decision Tree C4.5	Web Spam detection with low computational cost.	Graph-Regularization methods to be explored.
Geng *et al.*,2008 [11]	Web Spam Detection System	Classifier with Content and Link based features with Two-Stage Feature extraction.	WEBSPAM UK-2006	check figure 25	Decision Tree C4.5	Spam Detection efficiently with new Strategy	
Liu *et al.*,2008 [12]	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" websites	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.
Martinez *et al.*,2009 [14]	Language Model Analysis approach for Web Spam Detection	Different sources of information in a web page is collected along with the relationship between two linked documents are analysed each forming a different Language model	WEBSPAM-UK2006 and WEBSPAM-UK2007	check figure 28	Kullback-Leibler divergence for Formulation of Language Model, Meta-cost algorithm (cost-sensitive decision tree with bagging)	Web Spam efficiently detected with the application of Language Model Approach	The relationship between web pages in spite of web sites.
Benevenuto *et al.*,2009 [19]	Detection of video spammers and promoters	User Test collection by crawling YouTube , Analysing 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.
Dai *et al.*,2009 [16]	Historical information for Web Spam detection	Historical features of web site along with the current information of website is used to detect web spam.	2004 to 2007 Historical information from Wayback Machine, WEBSPAM-UK2007 for Classifier	30% better performance than Base Classifier of Current information	SVM classifier	-	Link-Based features and Different classification Methods can be deployed.
Abernethy *et al.*,2010 [72]	Webspam Identification Through Content and Hyperlinks(WITCH)	Used host features, Graph Regularizartion along with hyperlink Graph, Slack variables all at the training time	WEBSPAM-UK2006	0.963 % AUC	SVM with Graph Regularization	Highlighted an improtant issue which was ignored before that "Web Spam Detection Technique is more effective when tested with small training set as the number of hosts on the web are not as expensive as labelling".	WITCH can be applied to topical classification as well.
Wang *et al*,2010 [17]	Detecting spam information based on content trust model	uses quality and test features after generating evidence for web spam detection	as used in castillo *et al*, 2006, divided into two groups DS1 and DS2	F measure: DS1 - 0.835, DS2 - 0.765	SVM spam detection algorithm	Accurately detected Spam	Natural Language techniques can be used for recognising text that is artificially generated.
Araujo and Romo,2010 [15]	Quantified Link Analysis and Language Model Analysis approach for Web Spam Detection	Content, Link, Quantified Link features are combined	WEBSPAM-UK2006 and WEBSPAM-UK2007	check figure 30	Kullback-Leibler divergence for Formulation of Language Model, Meta-cost algorithm (cost-sensitive decision tree with bagging)	Better performance than the use of Language Model Approach only	The relationship between web pages inspite of web sites and Reducing the number of pages retrieved for each link and the number of links analyzed per page for better performance.
Tran *et al.*, 2011 [27]	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 [13]	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.
Wei *et al.*, 2012 [18]	Label Propagation Algorithm (LP)	used click-through bipartite graphs to detect Web spam	set of 2,100 spam sites and 1,153 non-spam sites	1,490 NONSPAM, 870 SPAM and remaining cant be accessed	PageRank and TrustRank	Efficient and effective in detecting Web spam pages/sites	solve the sparsity problem and more click-through features can be added to the framework.

Link spam

As discussed in the previous section, the search engine's working is only possible with a ranking algorithm such as - PageRank algorithm. This algorithm is highly dependent upon the hyperlink structure of the web. It can be easily concluded that the SERPs have a strong relationship with the structure of the web. So to get high ranking in SERPs, spammers try to manipulate the structure of web by using various techniques. Link spam is one of the techniques in which link farms are created for this purpose. Detection of link spam is not so easy. Work is done for link spam detection concerning different domains such as - Seed Selection, Social Networks and many more as discussed below:

Seed Selection :

Link farms detected before ranking the web pages can be beneficial. Same work has been done by detecting and dropping the affected links. The basic methodology used is the identification of domains of three incoming and three outgoing links. If they somehow match, then the page is considered in the link farm and exceeds the approach as *parentpenalty*. The process is continued till all the pages are added to the farm by the successive iterations. Punishing of these links is being done by deleting them before ranking [80]. The same algorithm of Seed Selection and *parentpenalty* has been extended by using the concept of Multi-level link structure. The outgoing links of other web pages within the same domain are also extracted. Link parsing sequence of MLSA is illustrated in Fig. 2.24. Four webpages A1,A2,A3,A4 are connected. In level 1, A1 is the candidate page to be analyzed, followed by its outgoing links that are A2 and A3. In Level 2, the outgoing link of A2, i.e., A4 is analyzed. The improvement in spam detection is shown in Fig. 2.25 [21]. Another technique is developed with some seed set already defined, which is followed by the random walk to detect the link spam depending upon the node probability. The nodes belonging to the same community are visited first by obtaining a higher probability than the nodes belonging to outside the community [81].

As discussed, the seed set propagation can help in link spam detection. But if the seed set is small as previously used, it can suffer from many issues. Selecting the large seed set manually is also not possible because of high cost, and lack of a single team of experts with

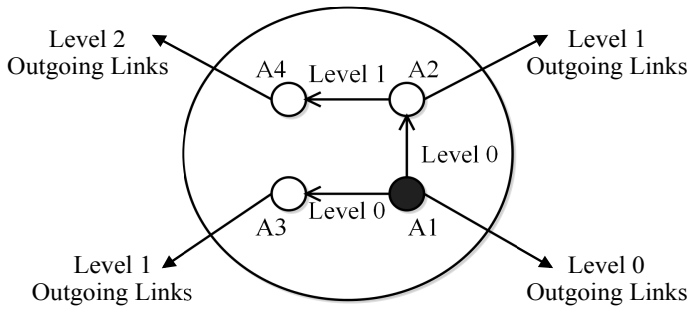


Figure 2.24: Link parsing sequence of MLSA [21]

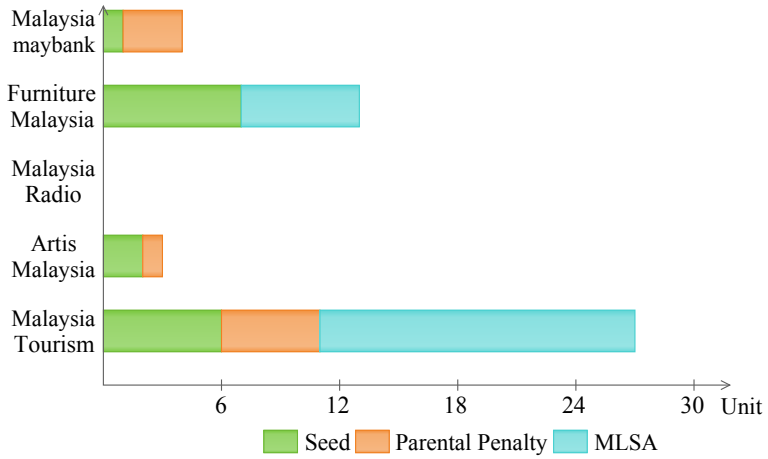


Figure 2.25: Number of spam pages identified using seed selection, parental penalty and MLSA [21]

vast knowledge about the web. ASE (Automatic Seed set Expansion) algorithm is proposed for constructing the broader seed set. If the trustworthy link is recommending another link, then it is assumed to be trustworthy. If a large number of trustworthy links are recommending the link, then it is called as joint recommendation link structure. This is how the seed set is expanded with probabilities and support degree [82].

Graph :

The structure of web highly influence the manipulated links. This has been given attention and web structure is revised followed by detection of link spam. The method used is LOD(Link Oriented Discrimination) and it is compared with PageRank algorithm with an example of six web pages [83].

Techniques for link spam detection are proposed by computing the probability of the web page against the graph of web and estimating rank propagation from web page directing it. To improve the accuracy of classification, graph clustering algorithms and predicted label propagation are used by the authors. The methodology used by them resulted with 80% spam pages, Thus proving that the state of the art in features shows tremendous results [84].

Trustworthiness and spamicity are the two measures of web page which can help in detection of hijacked links directing to spam sites. Normal page have high trustworthiness and low spamicity where as spam sites have low trustworthiness and high spamicity. These two measures are evaluated using the modified PageRank algorithm. After construction of spam seed set and trust seed set, two PageRank scores one positive and another negative are calculated [85]. Detecting the root instead of leaf is more robust as detecting the spam host instead of spam page. This work has been done by analyzing the link structure of spam pages. The host that are generating the out-links to the spam sites are known as spam link generators. Host is spam link generator or not, it is a classification problem and is evaluated by the various features such as - PageRank score, white score, spam score for depicting its trustworthiness [86].

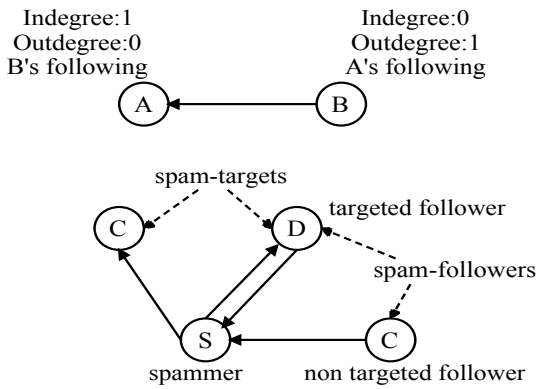


Figure 2.26: Terminology for the spammer's social neighborhood [22]

	Has Bio	Has URL	Profile Pic	Changed profile theme	Has Location	Has Lists
Top link farmers	87%	79%	96%	84%	84%	23%
Random sample	25%	14%	50%	40%	36%	4%

Figure 2.27: Characteristics from profile and activity of the top 100,000 link features [22]

The concept of using the neighboring pages for the detection of link spam has been laid in a framework and is named as Link Authorization Model. The relationship between the good and bad pages is disclosed by analyzing the link and content features. Based upon the relation, authorization of page is predicted whether it is linking to a good page or not. If authorization detected is false, then the threshold penalty is given to the outer page [87].

Social Networks :

Connectivity among people is mostly through twitter nowadays. Spammers try to take advantage by creating link farms. The reason why other people (non-spammers) follow spammers, has been addressed here and the terminology of spammer's social neighborhood is shown in Fig. 2.26. It has been found that the small set of twitters create large number of linked farms. Most of such users are the frequent users and follow other random users. Top linked farmers are compared with the random users for extracting those features which are comparable as shown in Fig. 2.27. Normal users do not provide bio information and links to external web pages whereas the spammers do [22].

Social network graph should not be able to provide the facility of link sharing between two individuals which tear the link privacy policy. To cover this issue, new scheme introduced is neighborhood randomization which treats the social network graph as directed and the local neighborhood can be the destination of any link. Controlling the neighborhood links can control the formulation of link farms than the other link perturbation techniques [88].

Miscellaneous :

Link spam is the technique for boosting the rank of the web page, called as the target page

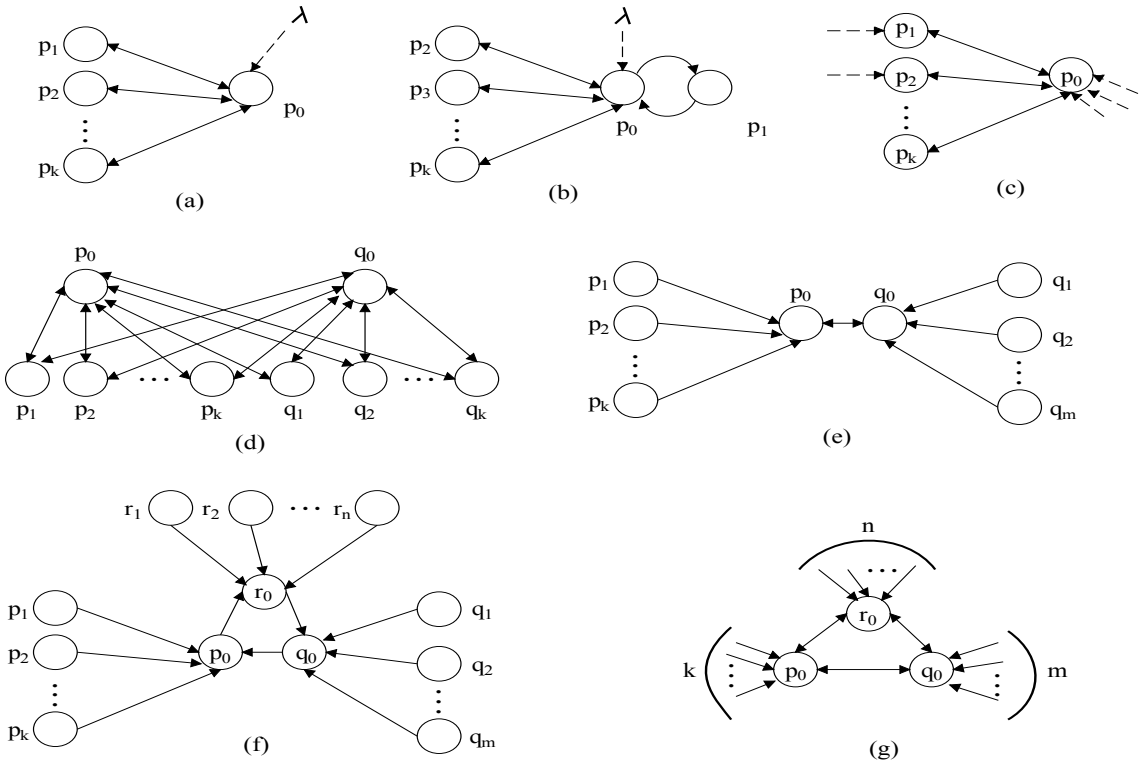


Figure 2.28: Link spam farm formulation [23]

and the pages that are forming the structure are known as link farm or link spam farm. There can be variances in the formulation of target page as shown in Fig. 2.28 and can be called as *Alliances* [23] as it can be single or multiple. In this figure, (a) Represents an optimal structure for a single spam farm with one target page. (b) Another optimal structure for a single spam farm with one target page. (c) Making boosting pages reachable through hijacked links. (d) Two spam farms with all boosting pages pointing to both targets. (e) Two spam farms with interlinked target pages. (f) Three spam farms with target pages forming a ring. (g) Three spam farms with target pages forming a complete core.

Calculating spam mass can also help in link farm detection. If the PageRank computed is high and is boosted by non-reputable pages, then its spam mass will also be high and if boosted by reputable pages, then its spam mass will be low. First approach is labelling, which only counts the in-links that can be easily manipulated. Second approach not only counts the in-links but it also checks the PageRank value of each contributor, it may also fail because looking only at immediate links does not satisfy the property of spam mass estimation. Third approach is following all the PageRank contributors of the in-links. Spam mass can be summarized as the contribution of direct or indirect neighbors in the PageRank of the spam node. An algorithm was designed to detect spam pages using spam mass [89].

Link spam can also be detected by treating it as a classification problem of machine learning based on random walk. This work is done on strongly connected graphs by defining some operators for classical regularization theory creation. As the algorithm Transductive Link Spam

detection can only work on strongly connected graphs, so the loosely connected components are removed during experiments from webspam-uk2006. The remaining two labels are spam or normal [90]. Minimum cut technique can also be used for link spam detection if the graphs are strongly connected [91].

Link farm detection as previously discussed, is competed with the new concept of page farms which is also used for link spam detection. A page is spam or not, idea behind this is how the page gathers its PAGERANK contribution. Page farm also gathers the same information and can be considered as the superset of link farm, so it can also help in formation of link spam farm. Page farms are gathered using greedy algorithm after evaluating its perfectness and statistics and is used not only for link spam detection but also for extracting the information that how a link spam farm is used for link spamming technique [92].

Another way to design web graph is based on sources and a Source-centric Link Analysis (SLA) model is designed. The link structure of web documents is analyzed within a parameterized framework, by developing this model that examines the different parameters contribution to the factor of time complexity and robustness. The conceptual techniques used by the spammers are also discussed such as - Hijacking, Honeypot, Collussion. These are the factors affecting the proposed source-centric link analysis (SLA) model, Source definition, Source-centric citation, Source size, Influence throttling, Application-specific parameters [93].

Spam if not detected at the time of ranking, can also be removed at the time of displaying search results. For this purpose, firstly the important features that are used for ranking of the web page are studied and then the feature selection is done. It contains both the content and link based features except two parameters, location and time. It is assumed that the pages are modified at the same time and same place. Then the test pages are generated. With these features, a classifier is trained. When applied to the search results, spam pages can be easily detected and thus removed [94].

Weight properties were introduced for link spam detection. Weight can be defined as the count of outgoing links between the two different hosts. Weight distribution function is shown in Fig. 2.29. Weight is computed by equation 2, where r1=030000, r2= 507053, r3= 000300, r4= 000050, r5= 020202, r6= 200030 and w is the summation of each row, w1= 3, w2= 20 , w3= 3, w4= 5, w5= 6, w6= 5 [24].

$$O_n = r_n \times \frac{1}{w_n} \tag{2.1}$$

Summarization of link spam detection techniques is presented in Table 2.3.

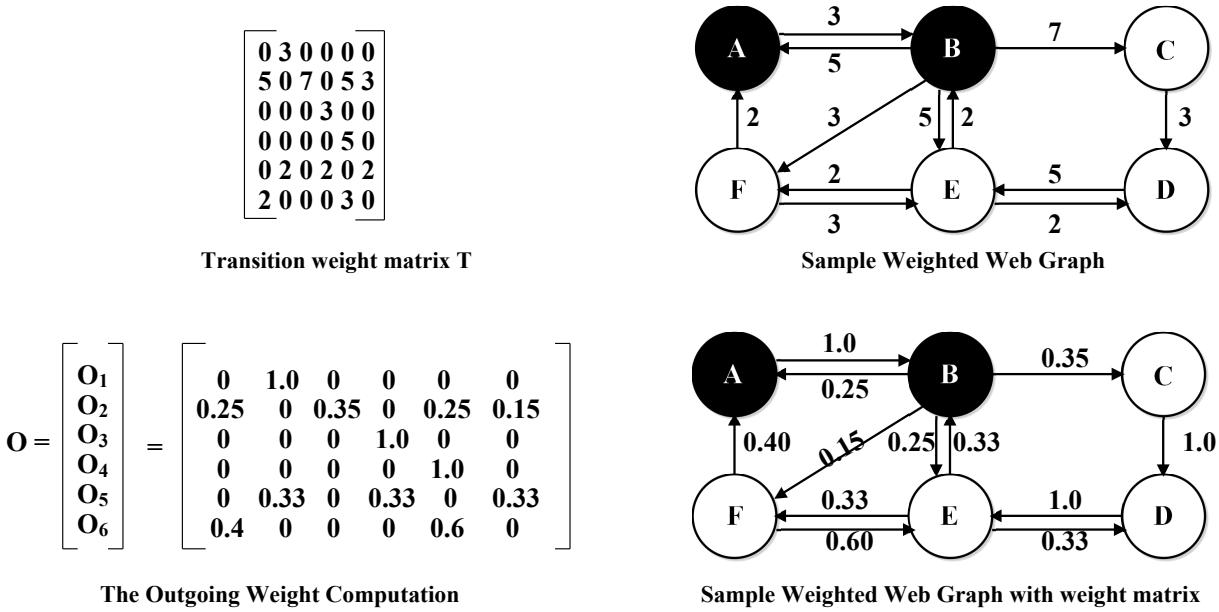


Figure 2.29: Weight property computation in web graph of six nodes [24]

Table 2.3: Link Spam Detection

Author	Algorithm	Dataset	Results	Findings/Advantage	Limitation
Wu and Davison ,2005 [80]	Link Spam Detection	yahoo search engine(2.1M Web pages, 412 queries) and search.ch(20M Web Pages)	Precision 72.6%	Can be used as ranking algorithm	Duplicate pages cannot be detected.
Gyongyi *et al.*, 2006 [89]	Link Spam Detection using Spam Mass	Yahoo search engine (73.3 million individual web hosts)	Heavy Link Farms detected	Robustness	Content information is untouched by this algorithm.
Tung *et al.*, 2006 [21]	Multi-level Link Structure Analysis (MLSA) for Link Spam Detection	Yahoo search engine (Five search terms are used, 500 URLs collected)	22 additional spam pages are detected using MLSA	improvement in link spam detection	Web page content weight is not incorporated in the algorithm.
Zhou *et al.*, 2007 [90]	Transductive Link Spam Detection	webspam-uk2006-1.2	-	Better than AntiTrust	Cannot work with weekly connected Graphs.
Wu and Chellapilla, 2007 [81]	Link Spam Detection by extracting link farms	Web Graph crawed in july 2006 from Live Search	95.12% precision	can be implemented with small seed set	Page-levels web graphs are not suitable for this approach.
Becchetti *et al.*, 2008 [84]	Two Link Farm Detection Algorithms	UK-2002 and UK-2006	87% and 63% accuracy	The use of regularization methods that exploit the topology of the graph gave tremendous results	Fault in Host-based approach
ZHOU and PEI, 2009 [92]	Link Spam detection using page farms	WEBSPAM-UK 2006	90% accuracy	Efficiently detection of Page Farms	Ignored Dangling Pages
Caverlee *et al.*, 2009 [93]	Source- centric Link Analysis (SLA) model	WB2001, UK2002, IT2004, UK2006	45% link farm detected	Proved better than PageRank in detection of link farm	Throttling values needs by be updated dynamically.
Chung *et al.*, 2010 [86]	Spam Link Generator Detection	Japanese Web archive (2004-2006)	Precision: 73% F-measure: 0.68	New spam techniques can be easily detected with time	Content features are not considered.
Zhang *et al.*, 2013 [82]	Automatic Seed set Expansion algorithm (ASE)	WEBSPAM-UK2007	Recall: .ac.uk domain from 59.2% to 55.9% , .gov.uk domain from 68.1% to 58.1%	used link structure for expanding the seed set	content based features are not included in the algorithm .
Goh *et al.*, 2014 [24]	Spam detetction technique with Weight properties	WEBSPAM-UK2007, WEBSPAM-UK2006	WEBSPAM-UK2007: Spam 344, Non-Spam 5709, WEBSPAM-UK2006: Spam 157, Non-Spam 3271	Better detection of spam	Only outgoing links are considered.
Eugene *et al.*, 2015 [87]	Link Authorization Model	WEBSPAM-UK2007	Spamicity: 0.77 detection, 0.60 demotion	Time and cost effective	Must adapt optimization Technique.

Table 2.4: Content Spam Detection

Paper number	Author	Algorithm	Features	Dataset	Results	Methods used for Experiments	Improvement
1.	Ntoulas *et al.*,2006 [25]	Web Spam Detection	1. Number of words in the page 2. Number of words in the page title 3. Average length of words 4. Amount of anchor text 5. Fraction of visible content 6. Compressibility 7. Fraction of page drawn from globally popular words 8. Fraction of globally popular words 9. Independent n-gram likelihoods 10. Conditional n-gram likelihoods	Total pages: 2364 spam + 14,804 non-spam, collected by the MSN Search crawler	1, 940 spam + 14440 non-spam + 788 incorrectly	C4.5 decision tree	Bagging and Boosting techniques
2.	Pera and Ng, 2008 [95]	Content spam detection	1. Title-Body Similarity 2. Similarity Threshold Value 3. Fraction of the Hidden Content 4. The Phrase-Similarity Value 5. The Phrase-Similarity Threshold Value 6. The Enhanced Phrase-Similarity Approach	WEBSPAM-UK2006	94.4% accuracy	The En-SimTB Threshold Value	
3.	Bhattarai *et al.*, 2009 [96]	Blog comment spam detection	1. Post-Comment Similarity 2. Word-duplication 3. Number of Anchor texts 4. Noun Concentration 5. Stopwords Ratio 6. Number of Sentences 7.	corpus created by Mishne and Carmel consisting of 50 random blog posts with 1024 comments posted to them	72% Spam Detected	Spam Similarity and Naïve Bayes Classifier	
4.	Ortega *et al.*, 2010 [97]	Web Spam Detection	1. Compressibility 2. Fraction of globally popular words 3. Average length of words	WEBSPAM-UK2006	Proved better than TrustRank Analysis	Content-based seed characterization	
5.	Wang *et al.*, 2010 [17]	Content Trust Model	1. Number of words in the page 2. Number of words in the page title 3. Average length of words 4. Amount of anchor text 5. Fraction of visible content 6. Compressibility 7. Fraction of page drawn from globally popular words 8. Fraction of globally popular words 9. Various features of the host component of a URL 10. IP addresses referred to by an excessive number of symbolic host names 11. Outliers in the distribution of in-degrees and out-degrees of the graph induced by Web pages and the hyperlinks between them 12. The rate of evolution of Web pages on a given site 13. Excessive replication of content	as used in castillo *et al*, 2006, divided into two groups DS1 (English Web pages) and DS2 (Chinese Web Pages)	F measure: DS1 - 0.835, DS2 - 0.765	SVM spam detection algorithm	Boosting technique
6.	Prieto *et al.*, 2012 [98]	Spam Analyzer And Detector (SAAD): Content Based classifier for Web Spam Detection	1. Number of words per page 2. Number of words in the title 3. Word length 4. Anchor words 5. Ratio of the visible content 6. Compression ratio 7. Number of common words 8. Independent n-gram probability 9. Dependent n-gram probability	Web Spam Corpus and WEBSPAM-UK 2006/7	Dataset 1 (22760 Pages): 2760 Spam and 20000 Non-Spam , Dataset 2 (100000 pages): 50000 Spam, 50000 Non-Spam	Decision Tree C4.5	Bagging and Boosting techniques.
7.	Dong and Zhou, 2012 [99]	Web Spam Detection	1. Distribution-based Topical Diversity Measure 2. Semantic-based Topical Diversity Measure	WEBSPAM-UK2007	F-Measure : 0.949 (RandomForest), 0.843(RandomTree)	Random Forest and Random Tree	Boosting technique

Content spam

Spammers uses various techniques to intrude the search engine rankings. Content Spam is the technique that modifies the content of the web page to attract the user query so as to achieve the better ranking in the search engine results. Keyword stuffing is one of its example. So it is quite beneficial to understand the relationship between the various spam content features and non spam content features. The correlation and correlation coefficient of content features has been analyzed. The features considered for experiments are: number of words in the page, number of words in the title, average word length, fraction of visible text, compression rate, entropy, fraction of anchor text and other content features. Their relationship can also help us to build good web spam detection system [100]. There are various authors who have tried to use this technique for spam detection as illustrated below in Table 2.4:

In paper 1, Various content features used detected 86% spam. It is believed that keyword stuffing can not be easily detected and can fool the spam detection algorithms as well. Heuristic methods forms different layers of the proposed web spam detection system [25]. Boosting and bagging techniques are also applied for improvement and results are shown in Fig. 2.30.

In paper 2, 94% of the spam is detected by the reliable and inexpensive proposed web spam detection tool. Relationship between multiple words has been experimented like title and content of web page. Bigrams contributes majorly in the development of this tool. Computational time is highly concentrated [95].

In paper 3, Comment spam in blogs has been detected by extracting the major content features like post-comment similarity, which can be further improved by expanding the keywords. Though this technique is not so good but can be used for social networks where spam comments can harm the system [96].

In paper 4, The content features are used to characterize the web graph and the score is computed for each node in the graph to detect the spam pages. This helps in demoting the spam pages by the proposed algorithm, which can be improved by using the content features for traversing the graph as well [97].

In paper 5, Content trust model has been build for web spam detection by considering the content features of web page like anchor text. The ranking has also been improved by deploying this model into ranking criteria which is used by the search engines [17]. The content trust modeling process has also been experimented while detecting spam in tagging of social network users [101].

In paper 6, The analysis of content features has been done while developing SAAD. The various content features for detection of web spam has been extracted. The relationship between the web pages is not taken into consideration. Decision tree is used for experiments. Bagging and boosting techniques have also been used for improvement [98].

In paper 7, proposed web spam detection technique based on topic models and analyze the topic diversity measures based upon these models. Various LDA models have been exper-

class	recall	precision
spam	82.1%	84.2%
non-spam	97.5%	97.1%

Recall and Precision of proposed classifier

class	recall	precision
spam	84.4%	91.2%
non-spam	98.7%	97.5%

Recall and Precision after Bagging Technique

class	recall	precision
spam	86.2%	91.1%
non-spam	98.7%	97.8%

Recall and Precision after Boosting Technique

Figure 2.30: Results by Analysing Content [25]

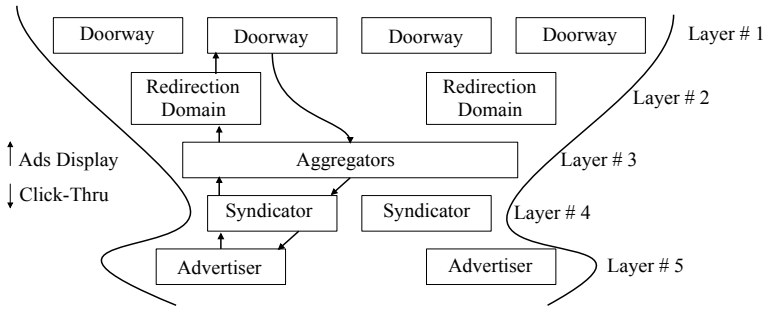


Figure 2.31: Spam double-funnel architecture [26]

imented and results proved that the content analysis according to topic helped a lot in spam detection [99].

Miscellaneous

A Double-funnel model is proposed for detection of **Redirection Spam**. It is a five layer model as shown in Fig. 2.31 which at each layer identifies the two keywords/object: First, targeted spammers and Second, the targeted advertisers. It has been observed three out of four unique URLs are spam after the categorization of top-15 domains into 4 major subcategories as shown in the Fig. 2.32 [26].

People going to remote Locations use GPS facility. There are different facilities like Gowalla and Foursquare that comes under the category of Location-based Social Networks(LBSN) [102]. It allows seekers to exchange geographic location but commenting over the location is nowadays prone to spam. This is known as *Tip Spam*. Experiments have been conducted on

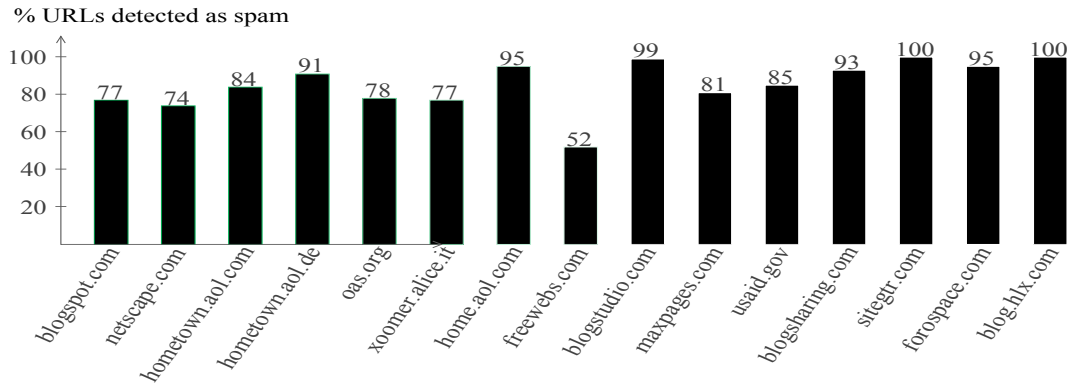


Figure 2.32: Top doorway domains and their spam percentages [26]

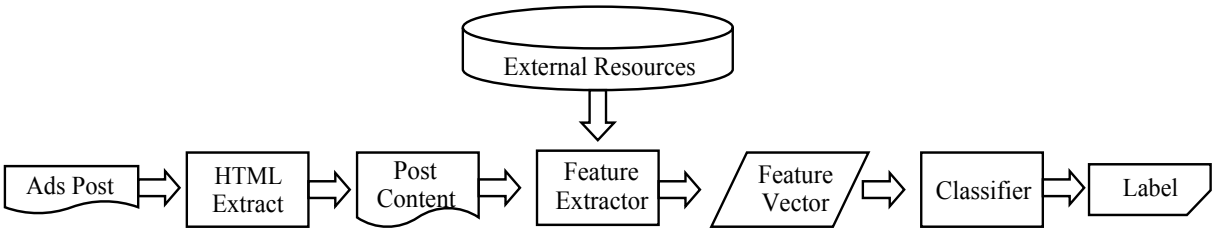


Figure 2.33: An overall framework for spam detection in online advertisements [27]

Apontador and Brazilian LBSN system. By analyzing the following attributes: content-based, user behavior, user post location attribute, relationship among users, various user behaviors, tip spam has been successfully detected [103].

An algorithm, SOcial network Aided Personalized (SOAP) and effective spam filter is proposed for the detection of *Email spam* in social networks. Firstly set of emails are manually checked that whether they are spam or legitimate. Then the bayesian spam classifier is trained with spam emails for the detection of spam keywords. With the probabilities computed for various spam keywords, email spam can be easily detected [104].

Online advertisements are also prone to spam by the Spammers. A spam detection approach is proposed for the detection of *online spam advertisements*. The content features are extracted from the dataset along with domain specific features. After labelling by the two judges, they are used for spam classification. Boosting and bagging techniques are used for performance improvement. The overall framework is described as in Fig. 2.33 [27].

Reviews about the products are available socially as they are posted online. Due to personal reasons such as - business profits, defaulters try to spoil the fair means by publishing spam reviews. For the detection of *Review Spam*, many techniques have been introduced, after which the review process has been drawn as shown in the Fig. 2.34. The features considered while detection of spam reviews are: content of review, meta-data of review, information about the product. Three different categories have been detected: spammers, review spam and spammer groups as shown in Fig. 2.35 and its analysis from year 2007 to 2014 is shown in Fig. 2.36. The different methods used for its detection are: finding duplicates, content based methods and

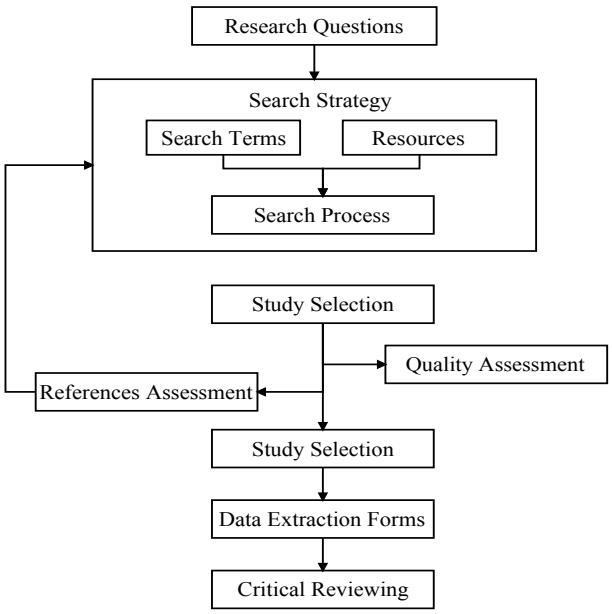


Figure 2.34: Review process [28]

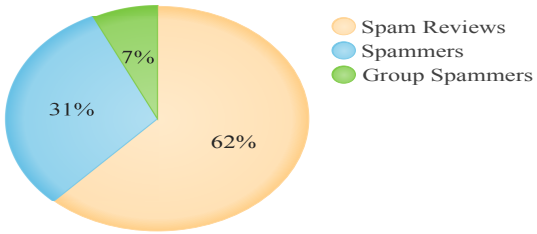


Figure 2.35: Distortion of proposed techniques in spam detection [28].

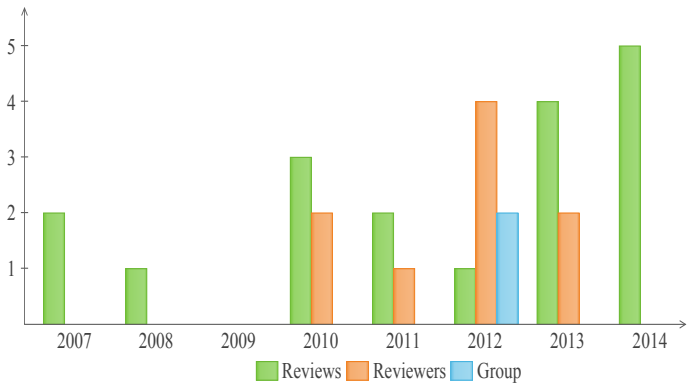


Figure 2.36: Distribution of research attention by publication year [28]

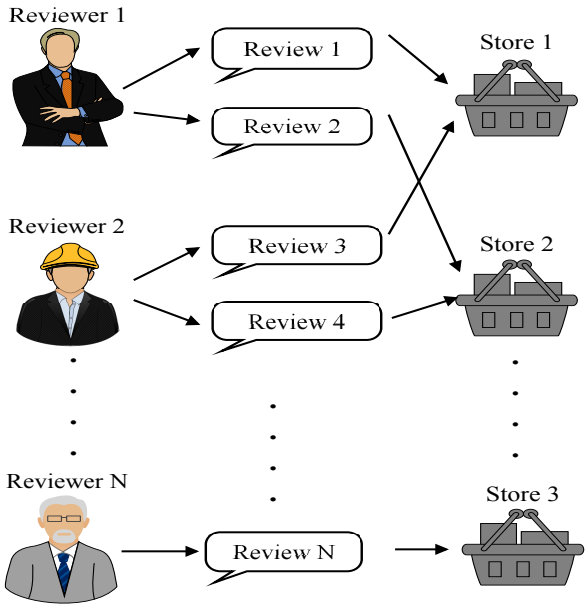


Figure 2.37: Relations between reviewer and store through graph nodes [28]

other methods. The relation between the reviewer and the store is also illustrated in the Fig. 2.37 [28].

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 belongs 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 [38].

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 [39]. All these features are considered in the model RankNet along with the different strategies used which are summarized in Table 2.7.

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 [105].

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 doing one certain action). The scheme has been validated in SVM [106].

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 [12]. Same work has been extended by extracting three new features namely Query Diversity (QD), Spam Query Number (SQN) and User-oriented TrustRank [13]. Click spam in the Ad networks have also been addressed [107].

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 [108].

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 [109]. 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 [110].

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 [111].

The user behavior analysis for different domains are summarized in Table 2.5.

Table 2.5: Summarized User Behavior analysis

Author	Algorithm	Description	Dataset	Features considered for Experiments	Results
Wang *et al.*,2016 [38]	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 2.6
Agichtein *et al.*,2006 [39]	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 [106]	Web Spambot detection	Spam robots have been detected by analyzing user behavior	Number of human records : 5555 , Number of Spambot records : 11039 , 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 [12]	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 [108]	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 (33150), UsageRanking (10000)	Visiting count and dwell time	0.317 at Normalized Discounted Cumulative Gain (NDCG) @ 10
Liu *et al.*, 2010 [111]	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 2.6: Depth dwell time prediction comparison [38]

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

Table 2.7: Strategies used for web search results prediction [39]

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

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.

2.2.2 PageRank based spam detection

Analyzing the PageRank of web page and its formulation deeply, helped in link spam detection. Benczur et al. [112] defines SpamRank as the share of unbiased distribution of PageRank by its supporters. It is done by penalizing the low quality supporters. The idea is generated by assuming that the honest page supporters are spread all over the web and spam page supporters contribute only to the target page. Supporter is judged by its quality for reliability. SpamRank aims at finding the feature set of a spam supporter web page. In this algorithm, the supporter of each page is selected by monte carlo personalized PageRank simulation. Then, the PageRank distribution is evaluated by the power law model to punish the spam supporters.

Link spamming has been renamed as topological spamming, with the same purpose of detection of link farms by Beccetti et al. [84]. The methodology proposed, computes the statistics of the neighbors for every node, and then uses this information for detecting link spam. In the experiments, it was concluded that the high ranking pages have large number of supporters. The complexity of large graph dataset was reduced by adopting the semi streaming model of computation. The truncated PageRank algorithm has been proposed for diminishing the immediate neighbor's contribution. Probabilistic counting algorithm has been used for making an estimate of the supporters.

Another variant of PageRank, i.e., DirichletRank solves the problem of link spam by analyzing the neighbors [35]. It is more resistant to link spam. The connection between the bogus page and target page indicates the occurrence of link spam and hence, detects it. Calculating spam mass can also help in link farm detection. If the PageRank computed is high and is boosted by non-reputable pages, then its spam mass is also to be high and if boosted by reputable pages, then its spam mass is low. First approach is labeling, which only counts the in-links that can be easily manipulated. Second approach not only counts the in-links but also checks the PageRank value of each contributor. It may also fail because looking only at immediate links does not satisfy the property of spam mass estimation. Third approach is following all the PageRank contributors of the in-links. Spam mass can be summarized as the contribution of direct or indirect neighbors in the PageRank of the spam node. An algorithm was designed to detect spam pages using spam mass [89].

2.2.3 Trust/Distrust based spam detection

As there are a number of malicious web pages, so trusting every web page is not practically possible. Trust among seeds of web graph form is a crucial measure and helps in spam

detection. An algorithm is introduced on the methodology of PageRank which follows the out-links rather than in-links. This algorithm helps to find the trust among the seed sets, hence it is named as TrustRank. It works by selecting the small seed set and manually judging them by the team of experts. They particularly determine the status of all the pages existing in the set by oracle function. Other pages are evaluated by applying the same decision tree generated previously which is called as approximate isolation [5]. The seed selection process has been simplified further by topical trustrank. For each web page, a trust score is evaluated by the trustrank algorithm. There can be multiple scores for a single page (a single page may lie in more than one topic) which are combined by simple summation or by quality biasness to generate the topical trustrank score [6]. In-link Growth Rate (IGR) is used for determining the increasing rate of in-links with comparison to original in-links. This modification is known as link Variable trustrank (LVTrustRank) [113].

A page is spam if it directs to another spam page. On the basis of this assumption, an algorithm proposed is BadRank by Kolda et al. [114]. BadRank is the weighted sum of scores of all the bad outbound links a web page points to. The aim is to remove such outbound links, so that the trusted page does not receive any BadRank score. Trust score is embedded in the computation of BadRank. The experiments resulting in differentiating are bad leaf links for incorporating trust. Another work done for propagating trust is Trust Propagation Rank(TPRank). It works in the same way as trustrank, but uses two different seed sets for the demotion of spam, reputable seed set and spam seed set. TPRank has been used for spam demotion and a new method proposed by following the approach of TPRank i.e., Trust Propagation (TP) spam mass. Equation 2.2 is used for computing TP spam mass: TP_SM stands for Trust Propagation (TP) spam mass, PR stands for PageRank, TP stands for Trust Propagation Rank [115].

$$TP\_SM = \frac{PR - TP}{PR} \tag{2.2}$$

The above discussed technique propagates either through trust or distrust. A page has either been proved as trustworthy or as dis-trustworthy. A new method proposed evaluates both the scores for a web page, i.e., T-Rank (Spam demotion) and D-Rank(Spam detection). The algorithm proposed for its computation is Trust-DistrustRank (TDR) algorithm. The trustworthy page is rewarded with more trust score whereas the dis-trustworthy page is penalized with the distrust score. T-Rank has been experimented and compared with PageRank, TrustRank, AVRank and LCRank. D-Rank has been experimented and compared with Inverse-PageRank and Anti-TrustRank. T-Rank and D-Rank, both outperforms [116]. The seed selection in trust/distrust propagation based algorithms suffers from a major issue i.e., selecting the large seed set manually. It is not convenient because of high cost, lack of single team of experts with vast knowledge about the web and is also a very time consuming task. ASE (Automatic Seed set Expansion) algorithm is proposed for constructing the larger seed set. If the trustworthy

link is recommending another link, then it is also assumed to be trustworthy. If large number of trustworthy links are recommending the link, then it is called as joint recommendation link structure. This is how the seed set is expanded with probabilities and support degree [82].

The comparison of all the variants of trustrank algorithm is summarized in [117]. All these algorithms are based on trust/distrust propagation. Either propagating through good seeds or bad seeds is presented in Table 2.8. The summarization of above discussed spam detection techniques either based on PageRank or Trust/Distrust is presented in Table 3.2.

Table 2.8: Performance of Trust/Distrust based spam detection models

Algorithm	Year	Good seed set	Bad seed set	Description	**Evaluation metrics**
TrustRank [5]	2004	√	×	Good pages are identified by propagating trust with inverse PageRank	compared with PageRank
ParentPenalty [80]	2005	×	√	Bad pages are detected as spam pages points to spam pages	compared with HITS algorithm
Anti-TrustRank [89]	2006	×	√	Distrust is propagated in reverse direction	
Topical TrustRank [6]	2006	√	×	Trust is propagated depending upon the topic	compared with PageRank and TrustRank
R-SpamRank [118]	2007	×	√	Inverse PageRank to propagate through distrust pages	
DiffusionRank [119]	2007	√	×	Works same as TrustRank	compared with PageRank and TrustRank
Link Variable TrustRank [113]	2008	√	×	Works same as TrustRank	compared with TrustRank
BadRank [114]	2009	×	√	Works for removing the bad out-links	
Trust Propagation Rank [115]	2014	√	×	Works same as trustrank but with two different seed sets, i.e., reputable and spam	compared with TrustRank
Trust Propagation spam mass [115]	2014	×	√	It is computed with the help of PageRank and TrustRank	compared with spam mass
Trust-DistrustRank [116]	2014	√	√	Computes both trust and distrust score using penalty mechanism	compared with D-Rank, Anti-Trust Rank and Inverse PageRank

2.2.4 Webpage features based spam detection

Spam can also be detected by analyzing the features of a web page. Spammers try to manipulate the web page content and linkage properties between two or more web pages. Content spam and link spam have been detected by the different web spam detection algorithms. Feature analysis has been done with respect to various aspects such as: neighbor analysis, link-based features analysis, content-based features analysis, extracted linguistic features, temporal analysis, hybrid features analysis and link diversity measures. All the techniques are summarized in Table 2.10.

Table 2.9: Web Spam Detection Algorithms

Author	Method	Description	Dataset	Execution time	Labels	Results	**Advantages**	**Disadvantages**	**Future work**
Benczur *et al.*, 2005 [112]	SpamRank	Rank for measuring the contribution in PageRank by spam supporters	31.2 M web pages of *.de* domain	17 hours (PageRank) + 4 hours(db creation) + 20mins (SpamRank)	Reputable: 70%, Spam: 16.5%, Advertisement: 3.7%, Weborg 0.8%, Non-existent: 7.9%, Empty: 0.4%, Alias: 0.3%, Unknown: 0.4%		Demotes the spam pages	Does not provide the updated ranking methodology	Incorporating SpamRank into a ranking function
Becchetti *et al.*, 2008 [84]	Truncated PageRank	Detected link spam by estimating spam supporters	18.5 M pages of UK-2002, 77.9 M pages of UK-2006	Approx 2 months	UK-2002: Normal 84%, Spam 16% , UK-2006: Normal 88% Spam 12%	F-measure: 0.870 (UK-2002), 0.630 (UK-2006)	Truncated the spam page contribution from PageRank score	Content based features were not considered while experimenting	Explore page level spam detection
Wang *et al.*, 2008 [35]	Dirichlet Rank	Solves the zero-one gap problem of PageRank, more stable and more resistant against link spam	1 .GOV dataset of the TREC and .UK Dataset	Depends upon the number of iterations	spam and no spam		Solves the zero-one gap problem of PageRank		
Gyongyi *et al.*, 2004 [5]	TrustRank algorithm	Evaluating the trust by following out-links	AltaVista crawled and indexed pages in August 2003	Depends upon number of web pages	Reputable- 56.3%, web organization- 3.7%, spam- 13.5%, others- 26.6%	Precision: 0.86, Recall: 0.55	TrustRank incorporates quality of site	-	Different seed selection processes
Wu *et al.*, 2006 [6]	Topical trustrank	topic selection resulted in better detection of spam sites.	Standford's WebBase Project and country specific web crawl courtesy of search.ch	-	spam and non spam	10% higher than trustrank	Reduce the ranking of spam pages in bigger communities	Boost the rankings of spam pages within smaller communities	Multi topic web pages for computation
Gyongyi *et al.*, 2006 [89]	Spam mass	Link Spam Detection using Spam Mass	Yahoo search engine (73.3 million individual web hosts)	-	good, spam, unknown, non-existent	-	It can handle irregular link structures	-	Content analysis for web spam detection
Kolda *et al.*, 2009 [114]	BadRank	Score computed by analyzing the outbound links	WEBSPAM-UK 2007	-	leaf self links, leaf bad links, self links, none	AUC: 0.7426-0.7497, SD: 0.0177-0.0220			
Leng *et al.*, 2014 [115]	TPRank, TPspam mass	TPRank for spam demotion, TP spam mass for spam detection	WEBSPAM-UK 2006, WEBSPAM-UK 2007		WEBSPAM-UK 2006: 5549 (Normal), 1924 (Spam) , WEBSPAM-UK 2007: 8980 (Normal), 501 (Spam)	Spam demotion: 10.623%, detection: 43.216%	Demotion and detection of web spam	-	Attempt to machine learning models for improvement
Zhang *et al.*, 2014 [116]	Trust-Distrust Rank algorithm	computes both trust and distrust scores for web page	WEBSPAM-UK 2007	15.38s	good hosts: 3169 , bad hosts: 134	T-Rank: 0.0453, D-Rank: 0.6270	Overcomes the disadvantages of TrustRank and Anti-Trust Rank	Does not include content features	Incorporating with other improvements of TrustRank, Anti-Trust Rank
Rungsawang *et al.*, 2007 [7]	Link Farm Effect Detection	Done by distributing the unbiased score in the web	YahooAPI	-	-		provides an automatic, adaptive way of combating link spam	Small dataset used for experiments	more link farm configurations with larger web graph can be explored.
Yang *et al.*, 2015 [8]	DRank algorithm	concentrate upon inbound hyper-links	WEBSPAM-UK2006	-	Spam and non-spam		analyzed the quantity, quality and diversity of their inbound hyper-links	-	Web searching can be improved by incorporating such methods into ranking methodology
Zhang *et al.*, 2013 [82]	Automatic Seed set Expansion algorithm (ASE)	used link structure for expanding the seed set	WEBSPAM-UK2007	-	Domain Labels: .org.uk, .co.uk, .ac.uk, .gov.uk	Recall: .ac.uk: 59.2% to 55.9%, .gov.uk: 68.1% to 58.1%	Useful for trust based propagation algorithms	Spam demoted is by spam manipulation	Expanding the spam seeds via a reasonable process.

Table 2.10: Web Page features used for web spam detection

Author	Dataset	Link based features	Content based features	Model	Cross Validation	Bagging	Results
Becchetti *et al.*, 2006 [120]	WEBSPAM-UK2002	Degree-based, PageRank, TrustRank, Truncated PageRank, TrustRank, Truncated TrustRank, Estimation of Supporters	-	Decision Trees	Ten fold	-	Detection rate: 80.4% to 81.4%, False Positive rate: 1.1% to 2.8%
Castillo *et al.*, 2007 [10]	WEBSPAM-UK2006	Degree-related measures, PageRank, TrustRank, Truncated PageRank, TrustRank, Truncated TrustRank, Estimation of Supporters	Number of words in the page, number of words in the title, average word length, fraction of anchor text, fraction of visible text, compression rate, corpus precision, corpus recall, query precision, query recall, independent trigram likelihood, entropy of trigrams	Decision trees, Stacked graphical learning	-	Done	F-measure: 0.723% to 0.763%, Detection rate: 78.7% to 88.4%, error rate: 5.7% to 6.3%
Abernethy *et al.*, 2008 [121]	WEBSPAM-UK2006	Host features, slack variables, graph regularization	-	Support vector machine	-	-	AUC: 0.963%
Becchetti *et al.*, 2008 [122]	WEBSPAM-UK2006	Degree-related measures, PageRank, TrustRank, Truncated PageRank, TrustRank, Truncated TrustRank, Estimation of Supporters	21 content features	Decision Tree	-	Done	F-measure: 0.723% to 0.763%, Detection rate: 78.7% to 88.4% , error rate: 5.7% to 6.3%
Piskorski *et al.*, 2008 [123]	WEBSPAM-UK2006, WEBSPAM-UK2007	-	Corleone- based features and Features extracted by General Inquirer(GI)	absDist, sqDist metric formulation	-	-	
Mahmoudi *et al.*, 2010 [124]	WEBSPAM-UK2007	-	-	LADtree, Neural Network, SVM , Naive Bayes	Ten fold	-	LADtree: 73.4%(ROC), 66.6%(F-measure) Neural Network: 72.7%(ROC), 67.3%(F-measure) SVM: 69.6%(ROC), 69.3%(F-measure) Naive Bayes: 67.5(ROC), 62.3(F-measure) Random Forest: 76.3%(ROC), 71.4%
Erdelyi *et al.*, 2011 [125]	WEBSPAM-UK2007	In-link Death Rate (IDR), In-link Growth Rate(IGR), Out-link Death Rate (ODR), Out-link Growth Rate(OGR), Change rate of clustering coefficient (CRCC), XJaccard, PSimRank	Ave, AveDiff, Dev, DevDiff, Decay	Bagged Logit-Boost, Decision Trees, Bagged Cost-sensitive Decision Trees, Logistic Regression, Random Forests, Naive Bayes	Five fold	-	AUC: 0.789%
Algur *et al.*, 2012 [126]	WEBSPAM-UK2007	Neighboring links of Target URL	Keyword redundancy, title keyword redundancy, meta tag keyword redundancy, anchor text keyword redundancy, URL keyword utility	-	-	-	Recall: 86%(Spam), 92%(Normal), Precision: 91.48%(Spam), 86.79%(Normal)
Gongwen *et al.*, 2012 [127]	WEBSPAM-UK2007	Link Diversity measures	Number of title words, Web pages compression ratio	Web page ranking	-	-	
Roul *et al.*,2016 [128]	WEBSPAM-UK2006	Personalized PageRank	Term Density test, POS(Part of Speech) ratio test	-	-	-	F-measure: 75.2%
Makkar *et al.*,2017 [129]	UK- 2011 web spam	-	Anchor text, meta character, meta word, unique word, total word, max length	Bagged CART, eXtreme Boosting technique, parallel random forest	Ten fold	Done	AUC: 0.912, Precision: 0.532, Accuracy: 90.39

Chapter 3

Issues in search engines ranking methodology

Information retrieval methods only consider the content of web page and ignore the link structure of the web. But, the new algorithm PageRank, considers the structure of web while calculating the score of a web page for its ranking [130].

The PageRank score of any page (or set of web pages on a topic known as community) is dependent on four main factors namely as [47]: *total number of pages in web, number of in-links, number of out-links and number of dangling pages*. A web page may contain a hyperlink to itself. There are many methods to compute PageRank such as- Power Method, Jacobi Algorithm, Gaussian Elimination, Gauss Seidal Method (Modified Jacobi Algorithm). Although Gauss Seidal is faster than Jacobi Algorithm but the data structure required to store the web matrix (if not fitted in main memory) cannot be handled by Gauss Seidal algorithm, so Jacobi algorithm is much better than Seidal algorithm [47]. Random walk-based techniques, such as - PageRank, encode the structure of the graph in the form of a transition matrix of a stochastic process from which the significance of the nodes in the graph is analyzed [131].

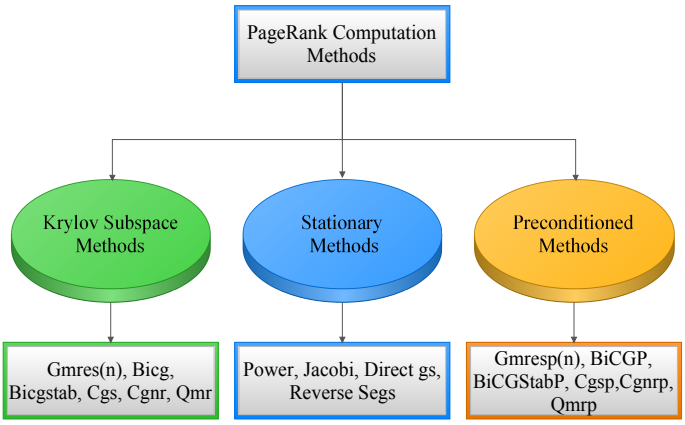


Figure 3.1: Methods used for computation of PageRank algorithm [2]

There is a requirement to update the PageRank vector because 40% of the web pages

changes in a week, i.e., the changing ratios of web pages fiddle among the pages of large content. PageRank vector is updated monthly by Google from scratch as previous PageRank vector is not at all useful for the next updation [130]. The web graph has atleast one non zero entry in a row, this is an assumption while calculating PageRank, which actually does not hold. There are pages with no out-link as well [35]. The PageRank algorithm assumes that the good page is pointed by the good page [132]. The PageRank is highly dependent upon the number of in-links [133]. Many techniques have been introduced for the improvement of PageRank computation such as - Extrapolation methods [134], Adaptive methods [135], Krylov subspace methods [29], Monte Carlo methods [136]. These techniques are summarized in Fig. 3.1.

Power method is the method which is based on web matrix and used for computing the PageRank score. The formula used for computing the PageRank score by power method is given by Eq. (1).

$$x_p = d \sum_{q \in pa[p]} \frac{x_q}{h_q} + (1-d) \tag{3.1}$$

Power Method is required because of the presence of large size web matrices and vectors. Power method also saves the storage space as only the current iteration of one vector is stored as compares to other method such as - GMRES where large number of vectors are stored. The random links are selected by the random surfer and the surfer may get bored after span of time and then he tries to explore some new pages. This probability to jump after getting bored is known as Damping factor *(d)* [32]. The convergence rate directly depends upon the value of d, which if chosen small, decreases the importance of a web page. The Eq. (1) is used by the power method for the computation of 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.

The value of d is an open research issue because of emerging web. To reduce the complexity of this method, either the number of iterations can be decreased or the workload of each iteration. This method does not compute the contribution of back button in the web browser. As PageRank is using the link structure of the web, so to depict its true picture it is advisable to select the value of d close to 1. This may lead to instability and convergence rate as well. Convergence rate of power method also depends on the web graph connectivity [130].

Few other ranking methods are also used for web document's ranking, for example, measuring recency ranking in aspect to freshness of the document mainly used for news updations [137]. Information about the search results of previous related queries also helps in improving the search results with the help of rank aggregation methods. Similar to other ranking algorithms like PageRank, it is also time consuming because of construction of the various win/loss graph but it came out with improved results when compared with the experiments performed for traditional query methodology [138]. Topic-related web pages can form a com-

munity which is returned by the query engine. This community may contain noisy data because of inter-linked web pages. Removing the noisy data from the community is a big challenge. It has been done efficiently as topic-related documents may also contain spam and irrelevant sites. The framework is proposed for the same [139]. Location can be another feature which can be included in the ranking algorithm for improving the web search as few queries are location dependent [140].

The main focus of this survey is to analyze PageRank algorithm which can be helpful in web spam detection. To achieve this, a complete updations of PageRank algorithm are studied and presented here. These PageRank updations take place for reducing the issues such as - convergence rate issue, deduction of computation cost, handling dangling nodes, prevention of web spam and structure of web graphs. It has been observed that the PageRank algorithm is adopted in various other domains such as multimedia, web graph, civil designs.

Convergence rate

Convergence rate is one of the important measures which influences the success of an algorithm. Power method requires very less storage space as it stores only two vectors at a time. Therefore, the convergence rate of power method is slow. Many techniques have been proposed to speed up the convergence law of power method such as - extrapolation method [134] [141] [142], adaptive method [135], and stationary methods [29]. Some of these methods are described as follows.

- Extrapolation method works on the basis of first eigenvalue of the Markov process and initializes it as 1 in its first iteration. With this, the non principal eigen-vectors can be easily computed by further iterations.

- Aitken extrapolation and epsilon extrapolation methods save the iteration time upto 38% as compared to the original power method. Quadratic extrapolation method is found to be more beneficial as it saves upto 59% of iteration time [134].

- The next algorithm is the updated extrapolation method based on Ritz values. In this method, firstly a vector is formed using extrapolation procedure. Then to combine the arnoldi method with extrapolation technique, it starts the new Arnoldi factorization with that vector. The resultant method is known as an Arnoldi-extrapolation algorithm for PageRank Acceleration. The key idea behind adopting this strategy is to switch over to Arnoldi iteration after extrapolation method [142].

- Adaptive methods does not consider the web pages which have been converged in previous iterations. Reordering may be costly but the reduction in matrix saves time and reduces the cost of subsequent iterations. This methodology results with an increase in the computational speed by 30% [135].

DAMPING FACTOR / METHODS	0.5	0.75	0.85	0.9	0.95	0.98
POWER	100	100	100	100	100	100
JACOBI	100	100	100	100	100	99.7
DIRECTGS	100	100	100	100	100	99.9
REVERSEGS	100	100	100	100	100	100
BiCG	42.3	10.4	5.9	5.7	5.7	5.6
BiCGP	100	99.5	98.0	95.3	87.6	78.0
BiCGstab	99.1	80	61.2	52.8	37.8	24.0
BiCGstabP	100	100	100	100	100	100
CGS	59.8	25.9	7.0	3.3	3.1	2.9
CGSP	100	100	99.3	98.2	92.3	82.1
CGNR	100	83.3	45.0	62.5	23.5	22.5
CGNRP	100	99.0	96.0	96.8	65.3	65.2
QMR	97.4	93.2	88.9	81.2	72.4	63.6
QMRP	100	99.9	99.5	98.7	94.4	85.4
GMRES(10)	100	100	100	100	94.6	66.7
GMRES10P	100	100	100	100	100	100
GMRES(20)	100	100	100	100	100	100
GMRES20P	100	100	100	100	100	100
GMRES(40)	100	100	100	100	100	100
GMRES40P	100	100	100	100	100	100

Figure 3.2: Comparison of stationary and non-stationary methods [29]

- Many stationary methods such as - Jacobi, Gauss-Seidal, Reverse Gauss-Seidal are proved to successful, but Gauss Seidal is found to be the best. Non-stationary methods also exists but can not perform as good as stationary methods. Among all the non stationary methods, BiCGStab can turn as same as Gauss-Seidal. The stationary and non-stationary methods are comparable [29] as in the Fig. 3.2.

- The web matrix can be partitioned into blocks for the parallel execution in power method and at each iteration the synchronization of all the blocks is done similar to Jacobi Splitting method. This splitting can be done in two ways either row-wise partitioning or non-zero elements partitioning. Both, the Relaxed (REL) and Extrapolated (EXT) methods, can be merged to the new algorithm called as Relaxed Extrapolated (RELEXT). This new formation, RELEXT, can reduce the iterations of power method at the greater extend [143].

- Improved monte carlo (MC) method states that whenever the crawler fetches new data in the web, it should be implemented in parallel. This lead to the resolution of PageRank updation. MC algorithms can start up either randomly or cyclically, but the random start is not as efficient as cyclic start. The information of all the visited pages is maintained by this algorithm [136].

- Two sampling methods proposed for this purpose are: Direct Sampling in Power Iteration (DSPI) and Adaptive Sampling in Power Iteration (ASPI). Both are based on Low-Rank

approximation of Transition Matrix. DSPI do the sampling of transition matrix only once and then uses it in the computation process while ASPI do the sampling of transition matrix many times with the sample adaptive rate adjusted in the computation process [144].

- Many algorithms are introduced to increase the convergence rate of power method irrespective of damping factor. The improved PageRank method called as Multiplicative Splitting Iteration method is introduced by using the methodology of linear system and the multi-splitting iterations [145].

- Another algorithm was developed where each web page with the help of linked links can update its own PageRank. It is known as randomized algorithm because it works at random times. Main emphasis in this algorithm was to improve the convergence rate of PageRank values. One agent/web page update its value by taking the average of all the values which are received by it at a particular instant. A webpage can send its score to various out-links and can receive the same from in-links. To reduce the communication and computation load, instead of waiting for the PageRank from all the links, one page can wait till the convergence of PageRank and then simultaneously can update. Same concept is modified by locally updating the PageRank by communicating with the neighbours [132].

- To reduce the computation load, web aggregation approach was developed. In this approach, the web is divided into various groups according to the criteria of links and each individual group then computes its PageRank. The value thus computed is the total value of whole group which is distributed among all the group members. In this approach, the request for PageRank is forwarded only to the outgoing links [133].

It has been observed concluded that the size of web should be considered while working for the convergence rate issue of Distributed Randomized PageRank Algorithms (DRPA) [146].

Computational cost

As the size of web is large, so a large number of iterations are required for the computation of PageRank. Various approaches are introduced to reduce the computational cost. The initiative taken for it is the reordering of PageRank algorithm. It is the process of locating zero rows in the Google matrix. This procedure is repeated till the sub matrix of non zero elements is formed. The computation is done on this sub matrix and subsequently on full matrix iteratively. This is the first updation of PageRank which works better in the presence of Dangling Nodes [147]. This approach of reordering has been improved with an Adaptive Reordered method by bringing the termination criteria for each iteration. It reduces the overhead of recursively reordering method [148]. Another approach known as Incremental Iteration approach

Type	Criterion (A)				Criterion (B)			
	Number of Total operations	Ratio	Improv ement	Accuracy	Number of Total operations	Ratio	Improv ement	Accuracy
Power iteration method	228764	1	-	-	228764	1	-	-
Incremental PageRank algorithm	209993	0.92	8%	0.99993	212362	0.93	7%	0.99994
Incremental iteration method	196083	0.86	14%	0.99218	163403	0.71	29%	0.99273
IPII	182432	0.80	20%	0.98405	151906	0.67	33%	0.98181

Figure 3.3: Computational cost of various PageRank methods and IPII [30]

is implemented. This approach uses the already used components to compute the uncovered components [36]. It is beneficial to join this approach with Incremental PageRank approach in which only the scores of calculated components and its dependent node's scores are considered. After combining, new approach thus formed is known as Incremental PageRank and Incremental Iteration method (IIPI). It decreases the communication load [30] as discussed in Fig. 3.3. MIRAM is used for distributed computational matrices of large size popularly known as power law networks. Computation of MIRAM method involves the Arnoldi algorithm as shown in Fig. 3.4 and its comparison with PageRank is given in Fig. 3.5. It clearly states MIRAM's outstanding performance because of presence of Hyper graph partitioning in its procedure [31].

Dangling pages

Dangling nodes are those which do not have hyperlinks. Presence of dangling nodes lead to loss of energy. If all the pages which are present in the web matrix are dangling pages, then the PageRank can be computed online. Computed PageRank value is approximately equal to the loss of total energy [47]. After making the matrix with the web pages, if any row contains all zeros, then it indicates the occurrence of dangling nodes. It is difficult to tell how exactly the dangling nodes are handled by Brin and Page [130].

In the original PageRank score calculation by Brin and Page, the dangling pages were just removed while calculating PageRank and then were added back after calculation [46]. This procedure actually increases the number of dangling pages instead of decreasing. The improved method for calculation of PageRank treats all the dangling nodes as a complete graph as it points to every other node and removes the damping factor from the calculation [32]. For example as in Fig. 3.6 , we can see the web martix gets changed by including or excluding dangling pages. Another methodology used is to jump from dangling page with the probability 1 to any random web page [149].

As computation of PageRank score includes dangling nodes, so handling them appropriately can reduce its complexity. Different authors have discovered methods for the same. There are many lumping algorithms in the literature such as - Lumping Dangling Nodes Algo-

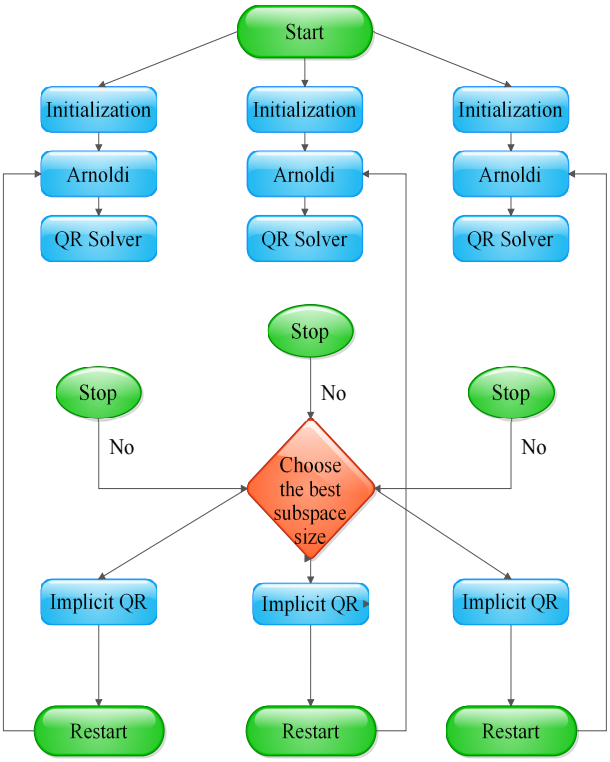


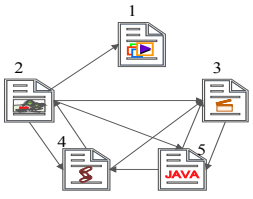
Figure 3.4: Overview of MIRAN [31]

	Power Method		MIRAN(4,8),k=2	
Damping Factor	**MVP**	**Execution Time**	**MVP**	**Execution Time**
0.85	239	27.29	56	22
0.90	401	48.53	62	21
0.95	725	83.56	62	27
0.99	3525	363.94	62	23

Figure 3.5: MIRAN versus original PageRank [31]

Link Structure Excluding Dangling Pages

Source Node	Out Degree	Destination Nodes
2	4	1,3,4,5
3	1	5
4	2	2,3
5	2	3,4



Link Structure Including Dangling Pages

Source Node	Out Degree	Destination Nodes
1	5	Null
2	4	1,3,4,5
3	1	5
4	2	2,3
5	2	3,4

Figure 3.6: Web structure of five nodes [32]

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

Figure 3.7: Summary of Lumping Algorithms [33]

rithm [150] . These algorithms focuses on the number of dangling nodes present in the web matrix as presented in Fig. 3.7 and are explained below.

A two-stage algorithm states that the non-dangling and dangling pages should be treated as different blocks. This formulation clearly depicts the importance of dangling pages [151]. The same concept of two different blocks was implemented using Jordan decomposition. It 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 [152]. Further the calculations were reduced by dividing the non-dangling pages in two blocks, weak non-dangling and strong non-dangling pages [153]. Dangling pages mainly arises from untruly sources and can be handled by the evaluating the prestige of web pages. Proportionate Prestige Score (PPS) algorithm computes the prestige score of a page and it propagates with every page it points to. This score is found to be more important than PageRank [34]. Process of calculating PPS has been explained in Fig. 3.8. It can be easily observed that the appropriate handling of Dangling nodes affects the PageRank calculations.

Web spam

With the emerging web, business organizations can gain huge amount of profits with the high ranking in search engine results. This is done by manipulating the link structure of web. Different tecniques are present in literature for detecting the web spam. Robust PageRank was introduced to fight against link spam by decreasing the ratio of contribution of influential

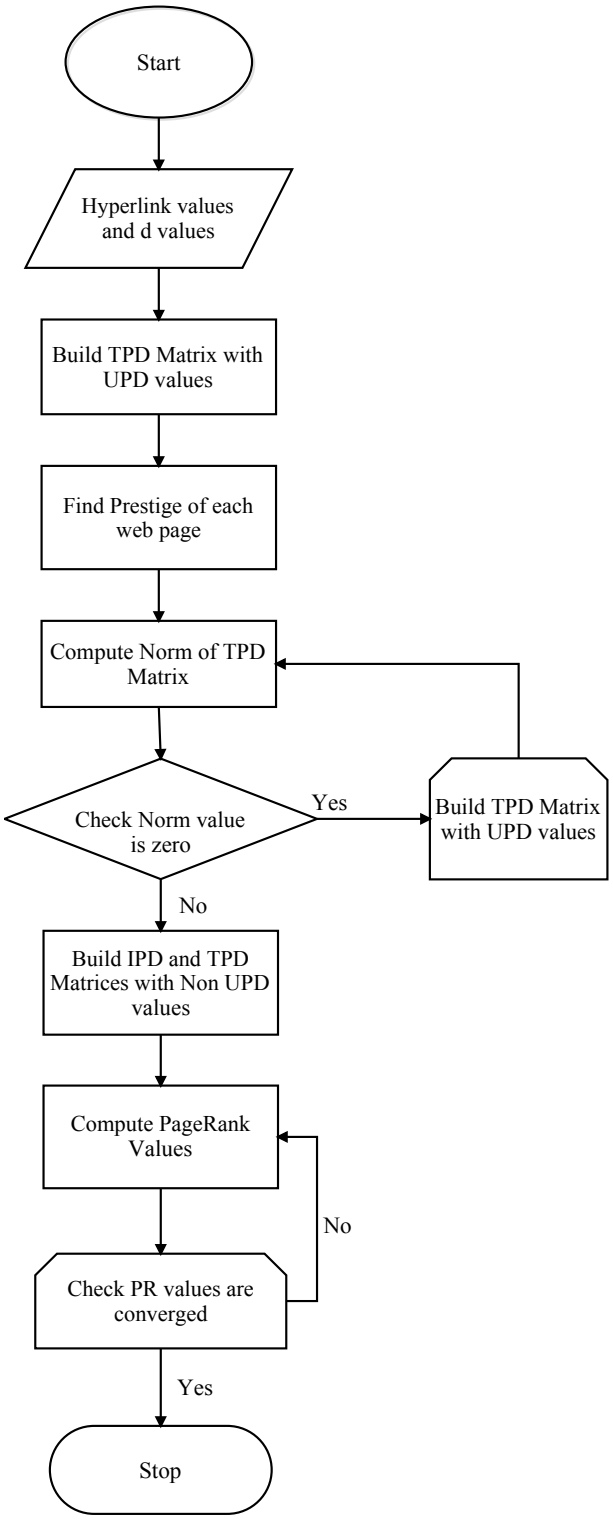


Figure 3.8: Flowchart of Proposed Prestige Score(PPS) algorithm [34]

METHODS	MAP	P@5	P@10
PageRank (improved)	0.134 (26%)	0.148 (32%)	0.11 (25%)
DirichletRank (improved)	0.140 (32%)	0.156 (39%)	0.116 (32%)

Figure 3.9: Dirichlet score versus PageRank score [35]

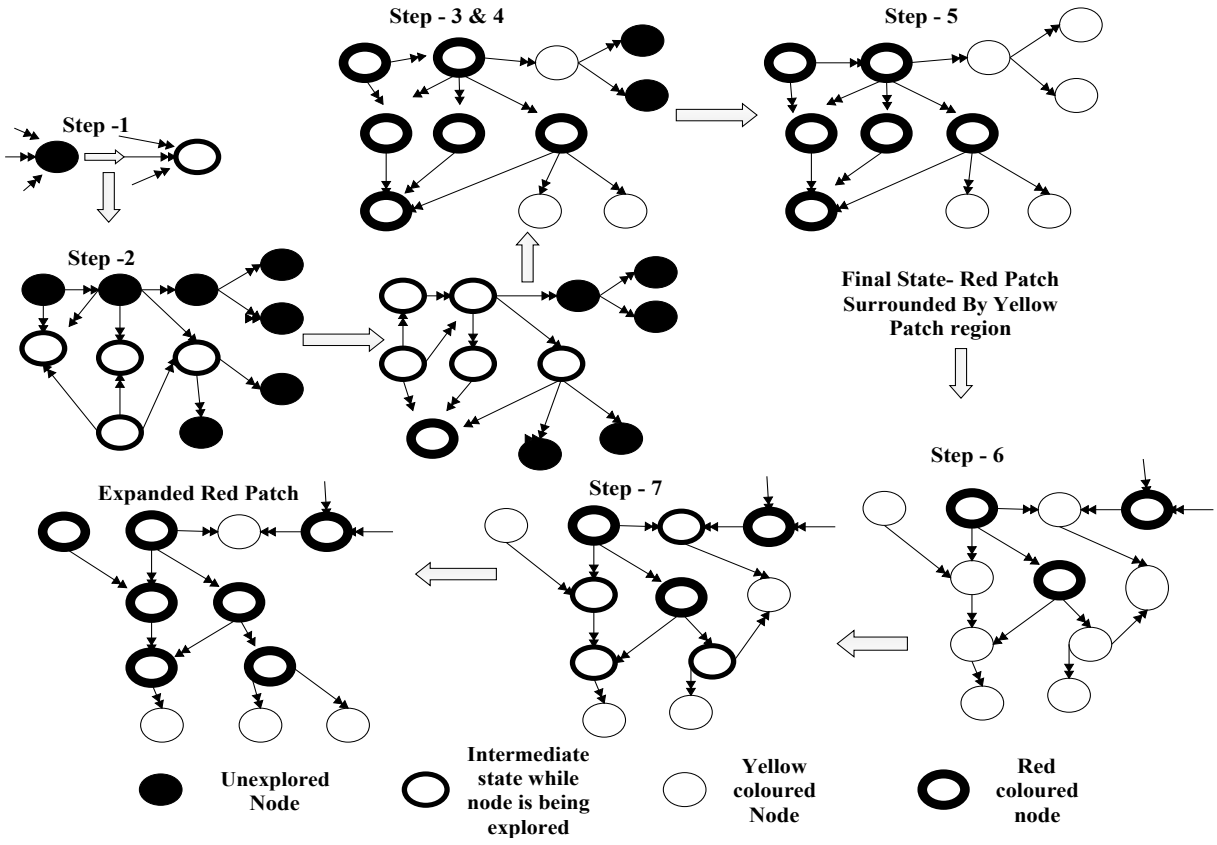


Figure 3.10: Divide and conquer steps [36]

nodes in the PageRank score [154]. Link Spamming is merely the result of the major problem of leakage known as zero-one gap problem. The PageRank of the page with no out-link and the PageRank of the page with one out-link, has a huge gap in their ranks. This has been given attention and solved by the updating PageRank algorithm by the new method known as *Dirichlet algorithm*. It is more stable and more resistant against link spamming. This algorithm works by following the procedure of visiting the out-links of the present page first. One more algorithm proposed simultaneously is Two-Stage Rank algorithm. It works to increase the convergence rate of power method. If artificial pages are created by spammer to increase the score of target page, then Dirichlet rank score is comparatively lower than PageRank as shown in Fig. 3.9. Thus, it needs more efforts by spammer, but can be trapped in case of high link farm [35].

Web graphs

PageRank computation not only depends upon the method used, but also on the structure of web graph. Handling the graph structure is done in various ways as discussed below.

- Power method for PageRank calculation uses graph decomposition methods only. It can be accelerated by using the partitioning schemes of hypergraph, i.e., one dimensional modeling and two dimensional modeling. These models were implemented to reduce the

Operation	Average Run Time (in ms)
Largest Red Patch PageRank	20
Yellow Patch Page PageRank	17
Divide and Conquer PageRank	38
Original PageRank	118
Divide and Conquer PageRank ran 3.1 times faster than Original PageRank	

Figure 3.11: Divide and conquer approach versus Naive Approach [36]

Number of Red Patches = 326		
	Number	**Percentage**
Total Edges in all Red Patches	33144	63.5 %
Total Edges in Yellow Patch	19039	36.5 %
Vertices in Red Patches	8699	71.25 %
Vertices in Yellow Patches	3510	28.75 %
Edges in Largest Red Patch	8839	16.94 %
Vertices in Largest Red Patch	552	4.5 %

Figure 3.12: Statistics of patch extraction [36]

communication load of each iteration [155].

- The same work has been extended by repartitioning the work load by processing the values on the different processors. The task is accomplished by the implementation of 1D and 2D compression schemes. The 2D scheme performed better load partitioning [156].
- The approach of distributing the web graph into domain of quasi-equivalent vertices and then computing the probability distribution of random walk was developed. It is the method of computing the PageRank scores with the formulation of intra-host and inter-host steps. It resulted in better approximation of PageRank scores [131].
- The PageRank issues are resolved by selecting the appropriate edges of graphs in polynomial time. This has been successfully proved by implementing the process of markov chain as stochastic shortest path problem. The same selection process is done with the greedy solution and named it as the PageRank iteration algorithm [157].
- To decrease the number of iterations of power method, treating the web graph as tree can be a solution. The tree is formed by core-tree decomposition and then applying the GMRES method on small tree width graph [158].
- VertexRank is the another algorithm used for assigning the weight to the vertices in the graph. In this, central vertex with the highest weight is placed first. All the other vertices are then assigned depending upon the depth of graph and number of edges connected. Thus, each vertex holds its position and is placed according to its coefficient. Finally, the position of each vertex is determined by the application of the classical force-directed algorithm [159].

- The another approach for dividing the graph into subgraph, is the Divide and Conquer approach. It follows the basic approach of PageRank that the vertices of one set does not depend upon the vertices of another set. The graph is divided into two patches, red and yellow. This approach solves the problem by reducing the size of graphs into subgraphs as shown in Fig. 3.10. It reduces the memory requirement. Thus, enabling the parallel processing of different approaches [36].

Thus, the problem of PagerRank calculation load has been solved by considering the properties and constraints of the web graph as shown in Fig. 3.11 and Fig. 3.12.

The hyperlink structure of the graph is not enough to measure the importance of a webpage. The importance of a web page may also 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's 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 [111]. Topic-sensitive PageRank computes more than one-score for each page. It highly depends upon the topic whereas in the original page rank only one score is computed. Topic-sensitive PageRank speeded the convergence rate by 25-300%. This algorithm performs context sensitive scoring as well [146].

The summarization of the updations for PageRank is presented in Table: 3.2. Here *D* refers to the Damping factor.

Table 3.1: PageRank algorithm analysis

Author	Method	Dataset	Evaluation Parameters	D	System Requirements	Execution Time	Advantages	Future_scope
Haveliwala, 2003 [146]	Multinomial Naive Bayes Classifier	120 Million pages, 280000 of 3 million URLs, 35 sample queries	OSim, KSim	0.75, 0.95	1533 Mhz AMD Athlon CPUs, 480 GB RAID-5, 2.5 GB of main memory	20 hours	Convergence Rate increased upto 25-300%	More sets of different topics
Kim and Lee, 2002 [32]	Improved PageRank Algorithm	10 million Korean Web Pages	Dampening Factor (d)	0.85	192 MB main memory , Operating system Accel Linux 6.1	Less than 30 minutes	Solved Norn-leak and RankSink Problem of PageRank	Web Crawling can enhance the Computation speed of PageRank
Eiron *et al.*, 2004 [149]	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
Lee *et al.*, 2003 [151]	Markov chain	451,237 webpages by Stanford WebBase Project in 2001	Dampening 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 [153]	Power Method	6012 Web Pages and 23875 hyperlinks	-	0.85	-	-	Reduces computational operations of PageRank vector	
Wang *et al.*, 2008 [35]	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	
Bradley *et al.*, 2005 [155]	Partitioning Tool Parkway2.1	India-2004, Stanford and Standford_Berkeley Web Graphs	GleZhu method	0.95	Beowulf Linux Cluster with 8 dual processor nodes. Each node has two Intel Pentium 4 3.0GHz processors and 2 GB of RAM.	Reduced upto 50% as compare to original PageRank	Reduces communication overhead per iteration	Faster parallel partitioning algorithms can be developed.
Broder *et al.*, 2006 [131]	Power Method on Host Graph	1446 million pages, 6650 million links, 31 million unique hosts, 241 million host-to-host edges	Number of passes over the entire graph	0.95	Alpha server with 4 CPUs, 32 Gigabytes memory	Reduced upto 50% as compare to original PageRank	Reduced number of passes over full links from 30 to 2	Different aggregates of web graph can be explored.
Maehara *et al.*, 2014 [158]	Naive GMRES method	Web Graphs and social networks: in-2004, it-2004, uk-2007-05, twitter-2010, web-BerkStan, web-Google, web-NotreDame, web-Standford, livejournal, flickr, orkut	Number of Iterations	0.85	Intel Xeon E5-2690, 2.90GHz CPU, 256GB memory , Ubuntu 12.04	2-4 times faster than PageRank algorithm	Number of iterations are 5 times less than Power Method	Several real networks can be explored via core-tree-decomposition
Dong *et al.*, 2015 [159]	VertexRank	Rome Graphs (with 10-100 vertices) USAir97, web-polblogs, bio-celegans, fb-messages	Number of iterations and Egde crossings		Windows 7, Intel Core i5-2400 at 3.10GHz, 2GB RAM	much lesser than classical method	more stable	Pruning algorithm for reducing the computation time
Liu *et al.*, 2010 [111]	BrowseRank	Three billion records, One Billion Unique URLs, 7500 queries	Precision, Mean Average Precision, Normalised Discount Cumulative Gain	0.95	-	-	Outperforms PageRank	This process can be used for detecting spam as well
Gu and Wang, 2013 [145]	Multiplicative Splitting Iteration (MSI)	Web-Standford, Amazon0505, Stanford-Berkeley	Beta1 , Beta2	0.95, 0.998	-	426.671s	Better than PageRank both in number of iterations and convergence rate	The evaluation parameters can be further optimised.
AVRAC HENKOV *et al.*, 2007 [136]	Monte Carlo Algorithm	INRIA Sophia Antipolis WebSite with 50000 Web Pages, 200000 hyperlinks	Convergence Rate	0.85	-	-	Removes the issue of update of PageRank Algorithm	Development of MC methods for other ranking algorithms such as HITS and SALSA.
Prabha and Vasantha, 2014 [34]	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	0.203ms	Improved quality of Search Engine Results	Can be used with other algorithms and Fuzzy Based Information System
Kamwar *et al.*,2003 [134]	Aitken Extrapolation Method	STANDARD.EDU link graph(280000 nodes, 3 million links, 12MB storage) , LARGEWEB Link graph (80M nodes, 3.6GB storage	convergence rate	0.99	AMD Athlon 1533MHz Machine, 6-way RAID-5 disk volume, 2GB main memory	38% less time than power method	Minimal overhead and cost effective	Kendalls residuals can be explored.
Kanwar *et al.*, 2003 [134]	Quadratic Extrapolation Method	STANDARD.EDU link graph(280000 nodes, 3 million links, 12MB storage) , LARGEWEB Link graph (80M nodes, 3.6GB storage	convergence rate	0.90, 0.95, 0.99	AMD Athlon 1533MHz Machine, 6-way RAID-5 disk volume, 2GB main memory	59% less time as compare to Power Method	Boosted the effectiveness of Power Method by eliminating the bottleNeck, the second and third eigenvector components of the current iteration of the power method than PageRank	Kendalls residuals can be explored.

Table 3.2: PageRank algorithm analysis

Author	Method	Dataset	Evaluation Parameters	D	System Requirements	Execution Time	Advantages	Future_scope
Kanwar *et al.*, 2004 [135]	Adaptive Method	STANDARD.EDU link graph (280000 nodes, 3 million links, 12MB storage), LARGEWEB Link graph (80M nodes, 3.6GB storage	convergence rate	0.85	AMD Athlon 1533MHz Machine, 6-way RAID-5 disk volume, 2GB main memory	18% less time as compare to Power Method	Cost effective method	
Arnal *et al.*, 2014 [143]	Parallel relaxed and extrapolated algorithm	it-2004, webbase-2001, uk-2007-05	Efficiency of Algorithm	0.85	HPC cluster of 26 nodes HP Proliant SL390s G7, 52 Intel XEON X5660 hexa-core at upto 2.8 GHz and 12MB cache per processor, 48GB RAM, CentOS LINUX 5.6 for x86 64 bit	-	Faster than PageRank	Speed up the convergence rate and time.
Liu *et al.*, 2015 [144]	Sampling Methods (DSPI and ASPI)	Wiki-Vote, Soc-Epinions, Slashdot-0902, Web-Stanford, Web-Google, Web-BerkStan	running time, memory usage, number of iterations till convergence	0.85	Intel Xeon x86-64 2.6GHz CPU, 32GB RAM	Wiki-Vote (DSPI:0.0379s, ASPI:0.0378s), Soc-Epinions (DSPI:0.0567s, ASPI:0.0539s), Slashdot-0902 (DSPI:0.2161s, ASPI:0.2056s), Web-Stanford (DSPI:1.0050s, ASPI:0.9950s), Web-Google (DSPI:47.8129s, ASPI:25.7321s), Web-BerkStan (DSPI:13.4836s, ASPI:11.8073s)	More accurate than other PageRank approximation methods	Computational efficiency can be improved.
Langville And Meyer, 2006 [147]	Reordering PageRank Algorithm	EPA.dat, CA.dat, NCSU.dat, ND.dat, SU450k.dat	Number of iterations and time	0.9	867 MHz Mac G4 with 1.5 GB RAM	EPA.dat-0.59s, CA.dat-1.22s, NCSU.dat-7.65s, ND.dat-130.54s, SU450k.dat-52.84s	Factor of nearly 6 speedup on one dataset	New PageRank acceleration techniques can be formulated with this technique other than Power Method.
Wu and Wei, 2010 [142]	Arnoldi-Extrapolation Algorithm	Stanford-Berkeley Web Matrix , 683446 pages , 7.6 million links	Computation time	0.85, 0.9, 0.95, 0.99	1.6 GZ dual core Intel(R) Pentium(R) processor, 1GB main memory	25.6s(d=0.85), 37.2s(d=0.9), 71.9s(d=0.95), 205.6s(d=0.99)	Improved Arnoldi-type Algorithm	Convergence of Arnoldi-type algorithm is research issue, optimal parameters of this algorithm.
Yu *et al.*, 2012 [33]	Dev-Lump 5	Wikipedia-20060925, Wikipedia-20070206, indochina-2004, Wb-edu, uk-2002	CPU Time ,size of Iteration Matrix	0.85, 0.90, 0.95, 0.99	Dell Workstation with four core Intel(R) Pentium(R) processor with CPU 3.2 GHz and Ram 16 GB, WINDOWS XP 64 bit operating system	Comparatively requires very less time as compare to Previous lumping algorithm	Dev-Lump 5 is more feasible and cheaper	MapReduce Infrastructure can be collaborated with this technique.
Cevahir *et al.*, 2011 [156]	Repartitioning models for Parallel PageRank and 1D, 2D compression schemes	Google, balkan, de-fr, webbase, indochina, arabic-2005, uk-2005, uk-union	Processing time	0.90	40 single-core Intel P4 3GHz processor with 64 Gbytes of RAM, AMD Opteron Dual Core 2.4 GHz processors.	642.4 seconds on 16 processors	Reduced preprocessing and communication overhead	Parallelization of LAW codes .
Ishii and Tempo, 2010 [132]	Distributed Algorithm for PageRank Computation	1000 webpages	mean-square convergence rate	0.99	-	-	reduced Communication overhead	Improvement of convergence rate with respect to Distributed algorithms.
Ishii *et al.*, 2012 [133]	Web Aggregation for Distributed algorithms	200 pages , 12 dense diagonal blocks with 5 to 30 pages each.	error bounds	0.5	-	-	reduced Communication overhead	Improvement of convergence rate with respect to Web Aggregation for Distributed algorithms.
Del *et al.*, 2007 [29]	Krylov subspace methods	1120 matrices	convergence rate	0.85	-	-	Restarted GMRES and preconditioned BiCGStab improves the rate of convergence	
Desikan *et al.*, 2006 [36]	Divide and Conquer approach for computing PageRank	Link graph from http://www.cs.umn.edu, 52183 edges, 12209 vertices	Runtime	-	-	3.1 times faster than original PageRank	efficient PageRank computation	study of efficient computation for the changing web .
Kin and Choi, 2015 [30]	IPII (Incremental PageRank Algorithm + Incremental Iteration method)	Hanyang university web site	Correlation coefficient	-	-	-	Reduction of Computational cost by IIPI	Explore more web pages.
Bu and Huang, 2013 [148]	Adaptive Reordered Method	wb-cs-stanford.mat , Standford.mat	Running time	0.85	MATLAB	Less than Reordering method	Fast Computational Speed due to efficient Stoping criteria	Other Acceleration methods other than Jacobi can be implemented.

3.1 Proposed work

Above all the issues of search engine's ranking methodology as discussed above, there is one major issue which effects the other issues as well, i.e., dangling pages.
Once the dangling pages are handled carefully while computing the rank scores of web pages, it would automatically resolve the other issues. Like convergence rate would be much better, computational cost would be reduced, and the web spam would also be prevented. This concern is focused in this proposal. This thesis presents two schemes for handling dangling pages, which automatically detects web spam.

1. Proposed scheme by analyzing user behavior: In this scheme, the user behavior is analyzed by two attributes, i.e., dwell time and click count. The PageRank algorithm is updated using these parameters and the system is trained using learning automata (LA). With the environment sensing capability of LA, the targeted dangling are punished. This scheme is detailed in chapter 4.

2. Proposed framework by detecting web spam: In the proposed cognitive framework, the ability to detect web spam automatically by search engine, is introduced. In this, the PageRank formula is modified by detecting malicious web pages. It is concluded that the malicious web pages are maximum of dangling pages. So, this framework handles the dangling pages efficiently. The framework is discussed in chapter 5 in detail.

3.1.1 Optimization

The decision to propose AI enabled system or some intelligent system is an emerging field as it does not only require choosing a machine learning or deep learning model, but analyzing its relevancy. The proposed system have also been designed keeping in view about the insights of the problem and addresses how this black box issue is solved. We have deeply analyzed the features of image, by extracting them manually. More complicated, but also potentially more powerful algorithms such as neural networks, ensemble methods including random forests, and other similar algorithms sacrifice transparency and explainability for power, performance, and accuracy. The corrective measures have been taken to avoid the over-fitting and under-fitting, i.e., by considering two sets (training and testing) and by adopting genetic algorithm. Genetic algorithm tries to search the neighborhood for the initial solutions that you have by heuristics method to get a best or optimal solution for the problem by search this solution search space. Also, crossover and mutation operators on genetic algorithm guarantee that you can improve the initial solutions to get local and global optima solutions.

Chapter 4

User Behavior Analysis

4.1 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. A 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 the 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 of the Internet by using various chat rooms and blogs for chatting. MySpace was the first social media website, launched in the early 2000s. It was the platform for making a profile and connecting with friends through it. It worked as a platform for websites like Facebook. Musicians like Colbie Caillat mostly used myspace for promoting their tracks. Then came the next boom for social networks, LinkedIn. LinkedIn was meant for professionals and businessman and is popular today also. For connecting people irrespective of their profession, Mark Zuckerberg came with 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, 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. 4.1. It demonstrates the yearly adaptations in social network zone.

User participation on the Internet is increasing at a rapid pace. It is reported from Internetlivestats [41] (an Internet survey company), that there are currently 3.4 billion Internet users in the world. These users access the Internet in different ways. Most frequent is through search

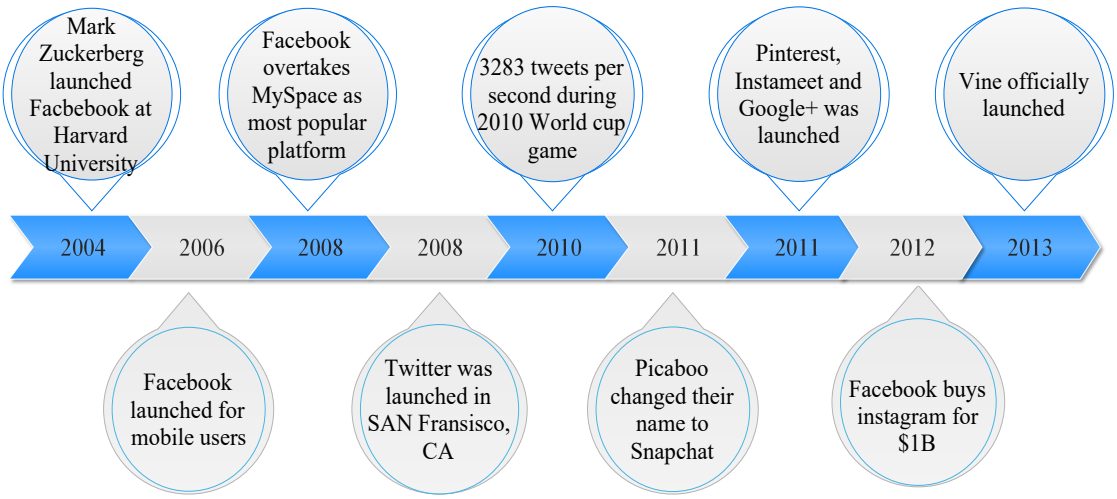


Figure 4.1: Social media evolution

engines. According to Netmarketshare's latest statistics [44], 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 [40], it has been observed that there are currently 1,003,887,790 web sites in the World Wide Web (WWW). All 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. 4.2(a). It can be judged by the survey, as shown in Fig. 4.2(b) that social media users are increasing rapidly.

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

Techniques are [5,27,28,103,104,154] by which the person known as spammer try to boost the rank of its website. These spam websites are complicated to detect. Sometimes, searching for authoritative sites consumes a lot of cost and time. The reason is that spam websites are boosted by gaining high ranks. Users who indulge in Internet surfing activities always expect valuable SERP. Since 1997, Google adopted the PageRank methodology for ranking websites and, it is updated periodically. PageRank is highly dependent on the number of in-links.

4.1.1 Energy management for web page ranking

The web is the collection of different communities(set of web pages) consisting of various topics. A community is connected to other pages on the web with the help of outgoing links. There are pages which do not connect outside the community and not even to other pages.

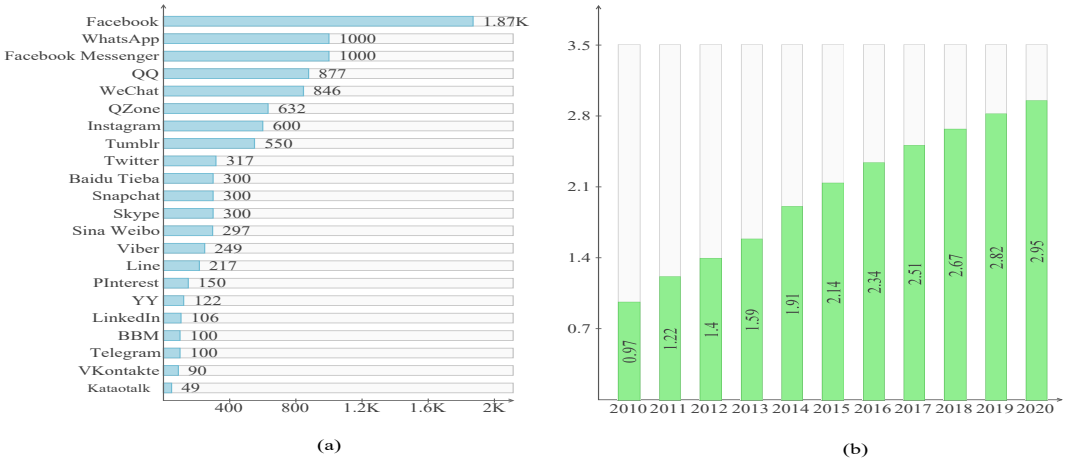


Figure 4.2: (a) Social network sites worldwide ranked by number of active users (in millions, as of January,2017) (b)Number of social media users worldwide from 2010 to 2020 (in billions)

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 [160] [161] [162]. The community's energy is dependent upon four components [47]:

$$E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp} \tag{4.1}$$

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 proves the loss of energy due to dangling pages. These equations can compute energies:

$$E_I^{in} = \frac{d}{1-d} \sum_{i \in in(I)} f_i x_i^* \tag{4.2}$$

$$E_I^{out} = \frac{d}{1-d} \sum_{i \in out(I)} (1-f_i) x_i^* \tag{4.3}$$

$$E_I^{dp} = \frac{d}{1-d} \sum_{i \in dp(I)} x_i^* \tag{4.4}$$

Where f_i is the fraction of hyperlinks of page i to the pages in I, concerning total outgoing links from page i. The number of pages going from the community and the number of dangling pages, both leads to the loss of energy. Dangling pages should be handled appropriately. Many authors [150], [151], [152], [153] 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 the user's behavior as discussed in Section 4 by the proposed scheme algorithm.

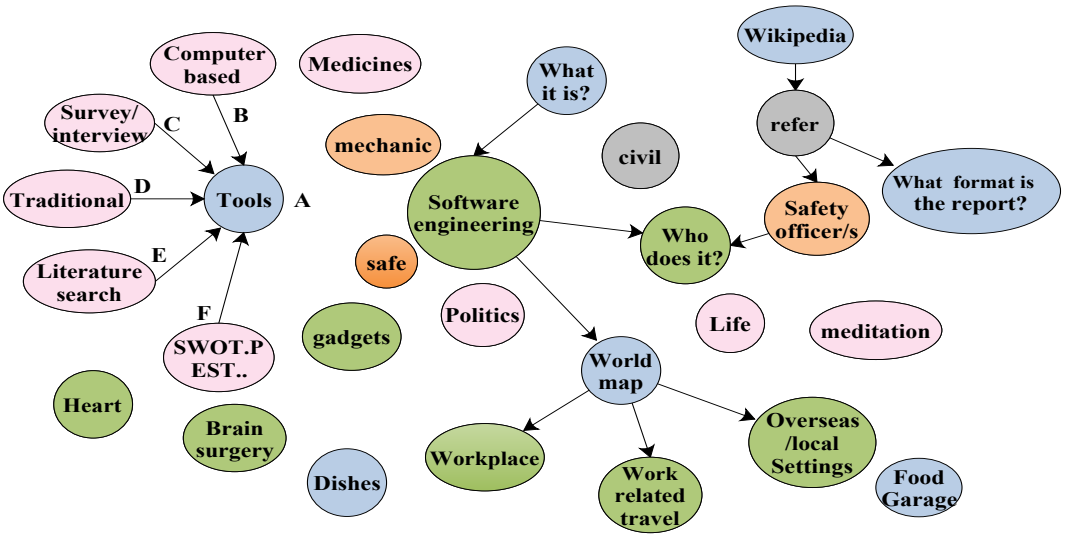


Figure 4.3: Web structure

Also, the reduced computational cost by the efficient ranking of web pages saves energy. It can be easily understood by Fig. 4.3, a sample of the dataset, there can be dangling pages existing in an 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.

4.2 Contributions

This Section presents our contributions to revising the SERP's. The 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 has been used which accept response as input from the environment and perform the action as output to the environment.
- The scheme has been validated with Microsoft Learning to Rank dataset.

4.3 System model

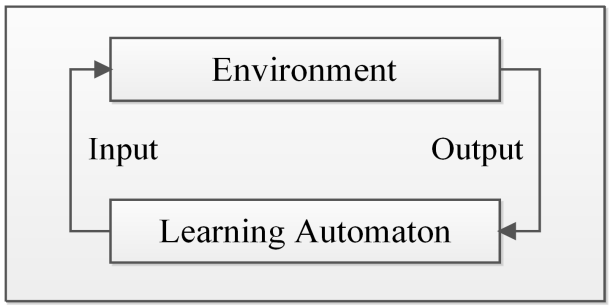


Figure 4.4: The working environment of learning automata

The approach used for our proposed system is discussed in this Section. It covers three terminologies, as shown in Fig. 4.4, LA, environment and input/output (action probability vector). In this, the LA system learns from the environment and accordingly update the action probability vector. These concepts are discussed below:

4.3.1 Learning Automaton

LA is a system having the potential to learn from the environment for taking action. This learning ability enhance the chances of updating the vector with the appropriate action. The action may be a reward or a penalty; it depends upon the environment. This system results with an optimal solution with a minimum number of penalties [163]. Structure of learning automaton is drawn by β, and α, where β is a set of output from the environment and $alpha$ is a set of actions. In β, 0 represents a reward, and 1 represents a penalty. With every learning from the environment, the action is taken. The action taken is random, which is based on probability distribution. The effect is then served to the environment as an input. The environment then decides by generating a reinforcement signal. Based on 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 the 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.

4.3.2 Environment

The environment in the region where the LA learn and react. The reaction may be positive or negative; it depends upon the output from the environment. The 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 until the optimal solution is achieved. The environment may be stochastic or probabilistic. In our proposed system, it is probabilistic.

4.3.3 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 [164], [165]. 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 [166], [167], [168]. The formulas for updating the probability are:

$$p_j(n+1) = (1-a)p_j(n), j \neq i, Y = 0 \tag{4.5}$$

$$p_j(n+1) = ap_j(n), j = i, Y = 0 \tag{4.6}$$

$$p_j(n+1) = p_j(n), Y = 0 \tag{4.7}$$

where a is the learning parameter.

4.4 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. 4.5.

The Algorithm 2 describes the proposed scheme, computes energy efficiency at the beginning and the end of the algorithm. It results in handling dangling pages. Following are the steps performed:

1. The energy flow within the web is computed in the step 2. $|I|$= 241527, $E_I^{in} = 0$, $E_I^{out} = 0$, (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)*12211.
   E_I= 241527 + (d/1-d)*12211.

2. The steps 3 to 6 computes the PageRank by using Eq. 1.

3. The steps 7 to 11 discards the pages other than dangling pages by the constraint of outlinks, the dangling pages have zero outlinks.

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.

procedure FUNCTION(PageRank)
  $E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp}$ ▷ Energy flow within web
  for $i = 1$ *to* n **do**
    Compute $PR(n)$ ▷ vector of PageRank values by using Eq. (1)
    Set $i \leftarrow i + 1$
  end for
  for $PR(n) = 1$ *to* n **do**
    if Outlink != 0 **then**
      Discard Node
    end if
  end for
  for $i = 1$ *to* n **do**
    Compute Revised PageRank $PR(n) = PR(n).D_t + C_c$ ▷ Using Dwell time and Click count
    Set $i \leftarrow i + 1$
  end for
  ▷ Entering LA scheme
Input : $PR(n)$
Output : 0 (reward), 1(penalty)
  $\forall P_i(n) = 0$ ▷ Initializing the population
  if 0 **then**
    Desirable response
    $P_i(n) = P_i(n) + P_\sigma$
    $\forall m \quad m \neq i \quad P_i(n) = (1 - P_i(n)) + P_\sigma$
  end if
  if 1 **then**
    Undesirable response
    $P_i(n) = P_i(n) - P_\sigma$
    $\forall m \quad m \neq i \quad P_i(n) = (1 - P_i(n)) - P_\sigma$
  end if
  $E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp}$ ▷ Energy flow after handling dangling pages
end procedure

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= 241527 + (d/d-1)*(12211-3403)

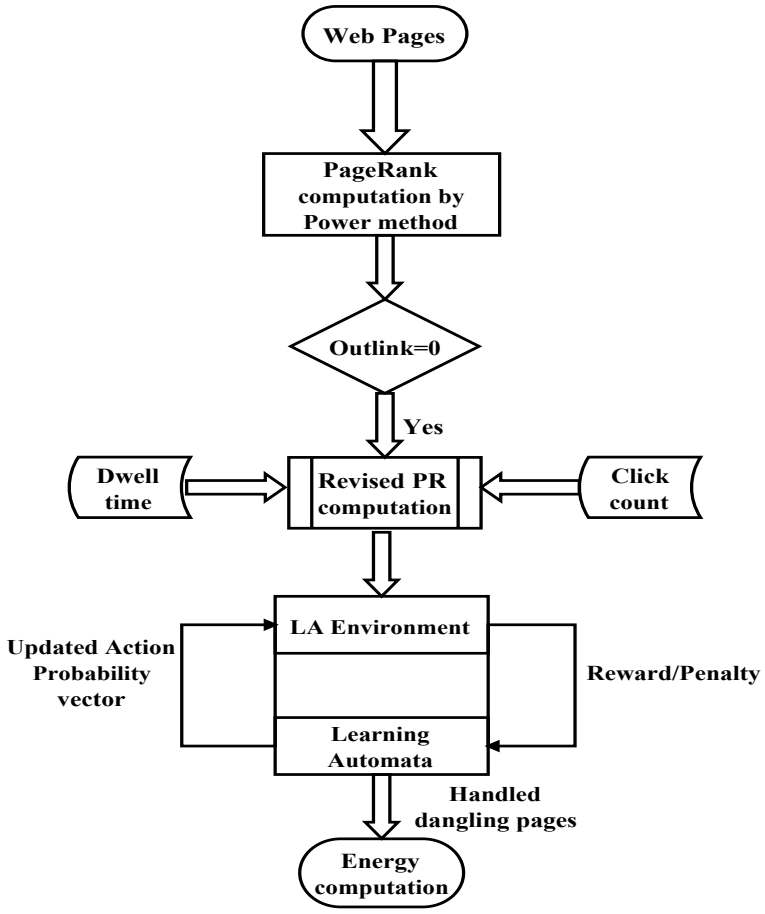


Figure 4.5: Proposed scheme

4.4.1 Initialize the population

As LA has 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.

4.4.2 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 the webpage. The importance of the webpage is computed by considering three factors: the 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 the Power method, is taken. Then, the 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.

4.4.3 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 the learning algorithm. Environment generates a reinforcement signal in which either a reward is granted, or a penalty is forced. According to the corresponding action, the probability vector is updated.

(a) Desirable response

$$P_i(n) = P_i(n) + P_\sigma \tag{4.8}$$

$$\forall m \quad m \neq i \quad P_i(n) = (1 - P_i(n)) + P_\sigma \tag{4.9}$$

In the equation 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 the equation 9 is executed, is selected randomly. Simultaneously, the rest of the query-url pair are updated with equation 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 equation 9, (1-p(i)) is updated by equation 10.

(a) Undesirable response

$$P_i(n) = P_i(n) - P_\sigma \tag{4.10}$$

$$\forall m \quad m \neq i \quad P_i(n) = (1 - P_i(n)) - P_\sigma \tag{4.11}$$

Penalty signal being received from the environment, Equation 11 and equation 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 equation 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.

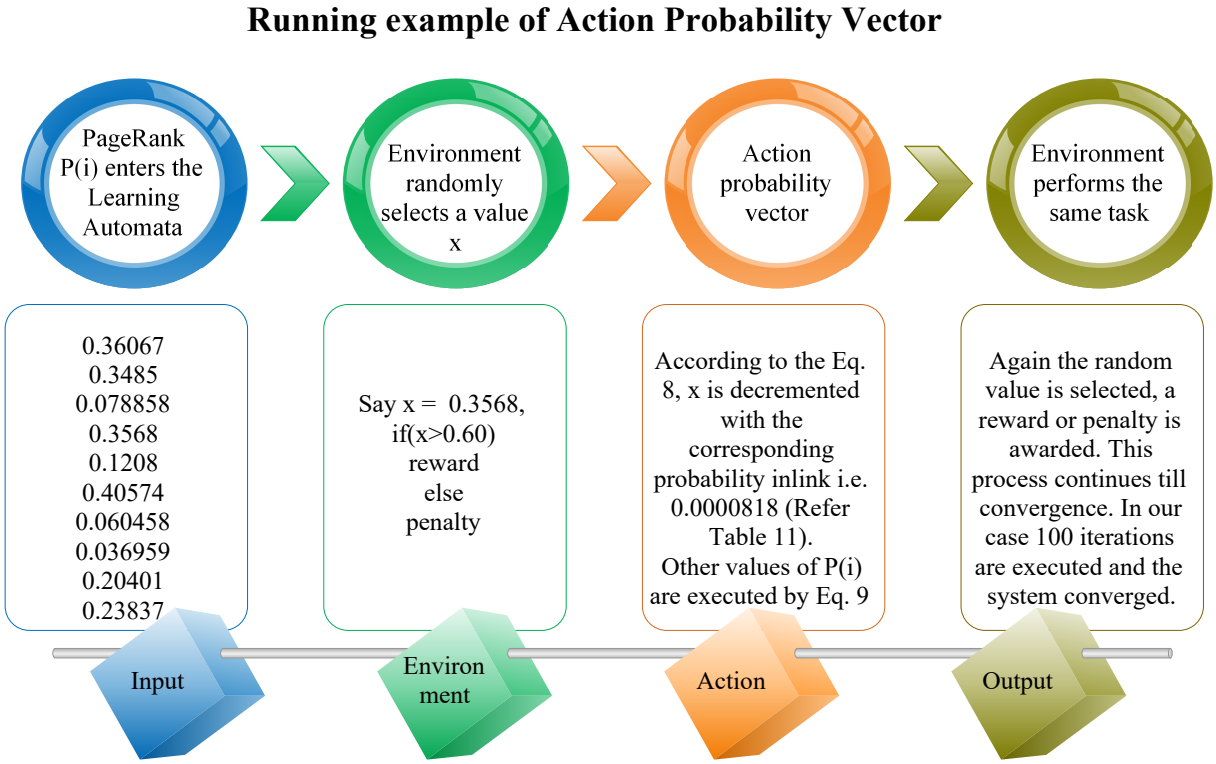


Figure 4.6: Working of action probability update vector

The working of the action probability update vector has been illustrated in Fig. 4.6. In this, we have taken the PageRank score of the first ten URL pairs. Random selection of the target URL pair is made by the environment, headed by reward or penalty. According to the value is updated along with updating the rest of the vector. Finally, system convergence is achieved after multiple iterations, which leads to an optimal solution. The results are discussed in Chapter 6.

Chapter 5

Updated PageRank algorithm for web spam detection

5.1 Introduction

In the era of computing, the information is stored, processed, and presented in a seamless and efficient form with the help of the Internet. A lot of brainstorming is going on by the researchers to bring innovations to improve the technology of the Internet further. Design of the Internet can be visualized as a network of interconnected computing devices for the exchange of information. Internet was originally started as a pilot project by the U.S department of defence. It was practically implemented with the support of ARPANET. Since the development of ARPANET (single network for communication) in the 1960s, the Internet underwent many changes. Later in the 1970s, the protocol TCP/IP (multiple networks for communication) was launched, which pushed the Internet to be used commercially. Another breakthrough for the Internet happened with the evolution by the World Wide Web. From the technical trends, it can be said that the Internet has gone through four different phases. These phases are popularly named as the Internet of Network (IoN), Internet of Computers (IoC), Internet of People (IoP), Internet of Things (IoT).

The first phase of the Internet is IoN, which started with the development of ARPANET. It enabled communication between two computers. The second phase, i.e., IOC originated with TCP/IP, which extended the network connectivity among more than two computers. IoP, as a third phase, established with the evolution of mobile phones, which led to the social connectivity amongst people. The fourth phase of the Internet, IoT, has emerged through communication among multiple objects with minimum human invention [169].

5.1.1 Internet of Things

Internet of things (IoT) is the term originated by Kevin Ashton in 1999 [170]. The strength of IoT is unquestionable and can be easily felt in routine activities. IoT plays a crucial role,

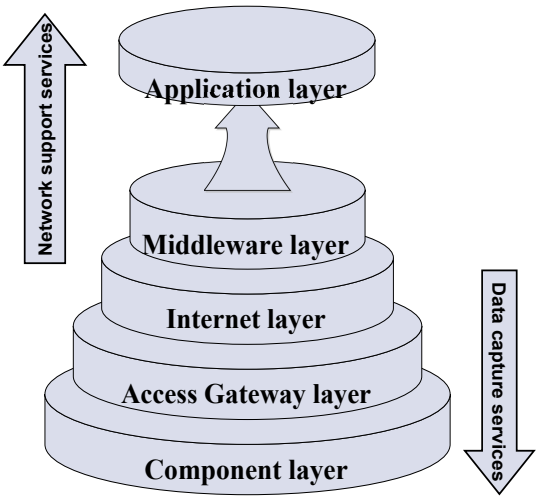


Figure 5.1: IoT architecture

from a simple task of switching on the light to a complex task of accessing the server from the remote area. Behavior of users towards IoT devices is noticeable in various fields like smart homes, healthy working environment, e-health, e-commerce, smart objects, and many more. 'IoT' can be understood by the terms 'Internet' and 'things'. It is generally defined as 'the wide variety of connected devices dependent upon protocols for unique identification and exchange of information'. IoT structure consists of five layers in which the Internet serves as the medium for data fetching from the component layer to the application layer [171]. The architecture of IoT is presented in Fig. 5.1 and the functionality of each layer is briefly elaborated below.

1. Component layer: The data sensors are deployed in this layer and are displayed in Fig. 5.2. These sensors are comprised of physical objects like RFID tags, sensor networks, and communication devices.

2. Access gateway layer: This is the layer in which data starts getting processed. The actions like message routing, encrypting, and publishing is performed at this level.

3. Internet layer: This layer consists of various protocols and is responsible for the transmission of data from one network to another.

4. Middle-ware layer: This layer acts as an interface between the application layer and the hardware layer. It is responsible for most critical functions of ensuring data integrity and data issues like semantic analysis and information discovery.

5. Application layer: This is the uppermost layer responsible for providing services to the users. The type of facility availed from this layer depends upon the type of industry, such as health, education, or manufacturing.

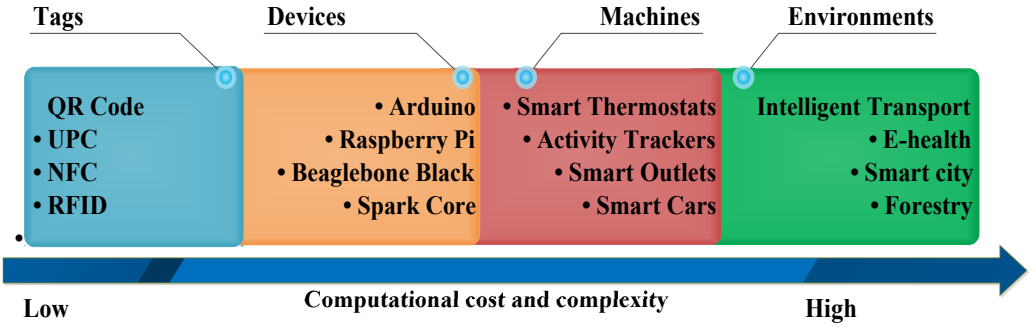


Figure 5.2: IoT Component Layer Devices

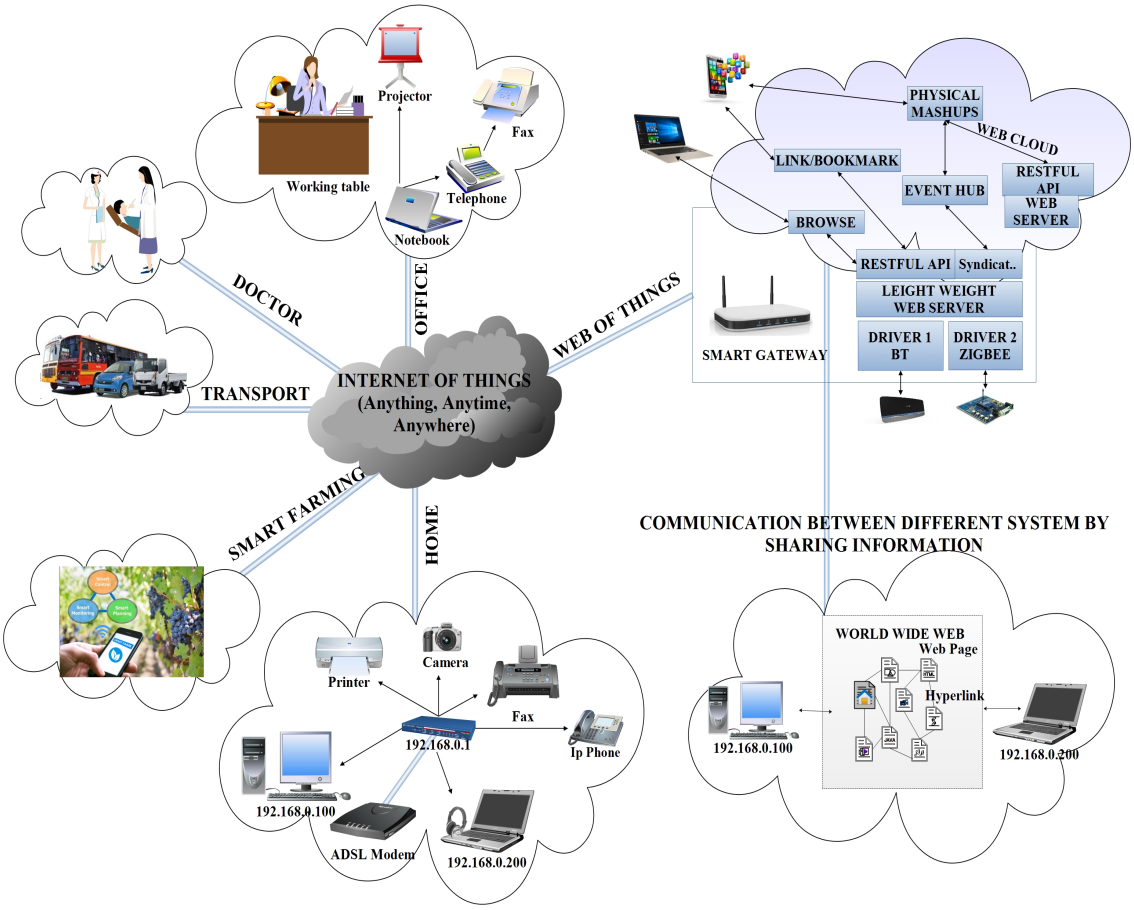


Figure 5.3: Web with framework of IoT

5.1.2 Cognitive Internet of Things

The new dimension of IoT is the Cognitive Internet of Things (CIoT), which gives the IoT objects, an ability to think, learn, understand, and react [172]. It is the process of integrating human cognition power into the system design. In CIoT, the IoT objects behave as agents and learn from the physical environment. It reduces human invention at a greater extent. IoT has also been defined as an intelligent physical-cyber-social (iPCS) by acting as a bridge between the physical world and the social world. The reasoning is applied in CIoT by analyzing the data

for inferring useful information required for concluding a decision. The recent methodologies adopted the technology of CIoT, which are discussed as follows.

- Cognitive management framework was proposed to empower the IoT in smart cities for collaborative objects by treating the real world as a virtual world. The framework uses cognition power for selecting the most relevant objects. A real case study was also presented to validate the framework more efficiently [173].
- CIoT was adopted to make the applications work intelligently, by learning through the actions, performing active decision making and finally responding to the environment with adaptive reaction [174].
- Another advancement in CIoT is Cognitive Radio Networks (CRN), which was first coined in 1999 by Joseph Mitola [175]. CRN was re-coined by Simon Haykin by reframing it with dynamic spectrum utilization management in 2005 [176].
- Website Preference Scale (WSPS) was developed by learning market strategies, using CIoT technology. The idea behind WSPS is to find the most preferred content elements for a website. It helped the website developers to achieve a higher number of visits [177].
- Cognitive M2M communication network was developed with a perspective of the protocol stack. It was implemented with the help of cognitive radio technology [178].
- Technological heterogeneity and complexity of IoT architecture was reduced by managing the IoT objects with cognitive management framework. The framework targeted to improve the energy-efficiency by selecting the most suitable objects from the set of diversified ones [179].
- real objects and virtual objects are responsible for providing IoT services. These objects were packed and framed with constraints to predict future potentials. The framework was implemented with the help of reasoning and machine learning paradigms [180]. Another proposal uses a machine learning model, neural network to forecast the future market trends [181].

5.1.3 Web of Things

As IoT becomes an active research area, different methods from various fields have been analyzed to adopt IoT technology. One dimension is considering IoT as Web of Things (WoT) [182]. In WoT, information sharing and device inter-operation are supported by the open Web standards. The need for web standards is taking a new edge with the increasing usage of mobile phones and laptops. This advancement of IoT has given rise to malware practices, which can be detected using the technique such as Analysis Evasion Malware Sandbox (AEMS) [183]. WoT, along with IoT, is presented in Fig. 5.3. This figure shows the ways in which the applications

such as health, farming, transport, and office types of equipment are dependent upon IoT. The paradigm of IoT, i.e., WoT, is presented, in which the *RESTFUL API* eliminates the need for protocols, and it can directly access the web. WoT has successfully simplified the process of information retrieval by different systems from the World Wide Web (WWW).

WWW is the medium for distributing data & information, and delivering services & products. Search engines are the platform for accessing the web efficiently. As the Internet consists of a large amount of information, so answering a user's query with a small set of result pages is a challenging job for search engines. After finding the relevant results, the filtered pages should also be ranked by the search engines. The search engines use a ranking algorithm for assigning a rank to the web pages. On the other end, there are enterprises like Airtel, BSNL, providing services to the people for accessing information through search engines at a nominal cost. With this cost-effective facility, people do not bother much to find authoritative websites. They usually click and open the websites present on the first page of the search results. The website developers being aware of this habit, try to take advantage by using various techniques such as animation to attract users for visiting their websites. However, embedding animation is ethical; whereas any attempt to forcefully push the site in the top result page is unethical. These kinds of unfair practices are followed by the spammers (the people who try to disturb the ranking methodology). For the prevention of spam websites, search engines keep on updating the ranking algorithms. The literature review states that the HITS [55], SALSA [56], DistanceRank [57] and PageRank [46] are the most successful ranking algorithms. According to Netmarketshare's latest statistics [184], the largest market share of search engine usage is achieved by Google (74.54%) Followed by Baidu(10.44%), Bing(7.92%), and Yahoo(5.53%). Google uses the PageRank algorithm for ranking the web pages. The proposed web spam detection technique in this proposal is for the PageRank.

Web spam

It is an unfair practice of altering the ranking methodology of search engines for getting on the top of search result pages. Content writers and website developers contribute to the successful implementation of spamming techniques. Such practitioners are known as spammers. The alternate definition of a webspam is "the unethical measures taken to improve the results of a search engine in favour of a considerable website". Website features, both link-based and content-based, are used for the formulation of spam. Sometimes, not getting the authoritative website in search results depict the link manipulation. There are various techniques meant for intruding spam, some of those are described below. The authors of [2, 10, 117] have attempted to detect the webspam. There are various categories of webspam [185] which depend upon the features used to inject the spam into the web pages. Three feature groups [16] were built for the detection of webspam, which is presented in Fig. 5.4.

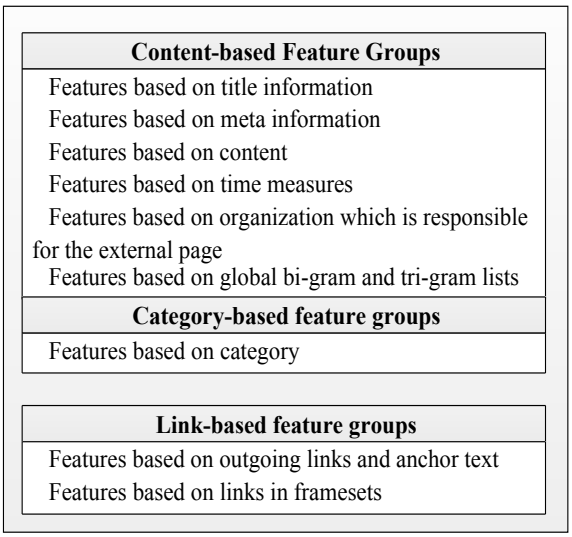


Figure 5.4: The feature groups used to build spam detection classifier [16]

Link spam

A search engine such as Google uses the ranking algorithm, PageRank, which depends on the link structure of the web. Search results by the PageRank are ranked according to the importance of web pages. The PageRank score of a web page depends on three measurements: total number of web pages in the web, the number of in-links to the web page, and the number of out-links of those in-links [47]. Spammers try to manipulate the link structure of the web for many reasons such as to increase the visit counts of the website, for financial gain, for competing with other sites, etc. The spammers create link farms, and the in-links of the websites are artificially increased. This kind of spam is known as link spam. Many authors [89, 92] have tried to detect the link spam. Moreover, the PageRank algorithm has also been updated periodically to identify the link spam.

Content spam

It refers to manipulating the contents of the web page to achieve a better ranking by the search engine. Keyword stuffing is one of its examples in which various keywords are inserted at different levels of the web page. It is a very beneficial strategy when the user query is keyword-based. This type of spam can be easily detected by automatic classifiers [25]. The features mostly used by the classifiers are number of words in the page, number of words in the title, average word length, a fraction of visible text, compression rate, and a fraction of anchor text. Authors [17, 98] have successfully detected content spam.

5.2 Contributions

This section describes our contributions to the proposed work. PageRank algorithm has been updated by considering the following remarks:

- A cognitive spammer framework is proposed which alters the computation of PageRank score by identifying spam nodes.
- 'Split by Over-sampling and Train by Under-fitting' (SOTU) approach has been introduced for balancing the class in dataset.
- The proposed framework has been validated by machine learning models. The best three models are ensemble followed by ten-fold cross-validation scheme.

5.3 Problem formulation

Search engines maintain the web pages periodically with the latest modifications such as addition of web sites, change in the content of a web page, etc. Crawler, a module of search engine is responsible for performing these modification tasks. Whenever, the web pages are added/modified in the web repository, it is the responsibility of ranking module in search engines to rank them. Cognitive ability helps the search engines to work more intelligently. PageRank is one of the ranking methodologies used by Google. PageRank considers in-links as an important measure while computing the rank score of the web pages. In-links considered by PageRank may be direct or indirect. Indirect links may be malicious because of the unknown origin or are created intentionally to promote the web page by gaining high rank.

The idea is generated for the demotion and detection of spam pages by ignoring the indirect malicious links. By neglecting these links, the PageRank score is undervalued as stated by Eq. 5.1.

$$P_i(PR) - P_i\alpha = \beta \tag{5.1}$$

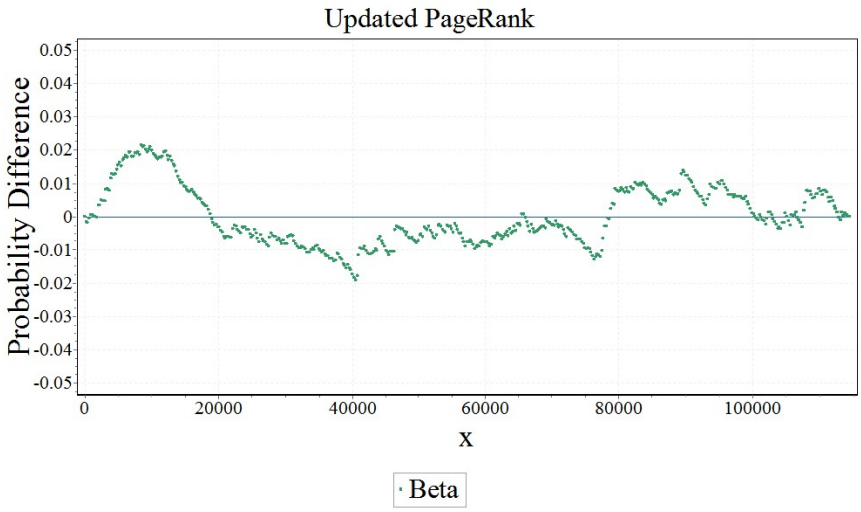


Figure 5.5: Distribution of probability difference for Updated PageRank

The probability difference states the occurrence of spam pages. It is defined more clearly with the help of probability difference graph in Fig. 5.5. It is the plot of the variation of empirical CDF from theoretical of a variable(here, it is the value of PageRank). Thus, the objective function can be expressed as a continuous curve by the Eq. 5.2.

$$Diff(x) = F_n(x) - F(x) \tag{5.2}$$

5.4 Proposed scheme

The linkage information depicts the neighborhood of a page. The proposed work focuses on to find if the neighbor pages are artificially created to boost the PageRank of the target page. Firstly, the challenge is how to deal with statistics to find the neighbors of the pages, or we can call them as supporters. Secondly, what to do with the information gathered, how to declare them as spam pages or different ways to punish them. The proposed scheme has been implemented with the following steps:

5.4.1 Supporters

Web pages are connected with each other through links. The link depicts the trust of one page on another. If one page has a hyper-link/direction to another page, it is called as referring/recommending. This is how the trust among web pages propagates. Say, if page x is recommending page y and page y is recommending page z, then, page x is directly connected to page y and indirectly connected to page z. All the recommendations that are referring to a web page are known as its supporters. It depicts the importance of neighbors in contributing to the PageRank score. In other words, we can say that the PageRank of a node completely depen-

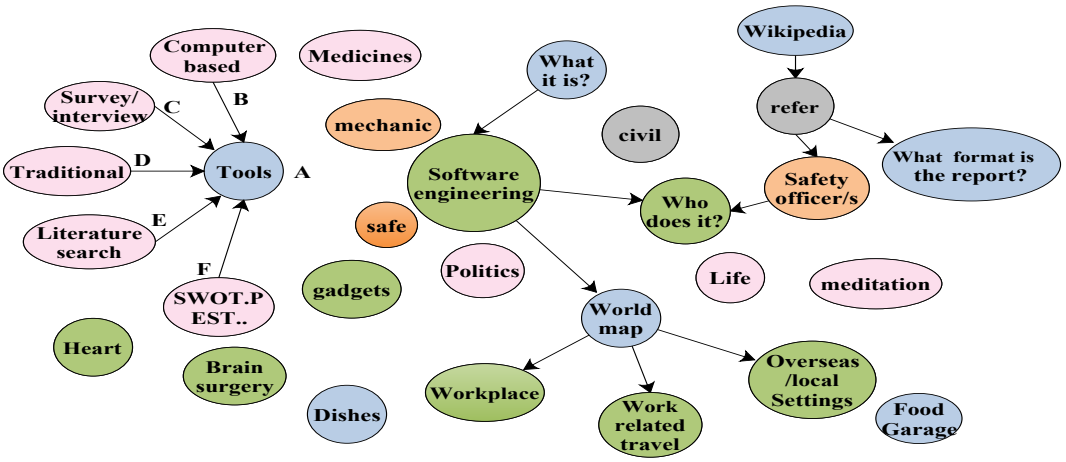


Figure 5.6: Web diagram [37]

dents on the PageRank of its supporters. The Eq 5.3 calculates the PageRank score calculation with the help of supporters, which is defined by Brin and Page [46].

$$PR_A = d((\frac{PR_B}{out-links_B}) + (\frac{PR_C}{out-links_C}) + (\frac{PR_D}{out-links_D}) + (\frac{PR_E}{out-links_E}) + (\frac{PR_F}{out-links_F})) + (1-d) \tag{5.3}$$

The above Eq. 5, clearly represents how the supporters B, C, D, E, F are contributing to the PageRank score of web page A. It can also be viewed in Fig. 5.6. It has been studied from the literature that the non-spam pages mostly point to other non-spam pages, and spam pages rarely point to non-spam pages [35].

5.4.2 Estimation of biased/unbiased supporters

The artificially created web pages are meant to increase the PageRank of a target page and are always differentiable from the web. These web pages form a link farm. The target page in this farm always gains the high PageRank score because of the presence of many in-links. As shown in Fig. 5.6, the target page A gets five in-links namely from B,C,D,E,F. So, these pages are forming a link farm and are easily differentiable from the web. But, the question is how to judge the supports as unreliable.

In this section, it is endeavoured to describe a method to evaluate reliable supporters or unreliable supporters (artificially created web pages). There are two variants of supporters, direct and indirect. The direct supporter is the source of the victim; this is due to the fact that the spammer may create a large number of websites consisting of thousands of web pages. But, this approach only controls a small portion of the web. The direct supporters have examined the connection between the spam website and artificially created websites. In Fig. 5.7, B, C,D,E, F are the direct supporters, and K is the indirect supporter. A, B, C, D, E, F forming a link farm,

but B, C, D, E, F are a direct supporter and K is the indirect supporter. The method proposed to evaluate the direct supporters .i.e, B, C, D, E, F with the help of out-links web pages. The number of out-links of a detected link farm is:

- B=2 (A, G)
- C=3 (A,H,I)
- D=3 (A,J,K)
- E=1 (A)
- F=1 (A)

It means that web pages E and F seems to be dedicatedly created for the promotion of web page A. Thus, E and F are considered as unreliable supporters and punished. In other words, the web pages with the one out-link are considered unreliable, where α=1. It has been concluded that the few direct contributors can be the victims for promoting a spam page. Therefore, the detection of such supporters and their contribution is necessary to judge the reliability of the web page. The supporters dedicatedly created for the promotion of a single website are judged to be biased and, are punished. The algorithm for computing the revised PageRank score has been discussed in the next section.

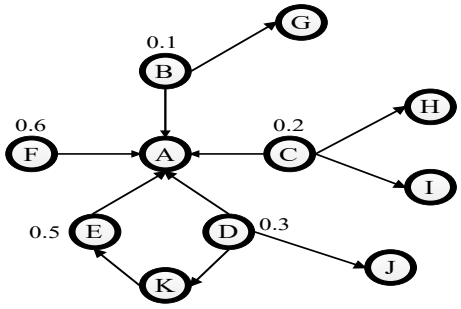


Figure 5.7: PageRank computation

5.4.3 Web model

The graph, *G=(V, E)* has been used for representing the structure of the web. The vertices depict the web pages, and the edges depict the connection between the vertices. There are two types of edges; out-link edges and in-link edges. The following notations are used in this article:
Nodes(N): Number of nodes in graph G.
Vertices's(V): represents the hosts or web pages.
edges (E): represents the link between the web pages.
in-degree edge (I): incoming links to the vertex.

out-degree edge (O): outgoing links from the vertex.
PR: PageRank score of the web page or host.
hp: Host page.
mp: Web page having maximum PageRank score.

5.4.4 Updated PageRank

Algorithm 2 Updated PageRank

Input: Graph including spam, non spam nodes.
Output: Detection of spam nodes.

procedure FUNCTION(PageRank)
  for $i = 1$ *to* n **do**
    Compute z_i ▷ vector of PageRank values by using Eq. (1)
    Set $i \leftarrow i+1$
  end for
  for $V_i = 1$ *to* n **do**
    if I and O == 0 **then**
      Discard Node
    end if
  end for
  $A[m,n] \leftarrow G'$ ▷ Graph after discarding nodes.
  create $D_{ij} = \begin{cases} 1 \; if(i,j) \in E \\ 0 \; otherwise \end{cases}$
  for $i = 1$ *to* n **do**
    Create jump probability matrix: $x_i = (Q - pAD((D - (D == 1))\ 0))/N$
    Set $i \leftarrow i+1$
  end for
  for $i = 1$ *to* n **do**
    if $x_i < z_i$ **then** ▷ by α probability
      Set $x_i \leftarrow 1$ ▷ Spam node
    else
      Set $x_i \leftarrow 0$ ▷ Non spam node
    end if
    Set $i \leftarrow i+1$
  end for
end procedure

The PageRank methodology is updated in order to avoid malicious pages the proposed Eq. 5.4 ignores the contributors which only target at promoting a single page. This formula demotes the PageRank score of a spam page. This means that the bogus pages are detected and demoted.

$$x = (Q - pAD((D - (D == 1))\ 0)) \tag{5.4}$$

The Eq. 5 has been used for computing PageRank score by the power method where Q is the identity matrix, p is constant 0.85, A is the sparse matrix and D is the matrix considering incoming and outgoing links. If the in-link has only one dedicated out link, it is ignored.

The algorithm 1 describes the steps followed to detect the spam nodes. In step 1, the graph of N nodes has been analyzed using bvgraph package in Matlab. In steps 2 to 5, the PageRank

of N nodes is computed. The steps 6 to 10 exclude the single pages from the collection, which do not have any out-link or in-link. In step 11, the matrix A is created after discarding the individual pages from the web. The sparse matrix D has been created in step 12. In step 14, the Jump probability matrix has been computed. In this new matrix x, the spam nodes are detected. If the in-link to the target node has only one dedicated out-link, it is therefore ignored. Thus, by ignoring the few in-links, the PageRank value is automatically decreased. Now, in step 18, the condition has been checked if the PageRank value is decreased by the probability range of 30% to 50%, then it is declared as spam node. This algorithm has helped to detect the link farms that targeted at single spam node.

5.4.5 Complexity analysis

The iteration in steps 2 to 5 and in steps 6 and 10 are from 1 to n. Also in step 14, when the jump probability matrix is computed, the traversing is done in n iterations. The analysis of algorithm complexity is done by measuring all the steps with their respective iterations.

Time Complexity: In this algorithm, Step 2 to 5 is the linear computation of PageRank score which takes O(n) time. The loop in steps 2-5 and steps 6-10 takes O(n) time in worst case. Steps 17 to 24 are conditional operations in LA system which takes O(1) time. Time complexity (TC) is computed as below:
=> TC= O(n)+O(n)+O(1)
=> TC= O(n)
Space Complexity: In this algorithm, input is fixed which does not exceed n, thus taking O(n) space. The loops also take O(n) space. The arithmetic operations take O(1) space. Space complexity(SC) is computed as below:
=> SC= O(n)+O(n)+O(1)
=> SC= O(n)

Chapter 6

Results and analysis

6.1 Results of proposed scheme by analyzing user behavior

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. 6.1. 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.

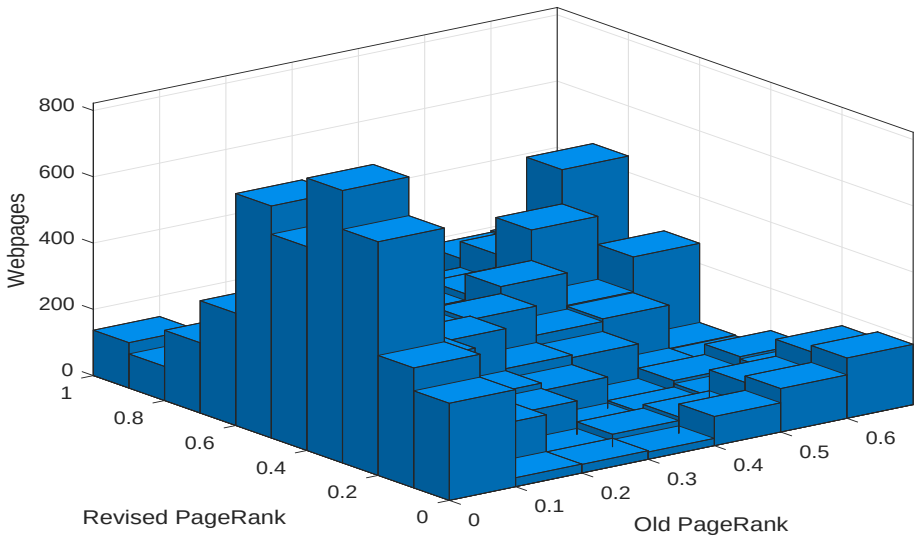


Figure 6.1: Comparison of old PageRank with revised PageRank

6.1.1 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' [186] (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 6.1. As, we have considered only the dangling pages for the experiments, So, out of 241527 query-url pair, only 12211 query-url pairs are considered.

Table 6.1: 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.

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.

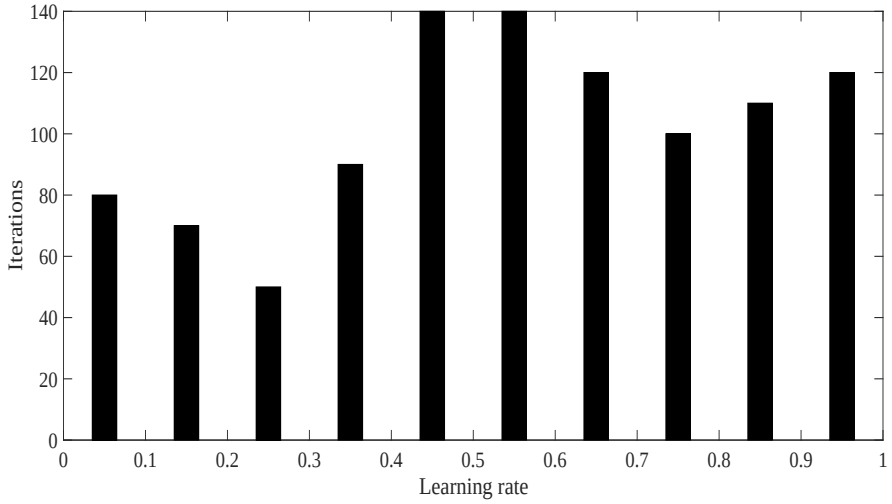


Figure 6.2: Learning Rate

6.1.2 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 computational 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 6.2. However, the implementation resulted with the iteration cycle with the change in learning rate which is clearly shown in Fig. 6.2. 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. 6.3.

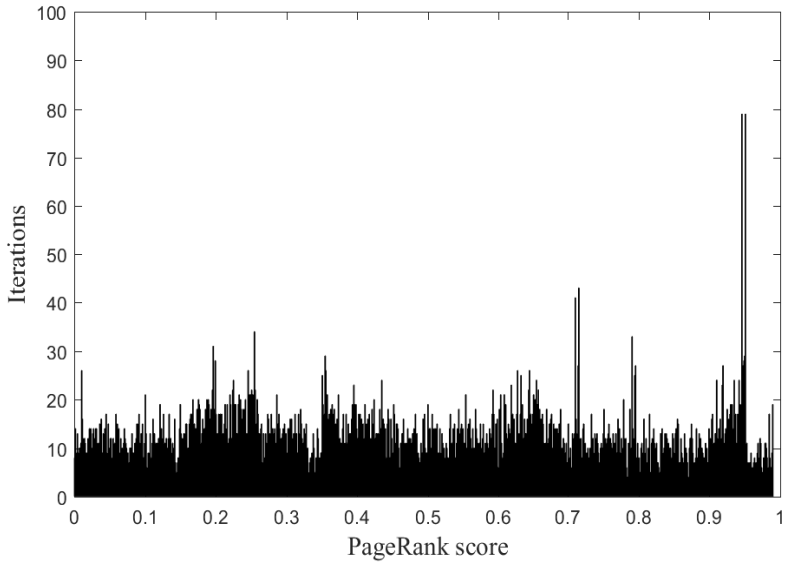


Figure 6.3: System Convergence

Table 6.2: 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

6.1.3 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 12211 dangling pages.

$$E_I = |I| + E_I^{in} - E_I^{out} - E_I^{dp} \tag{6.1}$$

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:

$$E_I = 241527 + (d/d - 1) * (12211 - 3403) \tag{6.2}$$

6.1.4 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 6.3 and results are presented in Fig. 6.4.

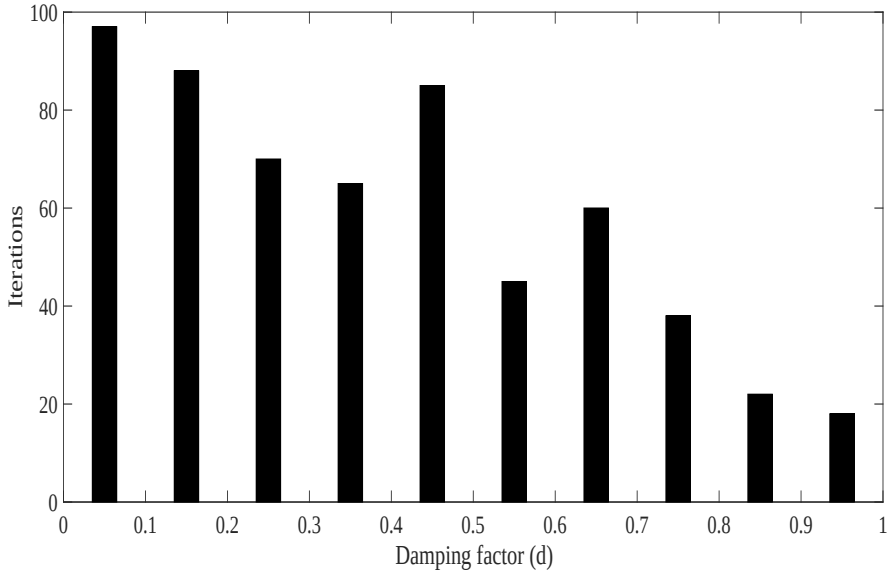


Figure 6.4: Damping factor

Table 6.3: 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

6.1.5 Impact of error on output

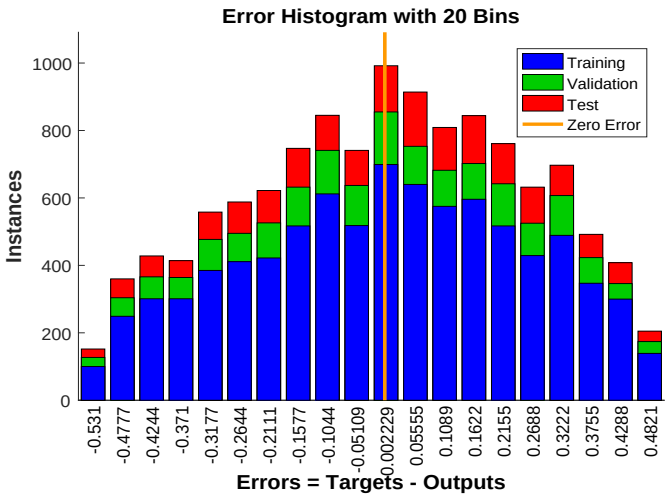


Figure 6.5: Error between target and output

The difference between the target and the output, states the occurrence of error and is depicted in Fig. 6.5. 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.

The error corresponding to the values of training, validation and testing(70:15:15), is depicted in Table 6.4. The correlation between input and error with target and output is presented in Fig. 6.6.

Table 6.4: 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

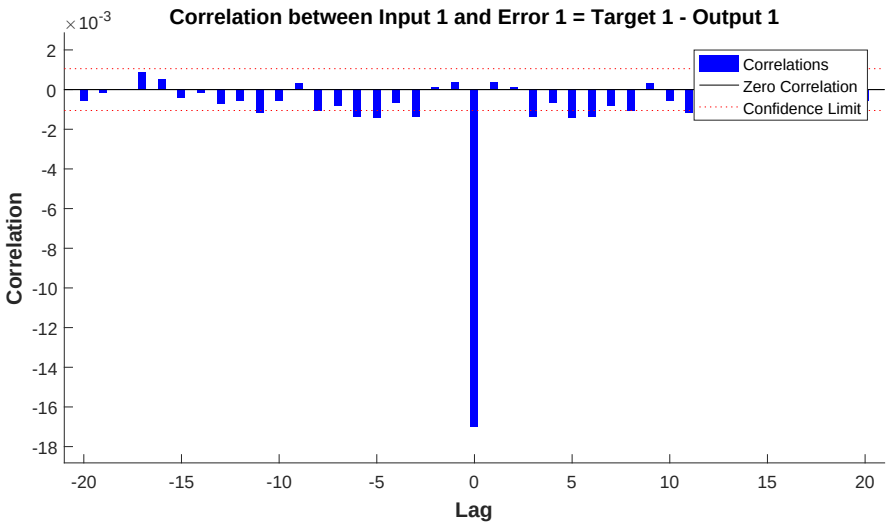


Figure 6.6: Input, output, target, error correlation

6.1.6 Machine Learning models

Three machine learning models has 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 choosen 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. 6.7.

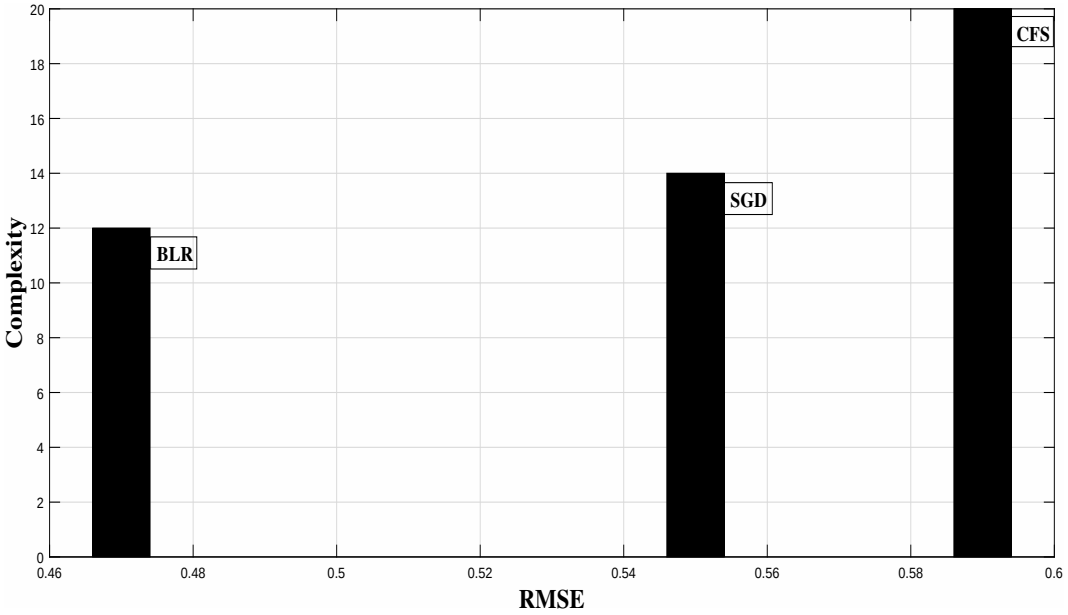


Figure 6.7: Performance of models

The performance of each model is depicted in Table 6.5.

Table 6.5: 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

As seen in the above table, Bayesian Regression performed best in all. As the value of alpha and beta are choosen randomly in this model, so the convergence didn't 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.

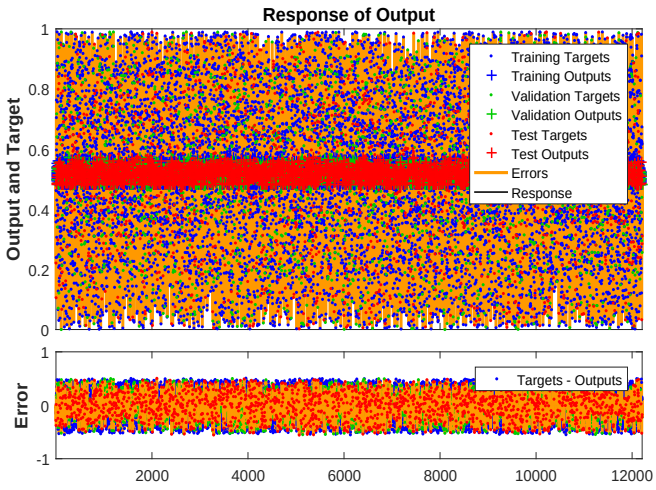


Figure 6.8: Overall performance of algorithm

Model performance is shown in Fig. 6.8. 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 12000).

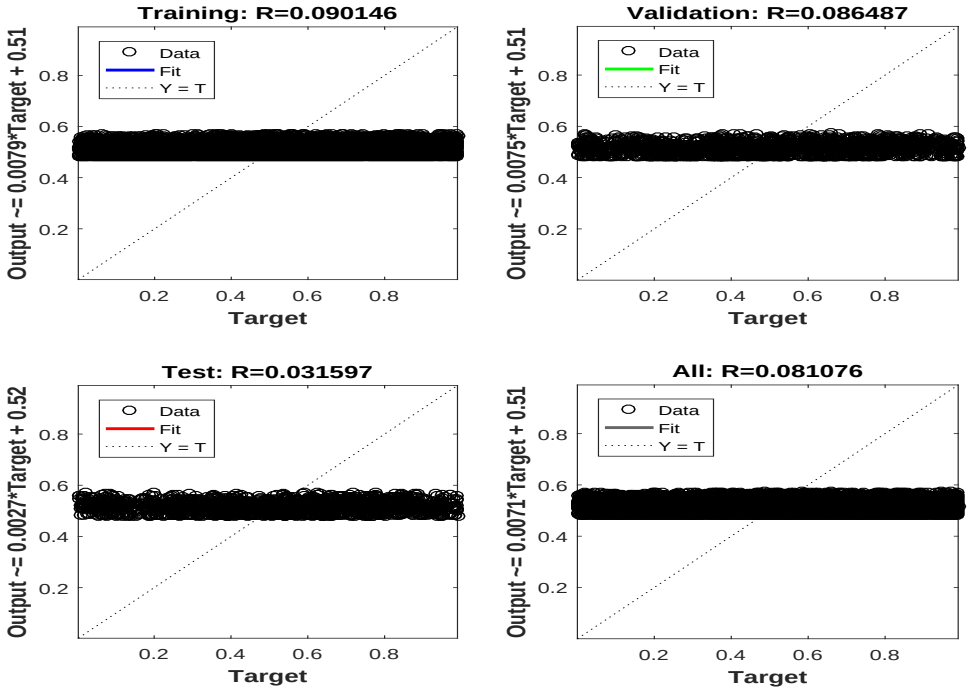


Figure 6.9: Relation between output and target

Regression R values are computed which depicts relation between the target and the output.

This relationship is presented in Fig. 6.9. 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.

6.1.7 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. 6.10.

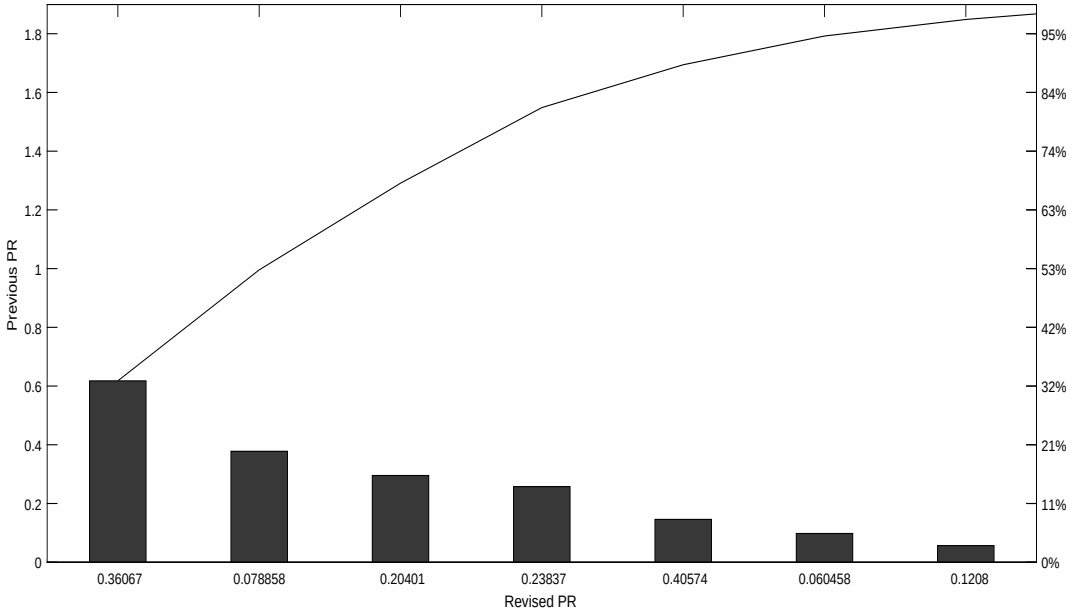


Figure 6.10: Comparison between previous PageRank with revised PageRank

The Table 6.6 represents the snapshot of the values generated in our experiments.

1. column: It represents the query-url pair number. Total number of pairs considered for experiments are 12211.

2. 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.

3. column: It represents the PageRank score computed with the Power method.

4. column: It represents the time user spends on web page. It is calculated using Gamma Distribution.

5. column: It represents the number of clicks on the web page. It is calculated using empirical probability method.

6. column: It represents the modified PageRank computed after analyzing user behavior attributes (dwell time, click count).

7. column: It represents the number of inlinks of each pair. It is normalized using empirical probability.

8. column: It represents the PageRank computed by the proposed scheme in LA environment.

Table 6.6: 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
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
12202	0	0.06982	15.2	1	0.41787	0.049955	0.23245
12203	0	0.00266	6.6	2	0.18594	0.41397	0.71406
12204	0	0.37256	3.8	3	0.47575	0.41397	0.07575
12205	0	0.25436	5.8	2	0.43764	0.000163	0.44026
12206	0	0.02942	13.3	5	0.074543	0.049955	0.87382
12207	0	0.14851	5.6	1	0.49656	0.009991	0.65642
12208	0	0.14809	7.8	2	0.33137	0.023995	0.71529
12209	0	0.01336	7.9	2	0.19664	0.0073704	0.31457
12210	0	0.00598	4.5	2	0.18926	0.41397	0.71074
12211	0	0.0325	5.1	4	0.098916	0.013103	0.30856

Findings/Significance of comparison between old PageRank and revised PageRank:

- 3403 dangling web pages out of 241527 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.

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 conditional operations in LA system which takes O(1) time. Time complexity (TC) is computed as below:
=> TC= O(n)+O(n)+O(1)
=> TC= O(n)
Space Complexity: In this algorithm, input is fixed which can not 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:
=> SC= O(n)+O(n)+O(1)
=> SC= O(n)

6.1.8 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 [31], revised PR (by effective user behavior measures) with the proposed method computed in LA environment. The experimental values are depicted in Table 6.7.

Table 6.7: 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

The Fig. 6.11 presents the analysis of four algorithms. The proposed scheme takes less than 50ms to execute for all the values of d.

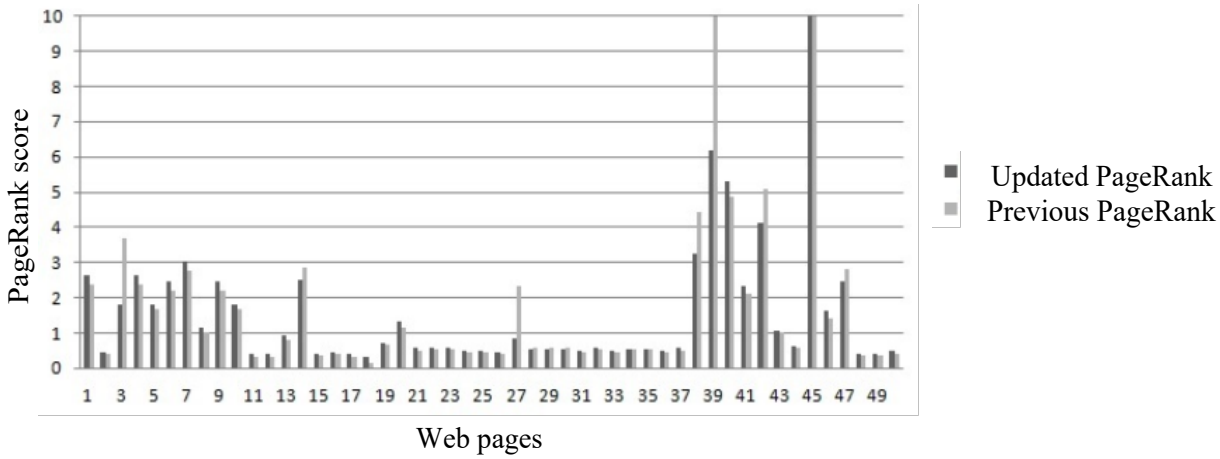


Figure 6.12: Detected spam pages

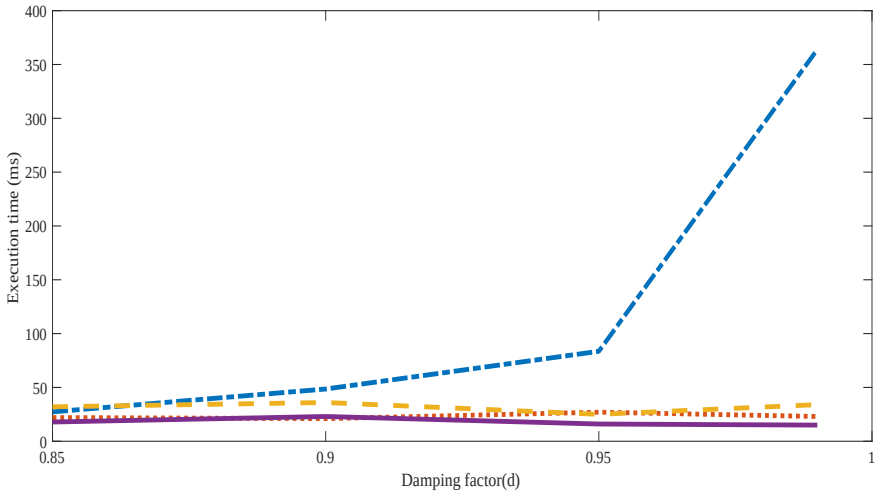


Figure 6.11: Execution time analysis of various algorithms

6.2 Results of proposed framework by updating PageRank algorithm

The PageRank algorithm has been updated with respect to web spam detection. The difference between the previous PageRank score and the updated PageRank score, detects the spam nodes. The first 50 values of jump probability matrix have been shown in Fig. 6.12. The figure clearly depicts the values decreased during the calculation of updated PageRank score. According to the proposed algorithm in previous section, if the PageRank value is decreased in the range defined, then it states the occurrence of spam. The web pages 3, 27, 39 are detected as spam pages.

6.2.1 Dataset

To experiment the proposed work, we have used publicly available dataset, WEBSPAM-UK2007 provided by the Laboratory of Web Algorithms, *Universita degli Studi di Milano*, along with the support of DELIS EU-FET research project. It is crawled in May 2007. Host level assessment has already been done by a number of volunteers. The labels were released in two different sets, SET1 containing 2/3 of the label hosts are used for training and SET2 containing 1/3 of the label hosts are used for testing.

The labels provided by judges, are named as 'spam','non-spam', 'borderline' and 'cannot classify'. The scores are also marked to take the final decision for classification, spam-1, non-spam-0, borderline-0.5, count of each category is represented in Table 6.8. Graphical presentation for both the sets is given in Fig. 6.13.

Table 6.8: Distribution of the number of pages reviewed by judges

SET	**Spam**	**Nonspam**	**Undecided**
SET1	222	3776	277
SET2	122	1933	149

The graph of WEBSPAM UK-2007 contains 105896555 vertices's, 556233648 edges, 12949672 dangling nodes, and 21563679 self-loops. However, due to large size of dataset, we have only considered the host graphs. This means, if one of the page in host is spam, then the whole host is marked as spam.

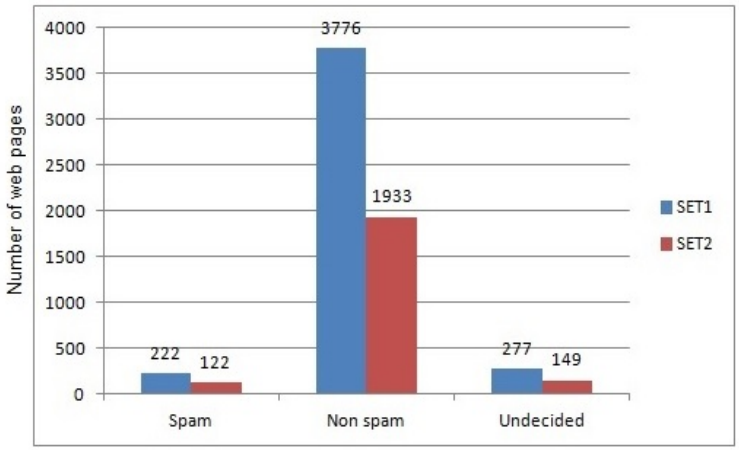


Figure 6.13: Distribution of dataset

Table 6.9: Beta distribution parameters of PageRank of home page

$\alpha 1$	$\alpha 2$	**a**	**b**
0.96834	0.95001	-1.1000E-14ă	1.1453E+5

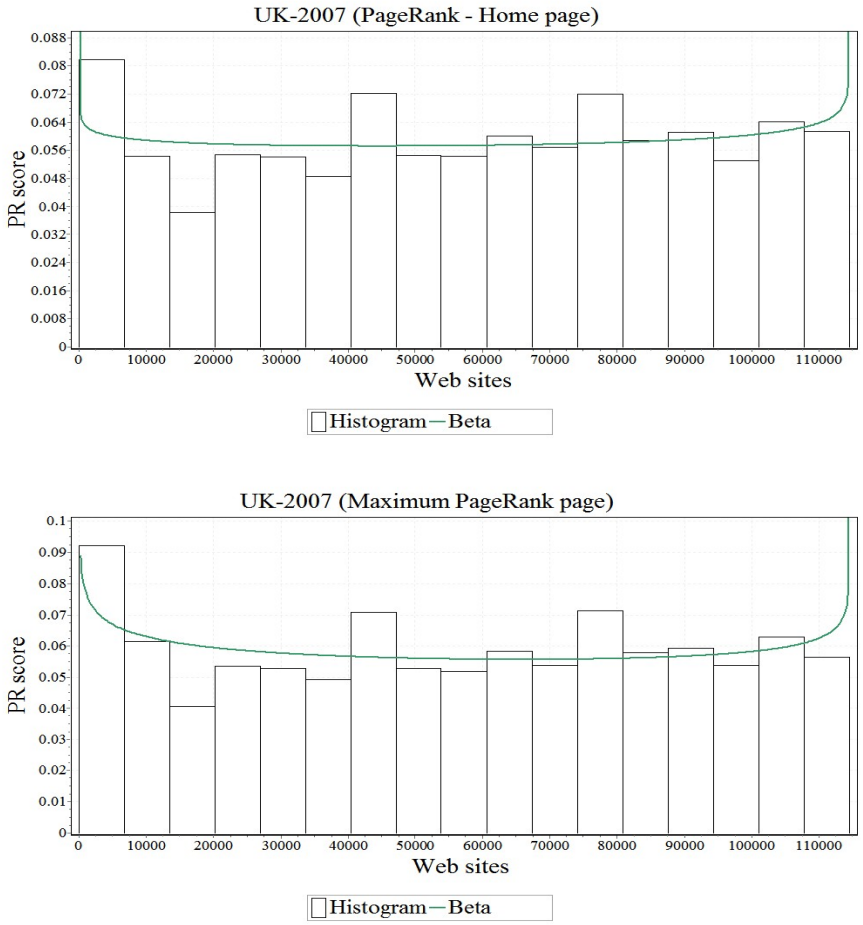


Figure 6.14: Histogram of PageRank values

6.2.2 PageRank

We computed PageRank with the help of power method, by estimating damping factor to be 0.85. Detailed description about the PageRank and the power method has been discussed in Section 2. As the power method considers the in-links links in its computation and many websites are having same number of in-links, so, these web sites share the same PageRank score. We have plotted the computed PageRank values for the dataset which has been discussed in the previous Section, by computing the beta distribution over it. The PageRank for home page of the website and the page with the maximum PageRank value are plotted in Fig. 6.14. The distribution evaluation parameters are discussed in Table 6.9 and Table 6.10.

Table 6.10: Beta distribution parameters of Maximum PageRank page

$\alpha 1$	$\alpha 2$	**a**	**b**
0.90719	0.93158	-1.1000E-14	1.1453E+5

6.2.3 Trustrank

Trustrank algorithm is used to compute the trust score of web pages. Detailed description about trustrank algorithm has been discussed in Section 2. We have plotted the trustrank values for the home page and the page with the maximum trustrank value by fitting beta distribution over it which is presented in Fig. 6.15. The distribution evaluation parameters are discussed in Table 6.11 and Table 6.12.

Table 6.11: Beta distribution parameters of trustrank of home page

$\alpha 1$	$\alpha 2$	**a**	**b**
1.1357	1.071	-1.1937E-14ă	1.1453E+5

Table 6.12: Beta distribution parameters of maximum trustrank page

$\alpha 1$	$\alpha 2$	**a**	**b**
1.0916	1.0559	-1.1937E-14	1.1453E+5

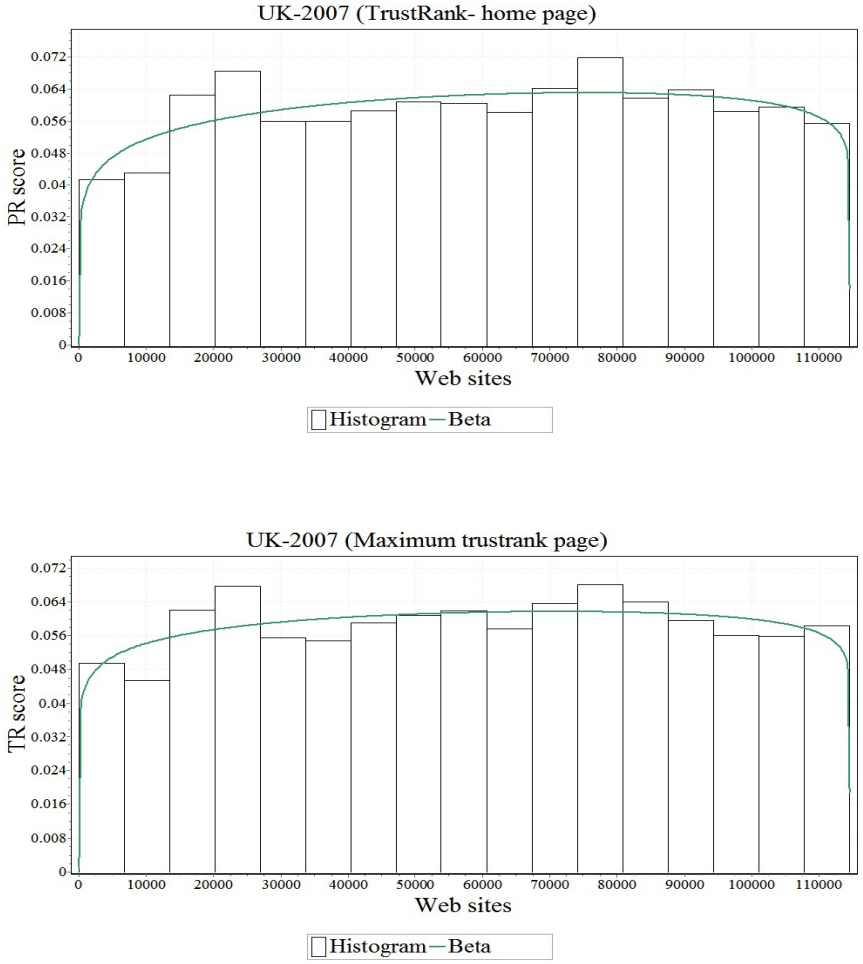


Figure 6.15: Histogram of trustrank values

6.2.4 Updated PageRank

The PageRank has been updated for the purpose of detection of spam nodes and demotion of malicious links. The detailed description about the algorithm is discussed in Section 4. As the updated PageRank algorithm ignores few in-links, so, updated PageRank scores are small when compared to PageRank score. Beta distribution is fitted over the updated PageRank score of home page and the page with the maximum score of updated PageRank. The distribution is presented in Fig. 6.16. The distribution evaluation parameters are discussed in Table 6.13 and Table 6.14.

Table 6.13: Beta distribution parameters of updated PageRank of home page

$\alpha 1$	$\alpha 2$	**a**	**b**
0.98031ă	ă0.96389	-1.1000E-14ă	1.1453E+5

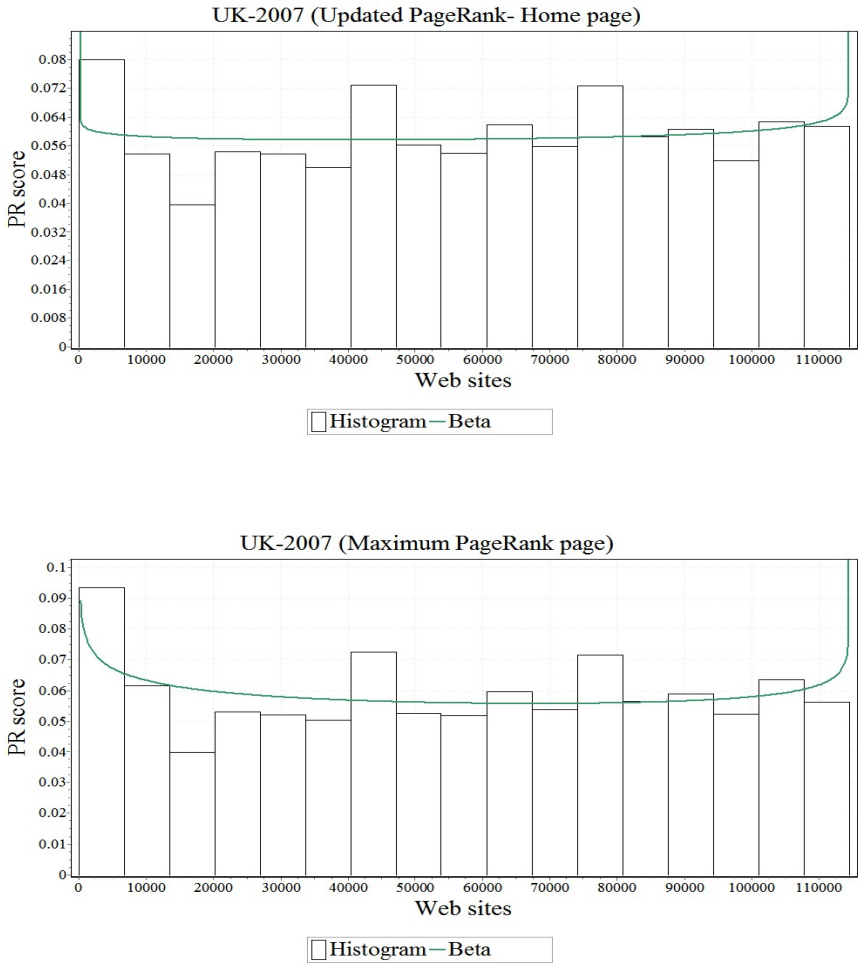


Figure 6.16: Histogram of updated PageRank values

Table 6.14: Beta distribution parameters of maximum value of updated PageRank page

$\alpha 1$	$\alpha 2$	**a**	**b**
0.90785ă	0.93698	-1.1000E-14	1.1453E+5

6.2.5 Comparison with existing techniques

The proposed scheme is compared with existing benchmark algorithms. All of the techniques which are able to detect the spam web pages and revise the rank score for better ranking of web pages are considered for the comparison with the proposed scheme. The evaluating parameter for the comparison is the rate at which spam is detected. The proposed scheme is able to detect 98% of spam as proved in the next section. Hence, it is proved to be beneficial for ranking the web pages. The Fig. 6.17 presents the analysis of four algorithms.

Table 6.15: Comparison of proposed algorithm with other algorithms

Author	**Method**	**Dataset**	**Features**	**Spam detection rate**
Becchetti *et al.*, 2008 [84]	Truncated PageRank	WEBSPAM-UK 2006	Link based features	88%
Kolda *et al.*, 2009 [114]	BadRank	WEBSPAM-UK 2007	Link and content based both	77%
Leng *et al.*, 2014 [115]	TP spam mass	WEBSPAM-UK 2006, WEBSPAM-UK 2007	-	43.216%
Zhang *et al.*, 2014 [116]	Trust-DistrustRank algorithm	WEBSPAM-UK 2007	Link based features	77%

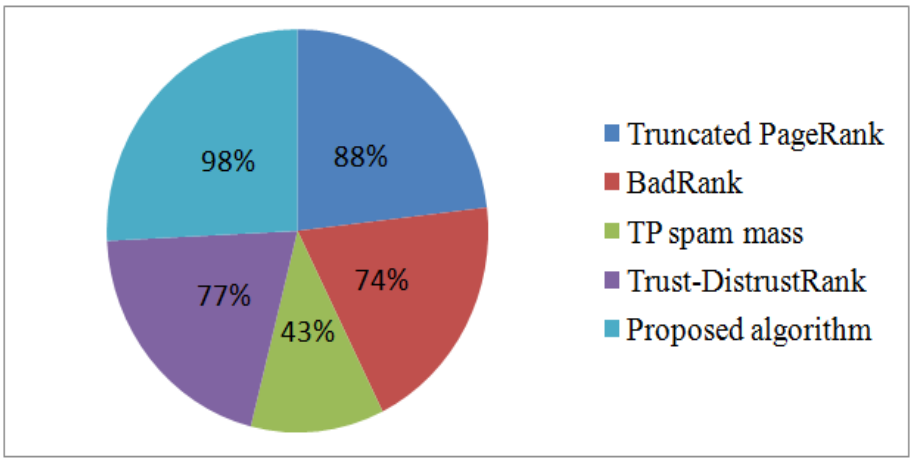


Figure 6.17: Spam detected by different spam detection algorithms

6.2.6 Machine learning models

The machine learning models have been used for the validation of proposed scheme. The R 'caret' package has been used to implement the various model building techniques. Before building the models, the data is preprocessed to remove redundancy and inconsistency. The data used for experiments is mentioned in Section 5.1. The methodology followed is discussed here below:

1. Feature selection: The proposed scheme deals with the spam detection by analyzing the link properties of the dataset. The dataset, detailed in Section 5.1, consists of 41 features and 114528 hosts. Before performing experiments, we need to remove redundant features by feature extraction and feature selection techniques. These techniques are generally used for improving learning performance, decreasing required storage and lowering computational complexity. Feature extraction is the technique of converting the high dimensional feature space into low dimensional feature space. Various types of feature extraction techniques exist in literature such as PCA [187] and Canonical Correlation Analysis (CCA).

   There are various feature selection techniques which are summarized in Fig. 6.18. Feature selection techniques depend upon the type of problem. The feature selection technique for the classification problem which follows supervised learning, are focused in finding the relevancy of feature with respect to the target class. The different types of features considered while choosing among the feature selection techniques are:

   - Flat features
   - Structured features
   - Streaming features

   We have ranked the features using one of the filter method, a flat feature technique, i.e., random forest. This random forest feature ranking by importance works on the paradigm of decision trees. It computes the gini index score to rank the features. This score represents the homogeneity in the data. The splitting of data into the nodes is done with the help of features. The gini value is computed for the root as well as for the leaves. Mean decrease gini value is generated after analyzing the difference between the various gini values. This value is highest for the most important feature. The feature importance score is computed and the results are presented in Table 6.16. The most important feature are extracted and formed a new feature set along with Hostid, updated_PageRank(output from Algorithm 1) and spam label. The optimal solution of the obtained features is discussed in Table 6.17.

Table 6.16: Feature importance score

Feature no.	**Feature name**	**Mean Decrease Gini value**
1	eq_hp_mp	7.5324557
2	assortativity_hp	1.1399570
3	assortativity_mp	2.6683472
4	avgin_of_out_hp	3.3025721
5	avgin_of_out_mp	3.6412640
6	avgout_of_in_hp	3.6675256
7	avgout_of_in_mp	3.5332456
8	indegree_hp	4.6953051
9	indegree_mp	4.7900811
10	neighbors_2_hp	2.0991127
11	neighbors_2_mp	2.0669512
12	neighbors_3_hp	2.5229768
13	neighbors_3_mp	2.2113250
14	neighbors_4_hp	2.6749084
15	neighbors_4_mp	2.3803912
16	outdegree_hp	4.6845567
17	outdegree_mp	4.6070761
18	pagerank_hp	5.2127952
19	pagerank_mp	5.2156659
20	prsigma_hp	3.5485915
21	prsigma_mp	3.7143720
22	reciprocity_hp	2.6912522
23	reciprocity_mp	2.7947788
24	siteneighbors_1_hp	2.9134713
25	siteneighbors_1_mp	2.4883176
26	siteneighbors_2_hp	3.7487207
27	siteneighbors_2_mp	3.5176911
28	siteneighbors_3_hp	3.0887188
29	siteneighbors_3_mp	3.1590898
30	siteneighbors_4_hp	1.8859393
31	siteneighbors_4_mp	3.0018736
32	truncatedpagerank_1_hp	1.9940961
33	truncatedpagerank_1_mp	2.0165576
34	truncatedpagerank_2_hp	4.8607264
35	truncatedpagerank_2_mp	4.1073029
36	truncatedpagerank_3_hp	4.0356649
37	truncatedpagerank_3_mp	4.2171884
38	truncatedpagerank_4_hp	4.2230141
39	truncatedpagerank_4_mp	4.3296526
40	trustrank_hp	3.7900802
41	trustrank_mp	3.7919782

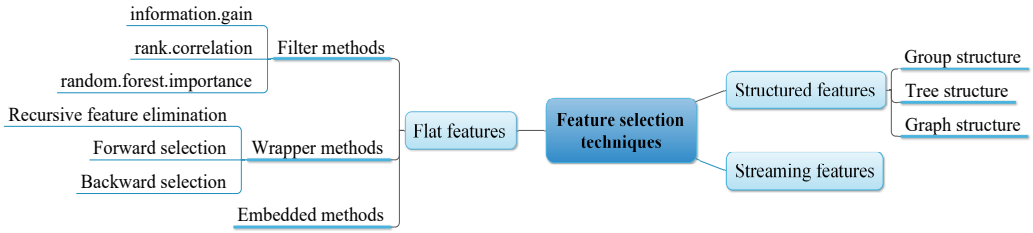


Figure 6.18: Feature selection techniques

Table 6.17: Features considered for experiments

No.	Features	Description
1	Hostid	The id for differentiating hosts.
2	Eq_hp_mp	indicating if the home page is the page with maximum PageRank or not.
3	In-degree_hp	In-degree of the host page.
4	In-degree_mp	In-degree of the page with maximum PageRank.
5	Out-degree_hp	out-degree of the host page.
6	Out-degree_mp	out-degree of the page with maximum PageRank.
7	PageRank_hp	PageRank score of the host page.
8	PageRank_mp	PageRank score of the page with maximum PageRank value.
9	updated_PageRank_hp	computed PageRank score of the host page.
10	updated_PageRank_mp	computed PageRank score of the page with maximum PageRank value.
11	spam label	web page is spam or not.

2. **Preprocessing for class balancing:** The dataset needs to be clean before processing it in models. The preprocessing is required to balance the data (same instances of multiple classes in a column), remove noisy data, fixation of NULL values and elimination of redundancies. The approach followed in this proposal for data cleaning is SOTU which is discussed in next section.

3. **Machine learning models:** The machine learning models are used for prediction, which are based upon the training and testing data. In the experiments, 15 different models with 11 sample sets are trained and tested. The steps followed are:

   - **Prediction model with SOTU:** The models are built by training with different sample sets. The models implemented are detailed in Table 6.18. The results of experiments after executing all the 15 models with 11 sample sets have been depicted in Tables 6.19 to 6.24 with different parameters.
   - **Ensemble classifier:** Based upon the overall performance as shown in Table 6.25, the models with the highest accuracy namely eXtreme Gradient Boosting, partDSA, Tree Based Ensembles, are ensemble to form the combined model. The ensemble model has resulted with improved parameters and is shown in Table 6.26. The ensemble model followed the ten fold cross validation scheme as shown in Table 6.27.

Table 6.18: Machine learning models

Model	Package	Tuning parameters	Method
Bagged MARS	Earth	nprune, degree	bagEarth
Bagged CART	ipred, plyr, e1071	None	Treebag
Bagged Model	Caret	Vars	Bag
Bayesian Generalized Linear Model	Arm	None	bayesglm
Boosted Linear Model	bst, plyr	mstop, nu	BstLm
Conditional Inference Tree	Party	Mincriterion	Ctree
eXtreme Gradient Boosting	Xgboost	nrounds, lambda, alpha	xgbLinear
Generalized Linear Model with Stepwise Feature Selection	MASS	None	glmStepAIC
k-Nearest Neighbors	Kknn	kmax, distance, kernel	Kknn
Least Angle Regression	Lars	Fraction	Lars
Parallel Random Forest	e1071,randomForest, foreach	Mtry	parRF
partDSA	partDSA	cut.off.growth, MPD	partDSA
Partial Least Squares	Pls	Ncomp	kernelpls
Partial Least Squares Generalized Linear Models	plsRglm	nt, alpha.pvals.expli	plsRglm
Tree-Based Ensembles	nodeHarvest	maxinter, mode	nodeHarvest

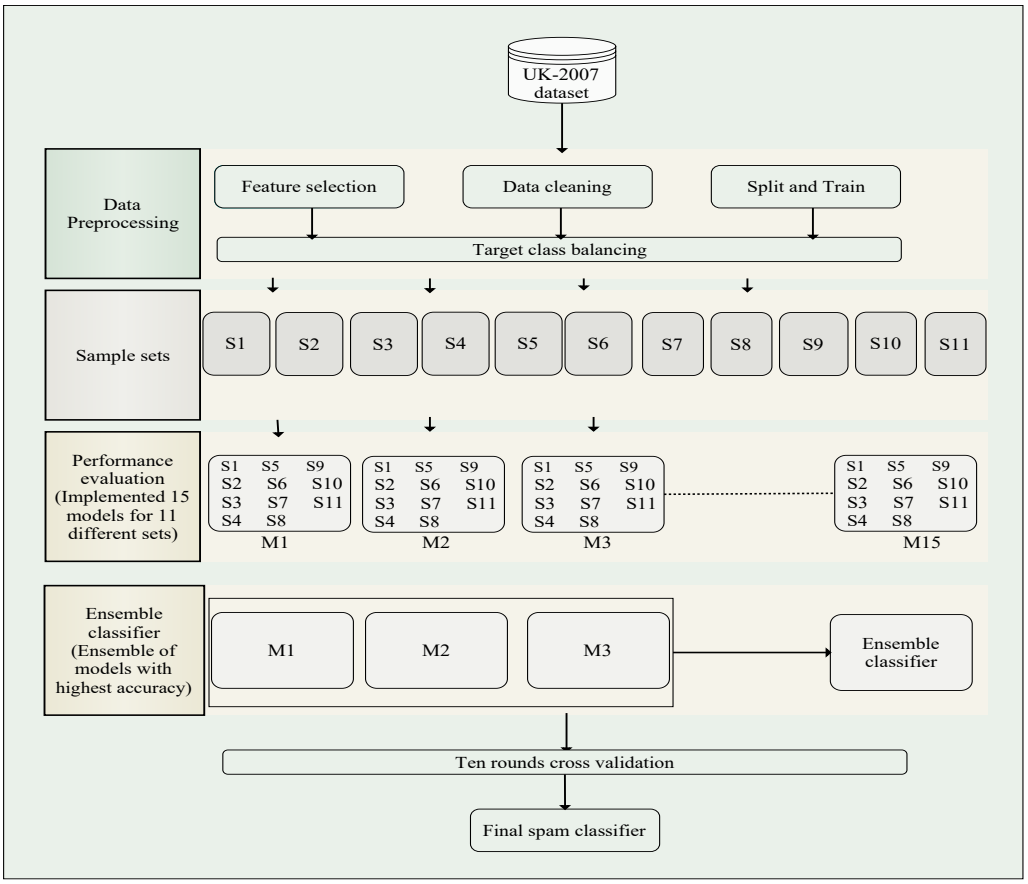


Figure 6.19: Architecture of 'Split by Over-sampling and train by Under-fitting' Approach

SOTU

The idea behind this approach is to handle inaccurate accuracy due to imbalanced classes of data. Like, the spam labels are propositioning less in comparison to other labels provided by the judges with the dataset. It can lead to biasing of the classifiers. It is specially handled when the minor class ,i.e, spam is the target class for the experiments. The approach followed is *Split by Over-sampling and train by Under-fitting* (SOTU). It is the over sampling method, by creating the multiple copies of minor class. The over fitting is avoided by training the model by a single sample set at a time. Each set contains equal instances of the minor class and the major class (the spam class and the non spam class with equal ratio). The complete dataset is divided into 11 buckets of sample sets. Each model is trained and tested 11 times with different sets. It has been therefore said, that the system is trained with unbiased data. The sample distribution approach in the proposed data cleaning approach (SOTU) is depicted in Fig. 6.19.

$$S = \sum_{i=1}^{d} \frac{x_m + x_n}{d} \tag{6.3}$$

Table 6.19: Performance of H with respect to models and samples

Set	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
S1	0.969	0.977	1	0.852	0.798	0.766	0.99	0.835	0.897	0.97	0.862	0.951	0.972	0.914	0.986
S2	0.985	0.96	0.986	0.897	0.683	0.856	0.97	0.847	0.873	0.985	0.918	0.959	0.959	0.872	0.945
S3	1	0.943	0.965	0.798	0.775	0.867	0.968	0.835	0.795	0.942	0.955	0.963	0.935	0.865	0.945
S4	0.961	0.926	0.917	0.937	0.826	0.853	0.971	0.89	0.762	0.865	0.904	0.958	0.958	0.945	0.959
S5	0.878	0.959	0.927	0.819	0.773	0.886	0.98	0.829	0.695	0.932	0.877	0.927	0.899	0.881	0.887
S6	0.907	0.978	0.892	0.767	0.701	0.873	0.979	0.779	0.732	0.875	0.737	0.928	0.878	0.815	0.904
S7	0.979	1	0.981	0.861	0.659	0.848	0.969	0.902	0.853	0.899	0.768	0.978	0.901	0.777	0.906
S8	1	0.346	0.986	0.223	0.104	0	0.959	0.33	0.76	0.113	0.714	0.986	0.351	0.07	1
S9	1	0.988	1	0.529	0.438	0.593	0.98	0.657	0.772	0.653	0.879	0.988	0.664	0.661	0.986
S10	0.989	0.98	0.986	0.77	0.699	0.801	0.99	0.956	0.807	0.845	0.847	0.934	0.882	0.741	1
S11	0.928	0.894	0.925	0.874	0.521	0.861	0.961	0.817	0.778	0.971	0.905	0.949	0.935	0.945	0.942
Avg.	0.963	0.904	0.960	0.757	0.634	0.745	0.974	0.7888	0.793	0.822	0.851	0.956	0.848	0.771	0.950

Note: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles, S1-S12= Sample sets

Table 6.20: Performance of Gini with respect to models and samples

Set	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
S1	0.987	0.999	1	0.966	0.899	0.859	0.993	0.927	0.981	0.974	0.935	0.97	0.986	0.984	0.991
S2	0.98	0.988	0.982	0.963	0.789	0.913	0.979	0.921	0.95	0.979	0.943	0.973	0.948	0.96	0.963
S3	1	0.965	0.98	0.932	0.841	0.919	0.978	0.921	0.907	0.929	0.962	0.977	0.935	0.926	0.962
S4	0.995	0.947	0.919	0.979	0.889	0.909	0.981	0.898	0.874	0.895	0.935	0.971	0.945	0.947	0.972
S5	0.925	0.976	0.977	0.893	0.838	0.924	0.989	0.902	0.872	0.908	0.94	0.965	0.858	0.901	0.921
S6	0.92	0.999	0.967	0.856	0.741	0.915	0.985	0.889	0.861	0.925	0.859	0.995	0.834	0.909	0.94
S7	0.999	1	0.999	0.914	0.776	0.899	0.978	0.938	0.93	0.857	0.829	0.988	0.863	0.825	0.934
S8	1	0.367	0.997	0.095	0.006	0	0.972	0.097	0.892	0.065	0.842	0.99	0.075	0.108	1
S9	1	0.992	1	0.45	0.483	0.696	0.986	0.562	0.874	0.495	0.96	0.998	0.52	0.61	0.99
S10	1	0.999	0.991	0.759	0.738	0.859	0.993	0.937	0.937	0.842	0.957	0.945	0.847	0.824	1
S11	0.908	0.918	0.901	0.886	0.676	0.903	0.976	0.876	0.899	0.959	0.958	0.97	0.942	0.951	0.96
Avg.	0.974	0.922	0.973	0.790	0.697	0.799	0.982	0.806	0.907	0.802	0.92	0.976	0.795	0.813	0.966

Note: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles, S1-S12= Sample sets

Table 6.21: Performance of AUC with respect to models and samples

Set	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	
S1	0.994	1	1	0.983	0.949	0.929	0.997	0.963	0.99	0.987	0.967	0.985	0.993	0.992	0.995	
S2	0.99	0.994	0.991	0.981	0.895	0.956	0.99	0.96	0.975	0.99	0.972	0.986	0.974	0.98	0.981	
S3	1	0.982	0.99	0.966	0.921	0.959	0.989	0.96	0.954	0.964	0.981	0.988	0.968	0.963	0.981	
S4	0.997	0.973	0.96	0.99	0.944	0.954	0.99	0.949	0.937	0.947	0.968	0.985	0.972	0.974	0.986	
S5	0.963	0.988	0.989	0.946	0.919	0.962	0.994	0.951	0.936	0.954	0.97	0.982	0.929	0.951	0.96	
S6	0.96	0.999	0.984	0.928	0.87	0.957	0.993	0.945	0.93	0.962	0.929	0.997	0.917	0.955	0.97	
S7	1	1	1	0.957	0.888	0.95	0.989	0.969	0.965	0.929	0.914	0.994	0.931	0.913	0.967	
S8	1	0.684	0.999	0.547	0.503	0.5	0.986	0.548	0.946	0.532	0.921	0.995	0.538	0.554	1	
S9	1	0.996	1	0.725	0.741	0.848	0.993	0.781	0.937	0.748	0.98	0.999	0.76	0.805	0.995	
S10	1	1	0.996	0.88	0.869	0.929	0.997	0.968	0.969	0.921	0.978	0.973	0.924	0.912	1	
S11	0.954	0.959	0.951	0.943	0.838	0.952	0.988	0.938	0.949	0.98	0.979	0.985	0.971	0.976	0.98	
Avg.	0.987	0.961	0.987	0.895	0.848	0.899	0.991	0.902	0.953	0.901	0.959	0.988	0.897	0.906	0.983	

Note: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles, S1-S12= Sample sets

Table 6.22: Performance of AUCH with respect to models and samples

Set	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	
S1	0.997	1	1	0.986	0.97	0.929	0.997	0.973	0.992	0.993	0.972	0.985	0.996	0.994	0.995	
S2	0.995	0.994	0.995	0.986	0.921	0.956	0.99	0.969	0.978	0.995	0.972	0.986	0.986	0.985	0.981	
S3	1	0.984	0.995	0.974	0.947	0.959	0.989	0.969	0.96	0.98	0.987	0.988	0.982	0.975	0.981	
S4	0.998	0.973	0.978	0.994	0.969	0.954	0.99	0.967	0.939	0.964	0.972	0.985	0.986	0.987	0.986	
S5	0.972	0.988	0.992	0.96	0.944	0.962	0.996	0.964	0.941	0.977	0.977	0.982	0.965	0.974	0.96	
S6	0.978	0.999	0.988	0.944	0.912	0.957	0.993	0.961	0.934	0.974	0.934	0.997	0.958	0.965	0.975	
S7	1	1	1	0.969	0.917	0.95	0.989	0.98	0.967	0.964	0.918	0.994	0.966	0.945	0.976	
S8	1	0.736	0.999	0.696	0.639	0.5	0.986	0.749	0.948	0.659	0.93	0.995	0.747	0.604	1	
S9	1	0.996	1	0.839	0.828	0.848	0.993	0.877	0.937	0.874	0.983	0.999	0.879	0.887	0.995	
S10	1	1	0.998	0.935	0.926	0.929	0.997	0.984	0.973	0.954	0.985	0.978	0.962	0.938	1	
S11	0.974	0.963	0.973	0.957	0.866	0.952	0.992	0.955	0.951	0.99	0.982	0.985	0.983	0.983	0.98	
Avg.	0.992	0.966	0.992	0.930	0.894	0.899	0.992	0.940	0.956	0.938	0.964	0.988	0.946	0.930	0.984	

Note: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles, S1-S12= Sample sets

Table 6.23: Performance of Youden with respect to models and samples

Set	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
S1	0.922	0.95	0.88	0.745	0.79	0.859	0.993	0.763	0.92	0.457	0.899	0.97	0.771	0.69	0.991
S2	0.918	0.971	0.922	0.755	0.673	0.913	0.979	0.721	0.896	0.546	0.943	0.973	0.731	0.699	0.963
S3	0.899	0.96	0.953	0.721	0.743	0.919	0.978	0.748	0.831	0.495	0.964	0.977	0.649	0.739	0.962
S4	0.924	0.947	0.931	0.835	0.8	0.909	0.981	0.691	0.775	0.748	0.934	0.971	0.75	0.738	0.972
S5	0.884	0.973	0.897	0.699	0.777	0.924	0.986	0.667	0.755	0.56	0.894	0.953	0.663	0.714	0.921
S6	0.926	0.888	0.907	0.703	0.759	0.915	0.985	0.697	0.778	0.657	0.809	0.934	0.713	0.685	0.933
S7	0.943	1	0.984	0.75	0.734	0.899	0.978	0.686	0.881	0.429	0.836	0.987	0.735	0.702	0.936
S8	0.919	0.462	0.943	0.183	0.013	0	0.972	0.198	0.719	0	0.789	0.99	0.21	0	1
S9	0.98	0.992	0.97	0.549	0.223	0.696	0.986	0.571	0.838	0.534	0.885	0.993	0.54	0.676	0.99
S10	0.96	0.977	0.971	0.648	0.255	0.859	0.993	0.768	0.8	0.714	0.872	0.957	0.79	0.614	1
S11	0.918	0.926	0.914	0.716	0.39	0.903	0.899	0.735	0.788	0.571	0.912	0.964	0.703	0.701	0.96
Avg.	0.926	0.913	0.933	0.664	0.557	0.799	0.975	0.622	0.816	0.519	0.885	0.969	0.621	0.632	0.966

Note: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles, S1-S12= Sample sets

Table 6.24: Performance of Accuracy with respect to models and samples

Set	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
S1	96.79	98	94.4	89.2	91.6	91.79	99.71	89.2	96.4	79.6	95.79	98.4	91.2	87.6	99.6
S2	96.8	98.8	96.8	88.8	85.2	95.16	99.14	88.4	95.6	82.4	97.68	98.8	88.4	86.4	98.4
S3	96	98.4	98	88.8	88	95.58	99.14	88.4	92.8	80	98.53	98.8	84.4	90.8	98.4
S4	96.81	98.01	97.21	90.8	92.03	95.17	99.15	88.45	90.44	89.64	97.27	98.8	89.64	88.84	98.8
S5	95.6	98.8	96	86.8	90	96.63	99.43	85.6	89.6	80.8	95.16	97.6	86.8	87.2	96.8
S6	96.81	95.6	96	87.2	89.6	96.21	99.43	87.2	89.6	85.6	92	96.4	84.8	88	97.2
S7	97.6	100	99.2	88.8	88	95.37	99.14	86.8	94.8	77.6	93.26	99.2	89.2	87.2	97.2
S8	96.79	77.8	97.59	67.47	60.24	59.2	98.85	65.86	87.95	57.43	91.33	99.6	66.67	59.04	100
S9	99.2	99.67	98.8	82.4	68	87.79	99.43	82	92.8	80.8	94.74	99.6	81.6	86.4	99.6
S10	98.41	99	98.01	84.46	70.92	94.33	99.72	91.24	90.84	87.25	94.54	98.01	91.24	83.67	100
S11	96.8	97	96.8	88.4	74.4	96	94.57	88.4	90.8	83.2	96.42	98.4	88	87.2	98.4
Avg.	97.05	96.46	97.16	85.73	81.63	91.2	98.88	85.59	91.96	80.40	95.15	98.51	85.63	84.76	98.58

Note: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles, S1-S12= Sample sets

Table 6.25: Summary of performance of all the models

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
M1	0.963	0.974	0.987	0.992	0.973	0.011	0.012	0.889	0.980	0.029	0.927	0.998	0.998	0.927	0.927	0.001	0.961	0.926	97.055	132.6
M2	0.904	0.922	0.961	0.966	0.926	0.030	0.035	0.764	0.939	0.035	0.913	1	1	0.913	0.913	0	0.946	0.913	96.46	7.182
M3	0.960	0.973	0.987	0.992	0.970	0.0127	0.014	0.910	0.978	0.027	0.934	0.999	0.999	0.934	0.934	0	0.965	0.933	97.16	58.20
M4	0.757	0.790	0.895	0.930	0.821	0.078	0.085	0.498	0.846	0.137	0.676	0.987	0.960	0.676	0.676	0.012	0.786	0.664	85.73	3.21
M5	0.634	0.697	0.848	0.894	0.728	0.120	0.135	0.275	0.719	0.181	0.562	0.995	0.9	0.562	0.562	0.004	0.668	0.557	81.63	12.31
M6	0.745	0.799	0.899	0.899	0.799	0.087	0.096	0.699	0.742	0.087	0.834	0.965	0.949	0.834	0.834	0.034	0.928	0.799	91.20	3.437
M7	0.974	0.982	0.991	0.992	0.982	0.007	0.008	1	0.995	0.011	0.982	0.993	0.991	0.982	0.982	0.006	0.986	0.975	**98.88**	66.07
M8	0.788	0.806	0.902	0.940	0.845	0.068	0.074	0.555	0.873	0.170	0.716	0.906	0.937	0.716	0.716	0.094	0.796	0.622	85.595	5.99
M9	0.793	0.907	0.953	0.956	0.844	0.068	0.074	0.472	0.870	0.080	0.846	0.970	0.951	0.846	0.846	0.03	0.894	0.816	91.96	3.649
M10	0.822	0.802	0.901	0.938	0.867	0.060	0.0643	0.578	0.858	0.209	0.610	0.908	0.946	0.610	0.610	0.091	0.710	0.519	80.39	2.243
M11	0.8514	0.92	0.959	0.964	0.891	0.044	0.052	0.654	0.909	0.048	0.891	0.993	0.990	0.891	0.891	0.006	0.937	0.885	95.156	3.902
M12	0.956	0.976	0.988	0.988	0.969	0.014	0.014	0.992	0.994	0.014	0.985	0.984	0.98	0.985	0.985	0.015	0.982	0.969	**98.51**	4.56
M13	0.848	0.795	0.897	0.946	0.887	0.047	0.054	0.455	0.893	0.171	0.712	0.909	0.942	0.712	0.712	0.090	0.795	0.621	85.63	2.328
M14	0.771	0.813	0.906	0.930	0.825	0.077	0.084	0.525	0.828	0.151	0.636	0.995	0.99	0.636	0.636	0.004	0.820	0.632	84.75	33.05
M15	0.950	0.966	0.983	0.984	0.966	0.014	0.016	0.860	0.987	0.014	0.966	1	1	0.966	0.966	0	0.982	0.966	**98.58**	139.05

Table 6.26: Performance of Ensemble model

M	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
EM	0.985	0.99	0.995	0.995	0.99	0.004	0.005	1	0.999	0.004	1	0.99	1	1	0.99	0	0.995	0.99	99.3	746.9

Table 6.27: 10 rounds of cross validation

R	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.985	0.99	0.995	0.995	0.99	0.004	0.005	1	0.999	0.004	0.99	1	1	0.99	0.99	0	0.995	0.99	99.6	754.1
2	0.986	0.991	0.995	0.995	0.991	0.004	0.005	1	1	0.004	0.991	1	1	0.991	0.991	0	0.995	0.991	99.6	681.6
3	0.986	0.991	0.995	0.995	0.991	0.004	0.005	1	1	0.004	0.991	1	1	0.991	0.991	0	0.995	0.991	99.6	729.3
4	0.97	0.979	0.989	0.989	0.979	0.008	0.01	1	0.999	0.008	0.979	1	1	0.979	0.979	0	0.989	0.979	99.2	18015
5	0.971	0.98	0.99	0.99	0.98	0.008	0.01	1	0.999	0.008	0.98	1	1	0.98	0.98	0	0.99	0.98	99.2	826
6	0.96	0.97	0.985	0.989	0.97	0.008	0.01	1	1	0.008	0.97	1	1	0.97	0.96	0	0.99	0.97	99.14	576
7	0.969	0.978	0.989	0.989	0.978	0.009	0.011	1	0.999	0.009	0.978	1	1	0.978	0.978	0	0.989	0.978	99.14	60.49
8	0.98	0.989	0.994	0.996	0.986	0.006	0.007	1	0.986	0.006	0.986	1	1	0.986	0.986	0	0.993	0.986	99.2	70.9
9	0.986	0.99	0.995	0.995	0.99	0.004	0.005	1	1	0.004	0.99	1	1	0.99	0.99	0	0.995	0.99	99.6	37.5
10	0.988	0.998	0.999	0.999	0.993	0.004	0.003	0.999	1	0.004	1	0.993	0.991	1	1	0.007	0.995	0.993	99.6	34.91

Note*: 1= H measure, 2=Gini, 3=AUC, 4=AUCH, 5=KS, 6=MER, 7=MWL, 8=Spec.Sens95, 9=Sens.Spec95, 10=ER , 13=Precision, 14= Recall, 15=TPR, 16=FPR, 17= F-measure, 18=Youden, 19=Accuracy, 20=Execution time

Note**: M1= Bagged MARS, M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting, M8= Generalized Linear Model with Stepwise Feature Selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel Random Forest, M12= partDSA, M13= Partial Least Squares, M14= Partial Least Squares Generalized Linear Models, M15= Tree-Based Ensembles

The Eq. 6.3 is used for the formulation of sample sets. S refers to the training file of dataset. x_m refers to the instances of minor class and x_n refers to the instances of major class. d is the number of sample sets, it depends upon the type of problem and number of features. In the case of classification problem, it is fixed to be an odd number. In the case of regression problem, it is fixed to be an even number. The number estimated should be close to the number of features (variation of 1 is suggested if needed). As spam detection problem is a classification problem, so it is estimated to be an odd number. In this proposal, the feature count is 11, which is already an odd number. d is fixed with the value 11 and sample sets formed are 11.

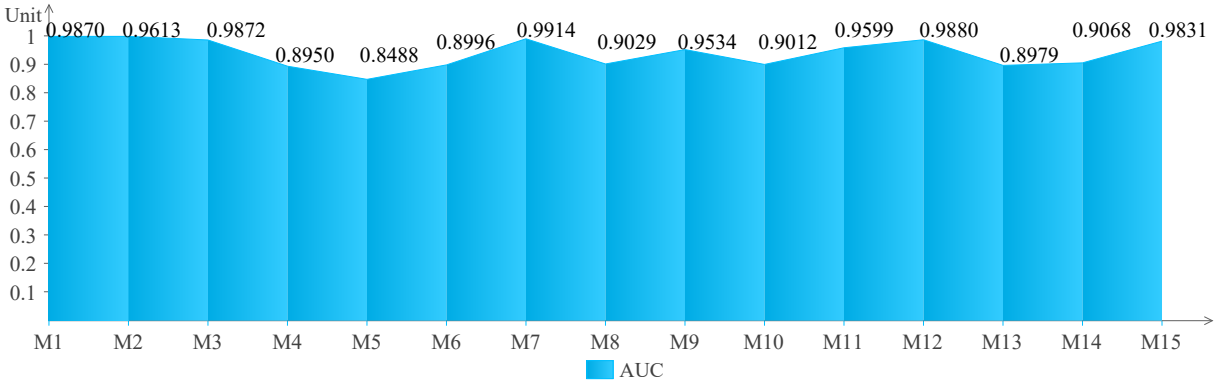


Figure 6.20: AUC comparison of machine learning models

Comparison of machine learning models

The results of different machine learning models are compared in terms of accuracy, execution time and AUC. The scenario adopted in this proposal targets the model to achieve the best performance within cost and in minimal time. Fifteen models on the basis of AUC are compared in Fig. 6.20. Extreme gradient boosting outperforms other models. Accuracy and execution time are two other parameters used for comparison and are presented in Fig. 6.21. Though many models finished their execution in very less time span, but judging them best would be unfair without considering their accuracy. It may be the case of rapid learning but inaccurate performance. In this aspect, partDSA outperforms others.

Note: M1= Bagged MARS , M2= Bagged CART, M3= Bagged Model, M4= Bayesian Generalized Linear Model, M5= Boosted Linear Model, M6= Conditional Inference Tree, M7= eXtreme Gradient Boosting , M8= Generalised_linear_model_with_stepwise_feature_selection, M9= k-Nearest Neighbors, M10= Least Angle Regression, M11= Parallel_Random_Forest, M12= partDSA, M13= Partial_Least_Squares, M14= Partial_Least_Squares Generalized_Linear_Models, M15= Tree Based_Ensembles

Training and testing datasets

The performance of machine learning model depends upon training and testing datasets. The experiments performed for the validation of proposed scheme uses the training and testing

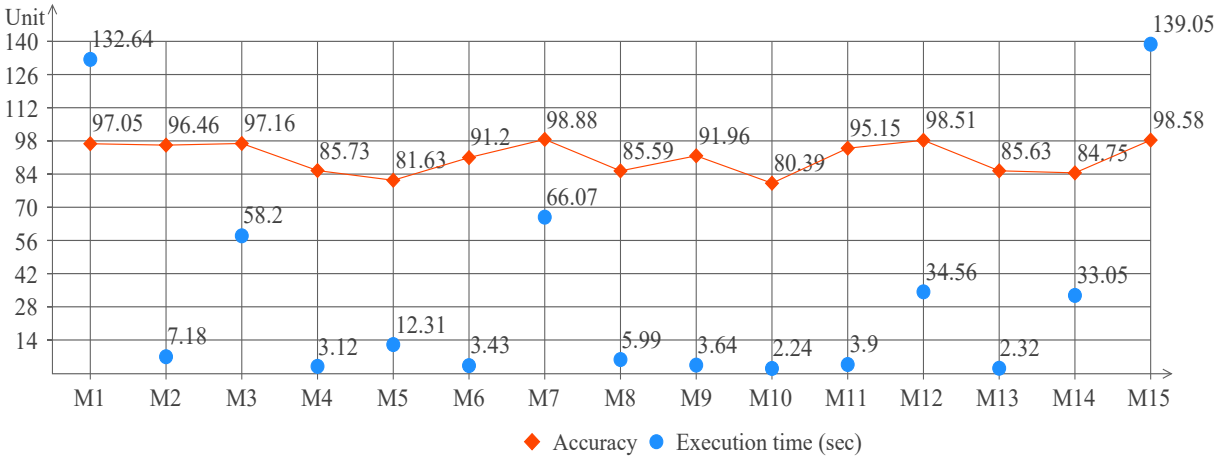


Figure 6.21: Accuracy and execution time comparison of machine learning models

data in the ratio of 70 and 30. However, if we change the training and testing ratios, the evaluation metrics also changes. This is because the working of machine learning model which greatly depends upon the formula and training data and testing data. The experiments performed by the variations in training and testing data are presented in Table 6.28.

Table 6.28: Variations in training and testing data

Models	**Accuracy (40:60)**	**Execution time (40:60)**	**Accuracy (50:50)**	**Execution time (50:50)**	**Accuracy (60:40)**	**Execution time (60:40)**	**Accuracy (70:30)**	**Execution time (70:30)**
M1	0.998	0.927	0.927	0.001	0.961	0.926	97.055	132.6
M2	1	0.913	0.913	0	0.946	0.913	96.46	7.182
M3	0.999	0.934	0.934	0	0.965	0.933	97.16	58.20
M4	0.960	0.676	0.676	0.012	0.786	0.664	85.73	3.21
M5	0.634	0.697	0.848	0.894	0.728	0.120	81.63	12.31
M6	0.745	0.799	0.899	0.899	0.799	0.087	91.20	3.437
M7	0.974	0.982	0.991	0.992	0.982	0.007	**98.88**	66.07
M8	0.788	0.806	0.902	0.940	0.845	0.068	85.595	5.99
M9	0.793	0.907	0.953	0.956	0.844	0.068	91.96	3.649
M10	0.822	0.802	0.901	0.938	0.867	0.060	80.39	2.243
M11	0.8514	0.92	0.959	0.964	0.891	0.044	95.156	3.902
M12	0.956	0.976	0.988	0.988	0.969	0.014	**98.51**	4.56
M13	0.848	0.795	0.897	0.946	0.887	0.047	85.63	2.328
M14	0.771	0.813	0.906	0.930	0.825	0.077	84.75	33.05
M15	0.950	0.966	0.983	0.984	0.966	0.014	**98.58**	139.05

Case Study

Analysis of different data cleaning approaches has been done. The techniques are summarized in Table 6.29.

Table 6.29: Different data cleaning approaches

No.	Approaches	Year	Technique	Description	Evaluation measures
1.	CNN [188]	1968	under sampling	-	-
2.	Tomek links [189]	1976	under sampling	-	-
3.	Balancing data [190]	1996	over sampling	created a new dataset with equal probabilities	error rate
4.	One sided selection [191]	1997	under sampling	under sampling the majority class	geometric mean
5.	SHRINK system [192]	1998	best positive region	positive classification of an overlapping region of minority and majority classes	ROC
6.	Solution for data mining [193]	1998	over and under sampling	over-sampling of the minority class with under-sampling of the majority class	Lift curve
7.	Learn to balance [194]	2000	focused resampling	introduced under-sampling, re-sampling and a recognition-based induction scheme	error rate
8.	Neighborhood Cleaning Rule (NCL) [195]	2001	under sampling	nearest neighbors that belong to the majority class are removed	Accuracy
9.	Synthetic minority oversampling technique (SMOTE) [196]	2002	over sampling	the minority class is over-sampled by creating 'synthetic' examples	ROC
10.	SMOTE + Tomek links [197]	2003	over sampling	Applied tomek links to create class clusters more accurately	Accuracy
11.	SOTU	Proposed scheme	over sampling	over sampling but avoided over fitting	Accuracy

6.2.7 Analysis of updated PageRank

The updated PageRank is deeply investigated by computing its cumulative distribution function, survival function, hazard function and, cumulative hazard function. We can figure out the updated PageRank, and it is summarized in Table 6.30.

Cumulative Distribution Function

The Cumulative Distribution Function (CDF) of a variable (here, it is the value of updated PageRank) can be expressed by the Eq. 6.4. The value on the y axis starts from 0, and propagates till 1. Beta distribution over the computed CDF is presented in Fig. 6.22.

$$F(x) = \int_{-\infty}^{x} f(t)dt \tag{6.4}$$

for -$\infty < x < \infty$

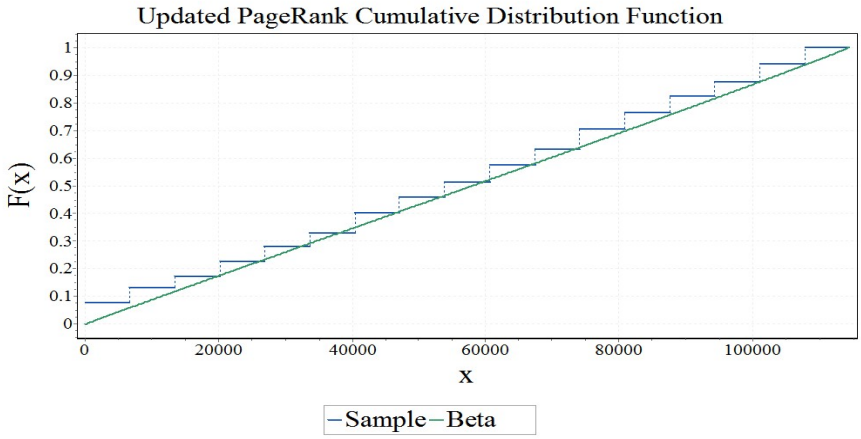


Figure 6.22: Distribution of cumulative distribution function for Updated PageRank

Survival Function

The Cumulative Distribution Function (SF) of a variable (here, it is the value of updated PageRank) can be expressed by the Eq. 6.5. The value on the y axis starts from 1, and propagates till 0. It is the inverse of CDF and can be comparable with it. Beta distribution of the computed survival function is presented in Fig. 6.23.

$$S(x) = 1 - F(x) \tag{6.5}$$

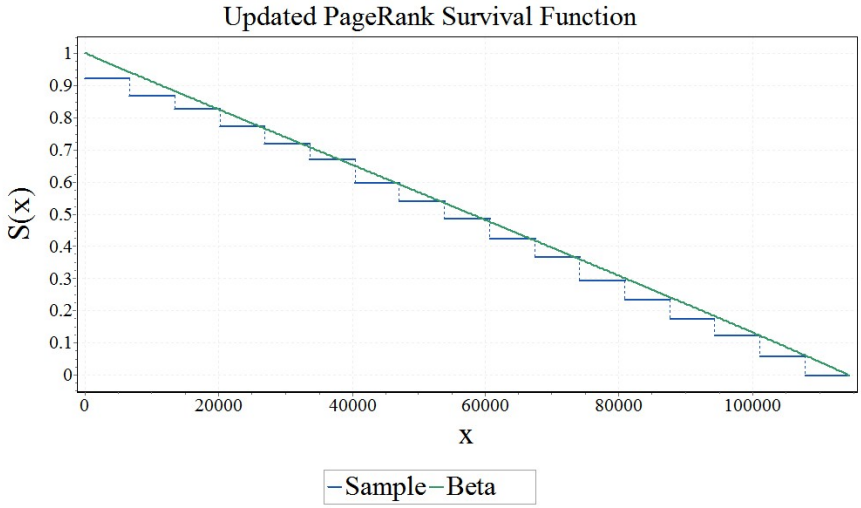


Figure 6.23: Distribution of survival function for Updated PageRank

Hazard function

The Hazard Function (HF) of a variable (here, it is the value of updated PageRank) can be expressed by the Eq. 6.7 . It is the ratio of probability density function (Eq. 6.6) with survival function. Beta distribution of the computed hazard function is presented in Fig. 6.24.

$$\int_a^b f(x)d(x) = Pr[a <= x <= b] \tag{6.6}$$

$$h(x) = f(x)/S(x) \tag{6.7}$$

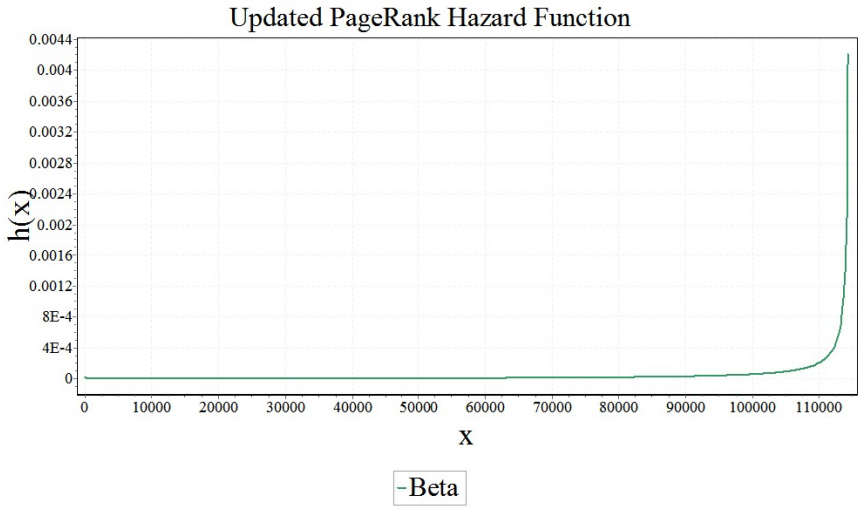


Figure 6.24: Distribution of hazard function for Updated PageRank

Cumulative Hazard function

The Cumulative Hazard Function (CHF) of a variable (here, it is the value of updated PageRank) can be expressed by the Eq. 6.8. It is the inverse of hazard function. Beta distribution over the computed CHF is presented in Fig. 6.25.

$$H(x) = \int_{-\infty}^{x} h(u)d(u) \tag{6.8}$$

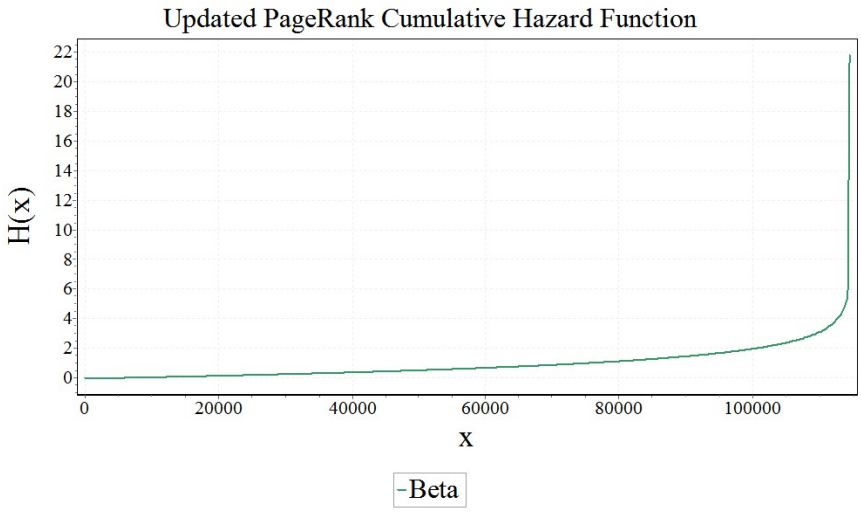


Figure 6.25: Distribution of cumulative hazard function for Updated PageRank

Table 6.30: Analysis of Updated PageRank

Parameter	**value**
Covariance of PageRank with updated PageRank	1.04701E-11
Correlation between PageRank and updated PageRank	0.986567548
F-Test between PageRank and updated PageRank	0
Harmonic mean of updated PageRank	2.94481E-09
Median of updated PageRank	2.13E-08
Mode of updated PageRank	1.79E-09
Skewness of updated PageRank	32.54902694

Different data sets used for the evaluation of updated PageRank

The experiments were performed on different data sets to evaluate the overhead of class balancing. The three different datasets used are:

- WEBSPAM-UK 2006
- WEBSPAM-UK 2007 [198]
- Microsoft Learning to Rank(MLR) [186]

It is observed from the Fig. 6.26 that the cost incurred while balancing the class increases with the increase in size of data. In some exceptional cases, such as MLR dataset, cost of class balancing does not increase with the increase in dataset. The reason behind this exception is the size of data. We can conclude that the if the dataset is large in size, the cost of balancing the class is also large.

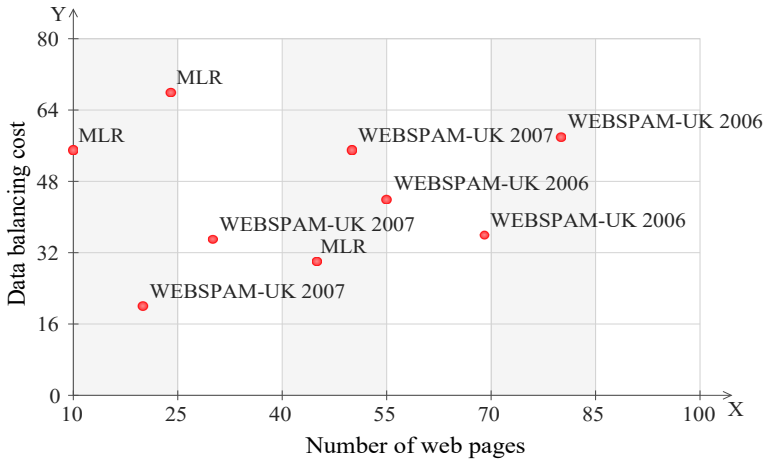


Figure 6.26: Increase in data balancing cost with the increase in data size of different datasets

6.2.8 Limitation of the proposed scheme in IoT environment

As the number of devices increases in IoT environment, so the searching of relevant information also becomes difficult. The proposed scheme not only detects the spam websites but also demotes the spam websites. The updated PageRank methodology does not allow the spam pages to enter into the process of PageRank score computation. Demoting the spam web site also increases the computational time of the PagRank score. In IoT environment, the number of devices increases, the computational time and searching time also increases. This increased cost in terms of computational time and searching time is presented in Fig. 6.27.

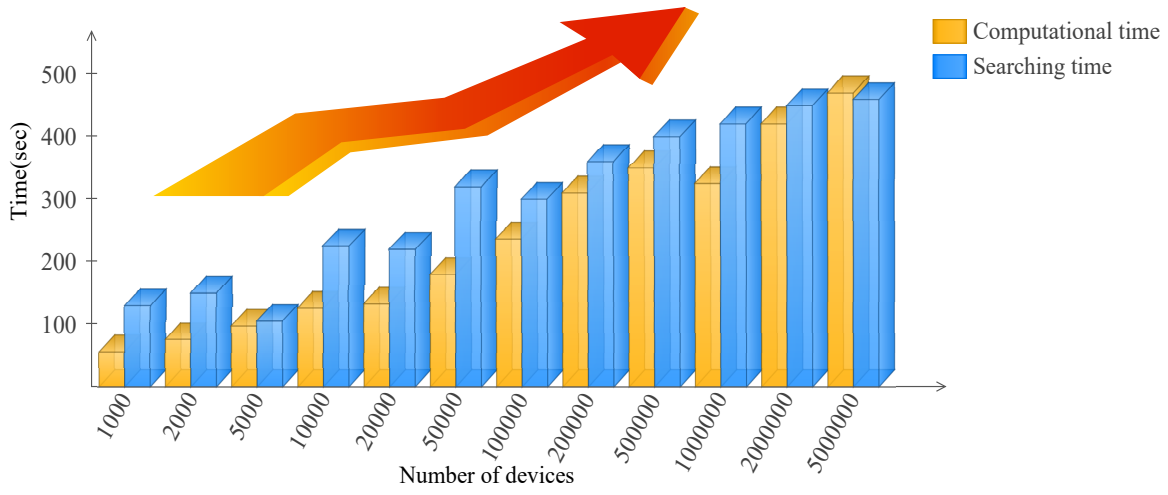


Figure 6.27: Increase in computational and searching time with increase in IoT devices

Chapter 7

Conclusion and Future Scope

This chapter gives the concluding remarks on the research work related to applying web spam detection techniques on the web pages for handling the dangling pages. Moreover, it also discusses some open issues which can be taken as future works in these domains.

7.1 Conclusion

This thesis introduces the search engine's ranking methodology. Google's ranking methodology, i.e., PageRank, is considered throughout the research work continued for accomplishing the objectives. Moreover, it also sheds light the challenges of this PageRank and illustrates the various applications of PageRank. The major issue of dangling pages in the working of the PageRank algorithm is resolved. The every if how of handling dangling web pages during the rank score computation is surveyed from the 1999 (PageRank's development time) till the date. It is proposed that the efficient handling of dangling pages is done in two ways. Both the proposed solutions are discussed.
In the first solution, we have proposed the scheme for ranking of webpages. We have paid attention to user browsing behavior, not only on the hyperlink structure of the graph. The Markov process of in-links 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 the second solution, the proposed framework enhances the search engine's ability to detect the link spam. This cognitive feature of the proposed framework, also forecasts the occurrence of spam. Spam detection framework modifies the PageRank algorithm by finding the biased and unbiased contributors in its score. The updated PageRank algorithm detects the

spam host if a single page is a spam or a website is a spam under that host. The probability by which the PageRank score is reduced, predicts the presence of spam. The spam detection models are build by the link features of web pages. The three most accurate models are analyzed and ensemble. The final spam detection classifier can achieve 99.6% accuracy. The proposed work makes a better accuracy rate when compared with the content-based spam detection classifier [25], which resulted in 2.2% false-positive rate. The spam nodes detected are mostly dangling nodes.

7.2 Future scope

The future research directions for web spam detection for efficient ranking of web pages can be focused on the following aspects:

- Web search can be improved by adopting query execution strategies such as the hub selection strategy. It not only improves the execution time but also can improve the ranking of search engine results [199].
- The trustworthiness of the webpage depends not only on the content of the page or its connectivity with other pages but also its source. Before ranking the page, it is beneficial to determine its source, which can be retrieved by using information extraction methods [200]
- Analyzing the graph structure can also help in improving search engine results by mining the relevant graph properties to be used by the data mining algorithms. Graph partitioning into different classes and sections highly affects the extraction of its properties [201].
- Ranking algorithms, such as HITS and PageRank, although they provide relevant results, ignore a very important aspect, location, which can be used for local searches based on the local popularity. It can be deployed easily in any ranking algorithm [202].
- Web graphs can be redefined depending on the diversity of topics in different webpages. Community formulation by clustering the webpages of similar topics can also improve the search results [203].
- Web searches are highly dependent on the crawling methodology and how the web directories are maintained. This task can be improved by webpage classification, which is a method of distributing webpages/sites according to their content and features [204].
- A web search is praiseworthy if it does not contain spam sites. Spam sites can be detected easily if the web graph structure is layered by adopting supervised or unsupervised approaches for its processing [205].

- Efficient graph construction by removing anomalies at the earlier stage can also improve the search results. There are many networks to be considered during the construction, but the appropriate one should be selected; otherwise, unknowingly, important edges may become disconnected [206].

- With little efforts on the server-side, host providers can prevent malicious webpages from entering the network. This effort is less problematic than the consequences based on the attacks caused by false webpages [207].

- Solving the local ranking problem can help obtain better search results. If a complete graph is not available at the time of the user query, it is always better to maintain the prediction table of URLs on the basis of previous searches performed irrespective of the location. It can also help in terms of proper resource utilization [208].

Bibliography

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

[2] A. Abbasi, F. Zahedi, S. Kaza *et al.*, "Detecting fake medical web sites using recursive trust labeling," *ACM Transactions on Information Systems (TOIS)*, vol. 30, no. 4, p. 22, 2012.

[3] Q. Qian, J. Li, J. Cai, R. Zhang, and M. Xin, "An anomaly intrusion detection method based on pagerank algorithm," in *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*. IEEE, 2013, pp. 2226–2230.

[4] M. Rafiei and A. A. Kardan, "A novel method for expert finding in online communities based on concept map and pagerank," *Human-centric computing and information sciences*, vol. 5, no. 1, p. 1, 2015.

[5] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen, "Combating web spam with trustrank," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 2004, pp. 576–587.

[6] B. Wu, V. Goel, and B. D. Davison, "Topical trustrank: Using topicality to combat web spam," in *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006, pp. 63–72.

[7] A. Rungsawang, K. Puntumapon, and B. Manaskasemsak, "Un-biasing the link farm effect in pagerank computation," in *21st International Conference on Advanced Information Networking and Applications (AINA'07)*. IEEE, 2007.

[8] B. Yang, H. Chen, X. Zhao, M. Naka, and J. Huang, "On characterizing and computing the diversity of hyperlinks for anti-spamming page ranking," *Knowledge-Based Systems*, vol. 77, 2015.

[9] Q. Gan and T. Suel, "Improving web spam classifiers using link structure," in *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web*. ACM, 2007, pp. 17–20.

[10] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri, "Know your neighbors: Web spam detection using the web topology," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 423–430.

[11] G. Geng, C. Wang, and Q. Li, "Improving web spam detection with re-extracted features," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 1119–1120.

[12] Y. Liu, R. Cen, M. Zhang, S. Ma, and 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.

[13] Y. Liu, F. Chen, W. Kong, H. Yu, M. Zhang, S. Ma, and L. Ru, "Identifying web spam with the wisdom of the crowds," *ACM Transactions on the Web (TWEB)*, vol. 6, no. 1, 2012.

[14] J. Martinez-Romo and L. Araujo, "Web spam identification through language model analysis," in *Proceedings of the 5th international workshop on adversarial information retrieval on the web*. ACM, 2009, pp. 21–28.

[15] L. Araujo and J. Martinez-Romo, "Web spam detection: new classification features based on qualified link analysis and language models," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 581–590, 2010.

[16] N. Dai, B. D. Davison, and X. Qi, "Looking into the past to better classify web spam," in *Proceedings of the 5th international workshop on adversarial information retrieval on the web*. ACM, 2009, pp. 1–8.

[17] W. Wang, G. Zeng, and D. Tang, "Using evidence based content trust model for spam detection," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5599–5606, 2010.

[18] C. Wei, Y. Liu, M. Zhang, S. Ma, L. Ru, and K. Zhang, "Fighting against web spam: a novel propagation method based on click-through data," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2012, pp. 395–404.

[19] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, and 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.

[20] P. Heymann, G. Koutrika, and H. Garcia-Molina, "Fighting spam on social web sites: A survey of approaches and future challenges," *IEEE Internet Computing*, vol. 11, no. 6, 2007.

[21] T. S. Tung, N. A. Yahaya, and S. Mustapha, "Multi-level link structure analysis techniqiue for detecting link farm spam pages," in *Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology*. IEEE Computer Society, 2006, pp. 614–617.

[22] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. P. Gummadi, "Understanding and combating link farming in the twitter social network," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 61–70.

[23] Z. Gyöngyi and H. Garcia-Molina, "Link spam alliances," in *Proceedings of the 31st international conference on Very large data bases*. VLDB Endowment, 2005, pp. 517–528.

[24] K. L. Goh, R. K. Patchmuthu, and A. K. Singh, "Link-based web spam detection using weight properties," *Journal of Intelligent Information Systems*, vol. 43, no. 1, pp. 129–145, 2014.

[25] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly, "Detecting spam web pages through content analysis," in *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006, pp. 83–92.

[26] Y.-M. Wang, M. Ma, Y. Niu, and H. Chen, "Spam double-funnel: Connecting web spammers with advertisers," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 291–300.

[27] H. Tran, T. Hornbeck, V. Ha-Thuc, J. Cremer, and P. Srinivasan, "Spam detection in online classified advertisements," in *Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality*. ACM, 2011.

[28] A. Heydari, M. ali Tavakoli, N. Salim, and Z. Heydari, "Detection of review spam: A survey," *Expert Systems with Applications*, vol. 42, no. 7, 2015.

[29] G. M. Del Corso, A. Gullí, and F. Romani, "Comparison of krylov subspace methods on the pagerank problem," *Journal of Computational and Applied Mathematics*, vol. 210, no. 1, pp. 159–166, 2007.

[30] K. S. Kim and Y. S. Choi, "Incremental iteration method for fast pagerank computation," in *Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication*. ACM, 2015, p. 80.

[31] Z. Liu, N. Emad, S. B. Amor, and M. Lamure, "Pagerank computation using a multiple implicitly restarted arnoldi method for modeling epidemic spread," *International Journal of Parallel Programming*, vol. 43, no. 6, pp. 1028–1053, 2015.

[32] S. J. Kim and S. H. Lee, "An improved computation of the pagerank algorithm," in *European Conference on Information Retrieval*. Springer, 2002, pp. 73–85.

[33] Q. Yu, Z. Miao, G. Wu, and Y. Wei, "Lumping algorithms for computing googles pagerank and its derivative, with attention to unreferenced nodes," *Information retrieval*, vol. 15, no. 6, 2012.

[34] V. L. Praba and T. Vasantha, "Efficient hyperlink analysis using robust proportionate prestige score in pagerank algorithm," *Applied Soft Computing*, vol. 24, 2014.

[35] X. Wang, T. Tao, J.-T. Sun, A. Shakery, and C. Zhai, "Dirichletrank: Solving the zero-one gap problem of pagerank," *ACM Transactions on Information Systems (TOIS)*, vol. 26, no. 2, p. 10, 2008.

[36] P. K. Desikan, N. Pathak, J. Srivastava, and V. Kumar, "Divide and conquer approach for efficient pagerank computation," in *Proceedings of the 6th international conference on Web engineering*. ACM, 2006, pp. 233–240.

[37] A. Makkar and N. Kumar, "User behavior analysis-based smart energy management for webpage ranking: Learning automata-based solution," *Sustainable Computing: Informatics and Systems*, 2018.

[38] C. Wang, A. Kalra, C. Borcea, and 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.

[39] E. Agichtein, E. Brill, S. Dumais, and 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.

[40] Netcraft, "March 2016 web server survey," https://news.netcraft.com/archives/2016/03/18/march-2016-web-server-survey.html, 2016 (accessed November 10, 2016).

[41] W. R. algorithm, "Internet users," http://www.internetlivestats.com/internet-users/, 2016 (accessed November 10, 2016).

[42] A. Celesti, M. Fazio, and M. Villari, "A study on join operations in mongodb preserving collections data models for future internet applications," *Future Internet*, vol. 11, no. 4, p. 83, 2019.

[43] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas, "Link analysis ranking: algorithms, theory, and experiments," *ACM Transactions on Internet Technology (TOIT)*, vol. 5, no. 1, pp. 231–297, 2005.

[44] N. Applications, "Desktop search engine market share," https://www.netmarketshare.com/search-engine-market-share.aspx?qprid=4qpcustomd=0, 2016 (accessed November 10, 2016).

[45] A.-J. Su, Y. C. Hu, A. Kuzmanovic, and C.-K. Koh, "How to improve your search engine ranking: Myths and reality," *ACM Transactions on the Web (TWEB)*, vol. 8, no. 2, p. 8, 2014.

[46] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." 1999.

[47] M. Bianchini, M. Gori, and F. Scarselli, "Inside pagerank," *ACM Transactions on Internet Technology (TOIT)*, vol. 5, no. 1, pp. 92–128, 2005.

[48] D. Hawking, "Web search engines. part 1," *Computer*, vol. 39, no. 6, pp. 86–88, 2006.

[49] J. Cho and H. Garcia-Molina, "Estimating frequency of change," *ACM Transactions on Internet Technology (TOIT)*, vol. 3, no. 3, pp. 256–290, 2003.

[50] Q. Tan and P. Mitra, "Clustering-based incremental web crawling," *ACM Transactions on Information Systems (TOIS)*, vol. 28, no. 4, p. 17, 2010.

[51] I. S. Altingovde and O. Ulusoy, "Exploiting interclass rules for focused crawling," *IEEE Intelligent Systems*, vol. 19, no. 6, pp. 66–73, 2004.

[52] J. Hirai, S. Raghavan, H. Garcia-Molina, and A. Paepcke, "Webbase: A repository of web pages," *Computer Networks*, vol. 33, no. 1, pp. 277–293, 2000.

[53] T. Haveliwala, A. Gionis, and P. Indyk, "Scalable techniques for clustering the web," 2000.

[54] F. Hunt, "Total number of websites and size of the internet as of 2013," http://www.factshunt.com/2014/01/total-number-of-websites-size-of.html, 2013 (accessed November 10, 2016).

[55] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM (JACM)*, vol. 46, no. 5, pp. 604–632, 1999.

[56] R. Lempel and S. Moran, "Salsa: the stochastic approach for link-structure analysis," *ACM Transactions on Information Systems (TOIS)*, vol. 19, no. 2, pp. 131–160, 2001.

[57] A. M. Z. Bidoki and N. Yazdani, "Distancerank: An intelligent ranking algorithm for web pages," *Information Processing & Management*, vol. 44, no. 2, pp. 877–892, 2008.

[58] T. Agryzkov, J. L. Oliver, L. Tortosa, and J. F. Vicent, "An algorithm for ranking the nodes of an urban network based on the concept of pagerank vector," *Applied Mathematics and Computation*, vol. 219, no. 4, pp. 2186–2193, 2012.

[59] S. Sharma, Y. Xu, A. Verma, and B. K. Panigrahi, "Time-coordinated multi-energy management of smart buildings under uncertainties," *IEEE Transactions on Industrial Informatics*, 2019.

[60] J. X. Parreira, C. Castillo, D. Donato, S. Michel, and G. Weikum, "The juxtaposed approximate pagerank method for robust pagerank approximation in a peer-to-peer web search network," *The VLDB Journal*, vol. 17, no. 2, pp. 291–313, 2008.

[61] L. Li, K. Ota, and M. Dong, "Deep learning for smart industry: Efficient manufacture inspection system with fog computing," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4665–4673, 2018.

[62] B. Perozzi, C. McCubbin, and J. T. Halbert, "Scalable graph clustering with parallel approximate pagerank," *Social Network Analysis and Mining*, vol. 4, no. 1, pp. 1–11, 2014.

[63] R. Interdonato and A. Tagarelli, "Multi-relational pagerank for tree structure sense ranking," *World Wide Web*, vol. 18, no. 5, pp. 1301–1329, 2015.

[64] F. Zhu, Y. Fang, K. C.-C. Chang, and J. Ying, "Scheduled approximation for personalized pagerank with utility-based hub selection," *The VLDB Journal*, vol. 24, no. 5, pp. 655–679, 2015.

[65] F. L. Cruz, C. G. Vallejo, F. Enrı, J. A. Troyano *et al.*, "Polarityrank: Finding an equilibrium between followers and contraries in a network," *Information Processing & Management*, vol. 48, no. 2, pp. 271–282, 2012.

[66] F. J. Ortega, J. A. Troyano, F. L. Cruz, C. G. Vallejo, and F. EnríQuez, "Propagation of trust and distrust for the detection of trolls in a social network," *Computer Networks*, vol. 56, no. 12, pp. 2884–2895, 2012.

[67] X. Tang and C. C. Yang, "Ranking user influence in healthcare social media," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, no. 4, p. 73, 2012.

[68] E. J. Yates and L. C. Dixon, "Pagerank as a method to rank biomedical literature by importance," *Source code for biology and medicine*, vol. 10, no. 1, p. 1, 2015.

[69] G. Wu, Y. Zhang, and Y. Wei, "Accelerating the arnoldi-type algorithm for the pagerank problem and the proteinrank problem," *Journal of Scientific Computing*, vol. 57, no. 1, pp. 74–104, 2013.

[70] Z. Yao, P. Mark, and M. Rabbat, "Anomaly detection using proximity graph and pagerank algorithm," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1288–1300, 2012.

[71] C. C. Yang and K. Chan, "Retrieving multimedia web objects based on pagerank algorithm," in *Special interest tracks and posters of the 14th international conference on World Wide Web*. ACM, 2005, pp. 906–907.

[72] J. Abernethy, O. Chapelle, and C. Castillo, "Graph regularization methods for web spam detection," *Machine Learning*, vol. 81, no. 2, pp. 207–225, 2010.

[73] P. Bedi *et al.*, "Parallel proactive cross domain context aware recommender system," *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 3, pp. 1521–1533, 2018.

[74] S. Sharma, J. S. Lather, and M. Dave, "Semantic approach for web service classification using machine learning and measures of semantic relatedness," *Service Oriented Computing and Applications*, vol. 10, no. 3, pp. 221–231, 2016.

[75] M. Erdélyi, A. Garzó, and A. A. Benczúr, "Web spam classification: a few features worth more," in *Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality*. ACM, 2011, pp. 27–34.

[76] V. M. Prieto, M. Álvarez, and F. Cacheda, "Saad, a content based web spam analyzer and detector," *Journal of Systems and Software*, vol. 86, no. 11, pp. 2906–2918, 2013.

[77] X. Liu, Y. Wang, S. Zhu, and H. Lin, "Combating web spam through trust–distrust propagation with confidence," *Pattern Recognition Letters*, vol. 34, no. 13, pp. 1462–1469, 2013.

[78] J. Fdez-Glez, D. Ruano-Ordas, J. R. Méndez, F. Fdez-Riverola, R. Laza, and R. Pavón, "A dynamic model for integrating simple web spam classification techniques," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7969–7978, 2015.

[79] M. Chakraborty, S. Pal, R. Pramanik, and C. R. Chowdary, "Recent developments in social spam detection and combating techniques: A survey," *Information Processing & Management*, 2016.

[80] B. Wu and B. D. Davison, "Identifying link farm spam pages," in *Special interest tracks and posters of the 14th international conference on World Wide Web*. ACM, 2005, pp. 820–829.

[81] B. Wu and K. Chellapilla, "Extracting link spam using biased random walks from spam seed sets," in *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web*. ACM, 2007, pp. 37–44.

[82] X. Zhang, W. Liang, S. Zhu, and B. Han, "Automatic seed set expansion for trust propagation based anti-spam algorithms," *Information Sciences*, vol. 232, pp. 167–187, 2013.

[83] W. Lee, "Discriminating biased web manipulations in terms of link oriented measures," in *International Symposium on Computer and Information Sciences*. Springer, 2005, pp. 585–594.

[84] L. Becchetti, C. Castillo, D. Donato, R. Baeza-Yates, and S. Leonardi, "Link analysis for web spam detection," *ACM Transactions on the Web (TWEB)*, vol. 2, no. 1, p. 2, 2008.

[85] Y.-j. Chung, M. Toyoda, and M. Kitsuregawa, "Detecting link hijacking by web spammers," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2009, pp. 339–350.

[86] ——, "Identifying spam link generators for monitoring emerging web spam," in *Proceedings of the 4th workshop on Information credibility*. ACM, 2010, pp. 51–58.

[87] O.-M. Eugene, Z. Fengli, O. K. Adu-Boahen, and B. E. Yellakuor, "Spam detection through link authorization from neighboring nodes," in *e-Technologies and Networks for Development (ICeND), 2015 Forth International Conference on*. IEEE, 2015, pp. 1–6.

[88] A. M. Fard and K. Wang, "Neighborhood randomization for link privacy in social network analysis," *World Wide Web*, vol. 18, no. 1, pp. 9–32, 2015.

[89] Z. Gyongyi, P. Berkhin, H. Garcia-Molina, and J. Pedersen, "Link spam detection based on mass estimation," in *Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment, 2006, pp. 439–450.

[90] D. Zhou, C. J. Burges, and T. Tao, "Transductive link spam detection," in *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web*. ACM, 2007, pp. 21–28.

[91] H. Saito, M. Toyoda, M. Kitsuregawa, and K. Aihara, "A large-scale study of link spam detection by graph algorithms," in *Proceedings of the 3rd international workshop on Adversarial information retrieval on the web*. ACM, 2007, pp. 45–48.

[92] B. Zhou and J. Pei, "Link spam target detection using page farms," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 3, no. 3, p. 13, 2009.

[93] J. Caverlee, S. Webb, L. Liu, and W. B. Rouse, "A parameterized approach to spam-resilient link analysis of the web," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 10, pp. 1422–1438, 2009.

[94] M. Egele, C. Kolbitsch, and C. Platzer, "Removing web spam links from search engine results," *Journal in Computer Virology*, vol. 7, no. 1, pp. 51–62, 2011.

[95] M. S. Pera and Y.-K. Ng, "Identifying spam web pages based on content similarity," in *International Conference on Computational Science and Its Applications*. Springer, 2008, pp. 204–219.

[96] A. Bhattarai, V. Rus, and D. Dasgupta, "Characterizing comment spam in the blogosphere through content analysis," in *Computational Intelligence in Cyber Security, 2009. CICS'09. IEEE Symposium on*. IEEE, 2009, pp. 37–44.

[97] F. J. Ortega, C. Macdonald, J. A. Troyano, and F. Cruz, "Spam detection with a content-based random-walk algorithm," in *Proceedings of the 2nd international workshop on Search and mining user-generated contents*.

[98] V. M. Prieto, M. Álvarez, R. López-García, and F. Cacheda, "Analysis and detection of web spam by means of web content," in *Information Retrieval Facility Conference*. Springer, 2012, pp. 43–57.

[99] C. Dong and B. Zhou, "Effectively detecting content spam on the web using topical diversity measures," in *Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01*. IEEE Computer Society, 2012, pp. 266–273.

[100] H. Ji and H. Zhang, "Analysis on the content features and their correlation of web pages for spam detection," *China Communications*, vol. 12, no. 3, pp. 84–94, 2015.

[101] I. Ivanov, P. Vajda, J.-S. Lee, and T. Ebrahimi, "In tags we trust: Trust modeling in social tagging of multimedia content," *IEEE Signal Processing Magazine*, vol. 29, no. 2, pp. 98–107, 2012.

[102] S. Moulik, S. Misra, and C. Chakraborty, "Performance evaluation and delay-power trade-off analysis of zigbee protocol," *IEEE Transactions on Mobile Computing*, vol. 18, no. 2, pp. 404–416, 2018.

[103] H. Costa, F. Benevenuto, and L. H. Merschmann, "Detecting tip spam in location-based social networks," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM, 2013.

[104] H. Shen and Z. Li, "Leveraging social networks for effective spam filtering," *IEEE Transactions on Computers*, vol. 63, no. 11, 2014.

[105] S. Xu, H. Jiang, and 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.

[106] P. Hayati, K. Chai, V. Potdar, and A. Talevski, "Behaviour-based web spambot detection by utilising action time and action frequency," *Computational Science and Its Applications–ICCSA 2010*, pp. 351–360, 2010.

[107] V. Dave, S. Guha, and Y. Zhang, "Measuring and fingerprinting click-spam in ad networks," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 175–186, 2012.

[108] M. Bilenko and 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.

[109] X. Yi, L. Hong, E. Zhong, N. N. Liu, and S. Rajan, "Beyond clicks: dwell time for personalization," in *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 2014, pp. 113–120.

[110] G. Dupret and 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.

[111] Y. Liu, T.-Y. Liu, B. Gao, Z. Ma, and H. Li, "A framework to compute page importance based on user behaviors," *Information retrieval*, vol. 13, no. 1, 2010.

[112] A. A. Benczur, K. Csalogany, T. Sarlos, and M. Uher, "Spamrank–fully automatic link spam detection work in progress," in *Proceedings of the first international workshop on adversarial information retrieval on the web*, 2005.

[113] Q. Chen, S.-N. Yu, and S. Cheng, "Link variable trustrank for fighting web spam," in *Computer Science and Software Engineering, 2008 International Conference on*, vol. 4. IEEE, 2008, pp. 1004–1007.

[114] T. G. Kolda and M. J. Procopio, "Generalized badrank with graduated trust," *Sandia National Laboratories, California*, 2009.

[115] A. Goh Kwang Leng, A. Kumar Singh, P. Ravi Kumar, and A. Mohan, "Tprank: Contend with web spam using trust propagation," *Cybernetics and Systems*, vol. 45, no. 4, pp. 307–323, 2014.

[116] X. Zhang, Y. Wang, N. Mou, and W. Liang, "Propagating both trust and distrust with target differentiation for combating link-based web spam," *ACM Transactions on the Web (TWEB)*, vol. 8, no. 3, p. 15, 2014.

[117] A. G. K. Leng, P. R. Kumar, A. K. Singh, and A. Mohan, "Link-based spam algorithms in adversarial information retrieval," *Cybernetics and Systems*, vol. 43, no. 6, pp. 459–475, 2012.

[118] C. Liang, L. Ru, and X. Zhu, "R-spamrank: a spam detection algorithm based on link analysis," *Journal of Computational Information Systems*, vol. 3, no. 4, pp. 1705–1712, 2007.

[119] H. Yang, I. King, and M. R. Lyu, "Diffusionrank: a possible penicillin for web spamming," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 431–438.

[120] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. A. Baeza-Yates, "Link-based characterization and detection of web spam." in *AIRWeb*, 2006, pp. 1–8.

[121] J. Abernethy, O. Chapelle, and C. Castillo, "Web spam identification through content and hyperlinks," in *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*. ACM, 2008, pp. 41–44.

[122] L. Becchetti, C. Castillo, D. Donato, S. Leonardi, and R. Baeza-Yates, "Web spam detection: Link-based and content-based techniques," in *The European integrated project dynamically evolving, large scale information systems (DELIS): Proceedings of the final workshop*, vol. 222, 2008, pp. 99–113.

[123] J. Piskorski, M. Sydow, and D. Weiss, "Exploring linguistic features for web spam detection: a preliminary study," in *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*. ACM, 2008, pp. 25–28.

[124] M. Mahmoudi, A. Yari, and S. Khadivi, "Web spam detection based on discriminative content and link features," in *Telecommunications (IST), 2010 5th International Symposium on*. IEEE, 2010, pp. 542–546.

[125] M. Erdélyi and A. A. Benczúr, "Temporal analysis for web spam detection: An overview." in *TWAW*, 2011, pp. 17–24.

[126] S. P. Algur and N. T. Pendari, "Hybrid spamicity score approach to web spam detection," in *Pattern Recognition, Informatics and Medical Engineering (PRIME), 2012 International Conference on*. IEEE, 2012, pp. 36–40.

[127] X. Gongwen, L. Xiaomei, Z. Zhijun, and X. LiNa, "Web spam detection based on link diversity and content features," *International Journal of Security and Its Applications*, vol. 10, no. 7, pp. 363–372, 2016.

[128] R. Kumar Roul, S. Rohan Asthana, M. Shah, and D. Parikh, "Detection of spam web page using content and link-based techniques: A combined approach," *Sadhana*, vol. 41, no. 2, pp. 193–202, 2016.

[129] A. Makkar and S. Goel, "Spammer classification using ensemble methods over content-based features," in *Proceedings of Sixth International Conference on Soft Computing for Problem Solving*. Springer, 2017, pp. 1–9.

[130] A. N. Langville and C. D. Meyer, "Deeper inside pagerank," *Internet Mathematics*, vol. 1, no. 3, pp. 335–380, 2004.

[131] A. Z. Broder, R. Lempel, F. Maghoul, and J. Pedersen, "Efficient pagerank approximation via graph aggregation," *Information Retrieval*, vol. 9, no. 2, pp. 123–138, 2006.

[132] H. Ishii and R. Tempo, "Distributed randomized algorithms for the pagerank computation," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 1987–2002, 2010.

[133] H. Ishii, R. Tempo, and E.-W. Bai, "A web aggregation approach for distributed randomized pagerank algorithms," *IEEE Transactions on automatic control*, vol. 57, no. 11, pp. 2703–2717, 2012.

[134] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub, "Extrapolation methods for accelerating pagerank computations," in *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003, pp. 261–270.

[135] S. Kamvar, T. Haveliwala, and G. Golub, "Adaptive methods for the computation of pagerank," *Linear Algebra and its Applications*, vol. 386, pp. 51–65, 2004.

[136] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova, "Monte carlo methods in pagerank computation: When one iteration is sufficient," *SIAM Journal on Numerical Analysis*, vol. 45, no. 2, pp. 890–904, 2007.

[137] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, C. Liao, and F. Diaz, "Towards recency ranking in web search," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 11–20.

[138] L. Li, G. Xu, Y. Zhang, and M. Kitsuregawa, "Random walk based rank aggregation to improving web search," *Knowledge-Based Systems*, vol. 24, no. 7, pp. 943–951, 2011.

[139] J. Eustace, X. Wang, and J. Li, "Approximating web communities using subspace decomposition," *Knowledge-Based Systems*, vol. 70, pp. 118–127, 2014.

[140] J. Zhao, P. Jin, Q. Zhang, and R. Wen, "Exploiting location information for web search," *Computers in Human Behavior*, vol. 30, pp. 378–388, 2014.

[141] C. Brezinski and M. Redivo-Zaglia, "The pagerank vector: properties, computation, approximation, and acceleration," *SIAM Journal on Matrix Analysis and Applications*, vol. 28, no. 2, pp. 551–575, 2006.

[142] G. Wu and Y. Wei, "An arnoldi-extrapolation algorithm for computing pagerank," *Journal of Computational and Applied Mathematics*, vol. 234, no. 11, pp. 3196–3212, 2010.

[143] J. Arnal, H. Migallón, V. Migallón, J. A. Palomino, and J. Penadés, "Parallel relaxed and extrapolated algorithms for computing pagerank," *The Journal of Supercomputing*, vol. 70, no. 2, pp. 637–648, 2014.

[144] W. Liu, G. Li, and J. Cheng, "Fast pagerank approximation by adaptive sampling," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 127–146, 2015.

[145] C. Gu and L. Wang, "On the multi-splitting iteration method for computing pagerank," *Journal of Applied Mathematics and Computing*, vol. 42, no. 1-2, pp. 479–490, 2013.

[146] T. H. Haveliwala, "Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search," *IEEE transactions on knowledge and data engineering*, vol. 15, no. 4, pp. 784–796, 2003.

[147] A. N. Langville and C. D. Meyer, "A reordering for the pagerank problem," *SIAM Journal on Scientific Computing*, vol. 27, no. 6, pp. 2112–2120, 2006.

[148] Y.-M. Bu and T.-Z. Huang, "An adaptive reordered method for computing pagerank," *Journal of Applied Mathematics*, vol. 2013, 2013.

[149] N. Eiron, K. S. McCurley, and J. A. Tomlin, "Ranking the web frontier," in *Proceedings of the 13th international conference on World Wide Web*. ACM, 2004, pp. 309–318.

[150] L. Li, X. Chen, and Y. Song, "The pagerank model of minimal irreducible adjustment and its lumping method," *Journal of Applied Mathematics and Computing*, vol. 42, no. 1-2, 2013.

[151] C. P.-C. Lee, G. H. Golub, and S. A. Zenios, "A fast two-stage algorithm for computing pagerank and its extensions," *Scientific Computation and Computational Mathematics*, vol. 1, no. 1, 2003.

[152] I. C. Ipsen and T. M. Selee, "Pagerank computation, with special attention to dangling nodes," *SIAM Journal on Matrix Analysis and Applications*, vol. 29, no. 4, 2007.

[153] Y. Lin, X. Shi, and Y. Wei, "On computing pagerank via lumping the google matrix," *Journal of Computational and Applied Mathematics*, vol. 224, no. 2, 2009.

[154] R. Andersen, C. Borgs, J. Chayes, J. Hopcroft, K. Jain, V. Mirrokni, and 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, pp. 69–76.

[155] J. T. Bradley, D. V. de Jager, W. J. Knottenbelt, and A. Trifunović, "Hypergraph partitioning for faster parallel pagerank computation," in *Formal Techniques for Computer Systems and Business Processes*. Springer, 2005, pp. 155–171.

[156] A. Cevahir, C. Aykanat, A. Turk, and B. B. Cambazoglu, "Site-based partitioning and repartitioning techniques for parallel pagerank computation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 786–802, 2011.

[157] B. C. Csáji, R. M. Jungers, and V. D. Blondel, "Pagerank optimization by edge selection," *Discrete Applied Mathematics*, vol. 169, pp. 73–87, 2014.

[158] T. Maehara, T. Akiba, Y. Iwata, and K.-i. Kawarabayashi, "Computing personalized pagerank quickly by exploiting graph structures," *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1023–1034, 2014.

[159] W. Dong, F. Wang, Y. Huang, G. Xu, Z. Guo, X. Fu, and K. Fu, "An advanced prepositioning method for the force-directed graph visualization based on pagerank algorithm," *Computers & Graphics*, vol. 47, pp. 24–33, 2015.

[160] T. Qiu, Y. Zhang, D. Qiao, X. Zhang, M. L. Wymore, and A. K. Sangaiah, "A robust time synchronization scheme for industrial internet of things," *IEEE Transactions on Industrial Informatics*, 2017.

[161] T. Qiu, R. Qiao, M. Han, A. K. Sangaiah, and I. Lee, "A lifetime-enhanced data collecting scheme for the internet of things," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 132–137, 2017.

[162] A. Alnoman, G. H. Carvalho, A. Anpalagan, and I. Woungang, "Energy efficiency on fully cloudified mobile networks: Survey, challenges, and open issues," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1271–1291, 2017.

[163] N. Kumar, S. Tyagi, and D.-J. Deng, "La-eehsc: Learning automata-based energy efficient heterogeneous selective clustering for wireless sensor networks," *Journal of Network and Computer Applications*, vol. 46, pp. 264–279, 2014.

[164] W. Saad, Z. Han, A. Hjorungnes, D. Niyato, and E. Hossain, "Coalition formation games for distributed cooperation among roadside units in vehicular networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 1, pp. 48–60, 2011.

[165] S. Misra, B. J. Oommen, S. Yanamandra, and M. S. Obaidat, "Random early detection for congestion avoidance in wired networks: a discretized pursuit learning-automata-like solution," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 40, no. 1, pp. 66–76, 2010.

[166] N. Kumar, S. Misra, and M. S. Obaidat, "Collaborative learning automata-based routing for rescue operations in dense urban regions using vehicular sensor networks," *IEEE Systems Journal*, vol. 9, no. 3, pp. 1081–1090, 2015.

[167] N. Kumar, J.-H. Lee, and J. J. Rodrigues, "Intelligent mobile video surveillance system as a bayesian coalition game in vehicular sensor networks: Learning automata approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1148–1161, 2015.

[168] N. Kumar, K. Kaur, A. Jindal, and J. J. Rodrigues, "Providing healthcare services on-the-fly using multi-player cooperation game theory in internet of vehicles (iov) environment," *Digital Communications and Networks*, vol. 1, no. 3, pp. 191–203, 2015.

[169] M. Fazio, R. Ranjan, M. Girolami, J. Taheri, S. Dustdar, and M. Villari, "A note on the convergence of iot, edge, and cloud computing in smart cities," *IEEE Cloud Computing*, vol. 5, no. 5, pp. 22–24, 2018.

[170] K. Ashton *et al.*, "That internet of things thing," *RFID journal*, vol. 22, no. 7, pp. 97–114, 2009.

[171] D. Bandyopadhyay and J. Sen, "Internet of things: Applications and challenges in technology and standardization," *Wireless Personal Communications*, vol. 58, no. 1, pp. 49–69, 2011.

[172] Q. Wu, G. Ding, Y. Xu, S. Feng, Z. Du, J. Wang, and K. Long, "Cognitive internet of things: a new paradigm beyond connection," *IEEE Internet of Things Journal*, vol. 1, no. 2, pp. 129–143, 2014.

[173] P. Vlacheas, R. Giaffreda, V. Stavroulaki, D. Kelaidonis, V. Foteinos, G. Poulios, P. Demestichas, A. Somov, A. R. Biswas, and K. Moessner, "Enabling smart cities through a cognitive management framework for the internet of things," *IEEE communications magazine*, vol. 51, no. 6, pp. 102–111, 2013.

[174] M. Zhang, H. Zhao, R. Zheng, Q. Wu, and W. Wei, "Cognitive internet of things: concepts and application example," *International journal of computer science issues*, vol. 9, no. 6, pp. 151–158, 2012.

[175] J. Mitola and G. Q. Maguire, "Cognitive radio: making software radios more personal," *IEEE personal communications*, vol. 6, no. 4, pp. 13–18, 1999.

[176] S. Haykin, "Cognitive radio: brain-empowered wireless communications," *IEEE journal on selected areas in communications*, vol. 23, no. 2, pp. 201–220, 2005.

[177] D. E. Rosen and E. Purinton, "Website design: Viewing the web as a cognitive landscape," *Journal of Business Research*, vol. 57, no. 7, pp. 787–794, 2004.

[178] A. Aijaz and A. H. Aghvami, "Cognitive machine-to-machine communications for internet-of-things: A protocol stack perspective," *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 103–112, 2015.

[179] V. Foteinos, D. Kelaidonis, G. Poulios, P. Vlacheas, V. Stavroulaki, and P. Demestichas, "Cognitive management for the internet of things: A framework for enabling autonomous applications," *IEEE Vehicular Technology Magazine*, vol. 8, no. 4, pp. 90–99, 2013.

[180] S. Sasidharan, A. Somov, A. R. Biswas, and R. Giaffreda, "Cognitive management framework for internet of things:a prototype implementation," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 538–543.

[181] S. Xiao, H. Yu, Y. Wu, Z. Peng, and Y. Zhang, "Self-evolving trading strategy integrating internet of things and big data," *IEEE Internet of Things Journal*, 2017.

[182] D. Guinard, V. Trifa, and E. Wilde, "A resource oriented architecture for the web of things," in *Internet of Things (IOT), 2010*. IEEE, 2010, pp. 1–8.

[183] M. Noor, H. Abbas, and W. B. Shahid, "Countering cyber threats for industrial applications: An automated approach for malware evasion detection and analysis," *Journal of Network and Computer Applications*, vol. 103, pp. 249–261, 2018.

[184] N. Applications, "Desktop search engine market share," https://www.netmarketshare.com/search-engine-market-share.aspx?qprid=4qpcustomd=0, 2016 (accessed January 17, 2018).

[185] Z. Gyongyi and H. Garcia-Molina, "Web spam taxonomy," in *First international workshop on adversarial information retrieval on the web (AIRWeb 2005)*, 2005.

[186] T. Qin and T. Liu, "Introducing LETOR 4.0 datasets," *CoRR*, vol. abs/1306.2597, 2013. [Online]. Available: http://arxiv.org/abs/1306.2597

[187] Y. Zhang, M. Chen, N. Guizani, D. Wu, and V. C. Leung, "Sovcan: Safety-oriented vehicular controller area network," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 94–99, 2017.

[188] P. Hart, "The condensed nearest neighbor rule (corresp.)," *IEEE transactions on information theory*, vol. 14, no. 3, pp. 515–516, 1968.

[189] I. Tomek, "Two modifications of cnn," *IEEE Trans. Systems, Man and Cybernetics*, vol. 6, pp. 769–772, 1976.

[190] A. S. Solberg and R. Solberg, "A large-scale evaluation of features for automatic detection of oil spills in ers sar images," in *Geoscience and Remote Sensing Symposium, 1996. IGARSS'96.'Remote Sensing for a Sustainable Future.', International*, vol. 3. IEEE, 1996, pp. 1484–1486.

[191] M. Kubat, S. Matwin *et al.*, "Addressing the curse of imbalanced training sets: one-sided selection," in *ICML*, vol. 97. Nashville, USA, 1997, pp. 179–186.

[192] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine learning*, vol. 30, no. 2-3, pp. 195–215, 1998.

[193] C. X. Ling and C. Li, "Data mining for direct marketing: Problems and solutions." in *KDD*, vol. 98, 1998, pp. 73–79.

[194] N. Japkowicz *et al.*, "Learning from imbalanced data sets: a comparison of various strategies," in *AAAI workshop on learning from imbalanced data sets*, vol. 68. Menlo Park, CA, 2000, pp. 10–15.

[195] J. Laurikkala, "Improving identification of difficult small classes by balancing class distribution," *Artificial Intelligence in Medicine*, pp. 63–66, 2001.

[196] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[197] G. E. Batista, A. L. Bazzan, and M. C. Monard, "Balancing training data for automated annotation of keywords: a case study." in *WOB*, 2003, pp. 10–18.

[198] L. B. P. B. M. S. Carlos Castillo, Debora Donato and S. Vigna, "Web spam collections," http://chato.cl/webspam/datasets/, 2007 (accessed November 10, 2016).

[199] S. Chakrabarti, A. Pathak, and M. Gupta, "Index design and query processing for graph conductance search," *The VLDB Journal*, vol. 20, no. 3, pp. 445–470, 2011.

[200] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang, "Knowledge-based trust: Estimating the trustworthiness of web sources," *Proceedings of the VLDB Endowment*, vol. 8, no. 9, pp. 938–949, 2015.

[201] G. Jeh and J. Widom, "Mining the space of graph properties," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 187–196.

[202] S. Asadi, X. Zhou, and G. Yang, "Using local popularity of web resources for geo-ranking of search engine results," *World Wide Web*, vol. 12, no. 2, pp. 149–170, 2009.

[203] Y. Dourisboure, F. Geraci, and M. Pellegrini, "Extraction and classification of dense implicit communities in the web graph," *ACM Transactions on the Web (TWEB)*, vol. 3, no. 2, p. 7, 2009.

[204] X. Qi and B. D. Davison, "Web page classification: Features and algorithms," *ACM Computing Surveys (CSUR)*, vol. 41, no. 2, p. 12, 2009.

[205] F. Scarselli, A. C. Tsoi, M. Hagenbuchner, and L. Di Noi, "Solving graph data issues using a layered architecture approach with applications to web spam detection," *Neural Networks*, vol. 48, pp. 78–90, 2013.

[206] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[207] H. Fryer, S. Stalla-Bourdillon, and T. Chown, "Malicious web pages: What if hosting providers could actually do something," *Computer Law & Security Review*, vol. 31, no. 4, pp. 490–505, 2015.

[208] M. Trevisiol, L. M. Aiello, P. Boldi, and R. Blanco, "Local ranking problem on the browsegraph," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015, pp. 173–182.