Paper ID (631)

Detecting Arabic Web Spam

Heider A. Wahsheh[1], and Mohammed N. Al-Kabi[2]

[1-2]*Computer Information Systems Department, IT & CS Faculty, Yarmouk University*

P.O. Box 566, 21163 Irbid, Jordan

[1]wahshehha09@student.yu.edu.jo

[2] mohammedk@yu.edu.jo

Abstract— *Web spamming defined as the actions that mislead search engines into ranking some pages higher than they deserve, this results inaccurate of the information quality on the Web, the amount of Web spam has increased and the spammer techniques is improved, all these reasons make the wrestling spam necessities.*

This paper discuss the current spamming techniques, ranking algorithms for Web pages, applying three algorithms that detected Arabic spam pages, and comparison between their different result, which show K-nearest neighbour is better than other used algorithms.

Index Terms— *spam, content based, Arabic spam, Arabic Web pages*

I. INTRODUCTION

For the wide spread of Web spamming that is used to mislead search engine users, the basis of the solution to this problem is that we know spammer methods, and algorithms associated with their work, which need to identify and fight the different techniques the hackers manipulate is growing as an insistent need.

The Arabic language is the official language of 22 countries, and the fifth most spoken language. It is one of the official languages of the United Nations. It is one of the Semitic languages, so it is written from right to left. It is based on 28 letters, where these letters are adopted by other languages such as Urdu, Persian, Malay, and Pashto [1].

The Arab internet users are found mainly in the Middle East and North Africa (MENA) and constitute around 5% of world population, and around 3.3% of the world internet users. Arabic Web materials do not exceed 1%, and most of these are published within blogs, so the number of Web pages with valuable information is small relative to large number of Arabic Web pages within blog [2].

One of the main problems facing search engines in MENA is the lack of large number of Arabic Web pages with valuable information, and this is clear within the free encyclopaedia (Wikipedia) which enables internet users to create and edit different articles, where the Arab contribution does not exceed 1% in best cases [3].

Many studies are conducted to explore different techniques to discover Web spam, but none of these is dedicated to Arabic Web pages. This study aims to detect Arabic spammed Web pages using content based analysis, by applying several algorithms and compare the results and achieve the best possible result to represent the best solutions.

There is no corpus for Arabic Web spam pages, so we enforced to collect around 400 Arabic Web pages, 202 of them is spammed Web pages, and assigning the attributes of them. Afterward the attributes of the spammed Web pages are analysed.

When we solve this problem we gain the benefits of saving time, effort and getting the required results fast and directly are the basic yields the Web users look forward and the motives at the anti-spammers side to build a high quality search systems.

II. LITERATURE REVIEW

Web spamming can be defined as the actions that mislead search engines into ranking some pages higher than they deserve [4], these results in degradation of the information quality on the Web, placing the users at risk for exploitation by Web spammers [5] and damaging the reputation of search engines as they weaken the trust of their users [6].

Many techniques are used by Web spammers to deceive the users, these types are classified as: link spam, content spam, and cloaking [4].

Link spams are considered as links between pages that are present for reasons other than merit, consisting of the creation of a link structure to take advantage of link-based ranking algorithms, such as PageRank, which gives a higher ranking to a Website the more other highly ranked Websites linked to it [7].

The most popular link-based Web ranking algorithms, like PageRank and TrustRank rely on a fundamental assumption that the quality of a page and the quality of a page's links are strongly correlated, that a high ranked Web page will be unlikely to contain lower quality links. This also opens doors for spammers to create link-based Web spam that manipulate links to the advantage of the Web spammers. Accordingly, two common link-spam scenarios are [5]:

Link hijacking: is a technique for link spamming, which legitimate reputable pages and inserting links that point to a spammer-controlled page, it appears to link analysis algorithms that the reputable page embraces the spam page [5].

Honeypots: this is an indirect way to spam a link by creating legitimate-appearing Websites which are called honeypots; a decoy or a trap to induce reputable pages to voluntarily link to these spammer-controlled pages. A honeypot can then pass linkages to spam pages [5].

Content spam or it is called term spam, are techniques that tailor the contents of text fields in order to make spam pages relevant for some queries [4]. Basically they tailor the contents of the text fields in HTML pages to make spam pages more relevant to some queries or repeating some important terms and dumping any unrelated terms [8].

The common text field for a page is the HTML tags including: the document body, the title, the Meta tags in the HTML header, and page URL. In addition, the anchor texts associated with URLs that point to pages [4].

Each of these text fields has a spamming target, for example, in the case of body spam. The spam terms are included in the document body and repeat them as a key stuffing. Arabic spammer sometimes uses English characters, which meets with the Arabic letters on the keyboard, repeating them seeking to raise the PageRank of the Arabic Web pages. This spamming technique is among the simplest and most popular ones, and it is almost as old as search engines themselves. In the title spam, higher weights to terms that appear in the title of a document are given, so Arabic spammers repeat the same words many times in the title and repeat the English characters; which that sharing the Arabic letter on keyboard clicks and put them in the title of Web pages in order to raise PageRank. The Meta tag spam appears in the document header. Because of the heavy spamming, search engines currently give low priority to these tags, or even ignore them completely.

And just as with the document title, search engines assign higher weight to anchor text terms, as they are supposed to offer a summary of the pointed document. Therefore, spam terms are sometimes included in the anchor text of the HTML hyperlinks to a page [4].

Some search engines also break down the URL of a page into a set of terms that are used to determine the relevance of the page. Spammers benefit from that by creating long URLs that include sequences of spam terms [4].

The algorithms used by search engines to rank Web pages based on their text fields use the fundamental term frequency-inverse document frequency (TF-IDF) metric used in information retrieval [4].

The TF-IDF function is a weight often used in text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the TF-IDF weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query [9].

With TF-IDF scores in mind, spammers can either make a page relevant for a large number of queries (to receive a non-zero TF-IDF score) by including a large number of distinct terms in a document, or make a page very relevant for a specific query (to receive a high TF-IDF score) by repeating some targeted terms [4].

Cloaking is a technique in which the content presented to the search engine crawls is different to that presented to the browser of the user [10].

This is done by delivering content based on the IP addresses or the user-agent HTTP header of the user requesting the page. When a user is identified as a search engine spider, a server-side script delivers a different version of the Web page, one that contains content not present on the visible page. The purpose of cloaking is to deceive search engines so they display the page when it would not otherwise be displayed [10].

Cloaking consists of sending different content to a search engine than to the regular visitors of a Web site. The version of the page that is sent to the search engine usually includes content spam, and can be detected using content-spam detection methods, or by comparing the indexed version of a page to the page that users actually see [6].

III. RANKING ALGORITHMS FOR WEB PAGES

As one of the most important algorithms in the modern search engines, PageRank algorithm has been widely used in many search engine systems [11].

It is a measure of Web page's relevance, first introduced by Brin and Page, the google's founders.

Google classifies the Web page's according to the pertinence scores given by PageRank, which are computed from the graph structure of the Web.

A page with a high PageRank will appear among the first items in the list of pages corresponding to a particular query. It is not surprising that some Webmasters want to increase the PageRank of their Web page's in order to get more visits from Web surfers to their Website. Since PageRank is based on the link structure of the Web, it is therefore useful to understand how addition or deletion of hyperlinks influences it [12].

PageRank could be thought of as a model of user behaviour. It assumes that there is a random surfer.

Starting from a randomly given Web page, people usually keeps clicking on the forward links, never hitting “back” but eventually get bored and start inputting another random Web page. PageRank computes the probability that the random surfer visits a page. The possibility for a Web page to be clicked is determined by several factors: the original importance of the Web page, this determines the possibility of a Web page to be started and the total number of Web pages that link to it, and the importance and the forward link number

of each of these Web pages. The PageRank algorithm could be presented as follow:

$$r(p) = \alpha \times \sum_{q:(q,p)\in\varepsilon} \frac{r(q)}{w(q)} + (1-\alpha) \times \frac{1}{N} \qquad (1)$$

Where $r(p)$ is the PageRank value for a Web page *p*; $w(q)$ is the number of forward links on the page *q*; $r(q)$ is the PageRank of page *q*; *N* is the total number of Web pages in the Web; α is the damping factor; $(q, p) \in \varepsilon$ means that Web page *q* points to Web page *p*.

A page can have a high PageRank if there are many pages that point to it, or if there are some pages with high PageRank pointing to it. This seems very reasonable and practical. However, it is vulnerable to some link-based spamming techniques.

Another rank algorithm is TrustRank. In contrary to the PageRank, this algorithm is based on forward links of Web pages, and assumes that a good Web page usually points to good Web pages, and seldom links to spam Web pages.
It selects a small set of known good pages as the seed pages. Then follow an approach similar to PageRank, the trust score is propagated via forward links to other Web pages. And finally, the pages with high trust scores are selected as good pages.

BadRank is an algorithm used to detect spam Web pages using a principle based on linking to bad neighbourhoods, that is, a page will get high BadRank value if it points to some pages with high BadRank values. While PageRank uses the backward links of a Web page, BadRank gathers information on the forward links of a Web page, so BadRank could be regarded as a reversion of PageRank. The formula of BadRank is given as:

$$BR(A) = E(A)(1-d) + d\sum_{i=1}^{n} \frac{BR(Ti)}{C(Ti)} \qquad (2)$$

where $BR(A)$ is the BadRank value of page A; Ti is a page that page A points to, with $BR(Ti)$ as its BadRank value; $C(Ti)$ is the total number of the backward links of page Ti; d is a damping factor; $E(A)$ is the original BadRank value for page A, which is determined by the spam filter.
Since there are no algorithms to calculate $E(A)$ and there is no clear way to combine BadRank values with other ranking methods such as PageRank is given, we can not judge the effectiveness of this approach [11].

The R-SpamRank algorithm aims to detect spam Web pages by gaining the spam rank value through forward links, which are the links of reverse direction used in traditional link-based algorithm, which means reverse spam rank.
The formula of the algorithm is:

$$RSP(A) = (1-\lambda)I(A) + \lambda\sum_{i=1}^{n} \frac{RSR(Ti)}{C(Ti)} \qquad (3)$$

$$I(A) = \begin{cases} 1 & \text{if A in blacklist} \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

where $RSR(A)$ is the R-SpamRank value of page *A*; λ is a damping factor, which is usually set to 0.85; $I(A)$ is the initial value for page *A*, it is set to 1 if page *A* in the original blacklist, otherwise 0; and Ti is the ith forward link page of page *A*, $C(Ti)$ is the number of in links of Page Ti, $RSR(Ti)$ is the R-SpamRank value of page Ti [11].

IV. Our Methodology

In our work, we want to be able to detecting Web spam, by classifying pages as spam or non spam pages, depending on

content based features using decision tree, Naive Bayes, & nearest neighbour algorithms as shown in Fig. 1.

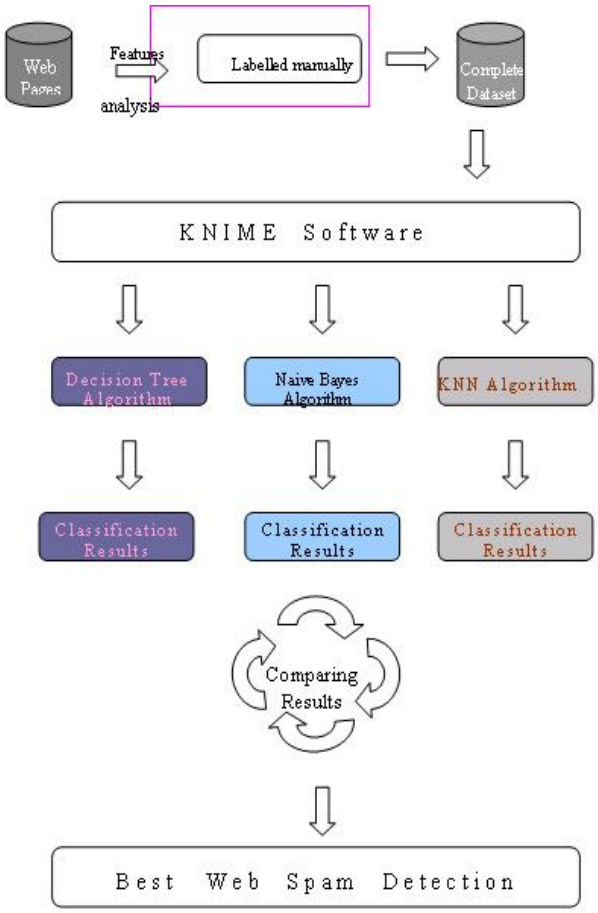


Fig. 1 Methodology Steps

A. Data Collection

Datasets are collected manually, using some online Web pages analysers taken according to Arabic Web pages features that are considered as indicators to the Web page text evaluation.

Our datasets consists of 402 Arabic Web pages, 202 of them Arabic spammed Web pages, where the attributes of the Web pages whether they are spam or non-spam are assigned by the authors. Afterward the attributes of the spammed Web pages are analysed.

B. Features

The features used for classification depends on content based of Web pages, these features serve as hints to spam detection, Such as the number of words in page, the number of words in the page's title, average sentence length (words) [8], these features can be used to detect key stuffing in the Web pages. Also the following new features have been proposed by the authors: Complexity factor of Web page within lexical density; which define in the computational linguistics as an estimated measure of content per functional (grammatical) and lexical units in total.

The formula of compute Lexical Density is:

$$Ld = (Nlex / N) \times 100 \qquad (5)$$

Where Ld is the analysed text's lexical density; $Nlex$ is a number of lexical word tokens in the analysed text; N is a number of all tokens in the analysed text [13]. Readability within Gunning-Fog Index; which measures the readability of language writing, the fog index is commonly used to confirm that text can be read easily by the intended audience. Texts for a wide audience generally need a fog index less than 12. Texts requiring near-universal understanding generally need an index less than 8 [14], the number of different words; which can be used with total count of words to detect the key stuffing in an efficient way and the number of characters in the all title tags in the Web pages.

C. Classification Algorithms

A decision tree is a decision support tool that uses a tree-like model or graph of decisions and their possible consequences. Decision trees are commonly used in operational research

specifically in decision analysis to help identify a strategy most likely to reach a goal [15].

A Naive Bayes classifier assumes that the presence or absence of a particular feature of a class is unrelated to the presence or absence of any other feature, an advantage of it is that it requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification [16].

While the nearest neighbour algorithms (KNN) attractive approach, it assigns the class label to the input pattern based on the class labels that majority of the K-closest (in some distance sense) neighbours from the training set posses. The advantages of these algorithms are summarized by their simple implementation, fastness to train using small training sets and it does not need any a priori knowledge about the structure of the training [17].

KNIME (version 2.1.2) software is used to analyse the datasets. These datasets serve as input to the decision trees, Naive Bayes, and nearest neighbour algorithms.

The KNIME software is read the dataset, partition it, then we use all the above algorithms each one alone, and compute the scorer of the accuracy and the error percentage of each algorithm.

When we used the decision trees with KNIME software we worked in many phases like reading the dataset & partition it into two sub datasets, one is taken by a decision tree learner as a rule to the other sub datasets to predict using decision tree predictor.

Then the predictor decision tree entered the last phase called scorer to compute the accuracy and error percent. In our dataset we achieved accuracy 96.273 %, error 3.727 %, the Fig. 2 show the decision tree results.

Category \ Prediction (DecTree)	non spam	spam
non spam	71	5
spam	1	84

Correct classified: 155 Wrong classified: 6

Accuracy: 96.273 % Error: 3.727 %

Fig. 2 The results of Decision Tree

As the decision trees, the Naive Bayes with KNIME software worked in many phases started from reading the dataset & partition it into two sub datasets, one is taken by a Naive Bayes learner as a rule to the other sub dataset to predict with Naive Bayes predictor.

The Naive Bayes predictor compute the count of spam and non spam Web pages, Gaussian distribution includes mean, standard deviation, and the rate for each attribute of the dataset.

Then the predictor entered to the scorer to compute the accuracy and error percent. In our dataset we achieved 95.031 % accuracy, 4.969 % error. The Fig. 3 summarizes the result of using Naive Bayes .

Category \ Winner(Naive Bayes)	non spam	spam
non spam	82	6
spam	2	71

Correct classified: 153 Wrong classified: 8

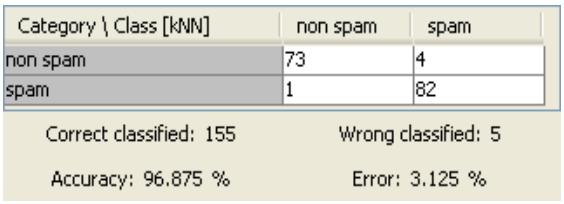
Accuracy: 95.031 % Error: 4.969 %

Fig. 3 Naive Bayes results

The last algorithm is the K-nearest neighbours. The classification performance of this algorithm usually varies significantly with different values of K, and the value of K implicitly indicates the space of the neighbourhood around the test pattern [17].

The K-nearest neighbour used with KNIME software and worked in many phases started from reading the dataset & partition it into two sub datasets, likes the last algorithms, two sub dataset entered into *K* Nearest Neighbour, assign in it number of neighbours to consider (*K*) .

When we used K=1 this mean we based to the closest neighbour and we achieved 96.875% accuracy, 3.125% error. It is important to know that the value of K can be found by a trade-off that is being made using trial and error procedures, so when we used K=3 we achieved 95% accuracy, 5% error, when we used K=5 we achieved 95.625% accuracy, 4.375 % error, and when we used K=15 we achieved 90.625 % accuracy and 9.375% error. Figures 4, 5, 6, and 7 show the differences of results depending on the value of K.

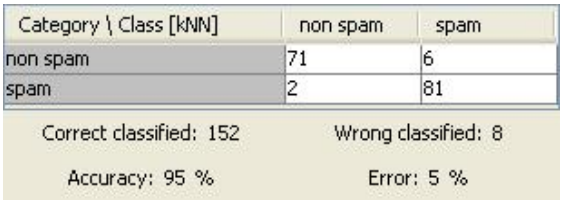


Category \ Class [kNN]	non spam	spam
non spam	73	4
spam	1	82

Correct classified: 155 Wrong classified: 5

Accuracy: 96.875 % Error: 3.125 %

Fig. 4 K-NN results when *K*=1

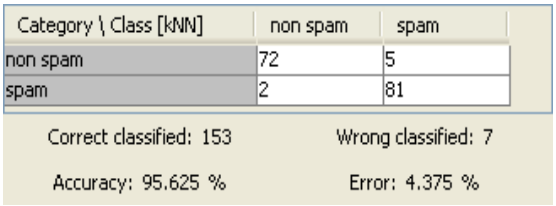


Category \ Class [kNN]	non spam	spam
non spam	71	6
spam	2	81

Correct classified: 152 Wrong classified: 8

Accuracy: 95 % Error: 5 %

Fig. 5 K-NN results when *K*=3

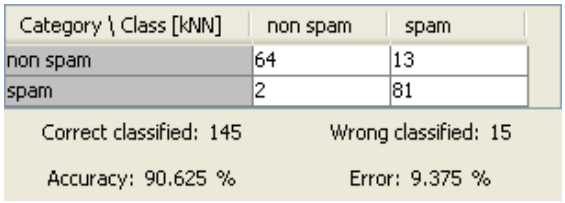


Category \ Class [kNN]	non spam	spam
non spam	72	5
spam	2	81

Correct classified: 153 Wrong classified: 7

Accuracy: 95.625 % Error: 4.375 %

Fig. 6 K-NN results when *K*=5



Category \ Class [kNN]	non spam	spam
non spam	64	13
spam	2	81

Correct classified: 145 Wrong classified: 15

Accuracy: 90.625 % Error: 9.375 %

Fig. 7 K-NN results when *K*=15

V. Conclusions and Future Work

In this paper we discussed the Arabic Web spam, main spam types, some spammer techniques and algorithms of raise ranking Web pages.

Three different algorithms are applied using KNIME software, and the results shows that the K-nearest neighbour algorithm when K=1 is better to be use than the Naive Bayes and decision tree, depending on the accuracy percentage of detecting Arabic Web spam pages .

In any case we need to test and apply more than one model and algorithm on larger dataset of Arabic Web pages as much as possible until we get a high degree of accuracy which enables us to identify Arabic spam pages more correctly depending on content based, and we need to find the ways to detect other spam types.

References

[1] Ryding K. ,(2005) A Reference Grammar of Modern Standard Arabic. http://bilder.buecher.de/zusatz/14/14749/14749960_vorw_1.pdf

[2] Arabic Speaking Internet Users Statistics. Visited on Jan 29, 2011, from http://www.internetworldstats.com/stats19.htm

[3] Gyongyi Z., Garcia-Molina H. & Pedersen J. (2004) Combating Web Spam with TrustRank. *Proceedings of the 30th International Conference on Very Large Databases (VLDB).*

[4] Gyongyi Z. & Garcia-Molina H. (2005) Web Spam Taxonomy. *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web*.
http://airWeb.cse.lehigh.edu/2005/gyongyi.pdf

[5] Caverlee J. & Liu L. (2007). Countering Web Spam with Credibility-Based Link Analysis. *Proceedings of the annual ACM Symposium on principles of Distributed Computing*. **26**, 157-166.

[6] Castillo C., Donato D., Gionis A., GioniS V. & Silvestri F. (2007) Know your Neighbors: Web Spam Detection using the Web Topology. *SIGIR*. 423 - 430 .

[7] Martinez-Romo J. & Araujo L. (2009) Web Spam Identification through Language Model Analysis. *AIRWeb*. 21-28

[8] Wang W., Zeng G. & Tang D. (2010) Using evidence based content trust model for spam detection. *Expert Systems with Applications*.

[9] Tf–idf .Visited on May 6, 2010, from
http://en.wikipedia.org/wiki/Tf–idf

[10] Lin J.L. (2009) Detection of cloaked Web spam by using tag-based methods. *Expert Systems with Applications*. **36**, 7493–7499.

[11] Liang C., Ru L. & Zhu X. (2005) R-SpamRank: A Spam Detection Algorithm Based on Link Analysis.
http://www.mts.jhu.edu/~marchette/ID08/spamrank.pdf

[12] Kerchove C., Ninove L., Dooren P. (2008) Maximizing PageRank via outlinks. *Linear Algebra and its Applications*. **429**, 1254–1276.

[13] Lexical Density. Visited on Jan 29, 2011, from
http://en.wikipedia.org/wiki/Lexical_density

[14] Gunning fog index. Visited on Jan 27, 2011, from
http://en.wikipedia.org/wiki/Gunning_fog_index

[15] Decision tree. Visited on Jan 27, 2011, from
http://en.wikipedia.org/wiki/Decision_tree

[16] Naive Bayes classifier. Visited on Jan 27, 2011, from
http://en.wikipedia.org/wiki/Naive_Bayes_classifier

[17] Sarkar M. (2007) Fuzzy-rough nearest neighbor algorithms in classification. *Fuzzy Sets and Systems*. **158,** 2134 – 21