*2013 International Nuclear Atlantic Conference - INAC 2013*
*Recife, PE, Brazil, November 24-29, 2013*
*ASSOCIAÇÃO BRASILEIRA DE ENERGIA NUCLEAR - ABEN*
***ISBN:*** **978-85-99141-05-2**

# NUCLEAR EXPERT WEB SEARCH AND CRAWLER ALGORITHM

**Thiago Reis[1], Antônio C. O. Barroso[2] and Benedito Filho D. Baptista[3]**

Instituto de Pesquisas Energéticas e Nucleares (IPEN/CNEN - SP)
Av. Professor Lineu Prestes, 2242
05508-000 São Paulo, SP, Brazil
[1] thiagoreis@usp.br
[2] barroso@ipen.br
[3] bdbfilho@ipen.br

## ABSTRACT

In this paper we present preliminary research on web search and crawling algorithm applied specifically to nuclear-related web information. We designed a web-based nuclear-oriented expert system guided by a web crawler algorithm and a neural network able to search and retrieve nuclear-related hypertextual web information in autonomous and massive fashion. Preliminary experimental results shows a retrieval precision of 80% for webpages related to any nuclear theme and a retrieval precision of 72% for webpages related only to nuclear power theme.

## 1. INTRODUCTION

Over the last years the World Wide Web, or simply the Web, has become a ubiquitous and habitual resource for everyday activities of diverse entities, from individuals to entire organizations, obtaining an exponential growth and becoming the largest information repository ever created. Facts and events, news, formal and informal knowledge, people opinions, social interactions are massively registered in general webpages, web blogs, web forums, e-commerce and social networking web sites by a growing number of entities, creating a huge and valuable public information repository. Therefore, the Web represents a new and relevant source of potential useful information, also for nuclear-oriented themes. Nuclear science and technology, nuclear industry and applications, nuclear knowledge management, communications and public acceptance of nuclear energy are examples of topics present in the Web, making it a unique source for the collection and analysis of nuclear-related information from global-wide perspective.

However, the Web is a decentralized and very dynamic environment, built of highly unstructured, noisy and heterogeneous data, and populated with a huge and increasing volume of information, making finding and retrieving its relevant information in large-scale automated manner for use and analysis of its content, surely non-trivial tasks. The mentioned facts present both challenges and opportunities for the research of methods and techniques able to search, retrieve, and evaluate web information. This article reports on web search and crawler algorithm that was developed to accomplish these goals and to retrieve useful webpages touching on nuclear themes, quickly and efficiently, by automatically browsing the Web.

## 2. METHODOLOGY

Information within the Web is mainly organized, presented, and accessed through **webpages** written in **HTML** code. A webpage can reference another webpage (the former called **source webpage**, the latter called **target webpage**) by using a **hyperlink** that connects, describes, and represents the relationship between them both. A hyperlink is defined inside source webpage HTML code by a text such as:

```
<a href="http://www.example.com">example domain</a>
```

Inside hyperlink HTML code there is an **URL** (uniform resource locator), which is the web address of target webpage. URLs are a form of naming and identification of resources (including webpages) on the Web.

Hyperlinks interconnect webpages and therefore the information across the Web, forming the underlying concept of **hypertext** which defines the structure of the Web and makes Web browsing possible. This Web hypertextual environment can be modeled as a directed, connected and sparse graph whose vertices correspond to webpages and edges correspond to hyperlinks that interconnect them [1][2][3][4]. This model is called **webgraph**. Previous studies [5][6][7] have shown important properties of the linkage structure of the webgraph, as summarized below:

- webpages that are linked to each other within the webgraph have a recommendation relation as the author of the source webpage referenced the target webpage by using a hyperlink;
- webpages that are linked to each other within the webgraph are likely to have related or similar content;
- hyperlink text (also known as anchor text) of a source webpage usually describes the target webpage to which it is linked to.

Web crawlers and search algorithms explore the webgraph by recursively retrieving webpages, extracting their hyperlinks, retrieving new webpages the extracted hyperlinks reference, and so on. The basic web search and crawling algorithm formalizes this process, as follows:

1. crawling algorithm begins with one or more URLs that constitute a **seed set**;
2. seed set is added to the **search frontier** – a list of URLs to retrieve;
3. crawler starts a **search iteration** by picking an URL from the frontier and retrieving the webpage at that URL;
4. retrieved webpage is parsed to extract its textual information and the hyperlinks within its HTML code;
5. new URLs inside the extracted hyperlinks are added to the frontier;
6. crawler goes back to step three and performs a new search iteration.

These steps are continuously performed until the search frontier gets empty or until some stop criteria is met.

Web search and crawling algorithms are based on two traditional graph search algorithms, known as **breadth-search** and **best-search**, and use the entire webgraph as them **search space**. To run an **exhaustive search** and retrieve all webpages starting from the seed set, a

web crawler must perform a breadth-first search by using a search frontier that implements a list ordered as a queue to put and pick URLs in **FIFO** (first-in, first-out) order. These crawlers are usually used to search and retrieve as maximum webpages as possible for web archiving and indexing purposes [8]. On the other hand, there are web crawlers designed to search and retrieve only webpages that have some specific content and so they carefully prioritize the URLs in search frontier, controlling the webgraph exploration process. These crawlers – usually called **preferential, focused or topical web crawlers** – focus their search in some predetermined topic or query of interest by estimating the probability that a hyperlink and its URL will lead to a relevant webpage before actually retrieving it. They perform **heuristic searches** by using best-first search algorithms and a search frontier that implements a list ordered as a **priority queue** to put and pick URLs in higher relevance probability order [8]. Search strategies for focused web crawlers are based in **web search heuristics** derived from the webgraph and linkage properties previously described. Previous researches related to web search and crawling methods and algorithms are found in: WebCrawler [9], Naïve Best-First [10], Taxonomy-based Classifier Crawler [11], Bayesian-based Classifier Crawler [12], Fish-Search [13][14], Shark-Search [15], PageRank [16], and InfoSpiders [17][18].

### 2.1. Methodological Framework

In this work, we designed a preferential web search and crawler algorithm which performs heuristic searches in the Web for nuclear-related webpages. The nuclear crawler is designed as an **expert system** [19] (also known as **knowledge-based system**) that integrates sub-components and gives to the crawler the nuclear expert knowledge needed to perform web searches for nuclear-related information, thus emulating web browsing and decision-making ability of a human nuclear expert.

This design comprises two main components: (1) a **knowledge base**, built as a **nuclear vocabulary** by a knowledge engineering process, gathered from the a priori knowledge of **nuclear experts**, which contains **keywords** carefully selected and strongly related to some predefined nuclear topics; and (2) an **inference engine** – a **feed-forward multilayer perceptron artificial neural network** – which performs hyperlinks relevance estimates and is used by nuclear crawler to guide its searches. Knowledge base keywords can be weighted to give more or less weight as they are more or less related to some predefined nuclear topic.

The nuclear crawler uses its knowledge base vocabulary, its neural network inference engine, and implements best-first search algorithms to perform searches in the Web hypertextual environment for webpages whose textual data is similar to a chosen nuclear topic. This topic is specified by a predefined weighted configuration of keywords selected from the nuclear vocabulary. This way, the nuclear crawler is able to find nuclear-related webpages within the large webgraph by following the hyperlinks which interconnect them. This design is illustrated in Figure 1.

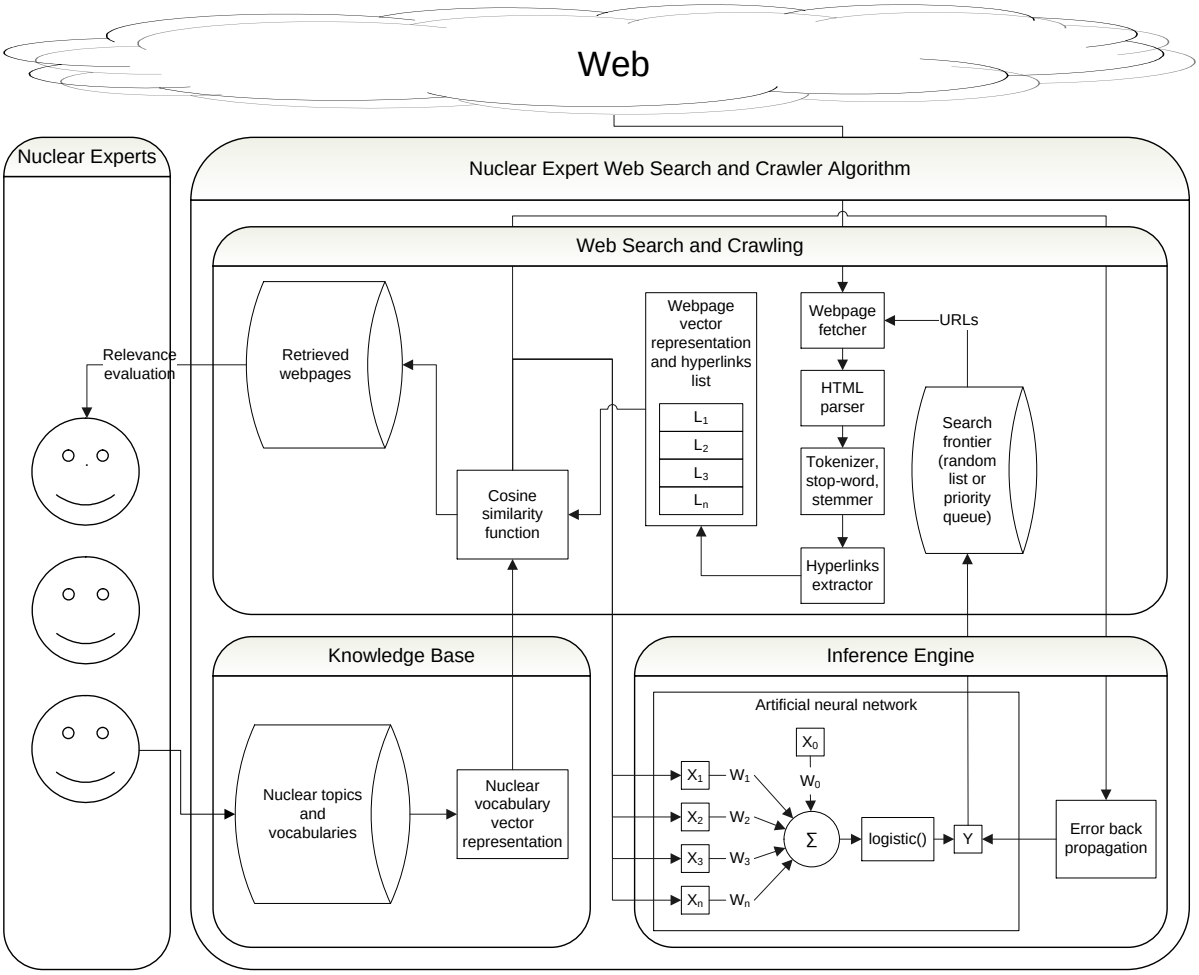


**Figure 1 – Nuclear expert crawler design diagram**

To identify if textual content of a webpage (or some part of it) is related to some nuclear topic vocabulary, the nuclear crawler builds a vector representation of the webpage and nuclear vocabulary by using the **vector space model** [4][20]. Webpage and nuclear vocabulary are viewed as vectors in a large dimensional space where each of their words corresponds to a dimension or axis in the vector space. In this representation, the text is treated as a **bag-of-words** because the ordering of words within text is ignored but the frequencies of each word are considered.

After build the vector representation of webpage and vocabulary texts, the nuclear crawler computes the **cosine similarity function** (Equation 1) of both vectors

$$sim(v,p) = \frac{\vec{V}(v) \cdot \vec{V}(p)}{\left|\vec{V}(v)\right|\left|\vec{V}(p)\right|} \tag{1}$$

where the numerator represents the dot product of vocabulary vector $\vec{V}(v)$ and webpage vector $\vec{V}(p)$, and the denominator is the product of their Euclidian lengths, thus length-normalizing the vectors to unit vectors and making possible to compare similar texts of different lengths. Higher cosine similarity values means that webpage textual content is more similar or more related to the nuclear vocabulary.

While the search process is running, neural network receives as inputs similarity values between nuclear vocabulary and text segments from the hyperlink and source webpage, computed by cosine similarity function, and returns as output the target webpage estimated relevance for some nuclear topic, called **webpage pre-score**, to which the hyperlink present in the source webpage leads. Two main web search heuristics, based on previously presented properties of linkage structure of the webgraph, are used to extract information about target webpage and to build neural network inputs: (1) the text surrounding the hyperlink in the source webpage and (2) the text within the hyperlink. The former is based on the property that webpages linked to each other are likely to have related or similar content and the latter is based on the property that hyperlink text usually describes the target webpage to which it leads.

After target webpage is retrieved from the Web, the nuclear crawler computes the cosine similarity value between nuclear vocabulary and target webpage text vector representations. This similarity value is called **webpage post-score** and is a more accurate relevance estimation, provided as an environment reinforcement signal and used to perform artificial neural network reinforcement learning by computing error back propagation algorithm.

These methods and techniques are combined and used together in the search and crawling algorithm.

### 2.2. Search and Crawling Algorithm

The nuclear crawler algorithm runs in two distinct steps: (1) **training step** and (2) **retrieval step**. Training step is used to build a sample of webpages used to train the crawler neural network by searching the webgraph in a partially random way and retrieving a more diversified set of webpages. Retrieval step is used to effectively search the Web for nuclear-related information by using a trained neural network.

In both steps the nuclear crawler receives as parameters: (1) a nuclear vocabulary containing keywords related to the nuclear topic being searched, (2) a seed set containing starting URLs, (3) the maximum number of webpages to retrieve, (4) a flag indicating if it is a training or retrieval step, and, if training, (5) the number of epochs for neural network training. Figure 2 shows nuclear crawler algorithm pseudo-code.

```
01  Nuclear-Expert-Crawler (knowledgebase, seedset,
02    maxsearch, train, epochs) {
03    if (train = true) {
04      create frontier as hyperlink_random_list;
05    else {
06      create frontier as hyperlink_priority_queue;
07    }
08    create retrieved as webpage_list;
09    create neuralnet as feedforward_neuralnet;
10    create searched as integer;
11    for-each (url in seedset) {
12      url.prescore = null;
13      frontier.add(url, null);
14      while (not frontier.empty and searched <= maxsearch) {
15        searched = searched + 1;
16        create webpage as webpage;
17        hyperlink = frontier.remove();
18        webpage.html = fetch(hyperlink.url);
19        webpage.source = hyperlink;
20        webpage.text = parse_text(webpage.html);
21        webpage.text = tokenize_text(webpage.text);
22        webpage.text = remove_stopword(webpage.text);
23        webpage.text = stemming_text(webpage.text);
24        webpage.prescore = hyperlink.url.prescore;
25        webpage.postscore = similarity(webpage.text, knowledgebase);
26        retrieved.add(webpage);
27        hyperlinks = parse_hyperlinks(webpage);
28        for-each (hyperlink in hyperlinks) {
29          if (not frontier.containsURL(hyperlink.url)
30            and not retrieved.containsURL(hyperlink.url)) {
31            prescore = neuralnet.classify(hyperlink);
32            hyperlink.prescore = prescore;
33            frontier.add(hyperlink, prescore);
34          }
35        }
36      }
37      frontier.clear;
38      searched = 0;
39    }
40    if (train = true) {
41      for-each (webpage in retrieved) {
42        neuralnet.learn(webpage);
43      }
44    }
45  }
```

**Figure 2 – Nuclear expert crawler algorithm pseudo-code**

The nuclear crawler runs, in both steps, a **search cycle** for each URL in the seed set (line 11). In each search cycle, first, the nuclear crawler initializes its search frontier by picking an URL from the seed set and putting it into search frontier (line 13). Next, the nuclear crawler starts a search iteration (line 14) by picking an URL from the frontier (line 17), fetching the webpage at this URL (line 18), processing and analyzing webpage text (lines 20 to 25), and extracting and adding to frontier the webpage hyperlinks (lines 27 to 35). Search iterations are repeated until nuclear crawler reaches the maximum number of webpages to retrieve parameter and then it starts a new search cycle for each remaining URL in seed set.

The search frontier implements a randomly shuffled list in the training step (line 4), where each frontier URL has a uniform probability of being picked at each search iteration. This makes the nuclear crawler training step search less greedy and provides a more diversified webpages training sample for the neural network training. For the retrieval step, a priority queue search frontier is used (line 6) to make the nuclear crawler able to retrieve webpages in descendent order of their pre-score ranking and, thus, performing a greedy search for nuclear-related webpages.

For each extracted hyperlink in search iteration (line 28), the nuclear crawler computes the target webpage pre-score (line 31 to 32), by analyzing source webpage and hyperlink texts with the cosine similarity function and feeding neural network inputs with similarity values. Then, each hyperlink is ordered according to its target webpage pre-score and added to search frontier (line 33).

Neural network inputs are computed as follows (line 31): first neural network input is the cosine similarity value between nuclear vocabulary and hyperlink text vector representations. Second to sixth neural network inputs are the cosine similarity values between nuclear vocabulary and the source webpage text blocks vector representations between current hyperlink and others intervening hyperlinks within the source webpage, up to a maximum window of five hyperlinks away. The last neural network input is the cosine similarity value between nuclear vocabulary and entire webpage text vector representations.

Webpage text processing and analysis is performed as follows: (1) webpage HTML code is parsed to extract webpage textual content (line 20); (2) extracted webpage text stream is tokenized to break it into words, symbols and numbers (line 21); (3) stop-words (extremely common words) are removed from tokenized text (line 22); (4) remaining words are processed by Porter stemming algorithm [21] reducing them to their stem form (line 23); (5) webpage pre-score is assigned by coping the pre-score of the hyperlink which leaded to it (line 24); (6) webpage post-score is computed by the cosine similarity value between nuclear vocabulary and processed webpage text vector representations (line 25).

When running a training step, the nuclear crawler lastly performs the neural network training by running error back propagation algorithm for each retrieved webpage, computing the error between the estimated webpage pre-score and the observed webpage post-score. Then, the trained neural network is saved to be used by the nuclear crawler in future retrieval steps.

### 2.3. Preliminary Experimental Results

In order to evaluate the nuclear crawler algorithm, we designed an experiment with the goal of retrieving webpages with textual content related to two main nuclear topics: (1) nuclear domain in general and (2) nuclear power specifically. A long run web search was performed over thousands of webpages in the Web and evaluation metrics were computed.

A weighted nuclear vocabulary composed of ten terms was used as nuclear crawler knowledge base for both nuclear topics, as shown in Table 1.

**Table 1 – Nuclear Vocabulary**

| Term | Weight |
|---|---|
| nuclear | 10 |
| energy | 5 |
| power | 5 |
| reactor | 2 |
| uranium | 2 |
| atomic | 1 |
| electric | 1 |
| technology | 1 |
| physics | 1 |
| fuel | 1 |

The seed set is built with URLs related to the two main nuclear topics defined and from three web sites: the Wikipedia[1], the World Nuclear Association[2], and the International Atomic Energy Agency[3]; as shown in Table 2.

**Table 2 – Seed Set**

| Seed URL | Nuclear topic |
|---|---|
| http://en.wikipedia.org/wiki/Nuclear_power | Nuclear power specifically |
| http://en.wikipedia.org/wiki/Nuclear_technology | Nuclear domain in general |
| http://world-nuclear.org/info/Current-and-Future-Generation | Nuclear power specifically |
| http://world-nuclear.org/info/Non-Power-Nuclear-Applications | Nuclear domain in general |
| http://www.iaea.org/NuclearPower | Nuclear power specifically |
| http://www-naweb.iaea.org/na | Nuclear domain in general |

Neural network and search parameters were defined as shown in Table 3.

**Table 3 – Neural Network and Search Parameters**

| Parameter | Value |
|---|---|
| Neural network input units | 7 |
| Neural network hidden layers | 1 |
| Neural network hidden neurons | 4 |
| Neural network output neurons | 1 |
| Neural network transfer function | logistic |
| Neural network learning rate | 0.1 |
| Neural network momentum rate | 0.5 |
| Neural network training epochs | 1000 |
| Search cycles (equals # URLs in seed set) | 6 |
| Search iterations (# webpages to retrieve) | 1000 |
| Max URLs in the frontier | unlimited |

[1] http://en.wikipedia.org
[2] http://world-nuclear.org
[3] http://www.iaea.org

Given these parameters, first, a nuclear crawler training step was executed and the neural network training was performed and, next, a nuclear crawler retrieval step was executed and evaluation metrics were computed. In summary, in both runs were performed 6 search cycles, one for each URL in the seed set, with 1000 search iterations for each cycle, thus retrieving a total of 12000 webpages.

The training step neural network sum of squared error is shown in Figure 3.

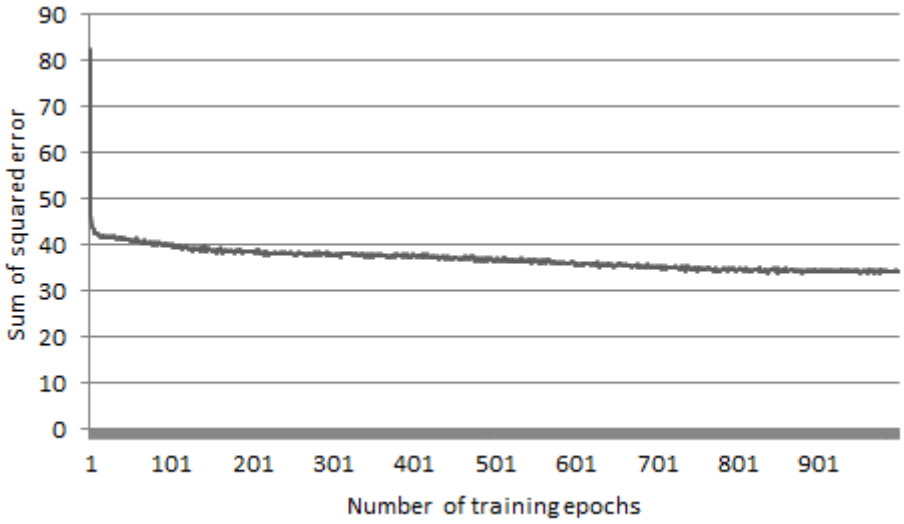


**Figure 3 – Neural network training error over epochs**

Figure 4 shows a comparison plot of the webpage pre-score, computed by the neural network, and webpage post-score, computed by cosine similarity function, in training and retrieval steps. Initially untrained neural network in training step computes high webpages pre-scores and, given that is used a randomly ordered frontier, the webpages post-scores are low, thus causing a high classification error as shown in Figure 5. In retrieval step, the trained neural network more accurately estimates the webpages post-scores, decreasing the classification error and makes the nuclear crawler more efficient to retrieve nuclear-related webpages.

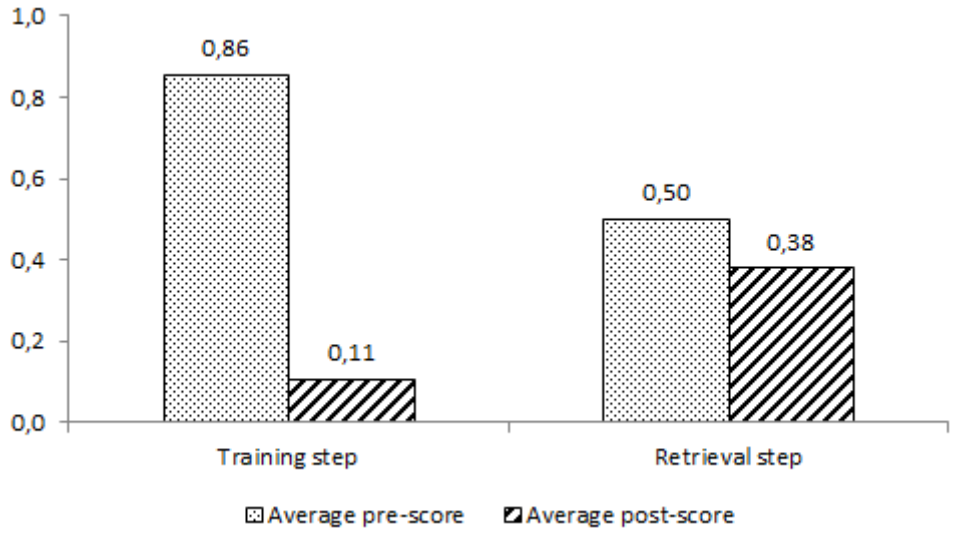


**Figure 4 – Comparison of neural network pre-score versus cosine similarity post-score in training and retrieval steps**

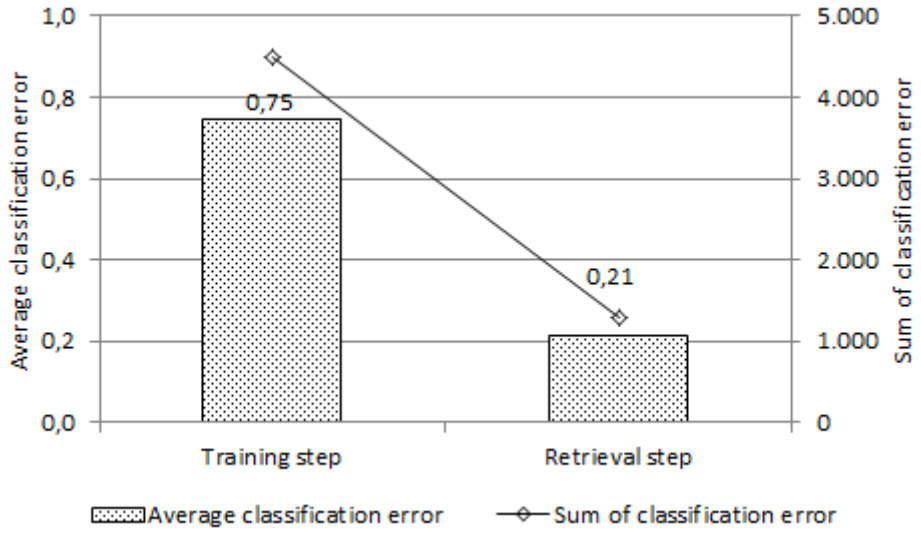


**Figure 5 – Comparison of training step classification error versus retrieval step classification error**

A small subset of 60 webpages was randomly selected from 6000 total retrieved webpages in the retrieval step to build an evaluation sample used to perform nuclear experts' relevance assessments. Two nuclear experts have manually and independently classified each evaluation sample webpage into both nuclear topics, marking yes if the webpages has a textual content related to the nuclear topic or no otherwise. A contingency table as shown in Table 4 was used to plot the nuclear experts' classification results.

**Table 4 – Nuclear Experts Relevance Evaluation Contingency Table**

| | | Nuclear Expert 2 | | |
|---|---|---|---|---|
| | | Yes | No | Total |
| **Nuclear Expert 1** | Yes | $a$ | $b$ | $q_1$ |
| | No | $c$ | $d$ | $q_0$ |
| | Total | $p_1$ | $p_0$ | $n$ |

Two evaluation metrics were computed from contingency table: (1) precision (Equation 2), which is the fraction of retrieved webpages that are relevant to the nuclear experts [4]; and (2) kappa statistic (Equation 5), which is a measure of the magnitude of agreement between nuclear experts, corrected for the rate of chance agreement [4]. Equation 3 is the observed proportion of the times the nuclear experts agreed and equation 4 is the probability that the nuclear experts agreed by chance [4].

$$Pr = \frac{a}{n} \tag{2}$$

$$P(A) = \frac{a + d}{n} \tag{3}$$

$$P(E) = \left(\frac{p_0}{n} * \frac{q_0}{n}\right) + \left(\frac{p_1}{n} * \frac{q_1}{n}\right) \tag{4}$$

$$K = \frac{P(A) - P(E)}{1 - P(E)} \tag{5}$$

Table 5 shows nuclear experts' relevance evaluation results for "nuclear domain in general" topic.

**Table 5 – Nuclear Experts Relevance Evaluation for "Nuclear Domain in General"**

| | | Nuclear Expert 2 | | |
|---|---|---|---|---|
| | | Yes | No | Total |
| **Nuclear Expert 1** | Yes | 48 | 1 | 49 |
| | No | 6 | 5 | 11 |
| | Total | 54 | 6 | 60 |

Table 6 shows nuclear experts' relevance evaluation results for "nuclear power specifically" topic

**Table 6 – Nuclear Experts Relevance Evaluation for "Nuclear Power Specifically"**

| | | Nuclear Expert 2 | | |
|---|---|---|---|---|
| | | Yes | No | Total |
| **Nuclear Expert 1** | Yes | 43 | 1 | 44 |
| | No | 8 | 8 | 16 |
| | Total | 51 | 9 | 60 |

Table 7 shows the evaluation metrics computed from nuclear experts' relevance assessments. In summary, the nuclear crawler algorithm has obtained 80% of precision and a kappa coefficient of 0.53 for "nuclear domain in general" topic, which represent a good precision with a moderate agreement. For "nuclear power specifically" topic, the nuclear crawler has obtained 72% of precision and a kappa coefficient of 0.55, which also represent a good precision with a moderate agreement.

**Table 7 – Evaluation Metrics Summary**

| **Measure** | **Nuclear Domain in General** | **Nuclear Power Specifically** |
|---|---|---|
| Precision | 0.80 | 0.72 |
| Kappa statistic | 0.53 | 0.55 |
| Observed agreement | 0.88 | 0.85 |
| Expected agreement | 0.75 | 0.66 |

## 3. CONCLUSIONS

Although the evaluation procedure was designed experimentally, focusing in nuclear experts relevance assessments, rather than formally over a statistically designed experiment, we consider these results a preliminary positive indicator that the nuclear crawler algorithm is both effective and efficient to search and retrieve relevant nuclear-related web information. Future improvements and assessments should consider testing variations on neural network and search parameters, performance comparison with others crawling algorithms and statistical-oriented results evaluation.

We expect as benefit of this research to develop and provide a potentially useful method for some nuclear-related purposes, such as: retrieving a fraction of the Web, running over limited computational resources, to retrieve webpages often to detect changes, real-time web searches, building large knowledge bases, corpus, and information repositories for post-retrieval analysis, web information discovery, etc.

## REFERENCES

1. A. Brodera, R. Kumarb, F. Maghoula, P. Raghavanb, S. Rajagopalanb, R. Statac, A. Tomkinsb, J. Wienerc, "Graph Structure in the Web," *Computer Networks*, **vol. 33**, pp. 309-320 (2000).

2. C. Cooper, A. Frieze, "A General Model of Web Graphs," *Proceedings on 9th Annual European Symposium on Algorithms (ESA)*, Aarhus, Denmark, August 28-31, pp. 500-511 (2001).

3. C. Cooper, A. Frieze, "Crawling on Web Graphs," *Proceedings on 34th Annual ACM Symposium on Theory of Computing (ACM STOC 2002)*, Montréal, Québec, Canada, May 19-21, pp. 419-427 (2002).

4. C. D. Manning, P. Raghavan, H. Schütze, *An Introduction to Information Retrieval*, Cambridge University Press, Cambridge, United Kingdom (2008).

5. B. D. Davison, "Topical Locality in the Web," *Proceedings of the 23rd Annual International ACM Conference on Research and Development in Information Retrieval (ACM SIGIR 2000)*, Athens, Greece, July 24-28, pp. 272-279 (2000).

6. B. D. Davison, "Topical Locality in the Web: Experiments and Observations," *Technical Report DCS-TR-414*, Department of Computer Science, Rutgers University (2000).

7. N. Craswell, D. Hawking, S. Robertson, "Effective Site Finding using Link Anchor Information," *Proceedings of the 24th Annual International ACM Conference on Research and Development in Information Retrieval (ACM SIGIR 2001)*, New Orleans, Louisiana, USA, September 9-12, pp. 250-257 (2001).

8. G. Pant, P. Srinivasan, F. Menczer, "Crawling the Web," *Web Dynamics*, **vol. 2**, pp. 153-177 (2004).

9. B. Pinkerton, "Finding what people want: Experiences with the WebCrawler," *Proceedings of the 1st International World Wide Web Conference*, Geneva, Switzerland, May 25-27 (1994).

10. F. Menczer, G. Pant, P. Srinivasan, "Evaluating Topic-driven Web Crawlers," *Proceedings of the 24th Annual International ACM Conference on Research and Development in Information Retrieval (ACM SIGIR 2001)*, New Orleans, Louisiana, USA, September 9-12, pp. 241-249 (2001).

11. S. Chakrabarti, M. van den Berg, B. Dom "Focused crawling: A new approach to topic-specific Web resource discovery," *Proceedings of the 8th International World Wide Web Conference*, Toronto, Canada, May 11-14, pp. 1623-1640 (1999).

12. M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, M. Gori, "Focused crawling using context graphs," *Proceedings of the 26th International Conference on Very Large Databases (VLDB 2000)*, Cairo, Egypt, September 10-14, pp. 527-534 (2000).

13. P. De Bra, R. Post, "Information retrieval in the World Wide Web: Making Client-based Searching Feasible," *Proceedings of the 1st International World Wide Web Conference*, Geneva, Switzerland, May 25-27, pp. 183-192 (1994).

14. P. De Bra, G. Houben, Y. Kornatzky, R. Post, "Information Retrieval in Distributed Hypertexts," *Proceedings of the 4th RIAO Conference*, New York, USA, October 11-13, pp. 481-491 (1994).

15. M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, S. Ur, "The Shark-Search Algorithm - An Application: Tailored Web Site Mapping," *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, April 14-18, pp. 317-326 (1998).

16. J. Cho, H. Garcia-Molina, L. Page, "Efficient Crawling Through URL Ordering," *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, April 14-18, pp. 161-172 (1998).
17. F. Menczer, "Complementing Search Engines with Online Web Mining Agents," *Decision Support Systems*, **vol. 35**, pp. 195-212 (2003).
18. F. Menczer, G. Pant, M. Ruiz, P. Srinivasan, "Evaluating Topic-driven Web Crawlers," *Proceedings of 24th Annual International ACM Conference on Research and Development in Information Retrieval (ACM SIGIR 2001)*, New Orleans, Louisiana, USA, September 9-12, pp. 241-249 (2001).
19. P. Jackson, *Introduction to Expert Systems*, Addison-Wesley, Boston, USA, (1998).
20. G. Salton, M. J. McGill, *An Introduction to Modern Information Retrieval*, McGraw-Hill, New York, USA (1983).
21. M. F. Porter, "An Algorithm for Suffix Stripping," *Program: electronic library and information systems*, **vol. 14**, pp. 130-137 (1980).